VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

ANALÝZA A NÁVRH EFEKTIVNÍHO ŘEŠENÍ PRO INTEGRACI WEB APPLICATION FIREWALL DO ARCHITEKTURY SOC

ANALYSIS AND DESIGN OF AN EFFECTIVE SOLUTION FOR INTEGRATION OF WEB APPLICATION FIREWALL INTO SOC ARCHITECTURE

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Vojtěch Hynek

AUTHOR

VEDOUCÍ PRÁCE — Ing. Anna Kubánková, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Bc. Vojtěch Hynek ***ID:*** 220890

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Analýza a návrh efektivního řešení pro integraci Web Application Firewall do architektury SOC

POKYNY PRO VYPRACOVÁNÍ:

Cílem této diplomové práce je analyzovat možnosti integrace Web Application Firewall (WAF) do architektury Security Operations Center (SOC). Práce se zaměří na identifikaci nejčastějších útoků na webové aplikace, zhodnocení stávajících řešení OpenSource WAF a výběr nejvhodnějšího řešení pro potřeby SOC. Dále se práce bude zabývat konfigurací vybrané WAF a tvorbou vlastních pravidel. Výsledkem práce bude nasazení WAF v testovacím a následně reálném prostředí a vyhodnocení vlivu WAF na prevenci a detekci kybernetických hrozeb v rámci SOC.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 21.5.2024

Vedoucí práce: Ing. Anna Kubánková, Ph.D.
Konzultant: Ing. Petr Vychodil

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Diplomová práce se zabývá problematikou integrace Webového aplikačního firewallu do prostředí Dohledového bezpečnostního centra. Výsledkem této práce je analýza současných možností integrace společně s identifikací nejčastějších útoků na webové aplikace. Pomocí provedené analýzy byl vybrán nejlépe vyhovující WAF společně se způsobem jeho integrace. Dále práce obsahuje podrobný popis zvolené integrace a její následné testování. Bylo provedeno testování správnosti fungování firewallu, jeho zátěžové testování a vliv na zpoždění v síti. Část práce popisuje také integraci WAF do reálného prostředí SOC. Integrace obnáší napojení na technologie správy záznamů a provozního monitoringu. Současně byla vytvořena vlastní integrace s platformou MISP, díky které je možné tvořit pro WAF dynamická pravidla. Součástí integrace je vytvoření vlastního parseru, korelačních pravidel a testovacího scénáře. Poslední část práce se věnuje analýze vlivu integrace WAF na prevenci a detekci kybernetických hrozeb, jejíž součástí je zhodnocení vznikajících výstrah za období jednoho měsíce.

KLÍČOVÁ SLOVA

kybernetická bezpečnost, dohledové bezpečnostní centrum, SOC, SOC jako služba, webový aplikační firewall, WAF, prevence a detekce kybernetických hrozeb, Modsecurity

ABSTRACT

The thesis deals with the issue of integration of Web Application Firewall into the environment of Supervisory Security Center. The result of this thesis is an analysis of current integration options along with identification of the most common attacks on web applications. Using the analysis performed, the best suited WAF was selected along with its integration method. Furthermore, the thesis contains a detailed description of the chosen integration and its subsequent testing. Testing of the correctness of the firewall, its stress testing and its effect on the network delay was performed. A part of the thesis also describes the integration of WAF into a real SOC environment. The integration involves connection to log management and traffic monitoring technologies. At the same time, a custom integration with the MISP platform has been developed, which makes it possible to create dynamic rules for the WAF. The integration includes the creation of a custom parser, correlation rules and a test scenario. The last part of the thesis is devoted to the analysis of the impact of the WAF integration on the prevention and detection of cyber threats, which includes an evaluation of the emerging alerts over a period of one month.

KEYWORDS

Cyber security, Security Operations Center, SOC, SOC as a service,Web Application Firewall, WAF, prevention and detection of cyber threats, Modsecurity

HYNEK, Vojtěch. *Analýza a návrh efektivního řešení pro integraci Web Application Firewall do architektury SOC s cílem zlepšení prevence a detekce kybernetických hrozeb*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2024, 89 s. Diplomová práce. Vedoucí práce: Ing. Anna Kubánková, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Vojtěch Hynek
VUT ID autora:	220890
Typ práce:	Diplomová práce
Akademický rok:	2023/24
Téma závěrečné práce:	Analýza a návrh efektivního řešení pro integraci Web Application Firewall do architektury SOC s cílem zlepšení prevence a detekce kybernetických hrozeb

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucí diplomové práce doktorce Anně Kubánkové za její ochotu vést moji diplomovou práci. Především tedy za její vstřícný přístup, cenné rady a věcné připomínky, kterými přispěla ke zdárnému dokončení této práce. Dále bych rád poděkoval odborníkům ze společnosti AXENTA, a. s. za mnoho hodin konzultací a cenné informace, které mi poskytli.

Obsah

Seznam obrázků

Seznam výpisů

Úvod

V době všudypřítomného připojení k internetu a stále větší všeobecné závislosti na webových technologiích se webové aplikace staly nedílnou součástí našeho každodenního života. Slouží jako základní nástroj ke komunikaci, elektronickému obchodování, sociální interakci a ukládání dat. Rozšířenost webových aplikací z nich však také činí atraktivní cíle pro škodlivé subjekty, které se snaží zneužít zranitelnosti pro finanční zisk, krádež dat nebo narušení služeb. Z tohoto důvodu je nutné webové aplikace chránit. K této ochraně lze využít například webový aplikační firewall (WAF), jemuž se tato práce z velké části věnuje.

Cílem této práce je vybrat nejlépe vyhovující řešení integrace WAF do bezpečnostního dohledového centra (SOC) s ohledem na jeho specifické požadavky a následně WAF nakonfigurovat nejlepším možným způsobem pro optimální detekci a prevenci kybernetických hrozeb. Posledním dílčím cílem je nasazení WAF do testovacího a posléze reálného prostředí a vyhodnocení vlivu WAF na prevenci a detekci kybernetických hrozeb v rámci SOC.

Práce je rozdělena do šesti částí. První část popisuje problematiku útoků a hrozeb na aplikační úrovni a přibližuje samotné fungování WAF a SOC. Druhá část se zaměřuje na analýzu současných možností integrace WAF do SOC. Dále je ve třetí kapitole provedena analýza potřeb a požadavků SOC pro samotnou integraci a výběr nejlépe vyhovujícího WAF a způsob jeho nasazení, ať už se jedná o vestavěné nasazení, nebo nasazení pomocí reverzní proxy. Čtvrtá kapitola se věnuje návrhu integrace a popisu jejího detailního provedení včetně testování a validace celého návrhu. Popisuje zátěžové testování i testování správnosti fungování WAF. Předposlední kapitola se věnuje integraci vybraného WAF do reálného prostředí SOC. V rámci ní je popsáno napojení na různé technologie v prostředí SOC. Tato napojení by měla zajistit co nejefektivnější integraci. V poslední kapitole je popsán scénář aktivního využití zvoleného WAF při zabezpečení sítě proti aktuálním zranitelnostem jednotlivých aplikací. Rovněž se v ní nachází vyhodnocení integrace WAF do SOC, které probíhá analyzováním výstrah, jež v rámci daného SOC vzniknou v průběhu jednoho měsíce, ve kterém bude nasazen vybraný WAF.

Hlavním přínosem celé práce by měl být návrh efektivního řešení, jež usnadní implementaci WAF do prostředí SOC. V práci by měl být kladen velký důraz na zlepšení detekce kybernetických hrozeb a vytvoření bezpečnějšího prostředí uvnitř komunikačních sítí.

1 Teoretický přehled

Tato kapitola diplomové práce pojednává o teoretických základech, jež jsou nutné pro pochopení problematiky, kterou se práce zabývá. Je zde popsán základní úvod do kybernetické bezpečnosti. Dále je v kapitole popsáno, jakým způsobem pracují dohledová bezpečnostní centra, jaké mají struktury a jaké technologie a metody využívají. Poslední část je věnována webovému aplikačnímu firewallu a hrozbám, před kterými dokáže komunikační sítě ochránit.

1.1 Kybernetická bezpečnost

Kybernetická bezpečnost v nejširším slova smyslu znamená ochranu sítí, zařízení a dat před neoprávněným přístupem, poškozením nebo ztrátou. Jedná se o rozsáhlý obor, jenž zahrnuje různé disciplíny, činnosti, hrozby a strategie, jejichž cílem je ochrana digitálního života a majetku.

Digitální svět se stal nedílnou součástí našeho každodenního života a kybernetická bezpečnost v něm tedy hraje zásadní roli. Hlavním důvodem je nepřetržité rozšiřování využití digitálních technologií, které v současnosti doprovází každého z nás. Může se jednat o nejrůznější digitální aktiva, jako je digitální měna, data, digitální služby, aplikace anebo neustálý přístup k počítačům prostřednictvím telefonů a nositelné elektroniky. Z toho vyplývá, že existují různé typy kybernetické bezpečnosti, z nichž je každý typ určen k ochraně jiných prvků digitálního ekosystému.

Význam kybernetické bezpečnosti nelze podceňovat. S rostoucí integrací digitálních aktiv do každodenního života roste taktéž míra kyberkriminality. Kybernetické útoky nebo jiná narušení bezpečnosti identifikovalo v roce 2022 39 % podniků a toto číslo se od té doby pravděpodobně nadále zvyšuje.

Kybernetická bezpečnost je důležitá pro všechny, nejen pro velké společnosti s obrovským množstvím dat. K narušení bezpečnosti může dojít i v malých firmách, a dokonce i u jednotlivců. Každý je potenciální obětí, a proto je kybernetická bezpečnost kritickým problémem všech [1].

1.1.1 Bezpečnost informací a systémů

Bezpečnost informací a bezpečnost systémů jsou dva zásadní aspekty kybernetické bezpečnosti. Často se používají zaměnitelně, ale mají výrazné rozdíly a odlišné oblasti zaměření, a proto v této podkapitole jsou podrobně popsány.

Zabezpečení informací neboli InfoSec se zabývá ochranou informací před neoprávněným přístupem, použitím, vyzrazením, narušením, změnou nebo jejich zničením. Tento typ zabezpečení se netýká pouze elektronických dat, ale také informací

ve fyzických formátech, jako jsou například papírové dokumenty.

Hlavním cílem zabezpečení informací je především zajištění důvěrnosti (Confidentiality), integrity (Integrity) a dostupnosti (Availability) informací, často označované jako triáda CIA:

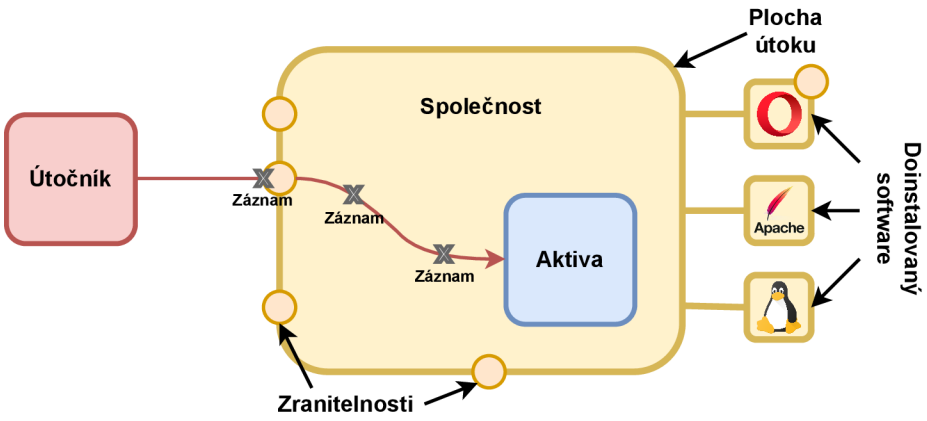
- **Důvěrnost** se zabývá tím, aby přístup k informacím měli pouze ti, kteří k tomu mají oprávnění.
- **Integrita** zajišťuje celistvost a správnost dat. Snaží se zabránit neoprávněným změnám dat.
- **Dostupnost** se zabývá zajištěním přístupu k informacím a datům. Usiluje o zamezení výpadků a útoků, které by mohly přístup k datům omezit.

Systémová bezpečnost je naopak podoblastí kybernetické bezpečnosti, která se zaměřuje na zabezpečení počítačových systémů, včetně ochrany celého systému před hrozbami a zranitelnostmi. To zahrnuje zabezpečení hardwaru, softwaru, dat, sítí a postupů, jimiž se k systémům přistupuje a které se používají.

Zabezpečení systémů se více zaměřuje na technické aspekty zabezpečení, jako je používání firewallů, antivirového softwaru a dalších technických opatření na ochranu systémů před kybernetickými hrozbami. Zahrnuje také procesy a metodiky spojené s udržováním bezpečnosti technologické infrastruktury [1].

1.1.2 Průběh kybernetického útoku

Kybernetický útok obvykle probíhá v několika fázích, z nichž každá má své odlišné charakteristiky. V průběhu těchto fází bude činnost útočníka obvykle generovat záznamy v různých systémech, jež mohou zahrnovat síťové záznamy, systémové záznamy a aplikační záznamy. Tyto záznamy mohou poskytnout cenné informace pro odhalení a vyšetřování útoku. Zkušení útočníci však často podniknou kroky k zahlazení svých stop, například záznamy vymažou, změní časové značky nebo použijí proxy servery, aby skryli svou skutečnou polohu. Než útočník projde všemi těmito fázemi může trvat i několik měsíců. Proto se vyplatí tyto záznamy pečlivě sledovat a pomocí jejich analýzy a hledání podezřelých vzorů a anomálií útočníka odhalit. Průběh útoku je vizualizován na obrázku 1.1. Zobrazuje, jakým způsobem v průběhu útoku vznikají záznamy a zároveň jak se při implementaci nových technologií rozšiřuje plocha útoku. Tato plocha označuje souhrn všech možných bezpečnostních rizik v prostředí organizace. Jde o souhrn všech potenciálních zranitelností (známých i neznámých) a kontrolních mechanismů ve všech hardwarových, softwarových a síťových komponentách [2].



Obr. 1.1: Diagram průběhu kybernetického útoku.

1.2 Bezpečnostní operační centrum (SOC)

Kyberprostor prostupuje veřejným i soukromým sektorem, překračuje geografické i státní hranice a hraje klíčovou roli při utváření globální ekonomiky, obchodu, kultury, společenského blahobytu a bezpečnosti. Největší výzvou při správě kyberprostoru je jeho ochrana před kriminálními a nepřátelskými živly, což představuje výzvu technickou i lidskou.

Pojem *operace kybernetické bezpečnosti* zahrnuje veškeré činnosti věnované zabezpečení určitého segmentu kyberprostoru. Tyto činnosti představují neustálé monitorování a analýzu hrozeb a incidentů, proaktivní nebo reaktivní opatření proti vznikajícím hrozbám nebo obnovu po incidentech. Vzhledem k tomu, že je z podstaty věci nemožné, aby kyberprostor plně zabezpečil pouze jeden subjekt, existuje zásadní potřeba koordinace mezi operačními středisky kybernetické bezpečnosti. Tato operačními střediska provozovaná veřejným i soukromým sektorem vzájemně úzce spolupracují.

Jak již bylo naznačeno, SOC slouží jako centralizovaný bezpečnostní subjekt, jenž pomáhá společnostem při identifikaci, vypořádávání a zmírňování bezpečnostních útoků. V závislosti na požadavcích podniku nebo klienta může SOC převzít také řízení technických kontrol. Hlavní odpovědností SOC je nepřetržité monitorování a zlepšování bezpečnostní pozice organizace tím, že provádí operace kybernetické bezpečnosti pomocí využití technologií a přesně definovaných procesů a postupů [3]. Spojitost mezi personálem, technologiemi a procesy je vyobrazena na obrázku 1.2.

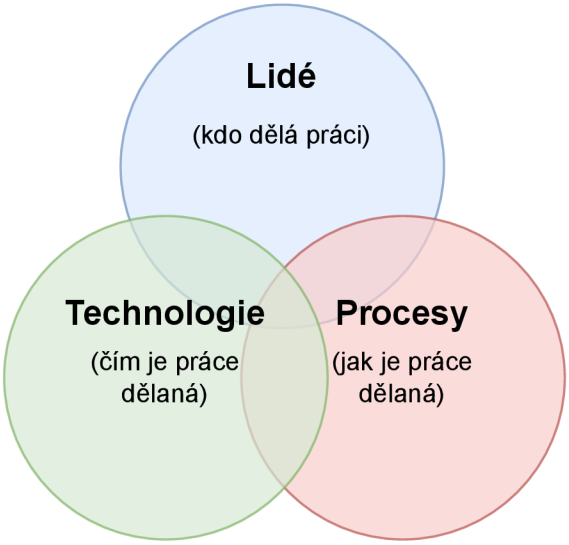
1.2.1 Technologie

Z technologického hlediska má každý SOC dvě základní softwarové části: systém pro správu záznamů (LM) a systém pro správu bezpečnostních informací a událostí pro korelaci (SIEM). Vyspělejší SOC využívají také orchestrační software (SOAR), který propojuje systémy SIEM, informační manažerské systémy (IMS) a další bezpečnostní nástroje a iniciuje aktivní reakce.

- **LM** je základním stavebním kamenem při řešení kybernetické bezpečnosti v mnoha organizacích i bezpečnostních centrech. Stará se o shromažďování a organizaci dat záznamů generovaných různými systémy a aplikacemi. Tento organizovaný přístup usnadňuje účinné monitorování a analýzu událostí souvisejících se zabezpečením. Hlavní odpovědností správy záznamů je zajistit dostupnost, integritu a důvěrnost dat záznamů. To zahrnuje systematické ukládání záznamů, které umožňuje jejich rychlé a efektivní vyhledání v případě potřeby reakce na incident, forenzní analýzy nebo pro účely dodržování interních předpisů nebo zákonů.
- **SIEM** hraje v SOC klíčovou roli, neboť řeší významnou část technologických požadavků. Jeho hlavním úkolem je centralizovaně shromažďovat data důležitá z hlediska bezpečnosti a usnadňovat bezpečnostní analýzy prostřednictvím korelace událostí záznamů. Kromě toho nabízí SIEM další funkce, jako je obohacování dat o kontext, normalizace heterogenních dat a poskytování možností reportování a upozorňování. SIEM také usnadňuje sdílení informací o hrozbách připojením k platformám pro výměnu informací o kybernetických hrozbách. Navíc aktivně zapojuje lidské bezpečnostní analytiky tak, že nabízí možnosti vizuální bezpečnostní analýzy. Nedílnou součástí tohoto komplexního systému je správa záznamů [4].
- **SOAR** je dalším krokem ve vývoji SOC. Řeší širokou škálu technologických potřeb. Jeho hlavní funkce zahrnuje orchestraci a automatizaci bezpečnostních procesů s cílem zvýšit efektivitu a účinnost. Konsolidací dat souvisejících se zabezpečením umožňuje SOAR pokročilou analýzu prostřednictvím korelace událostí, což podporuje proaktivní přístup k zabezpečení. Kromě automatizace poskytuje SOAR funkce, jako je obohacování kontextu, normalizace dat, robustní mechanismy reportování a upozorňování [5].

1.2.2 Lidé

Tým SOC se skládá z několika rolí, z nichž každá má své vlastní odpovědnosti a dovednosti. Tento tým musí efektivně komunikovat se zainteresovanými stranami a vrcholovým vedením, aby mohl eskalovat a předávat hrozící rizika a problémy během běžného provozu a za mimořádných situací.

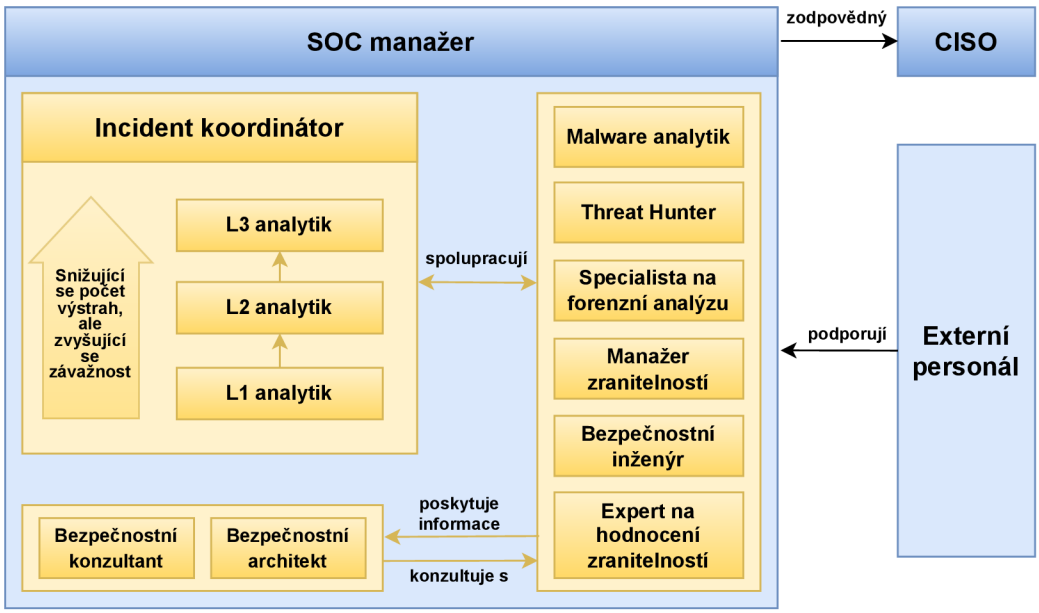


Obr. 1.2: Diagram vizualizující vztah lidí, technologií a procesů.

- **L1 analytik** zpracovává bezpečnostní hrozby v reálném čase a většinou je prvním členem týmu, jenž se s hrozbami setkává. Postupuje podle svých znalostí a dovedností, případně využívá takzvané runbooky, podle kterých postupuje. Jedná se o detailní popis postupu, kterým by se měl Triage Specialista řídit. Runbooky jsou tvořené L2, nebo L3 členy týmu. Pokud by daný problém L1 analytik nedokázal vyřešit, může shromáždit všechny zatím získané informace a problém eskalovat výše na L2 analytika. Dále je první vrstva analytiků zodpovědná za provozní monitoring, při kterém dohlížejí na správné fungování celé architektury SOC s pomocí různých alarmů a výstrah, které hlídají. L1 analytik by měl ovládat znalosti týkající se počítačových sítí a základních principů bezpečnosti. Mezi jeho měkkými dovednostmi by měla být pečlivost a vytrvalost [6].
- **L2 analytik** provádí hloubkovou analýzu bezpečnostních hrozeb. Zpracovává eskalované incidenty od svých L1 kolegů a provádí taktéž úpravu a nastavování různých bezpečnostních výstrah a korelačních pravidel. Aby tuto roli mohl, plnit musí mít velmi široký přehled o trendech v kyberbezpečnosti a neustále se vzdělávat v oblasti aktuálních hrozeb. Dále sepisuje již zmíněné runbooky tak, aby se omezily eskalace incidentů a L1 analytici dokázali většinu vyřešit sami. Pokud se analytik na úrovni 2 setká se závažným problémem, konzultuje ho s dalšími analytiky 2. úrovně, nebo je problém eskalován až na L3 analytika. L2 analytik by měl mít velmi dobré znalosti v principech fungování celého SOC. Pro tuto úroveň je velmi důležité zachovat klid i při stresových situacích a mít analytický a zkoumavý způsob myšlení [6].
- **L3 analytik** je nejzkušenějším pracovníkem v SOC. Řeší nejsložitější inci-

denty, jež byly eskalovány přes všechny nižší úrovně. Skládá se z expertů, kteří jsou specialisté na určité oblasti, například penetrační testování nebo analýzu škodlivého kódu. Díky tomu můžou druhé úrovni analytiků sloužit jako opora při specifických dotazech na dané problematiky. Hlavní náplní L3 analytiků je provádění aktivní činnosti pro hledání bezpečnostních slabin a rizik dané organizace. To je prováděno například pomocí penetračního testování anebo skenů zranitelností. Při následném nalezení nějakého nedostatku L3 analytik navrhne a implementuje řešení, jež daný nedostatek odstraní [6].

- **SOC manažer** dohlíží na tým SOC a v případě potřeby poskytuje technické poradenství. Jeho hlavní odpovědnost však spočívá v efektivním řízení týmu. To zahrnuje činnosti, jako je nábor, školení a hodnocení výkonnosti členů týmu. Dále odpovídá za synchronizaci mezi analytiky a inženýry. Rovněž řídí a organizuje reakci společnosti na hlavní bezpečnostní hrozby.



Obr. 1.3: Diagram vizualizující strukturu SOC.

V rámci této organizační struktury navíc existuje další manažerská pozice na vysoké úrovni: koordinátor reakce na incident. Tato role se věnuje organizování všech činností spojených s reakcí na incidenty a zajišťuje soudržný a efektivní přístup k řešení těchto bezpečnostních incidentů.

Různí bezpečnostní specialisté spolupracují s analytiky SOC, aby zajistili efektivní provoz. Analytici malwaru přispívají prováděním reverzního inženýrství malwaru, a poskytují tak zásadní informace pro reakci na incidenty. Lovci hrozeb aktivně vyhledávají potenciální hrozby v rámci organizace i mimo ni a využívají k tomu

techniky, jako jsou revize protokolů a analýza dat Threat Intelligence (TI).

Další zásadní roli hrají bezpečnostní inženýři (SE), kteří vyvíjejí, integrují a udržují nástroje SOC. Definují požadavky na nové nástroje, spravují přístup k nástrojům a starají se o konfiguraci a instalaci klíčových bezpečnostních systémů. Bezpečnostní inženýři se podílejí také na psaní a aktualizaci detekčních pravidel pro systémy SIEM. Jedná se většinou o odborníky na určitou oblast [7].

1.2.3 Procesy

Procesy jsou důležitou součástí činnosti SOC, protože poskytují strukturu a pokyny pro reakci na bezpečnostní incidenty a přípravu na ně. Jedním ze způsobů, jak strukturovat procesy SOC, je životní cyklus reakce na incidenty, jenž zahrnuje čtyři kroky: přípravu, detekci a analýzu, omezení, eliminaci a obnovu a činnost po incidentu. Tento rámec je uveden v normě ISO/IEC 27035:2016 a v příručce Computer Security Incident Handling Guide od Národního institutu standardů a technologií (NIST).

Příprava zahrnuje vytvoření zásad, postupů a nástrojů pro podporu reakce na incident. Detekce a analýza obsahuje identifikaci a vyhodnocení potenciálních bezpečnostních incidentů. Omezení, eliminace a obnova pojednává o omezení incidentu, odstranění hrozby a obnovení běžného provozu. Činnost po incidentu zahrnuje analýzu incidentu s cílem identifikovat získané klíčové zkušenosti a zlepšit budoucí reakci na podobné incidenty [6].

Je důležité vytvoření komplexního porozumění procesům, úkolům a rozhraním SOC a integrování procesů SOC s ostatními podnikovými procesy. Procesy SOC také vyžadují vysokou úroveň automatizace, aby bylo možné zpracovat velký objem výstrah a dat generovaných různými bezpečnostními nástroji. Nedostatečná automatizace může vést k pomalejší reakci a zvýšit riziko bezpečnostních incidentů [7].

1.3 Firewall

Firewall je klíčovou součástí zabezpečení sítě. Monitoruje a řídí příchozí i odchozí síťový provoz na základě sady předem definovaných bezpečnostních pravidel a funguje jako bariéra mezi důvěryhodnou vnitřní sítí a nedůvěryhodnými vnějšími sítěmi, jako je například internet. Tato bariéra je zásadní pro ochranu citlivých dat a systémů před neautorizovaným přístupem a kybernetickými útoky.

Firewally existují ve více typech podle svých struktur a funkcí. Například firewally paketové vrstvy analyzují provoz ve vrstvě transportního protokolu a hledají škodlivý kód, jenž by mohl infikovat síť nebo zařízení. Firewally na úrovni okruhů

zkoumají data, která prochází během procesu handshake mezi systémy. Brány firewall aplikační vrstvy zajišťují, že na aplikační úrovni existují pouze platná data a teprve poté je propouští. Brány firewall na úrovni proxy serverů, známé také jako brány na aplikační úrovni, zachycují a zkoumají všechny informace, jež do sítě vstupují nebo z ní vystupují. Tímto způsobem poskytují vysoký stupeň kontroly dat, což umožňuje podrobnější filtraci a zabezpečení proti pokročilým hrozbám [8].

1.3.1 Web Application Firewall (WAF)

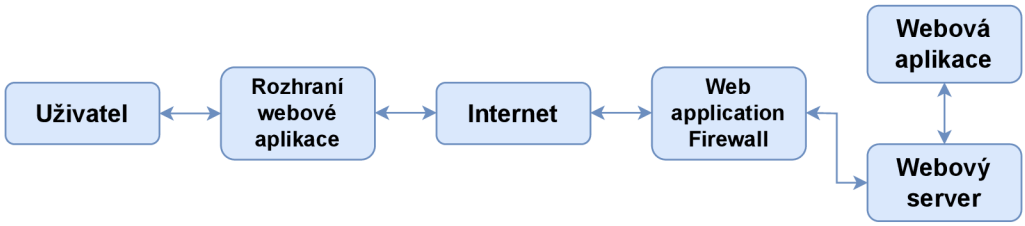
Web Application Firewall zkráceně WAF je brána firewall založená speciálně na filtrování, blokování a monitorování HTTP komunikace mezi internetem a webovou aplikací nebo webovou stránkou. WAF kontroluje každý paket pomocí nastavených pravidel, s nimiž se snaží analyzovat logiku aplikace na 7. vrstvě modelu ISO-OSI. Nejčastěji se jedná o GET a POST požadavky. GET slouží k získání dat z webového serveru a POST k odeslání dat na webový server. Většinou je využívaný velkými společnostmi pro ochranu před útoky nultého dne, nákazou škodlivým softwarem a různými webovými útoky, jako jsou například SQL Injection nebo Cross-site scripting (XSS). WAF se dělí na následující tři typy [9]:

- **Network based** – jedná se o hardwarové zařízení nasazené uvnitř lokální sítě co nejblíže danému webovému serveru. Dosahuje nejnižší latence a většinou i umožňuje replikaci pravidel a nastavení, čímž zprostředkovává rozsáhlé nasazení ve velkých sítích a následnou snadnou konfiguraci a správu. Network based WAF jsou nejdražší variantou a vyžadují také uložení a údržbu fyzického zařízení.
- **Host based** – mohou být plně integrované do softwaru webové aplikace. Jedná se o řešení s nižšími náklady a většími možnostmi přizpůsobení dané aplikaci. Mezi nevýhody patří využití zdrojů na místním serveru, na němž běží webová aplikace, složitější implementace a také potřeba většího množství personálních zdrojů.
- **Cloud based** – jedná se o řešení, které je nejsnazší implementovat. Stačí pouze přesměrovat provoz webové aplikace. Také je to řešení velmi levné, protože uživatel nemá žádné počáteční náklady a za zabezpečení jako službu platí pouze měsíčně nebo ročně. Uživatel se nemusí starat ani o aktualizování systému, před nejnovějšími hrozbami je neustále chráněn. Z některých funkcí nebo z celé implementace se však může stát černá skřínka, do které uživatel nemá vhled.

Existují tři hlavní přístupy, jakými WAF analyzuje a filtruje obsah v HTTP požadavcích, aby se omezila anebo v ideálním případě eliminovala škodlivá aktivita v síti dříve, než se dostane na server ke zpracování. Mezi tyto přístupy patří [10]:

- **Whitelisting** – ve výchozím režimu dochází k odmítání všech požadavků a povolují se pouze takové, které jde považovat za důvěryhodné. Například pomocí seznamu s důvěryhodnými IP adresami, o nichž je dobře známo, že jsou bezpečné. Tento přístup je méně náročný na zdroje, ale jeho nevýhodou je, že může dojít k neúmyslnému blokování neškodlivého obsahu. Jedná se tedy o velmi široký, ale nepřesný přístup.
- **Blacklisting** – u tohoto přístupu se používají přednastavené signatury k blokování škodlivého provozu. Je velmi přesný a pomocí specifických vlastností paketů lze rozhodnout o jeho zahození. Je to vhodné řešení například u webových stránek, jež přijímají velké množství paketů od různých zdrojů, u kterých nemůžeme ověřit jejich důvěrnost. Nevýhodou tohoto řešení je velký nárok na zdroje a vyžaduje více informací než Whitelisting.
- **Hybrid security** – tento přístup využívá a kombinuje prvky obou výše zmíněných přístupů.

Samotný WAF představuje klíčovou složku v zabezpečení webových serverů a aplikací. Neposkytuje pouze ochranu proti útokům na webové aplikace a omezení rychlosti zdánlivě škodlivého provozu. Mezi jeho důležité součásti dále patří i monitorování a generování záznamů, jež může být zásadní v následné analýze potenciálních útoků a hrozeb. Často se jedná i o specializované algoritmy nebo profilování aplikací, které dokáže vyhodnotit nestandardní provoz, a upozornit tak na různé anomálie v běžném provozu. Protože jsou systémy WAF nakonfigurovány na okraji sítě, můžou poskytnout Sítím pro distribuci obsahu (CDN) ukládání webových stránek do mezipaměti, a tím zkrátit dobu jejich načítání [11].



Obr. 1.4: Diagram implemetace WAF.

1.4 Kybernetické hrozby v komunikačních sítích

Kybernetické hrozby v komunikačních sítích představují významný problém, protože mohou narušit komunikaci, způsobit finanční ztráty, ohrozit soukromí a bezpečnost dat. Tyto hrozby lze rozdělit dle jejich typu [12]:

- **Malware** je označení užívané pro škodlivý software. Jedná se o nejčastější hrozbu v kybernetické bezpečnosti. Pod malware spadá široká škála škodlivého softwaru, například spyware, ransomware, viry, červi a trojské koně, které se šíří napříč sítí a mohou způsobit ztrátu dat, ztrátu finančních prostředků, odposlech důvěrných informací nebo narušení provozu. Malware často využívá sofistikovaných metod, aby se vyhnul detekci a infikoval další zařízení.
- **Sociální inženýrství** je dalším důležitým faktorem v kybernetických hrozbách. Jedná se o taktiku, při níž útočník využívá manipulaci, klamání a ovlivňování, aby získal kontrolu nad systémem, osobní informace nebo finanční prostředky. Nejrozšířenější metodou sociálního inženýrství je phishing, nejčastěji se jedná o falešné e-maily a webové stránky určené k oklamání oběti.
- **Útoky na webové aplikace** jsou považovány za značnou hrozbu, neboť tyto aplikace často slouží jako vstupní brány do komunikačních sítí. Kybernetičtí útočníci využívají různé sofistikované techniky, mezi něž patří SQL Injection (SQLi), Cross-Site Scripting (XSS) a Cross-Site Request Forgery (CSRF), s cílem proniknout do webových aplikací. Výsledkem těchto útoků může být únik citlivých dat, poškození integrity aplikace nebo celého webového serveru. Útoky na webové aplikace jsou podrobněji popsány v následující kapitole 1.4.1.
- **DoS a DDoS útoky** jsou útoky zaměřené na přetížení sítě nebo služeb, s cílem vyřadit je z provozu anebo snížit jejich kvalitu. Provádí se prostřednictvím zaslání velkého počtu nelegitimních i legitimních dotazů na server, jenž se odbavováním požadavků zatíží nebo může dojít až k jeho pádu (DoS). K takovému útoku může být využito i více infikovaných zařízení (DDoS).
- **Zneužití datového provozu** může útočníkům poskytnout citlivé informace. K tomu lze použít specializovaný software zvaný packet sniffer, který zkoumá datové pakety a shromažďuje z nich důležité informace. Další rozšířenou metodou, sloužící k zneužití datového provozu, je takzvaný Man in the Middle útok (MitM). Jedná se o techniku, kdy se útočník umístí mezi komunikaci dvou stran, což mu umožní filtrovat, krást a upravovat danou komunikaci.

Kybernetické hrozby mohou být pasivní nebo aktivní. Pasivní hrozby zahrnují činnosti, jako je odposlech a nečinné skenování, jejichž cílem je zachytit síťový provoz. Na druhé straně aktivní hrozby zahrnují činnosti, při kterých se útočník snaží provádět příkazy narušující běžný provoz sítě, jako jsou útoky DoS a útoky SQL injection. Aby útočníci dosáhli svého cíle, musí k provedení série útoků často využít více typů hrozeb. Například se může nejdříve jednat o zaslání phishingových e-mailů za účelem kompromitace větší sady počítačů, na které se následně nainstaluje útočníkem řízený malware. To útočníkovi v architektuře sítě poskytne možnost pohybu do stran, čímž se zmenší pravděpodobnost jeho odhalení a zvýší pravděpodobnost opětovného vstupu do dané sítě. Útočník poté může využít různých zranitelností

a útoků, aby dosáhl svého cíle, ať už to je tiché shromáždění dat, získání vyšších oprávnění anebo narušení provozu v síti [13].

1.4.1 Typy útoků na webové aplikace

Webové aplikace nejsou navzdory své funkčnosti a pohodlí imunní vůči různým bezpečnostním hrozbám. Tyto hrozby nebo útoky mohou ohrozit integritu, dostupnost a důvěrnost webové aplikace a jejích dat. Níže jsou zmíněny některé nejčastější typy útoků na webové aplikace.

SQL Injection

SQL Injection (SQLi) je technika útoku, při níž útočník do dotazu vloží škodlivý kód SQL, který aplikace následně provede. Tím může útočník různými způsoby manipulovat s databází, například číst nebo modifikovat citlivá data v databázi, provádět administrátorské operace nad databází a mnoho dalších operací. Níže je uveden příklad špatně vytvořeného SQL dotazu v teoretickém scénáři, kde existuje databáze obsahující kombinaci uživatelského jména a hesla, jež slouží pro přihlášení uživatele do aplikace.

```
SELECT * FROM users WHERE username = '" + username + " ' AND password
                        = '" + password + " ';
```

Tento kód není bezpečný, protože útočník může zadat hodnoty, které dotaz změní, například `username='admin' --` a `password=[jakákoliv hodnota]`, což povede k dotazu:

```
SELECT * FROM users WHERE username = 'admin' --' AND password =
                        '[jakákoliv hodnota]';,
```

čímž by se efektivně obešla kontrola hesla.

Útoky SQLi představují pro webové aplikace významnou hrozbu, existuje ale velká řada způsobů, jak se před nimi efektivně chránit. Například pomocí předpřipravených příkazů s využitím parametrizovaných dotazů, přidáním takzvaných escape znaků před všechny meta znaky nebo filtrováním legitimních SQL dotazů pomocí WAF [14].

Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) je typ útoku, při němž útočník vkládá škodlivé skripty do webových stránek zobrazovaných jinými uživateli. Tyto skripty mohou krást uživatelská data, manipulovat s webovým obsahem nebo přesměrovávat uživatele na

Výpis 1.1: Ukázka zranitelné HTML konstrukce pomocí XSS.

```
<p>Hello, <span id="name"></span>!</p>

<script>
document.getElementById('name').textContent = name;
</script>
```

škodlivé stránky. Útoky XSS lze rozdělit do tří typů: odražený, uložený a XSS založený na objektovém modelu dokumentu (DOM).

Ve výpisu 1.1 je zobrazena jednoduchá ukázka XSS za předpokladu, že obsah proměnné `name` pochází z neověřeného zdroje od útočníka. Pokud daná proměnná obsahuje skript typu `<script>[škodlivý kód]</script>`, tak se tento skript spustí v prohlížeči každého, kdo na stránku přistoupí [15].

Cross-Site Request Forgery (CSRF)

CSRF (Cross-Site Request Forgery) je kybernetický útok, při němž útočník zneužije důvěryhodnost uživatele na jedné webové stránce k provedení neoprávněných akcí na jiné webové stránce, na kterou má uživatel přístup. Obranou proti CSRF je použití náhodně generovaných tokenů v každém požadavku, což útočníkovi znemožňuje vytvořit platný požadavek bez znalosti tohoto tokenu.

Výpis 1.2: Ukázka teoretického CSRF útoku.

```
<img src="http://bank.com/withdraw?account=bob&amount
   =1000000&for=bob" width="0" height="0" />
```

Ve výpisu 1.2 je zobrazen jednoduchý teoretický CSRF útok. Při tomto útoku se očekává, že je uživatel také v prohlížeči přihlášen ke svému bankovnictví. Tento požadavek by poté mohl provést výběr jeho finančních prostředků bez jeho vědomí [16].

Server-side Remote Code Execution (RCE)

Remote Code Execution (RCE) je kybernetický útok, při kterém útočník získává možnost vzdáleně provádět svůj vlastní kód na serveru nebo v prostředí hostujícím webovou aplikaci. Obranou proti RCE útokům je pravidelná aktualizace systému, správná konfigurace a audit kódu na přítomnost bezpečnostních zranitelností.

Útok RCE lze teoreticky snadno provést pomocí python prostředí viz výpis 1.3. Pokud útočník do proměnné `address` vloží jakoukoli ip adresu, za kterou bude následovat oddělovač `";"` s dalším příkazem, může spustit jakýkoliv příkaz. Například

Výpis 1.3: Ukázka teoretického RCE útoku.

```
import os
os.system("ping " + address)
```

string `"8.8.8.8; ifconfig"` by provedl ping na DNS server společnosti Google a následně zobrazil informace o dostupných síťových adaptérech [17].

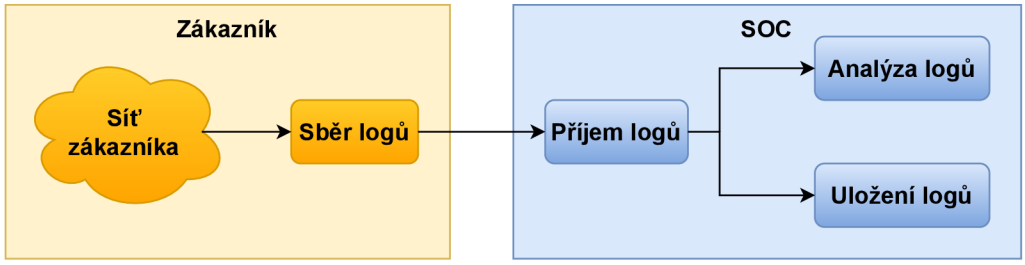
2 Analýza současného stavu

Integrace WAF do architektur SOC se stala nezbytnou součástí obrany proti široké škále hrozeb, včetně zranitelností OWASP Top 10, hrozeb nultého dne a neznámých zranitelností aplikací. Zvolená integrace se využívá stále více, protože umožňuje větší vhled do probíhající síťové komunikace a zabraňuje tak útokům, jež dokáží obejít tradiční síťové firewally. Tato kapitola se zabývá analýzou současných způsobů integrace Webového aplikačního firewallu do architektury Kybernetického bezpečnostního centra. Dále detailně popisuje přehled existujících řešení WAF, jejich zástupců a v neposlední řadě aktuální možnosti testování WAF pomocí nejčastěji používaných webových zranitelností a útoků.

2.1 Současné možnosti integrace WAF do SOC

Cloud computing se stal významným trendem v oblasti sítí díky své schopnosti poskytovat podnikům větší flexibilitu, nákladovou efektivitu a škálovatelnost. Tento trend se projevil i v oblasti síťových firewallů. Cloudové systémy WAF mohou být plně spravovány poskytovatelem daného řešení, samospravovány nebo poskytovány hybridním způsobem, tedy kombinací dvou předchozích možností.

Dalším trendem je využívání automatizace v SOC. Automatizované procesy mohou celý proces detekce kybernetických hrozeb urychlit, a snížit tak dobu rychlosti reakce na daný incident. Rovněž také dokáží usnadnit a zpřesnit identifikaci hrozeb. Automatizace v prostředí SOC je dosaženo pomocí využití pokročilých řešení, jako jsou například SIEM nebo SOAR [18]. Tato řešení byla podrobněji popsána v kapitole 1.2.1.



Obr. 2.1: Diagram architektury SOC jako služba.

Integrace WAF do SOC se bude lišit na základě toho, zda společnost provozuje vlastní SOC nebo využívá SOC jako službu, viz obrázek 2.1. Záleží také na typu WAF, jenž bude do architektury integrován, viz podkapitola 1.3.1.

Pokud se jedná o scénář, v němž společnost provozuje vlastní SOC, bude implementace snadnější. Zejména díky skutečnosti, že společnost zná svoji síť velmi dobře

a může WAF do sítě implementovat i nastavit sama. Pokud by se jednalo o scénář se SOC jako služba a externí SOC by WAF do sítě implementoval, musely by mu být poskytnuty informace o dané síti a současně umožněn veškerý potřebný přístup k ní. Nasazení by pak bylo složitější a trvalo by pravděpodobně déle.

V druhé rovině se nachází problematika typu firewallu, který je nasazen. Pro potřeby SOC jako služby by se nejspíše jednalo o nasazení Network based nebo Cloud based firewallu. Implementace Host based firewallu by mohla připadat v úvahu při využití vlastního SOC společnosti. Vzhledem k velmi snadné komunikaci a spolupráci s vlastním IT týmem společnosti, který by Host based firewall dokázal lépe a snadněji implementovat přímo do aplikace. Nasazení Cloud based firewallu by bylo velmi snadné, jednalo by se pouze o přesměrování komunikace s webovým serverem přes cloudovou službu. U Network based firewallu by nasazení bylo mírně složitější a WAF by se musel implementovat přímo do sítě společnosti, viz obrázek 1.4. Na obrázku můžeme vidět, že by se WAF nasazoval přímo před webový server tak, aby požadavky na webové aplikace nebo stránky procházely nejprve přes WAF a až poté byly webovým serverem zpracovávány.

U všech výše zmíněných možností nasazení je nutné současně provést druhou část implementace. Jedná se o správné nastavení, importování nebo vytvoření všech požadovaných pravidel tak, aby chování WAF odpovídalo předem definovaným požadavkům. Mezi toto nastavení také spadá správný sběr záznamů, jejich následné uložení a zpracování. Toto nastavení je pro SOC klíčové, a proto je důležité věnovat mu velkou pozornost [6].

2.2 Přehled existujících řešení a jejich srovnání

Hlavním rozdílem mezi existujícími řešeními je, zda se jedná o komerční nebo open-source variantu. Obě možnosti mají své výhody i nevýhody, proto je nutné provést analýzu, jež se zaměřuje na otázku, jaké řešení je pro danou společnost a situaci nejvhodnější.

Komerční varianty mohou být pro některé menší společnosti finančně náročné, ale poskytují ve většině případů specializovanou zákaznickou podporu včetně pravidelných aktualizací, záplat a podpory od odborníků. Open-source WAF jsou obvykle zdarma, což z nich činí atraktivní volbu pro organizace s omezeným rozpočtem, a také nehrozí žádné riziko, že bude produkt vázán konkrétními omezeními daného výrobce. Nevýhodou ale bývá absence zákaznické podpory, jež je často nahrazena méně spolehlivou komunitní podporou [19].

2.2.1 Komerční WAF

AWS WAF

AWS WAF je bezpečnostní služba určená k ochraně webových aplikací před běžnými webovými zranitelnostmi, které by mohly ovlivnit dostupnost aplikace, ohrozit její zabezpečení nebo spotřebovat nadměrné množství zdrojů. Tuto službu lze nasadit bez počátečních nákladů a s nízkými provozními náklady, kdy se cena služby odvíjí od skutečně spotřebovaných zdrojů. Nasazení je velmi jednoduché, rychlé a snadno se škáluje. Bohužel se jedná o službu, kterou lze chránit pouze webové aplikace a stránky, jež jsou postaveny na AWS. Dále se jedná o cloudové komerční řešení, které přináší všechny jeho výhody i nevýhody [20].

Nginx App Protect

Nginx App Protect představuje komplexní řešení zabezpečení pomocí WAF a proti DoS útokům. Díky této kombinaci není firewall zbytečně zatěžován případným DoS útokem, o který se postará a zablokuje ho předsazená vrstva ochrany. Slouží k ochraně aplikací a rozhraní pro jejich programování (API). Řešení vyvíjí společnost F5 Networks, jež je také autorem populárního open-source projektu NGINX. Možnost nasazení v mnoha různých prostředích představuje jeho velkou výhodou, včetně možné aplikace ve veřejných a soukromých cloudech nebo virtuálních strojích. Návrh řešení je uzpůsoben pro snadné škálování, díky kterému se hodí pro malá i rozsáhlá nasazení. Mezi hlavní nevýhody patří obtížnější konfigurace a obecné nevýhody plynoucí z uzavřenosti kódu, jako například menší míra přizpůsobení konkrétním požadavkům nebo potencionální omezení výrobcem [21].

Cloudflare

Cloudflare WAF je cloudové bezpečnostní řešení určené k ochraně webových aplikací před širokou škálou kybernetických hrozeb. Dokáže analyzovat provoz a filtrovat škodlivé požadavky na základě vlastních pravidel a strojového učení. Díky tomu efektivně brání společnost před běžnými zranitelnostmi a známými útoky. Hlavní výhodu tohoto řešení představuje integrace s CDN a možnost škálovat řešení pro různé velikosti společností nebo týmů. Mezi jeho nevýhody se řadí velmi omezená možnost přizpůsobení jak oproti open-source řešením, tak i oproti ostatním komerčním variantám. Další nevýhoda vyplývá z cloudové podstaty tohoto řešení, není jej možné implementovat na vlastním webovém serveru. Implementace je velmi jednoduchá, stačí pouze přesměrování své domény na stránky Cloudflare a aktivace vybraného plánu a WAF [22].

2.2.2 Open-source WAF

ModSecurity

Jedná se o open-source WAF, jenž poskytuje ochranu před řadou útoků na webové aplikace. Umožňuje sledování provozu HTTP a analýzu v reálném čase s minimálními změnami stávající infrastruktury.

Mezi hlavní klíčové funkce ModSecurity patří monitorování zabezpečení a řízení přístupu k aplikacím. To je umožněno pomocí přístupu k datovým tokům HTTP provozu a jejich kontrolou. Společně s mechanismem trvalého ukládání dokáže ModSecurity pomocí této funkce provádět korelace událostí v čase. Další důležitou funkcí je kompletní sběr záznamů o provozu, jež umožňuje zaznamenávat nezpracovaná data provozu pro budoucí analýzu nebo forenzní účely. ModSecurity napomáhá také takzvanému hardeningu webových aplikací, jehož cílem je jejich zabezpečení a co největší ztížení zneužití slabých míst pro útočníky. ModSecurity rovněž dokáže průběžně hodnotit bezpečnost, odhalit slabá místa zabezpečení a varovat před nimi dříve, než je útočník dokáže zneužít. Poslední funkcí je velká míra přizpůsobení pomocí vlastního jazyka pravidel, který umožňuje přizpůsobit jeho funkce konkrétním potřebám společnosti [23].

Velkou předností využití ModSecurity jako WAF je již zmíněná možnost přizpůsobení. K tomu napomáhá možnost tvorby vlastních pravidel, díky nimž si může společnost WAF nastavit a odladit tak, aby se minimalizovala tvorba falešně pozitivních bezpečnostních událostí. ModSecurity funguje na široké škále operačních systémů včetně těch nejpoužívanějších, jako je Windows, Linux a Mac OS. Vzhledem k této široké podpoře a faktu, že se jedná o open-source produkt, je implementace do stávající infrastruktury velmi jednoduchá a efektivní.

ModSecurity podporuje dvě různé možnosti nasazení: vestavěné nasazení a nasazení pomocí reverzního proxy serveru.

- **Vestavěné nasazení** znamená spuštění ModSecurity v rámci procesu webového serveru. Tento režim umožňuje ModSecurity chránit místní webový server a zároveň spotřebovávat místní zdroje, jako jsou procesor a paměť RAM. Správa souborů se záznamy a konfigurace se však může stát náročnou, pokud je k dispozici více instalací. Hlavní výhodou vestavěného nasazení je jednoduchost a přímá integrace s webovým serverem. Toto řešení vyhovuje velmi dobře samostatným serverům nebo společnostem s malým počtem serverů, jež lze spravovat samostatně. Mezi nevýhody vloženého nasazení patří potenciální spotřeba prostředků, protože běží v rámci procesu webového serveru, a složitost správy více instalací. Je také důležité uvědomit si, že v tomto režimu může ModSecurity chránit pouze místní webový server, na němž byl implementován.
- **Při nasazení pomocí reverzního proxy serveru** funguje ModSecurity jako

jeho součást. Toto nastavení umožňuje jediné instalaci ModSecurity chránit libovolný počet webových serverů. Hlavní výhodou tohoto režimu nasazení je, že umožňuje centralizovanou správu a ochranu více back-endových serverů. Poskytuje také další vrstvu abstrakce a kontroly pro řízení síťového provozu a nabízí výhody vyrovnávání zátěže. Nevýhodou je, že nasazení reverzního proxy serveru může přinést dodatečné zpoždění kvůli dalšímu skoku v síťovém provozu. Vyžaduje také robustnější infrastrukturu a návrh sítě, aby se reverzní proxy server nestal úzkým hrdlem nebo jediným bodem selhání [24].

Volba nejvhodnějšího řešení by měla vycházet z cílů, požadavků a specifik infrastruktury společnosti. Vestavěné nasazení představuje přímočaré řešení, které je ideální pro menší konfigurace. Na druhé straně, implementace reverzního proxy serveru poskytuje rozšířenou kontrolu a je škálovatelná, což ji činí vhodnou pro rozsáhlejší infrastruktury.

Open-AppSec

Open-AppSec je sada nástrojů vyvinutá k zajištění bezpečnosti aplikací. Je součástí širšího projektu známého jako Application Security (AppSec), jenž se zaměřuje na využití softwaru, hardwaru, technik, osvědčených postupů a procedur k ochraně počítačových aplikací před vnějšími bezpečnostními hrozbami. Open-AppSec je projekt s otevřeným zdrojovým kódem, který zahrnuje několik repozitářů, z nichž každý byl vyvinut v jiném programovacím jazyce. Hlavní kód a logika Open-AppSec jsou vyvinuty v jazyce C++. Díky tomu, že je Open-AppSec šířen jako projekt s otevřeným kódem a využívá různé programovací jazyky, je velmi flexibilní a přizpůsobivý různým vývojovým prostředím. Tyto vlastnosti ale zároveň přinášejí i svá negativa. Při aktualizacích a bezpečnostních záplatách se musí spoléhat na komunitu, jež musí aktualizaci vytvořit a implementovat. To znamená, že hrozí rychlé zastarání projektu. Dále je pro správnou implementaci potřeba vyšší úroveň znalostí, a to z důvodu, že projekt využívá vyšší množství programovacích jazyků [25].

WebKnight

WebKnight je open-source WAF, jenž vyvinula společnost AQTRONiX. Je určený pro webové servery Internet Information Services (IIS) společnosti Microsoft. Jeho hlavní funkcí je blokování škodlivých požadavků skenováním veškerého příchozího provozu na IIS servery. Mezi hlavní funkce tohoto WAF patří administrační rozhraní pro správu a zobrazování statistik. Umožňuje zároveň zaznamenávat všechny požadavky zpracované nástrojem WebKnight včetně těch, které byly zablokovány kvůli potenciálním hrozbám, čímž se při následné analýze provozu v dané síti stává tento nástroj uživatelsky velice přívětivým. Dále je možné provádět aktualizace za běhu

nástroje bez nutnosti restartu IIS serveru po provedení změn. Účinnost systému WebKnight, stejně jako jakéhokoli jiného systému WAF, závisí na jeho konfiguraci. Nesprávně nakonfigurovaná pravidla mohou vést k falešně pozitivním výsledkům, kdy je legitimní provoz označen jako škodlivý, nebo k falešně negativním výsledkům, kdy škodlivý provoz projde bez povšimnutí. Taktéž tento nástroj nabízí stejné výhody a nevýhody vyplývající z jeho open-source architektury –konkrétně tedy jeho velkou míru přizpůsobení potřebám dané společnosti a zároveň schopnost postrádat komplexní podporu a pravidelné aktualizace. Implementace řešení WebKnight můžeme rozdělit na tři základní kroky. Povolení filtrů a rozšíření ISAPI. Stáhnutí a nainstalování nástroje WebKnight. Konfigurace nástroje WebKnight. Jedná se o všestranný a robustní open-source WAF pro IIS servery [26].

Shadow Daemon

Shadow Daemon je sada nástrojů s otevřeným zdrojovým kódem určená k detekci, záznamu a prevenci útoků na webové aplikace. Jedná se technicky vzato o WAF, jenž zachycuje požadavky a filtruje škodlivé parametry, čímž zajišťuje bezpečnost a integritu webových aplikací. Shadow Deamon je modulární systém. To znamená, že byl navržen tak, aby bylo možné přidávat a odebírat různé moduly nebo rozšíření, a zvýšit tak úroveň bezpečnosti. Jedná se například o moduly, jež umožňují provádět detekci útoků, zpracování záznamů, reagovat na dané útoky a mnoho dalšího. Jedná se o aplikace a rozšíření, které jsou napsané v PHP, Perl a Python. Na rozdíl od jiných WAF Shadow Deamon neblokuje škodlivé požadavky úplně. Místo toho se snaží odstranit potenciálně škodlivé části požadavků, a tím omezit falešně pozitivní případy, jež by mohly negativně ovlivnit práci klienta. Další výhodou tohoto řešení je snadná instalace a správa díky jeho přehlednému webovému rozhraní. Oproti tomu nevýhodou tohoto řešení může být obtížnější konfigurace, jež vyžaduje určité znalosti a pochopení jeho vlastností a funkcí [27].

2.3 Aktuální možnosti testování WAF

Testování WAF je nezbytnou součástí zajištění bezpečnosti webových aplikací. Aby bylo zajištěno, že je WAF dobře implementován a nastaven, je třeba jej důkladně otestovat. To zahrnuje různé kroky včetně testování výkonu reálného provozu, automatického testování správnosti konfigurace WAF a kontrol po nasazení. Plnění těchto testů může probíhat ručně, ale není to úplně vhodné, proto se častěji využívají různé automatické nástroje.

2.3.1 Zátěžové testování pomocí simulovaného provozu

Testování výkonu v simulovaném provozu je nezbytné pro pochopení toho, jak se WAF bude chovat po následném nasazení do reálného prostředí. Pro tento typ testování lze použít nástroj WAF Bench (wb). Jedná se o nástroj založený na A/B metodě, jenž se používá k testování výkonu WAF. A/B metoda je založená na testování dvou verzí, aby se zjistilo, která z nich funguje lépe. Nástroj WAF Bench je tedy určen k měření efektivity a účinnosti systémů WAF pomocí simulace různých typů přenosů HTTP. Lze jej použít k vyhodnocení vlivu WAF na efektivitu a škálovatelnost webových aplikací [28].

2.3.2 Testování WAF detekcí

Při tomto druhu testování se kontroluje, jak dokonale je WAF schopný detekovat škodlivý provoz. Jedná se o schopnost odhalení běžných útoků, ale taktéž o zamezení případů špatně zablokovaného legitimního provozu. Jinými slovy je nutné snažit se o co největší omezení falešně negativních případů, kdy WAF nedokáže zablokovat skutečný útok. A zároveň je nutné snažit se o omezení falešně pozitivních případů, kdy WAF nesprávně blokuje legitimní provoz, čímž může docházet k zablokování legitimní komunikace uživatele.

Pro tento typ testování je možné využít různé typy nástrojů, například nástroj Damn Vulnerable Web Application (DVWA). Jedná se o velice zranitelnou aplikaci, jejíž hlavním účelem je sloužit jako pomůcka pro testování automatických procesů, zabezpečení aplikací a dovedností profesionálů v legálním prostředí. Je napsaná v programovacím jazyce PHP a pro své databáze využívá systém MySQL. Tuto aplikaci lze nasadit na webovém serveru chráněném pomocí WAF a prostřednictvím speciálně vytvořené škodlivé komunikace otestovat detekční schopnosti nasazeného WAF [29].

2.3.3 Kontrola po nasazení

Kontrola a testování WAF po nasazení jsou velmi důležité. Pro pomoc s vyhodnocením účinnosti implementovaného řešení lze použít sadu základních pravidel OWASP ModSecurity CRS. Tato sada je navržená speciálně pro ModSecurity WAF, ale je možné ji využít i s jinými kompatibilními firewally. Jedná se o sadu obecných pravidel pro detekci útoků navržených tak, aby chránila webové aplikace před širokou škálou útoků, včetně deseti nejčastějších útoků OWASP. Zároveň jsou pravidla v této sadě optimalizována tak, aby vznikal co nejmenší počet falešně pozitivních výstrah. Po nasazení je vhodné tato pravidla nakonfigurovat, aby se odhalila případná chyba v implementaci firewallu. Případně je lze také využít jako základní konfiguraci pro

následné testování, jež se tímto způsobem urychlí. Je také vhodné z těchto pravidel vycházet a dále je upravit a přizpůsobit podle požadavků organizace. Je důležité úpravu pravidel neopomenout, protože by mohla být útočníkem zneužita k tomu, aby měl komplexní představu o nastavení daného firewallu a bylo by pro něj pak snadnější ho obejít, a vyhnout se tak svému odhalení nebo zaslat škodlivý dotaz do chráněné webové aplikace.

3 Potřeby a požadavky pro integraci WAF do SOC

Jak již bylo zmíněno v kapitole 2.1, SOC je zodpovědný za údržbu, monitorování a ochranu bezpečnosti informací v organizaci. Slouží jako centrum zpravodajství, shromažďuje informace v reálném čase z různých prostředků a využívá je k identifikaci bezpečnostních událostí a k účinné a včasné reakci na ně. Aby mohl tyto úkoly plnit, využívá k tomu různé technologie. Jednou z těchto technologií může být WAF, jenž plní klíčovou roli při ochraně kritických webových aplikací před celou řadou hrozeb. SOC má ale svoje specifické potřeby a požadavky na tyto technologie. Tato kapitola se zaměřuje na jejich analýzu a zároveň také na rizika a hrozby spojené s integrací těchto technologií do infrastruktury.

3.1 Analýza potřeb a požadavků

- **Viditelnost** je jedním ze základních požadavků na integraci WAF do SOC. Integrace by měla zajistit úplný vhled do komunikace a činnosti daného WAF, což zahrnuje údaje týkající se potenciálních útoků, falešných poplachů a narušení bezpečnosti. Čím větší viditelnost samotnému WAF poskytneme, tím větší viditelnost bude mít SOC a kvalita poskytované služby se zvýší. Aby bylo dosaženo co nejvyšší viditelnosti, je důležité, aby všechna komunikace směřující na webové aplikace procházela přes implementovaný WAF a zároveň aby bylo správně nastaveno vytváření záznamů daného firewallu. Viditelnost může být značně ovlivněna i ostatními faktory, například pokud nebude mít SOC dostatečné zdroje, tak nebude schopný zpracovávat všechny záznamy nebo zpracovávat všechnu komunikaci s webovými aplikacemi. Viditelnost dokáže významně ovlivnit také šifrování dat.
- **Ochrana** je hlavním požadavkem pro integraci WAF do SOC. Systém WAF by měl nabízet spolehlivou ochranu proti běžným zneužitím či zranitelnostem a měl by být schopen zabránit útokům, jež obvykle obcházejí tradiční síťové firewally. Pro správné plnění této funkce nejvíce záleží na jeho konfiguraci a vytvoření efektivních pravidel. Snažíme se docílit co největší ochrany s ohledem na spolehlivost a efektivitu. Správná implementace WAF ovlivní velké množství faktorů, pro požadavek ochrany jsou nejdůležitější následující dva:
  - **Detekce hrozeb** je opět velmi závislá na kvalitě napsaných pravidel. Snažíme se, aby detekce hrozeb byla co nejpřesnější a pokud možno aby byly všechny odhaleny. Zároveň by měl být co nejmenší výskyt falešně pozitivních výsledků, jež by dále pouze zmenšovaly efektivitu daného ře-

šení. Aby bylo dosaženo co nejpřesnějších detekčních schopností, je důležité udržovat sady pravidel aktuální, tak aby chránily i před nejnovějšími hrozbami a zároveň netvořily zbytečné falešně pozitivní případy.
  - **Reakce na hrozby** umožňuje zrychlit a zautomatizovat proces vypořádávání se s hrozbami. Pokud tuto schopnost WAF nabízí, jedná se o značné ulehčení pro SOC. Ten je následně schopný rychleji reagovat na hrozby, ať už se jedná o skutečné útoky, nebo falešné poplachy. Mezi takovéto reakce může patřit zablokování potenciálně nebezpečné komunikace, případně její omezení.
- **Rozsah generování záznamů** je požadavkem, který výrazně ovlivňuje všechny již zmíněné požadavky. Od integrace požadujeme co nejširší rozsah, takový, aby nedošlo k ovlivnění spolehlivosti a efektivity. Pokud by k ovlivňování docházelo, je potřeba zaměřit se na generování těch nejrelevantnějších záznamů. Jedná se jak o nastavení tvorby záznamů v případě zablokování komunikace, detekci potenciálně nebezpečné komunikace, povolení komunikace, tak i o vytváření záznamů různých výstrah a výkonu daného WAF.
- **Napojení zdrojů** by mělo být maximálně kompletní, aby se zajistila optimální viditelnost. Jak již bylo dříve zmíněno, veškerá komunikace s webovým serverem, jež vyžaduje monitorování nebo ochranu, by měla projít skrze integrovaný WAF, kde bude náležitě analyzována. Tento požadavek má významný dopad na ostatní aspekty integrace. Před připojením nových zdrojů je důležité pečlivě ověřit, zda existující zařízení a technologie dokážou toto napojení zvládnout bez negativního dopadu na výkon. V případě potřeby může být nezbytné přidat další výpočetní kapacity nebo provést optimalizaci systémů.
- **Výkonnost** dané integrace je ovlivněna mnoha faktory. Zejména rozsahem generování záznamů, kvalitou ochrany a rozsahem viditelnosti. Výkonnost integrovaného systému má zásadní význam pro zachování účinnosti a efektivity operací SOC. Systém WAF by měl být schopen zvládat vysoké objemy provozu, aniž by způsoboval problémy s výkonem. V ideálním scénáři budeme mít dostatek výkonu a výpočetní síly, aby výkonnost nebyla ovlivněna. V reálném prostředí se bude muset jednat o kompromis mezi těmito požadavky.
- **Efektivita** je dalším velmi důležitým požadavkem, který je nutno při integraci webového firewallu zvážit. Při různých způsobech nasazení WAF se efektivita zpracovávání požadavků může velmi lišit. Stejným způsobem se bude lišit i zpoždění, jež vznikne zpracováním komunikace mezi klientem a webovým serverem. Při integraci se samozřejmě snažíme dosáhnout co nejvyšší efektivity a co nejrychlejšího zpracování webové komunikace. Tím bude dosaženo i nejmenšího možného zpoždění. V praxi, kde má nejvyšší vliv způsob nasazení firewallu, je ale velmi náročné zpoždění a efektivitu zlepšit. Různé

způsoby nasazení byly podrobně popsány v kapitole 2.1. Při nasazení firewallu jako proxy vzroste značně efektivita možným využitím vyvažování zátěže společně s navázáním SSL spojení. Na druhou stranu tím vznikne v komunikačním schématu mezi klientem a serverem další skok, jenž může zvýšit latenci dané komunikace [30].

3.2 Identifikace rizik integrace

Integrace jakékoliv nové technologie přináší mnoho nových hrozeb a rizik. Počínaje samotným rozšířením plochy útoku, kterou by mohl útočník využít, až po rizika spojená například se špatnou konfigurací. Tato rizika mohou výrazně ohrozit fungovaní jak SOC, tak i dané společnosti. Proto byla v této podkapitole jednotlivá rizika sepsána a při implementaci s nimi bylo náležitě nakládáno.

- **Plocha útoku** se přirozeně zvýší při jakékoliv integraci nové technologie. Z tohoto důvodu je nutné využít ověřené postupy, jež toto riziko výrazně zmenší. Není možné ho úplně odstranit, ale jeho zmenšení na zanedbatelnou úroveň je dostačující. Mezi tyto nejvhodnější postupy se řadí komplexní analýza rizik, princip nejmenších oprávnění, průběžné monitorování a správa záplat.
- **Chybné nasazení** může ovlivnit správné fungování celé integrace. Jde o velmi podstatný krok pro následnou konfiguraci, proto je nesmírně důležité, aby byl způsob, jakým bude integrace provedena, důkladně popsán a analyzován. Tím se, pokud možno, předejde velkému množství chyb, které by sice nemusely být na první pohled patrné, ale mohly by ovlivnit fungování celého systému. V této práci se detailnímu návrhu řešení integrace věnuje kapitola 4. Dalším způsobem, jak toto riziko omezit, je následné provedení kontroly a testů, které by měly případné chyby odhalit. Kontrola správnosti nasazení je velmi náročná, zejména kvůli specifickým vlastnostem jednotlivých integrací, jež přinášejí unikátní chyby a problémy.
- **Chybná konfigurace** je dalším velkým rizikem, které se při integraci nové technologie může objevit. Mohla by výrazně ovlivnit detekci bezpečnostních incidentů nebo dokonce až omezit legitimní požadavky klientů. Chybnou konfiguraci lze poměrně snadno odhalit a opravit za použití testovacích scénářů, v nichž můžeme simulovat legitimní a nebezpečnou komunikaci, a tím následně i vyhodnotit správnost konfigurace. Detailnímu testování jak samotného nasazení, tak i detekce se věnuje kapitola 4.2 této práce.
- **Chybné detekce** velmi úzce souvisí s chybnou konfigurací či chybným nasazením, pokud bude obojí implementováno správně, nemělo by k chybné detekci dojít. Musíme však počítat s tím, že k chybné detekci dojít může, a proto je dobré mít nastavený určitý proces vyhodnocení správnosti detekcí, pomocí

kterého můžeme jejich počet kontrolovat a v případě nějakého vychýlení včas reagovat [31].

3.3 Výběr nejlépe vyhovujícího WAF

Při výběru nejlépe vyhovujícího WAF pro integraci do SOC bylo vycházeno z potřeb a požadavků popsaných v rámci kapitoly výše. Byly také zohledněny specifické požadavky interního SOC a jeho alternativy, ve které je SOC nabízen jako služba. Dále byl v potaz brán také způsob, jakým bude WAF nasazen.

Nejdůležitějším kritériem je možnost rozsáhlého nasazení, zejména pro větší SOC struktury. Pro zajištění ochrany napříč celou společností musí WAF chránit několik webových serverů nebo webových aplikací současně. Z toho vyplývá požadavek na možnost správy více instancí najednou anebo možnost nasazení jako proxy.

Dalším důležitým kritériem, jež z výše uvedeného vyplývá, je robustnost a efektivita. Z velkého množství webových serverů a aplikací, které budou chráněny, bude vznikat taktéž velké množství komunikace, která bude muset být analyzována.

Třetím kritériem je snadnost integrace do již vytvořené technologické struktury SOCu a míra možného přizpůsobení daného WAF. Každá společnost a její SOC má speciální požadavky, jež lehce upravují jeho strukturu a využívané technologie. Proto je důležité, aby se konkrétní WAF dokázal této struktuře přizpůsobit, a tím správně plnit svoji funkci.

Tato popsaná kritéria nejlépe splňuje ModSecurity. Lze ho nasadit jak v režimu vestavěného serveru, tak i v režimu reverzního proxy serveru, čímž dokáže zajistit ochranu napříč celou organizací. Další důležitou vlastností tohoto WAF je míra přizpůsobení a možnost použití nejrůznějších robustních funkcí. To je zapříčiněno vlastním flexibilním jazykem pro tvorbu pravidel a otevřeností zdrojového kódu. Díky těmto vlastnostem je snadné ModSecurity přizpůsobit nejrůznějším požadavkům společnosti a webových aplikací. Umožňuje širokou škálu využití od monitorování aplikací v reálném čase, přes úplné zaznamenávání informací o provozu, až po blokování hrozeb. Dále je Základní sada pravidel (CRS) navržena pro optimální výkon a pomocí přizpůsobení vlastních pravidel se značně urychlí doba ověřování dotazů, což ModSecurity činí vysoce efektivním WAF.

Největší nevýhodou ModSecurity, jež rozhodování o výběru nejlépe vyhovující WAF pro tuto práci komplikovala, je fakt, že k 1. červenci roku 2024 končí oficiální podpora od společnosti Trustwave. ModSecurity se opět stane plně komunitně vytvářeným a udržovaným softwarem. Je tedy možné, že jak aktualizace přinášející nové funkcionality, tak i bezpečnostní aktualizace budou značně omezeny. Komunitní vývoj avšak funguje u velké řady produktů, proto i přes tyto informace byl pro potřeby této práce zvolen právě ModSecurity.

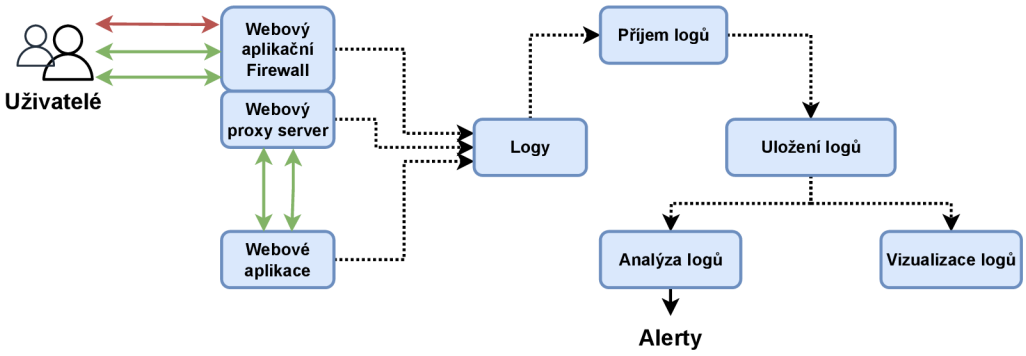
4 Návrh řešení implementace WAF v testovacím prostředí

Tato kapitola se věnuje navržení řešení implementace webového aplikačního firewallu do testovacího prostředí. Je rozdělena do tří hlavních částí, které se zabývají podrobným popisem vytvořeného návrhu, detaily samotného provedení integrace do testovacího prostředí a validací zvoleného návrhu.

4.1 Konceptuální návrh

Diagram 4.1 zobrazuje základní testovací implementaci, jež byla v rámci této diplomové práce vytvořena. Jedná se o řešení požadavků na webovou aplikaci až po agregaci záznamů, které jsou implementací vytvářeny. Druhá část diagramu, týkající se zpracování a vizualizace záznamů, byla demonstrována pomocí takzvaného ELK Stack. V následujících částech této kapitoly je detailně popsáno, jakým způsobem byla implementace provedena.

Testovací implementace byla provedena ve speciálním subnetu SOC na jednom virtuálním serveru. Jedná se o subnet připojený k internetu pomocí šifrovaného VPN kanálu, jenž neobsahuje žádné prostupy do produkčního prostředí, které kvůli tomu nemůže být tudíž nijak ohroženo. Na virtuální server byl nainstalován operační systém Ubuntu Server 22.04. Zároveň má k dispozici 4 CPU, 8 GB RAM a 80 GB HDD. Jedná se o dostatečný výkon pro testovací účely.



Obr. 4.1: Diagram implementace WAF v testovacím prostředí.

První implementovanou částí byl Apache HTTP server sloužící k hostování zranitelné webové aplikace, jejž byl následně rozšířen o WAF ModSecurity a funkcionalitu reverzního proxy serveru. Toto rozšíření bylo velmi snadné, protože ModSecurity je rozšiřující modul Apache HTTP serveru. Jako jednoduchá zranitelná aplikace byla využita DVWA, která nadmíru splňuje požadavky na následné testování funkčnosti

celé integrace. Nastavení reverzní proxy bylo zvoleno zejména z důvodu, že v reálném prostředí bude potřeba směrovat veškerou komunikaci přes nasazenou WAF a tento způsob konfigurace je k tomuto účelu nejvíce vyhovující. Díky němu bude možné ochranu WAF škálovat jak na nové technologie v rámci SOC, tak i na technologie zákazníků SOC.

Následně byl nasimulován provoz směřující na webovou aplikaci. Simulování provozu probíhalo ve dvou fázích, první fází byl zátěžový test a druhou simulace útoků na webové aplikace. Zátěžového testování bylo dosaženo za pomoci nástroje Siege a simulace byly provedeny pomocí vlastních python skriptů.

Poslední částí, kterou se tato integrace zabývá, je generování záznamů všech použitých technologií. Jedná se o nastavení správného generování záznamů a popis, jakým způsobem a kam dané technologie záznamy ukládají. Tyto informace jsou velmi důležité především pro implementaci integrace do reálného prostředí. V tomto reálném prostředí je kritické, aby mezi všemi technologiemi probíhala výměna důležitých informací správně. Všechny záznamy byly následně zpracovány pomocí ELK Stack. Tímto způsobem byl demonstrován způsob sběru a následné analýzy požadovaných záznamů.

4.1.1 Nasazení webového serveru se zranitelnou aplikací

Základním prvkem celého řešení je webový aplikační server. Na tomto serveru dojde k nainstalování zranitelné aplikace, jež poslouží pozdější validaci a testování řešení. Dále zde bude nasazen daný firewall sloužící k filtrování škodlivé a legitimní komunikace proudící od klienta k aplikaci.

K těmto účelům byl vybrán světově nejrozšířenější webový server Apache HTTP. Před konkurencí je upřednostňován díky své flexibilitě, spolehlivosti a rozsáhlé komunitní podpoře. Nabízí také funkce jako například dynamické načítání modulů a rozsáhlé možnosti integrace s dalším softwarem. Těchto funkcionalit plně využijeme při implementaci dalších částí integrace WAF do SOC.

Server Apache má na Ubuntu snadnou instalaci. Apache je k dispozici ve výchozích softwarových repozitářích Ubuntu, tudíž je možné jej nainstalovat pomocí běžných nástrojů pro správu balíčků. Nutností zůstává primárně aktualizace indexů balíčků a následná instalace Apache HTTP serveru a všech potřebných závislostí. Toho bylo dosaženo pomocí příkazů `sudo apt update` a `sudo apt install apache2`. Po instalaci je třeba povolit a spustit danou službu. V případě restartování serveru dojde k přerušení služby. Proto se doporučuje spustit současně také příkaz `enable`. Příkazy pro spuštění a povolení služby Apache jsou následující:

```
systemctl enable apache2
systemctl start apache2.
```

Nyní lze ověřit, zda je webový server správně nainstalovaný, a to pomocí příkazu `systemctl status apache2` nebo otevřením webového prohlížeče s adresou `http://localhost/` na daném serveru, případně s adresou tohoto serveru na jiném zařízení připojeném do stejné sítě [32].

Po ověření funkčnosti webového serveru následovalo doinstalování všech potřebných závislostí:

- mariadb-server
- mariadb-client
- php
- php-mysqli.

Poté byla povolena a spuštěna doinstalovaná služba mariadb.

Instalace zranitelné aplikace

Dalším krokem práce bylo nasazení zranitelné aplikace, pomocí níž se vybraný WAF testoval a validoval. Archiv se zranitelnou aplikací byl stažen z tohoto odkazu `https://github.com/ethicalhack3r/DVWA/archive/master.zip` do adresáře `/var/www`. Po extrahování se složka `DVWA-master` přejmenovala na `html`.

K vytvoření konfigurace DVWA aplikace byla využita defaultní konfigurace, která se nachází ve složce `config` v souboru `config.inc.php.dist`. V této konfiguraci byly pouze změněny dvě části. Jedná se o nastavení přístupu do databáze, kde byly změněny základní přihlašovací údaje a základní nastavení zabezpečení aplikace. To bylo upraveno z *impossible* na *low*. Nastavení zabezpečení udává, jak moc bude daná aplikace zranitelná. Pro potřeby této práce se nejvíce hodí úroveň *low*, jež proces testování a validace významně usnadní. Konkrétní nastavení je zobrazeno v následujícím výpisu.

Výpis 4.1: Ukázka změněného nastavení v defaultní konfiguraci config.inc.php.

```
...
$_DVWA = array();
$_DVWA['db_server'] = getenv('DB_SERVER') ?: '127.0.0.1';
$_DVWA['db_database'] = 'dvwa';
$_DVWA['db_user'] = 'vhynek';
$_DVWA['db_password'] = 'aXj464y54Kl';
$_DVWA['db_port'] = '3306';
...
$_DVWA['default_security_level'] = 'low';
...
```

Podle vyplněných údajů ve výše zmíněné konfiguraci byla vytvořena MySQL databáze. Vytvoření proběhlo pomocí dříve nainstalované relační databáze MariaDB a následujících příkazů:

```
mysql -u root
create database dvwa
grant all on dvwa.* to vhynek@localhost \
 identified by 'aXj464y54Kl'.
```

Následně bylo třeba upravit určité nastavení Apache serveru a změnit oprávnění některých složek DVWA aplikace tak, aby vše korektně fungovalo. Jedná se o nastavení v souboru `php.ini`, jenž se nachází ve složce `/etc/php/8.1/apache2`. V tomto souboru byly změněny položky `allow_url_fopen` a `allow_url_include` na hodnotu *On*. První položka povoluje PHP scriptům použití funkcí jako `fopen()` k otevírání souborů pomocí URL, což poté umožní vzdálený přístup k souborům. Druhá položka dovoluje PHP skriptům zahrnovat soubory ze vzdálených serverů pomocí URL adres. Dále byly na hodnotu `On` změněny rovněž položky `display_errors` a `display_startup_errors`. Jedná se o konfigurační parametry PHP, které řídí, zda se chybová hlášení a varování zobrazí přímo ve výstupu spuštěných PHP scriptů [29].

Výpis 4.2: Ukázka změněného nastavení v základní konfiguraci php.ini.

```
...
allow_url_fopen = On
allow_url_include = On
...
display_errors = On
display_startup_errors = On
...
```

Následně byla změněna oprávnění k zápisu pomocí příkazu `chmod a+w` v následujících složkách.

- hackable/uploads
- external/phpids/0.6/lib/IDS/tmp/phpids_log.txt
- config/.

Nastavení reverzní proxy

Předposledním důležitým krokem je nastavení Apache HTTP serveru jako reverzní proxy. Reverzní proxy server se obvykle používá k předávání požadavků na jiný

server nebo službu. V této práci je reverzní proxy server využitý obdobným způsobem. Umožňuje přesměrování veškeré komunikace, kterou chceme kontrolovat pomocí WAF, na reverzní proxy server, na nějž je zmíněný WAF nasazený. Pomocí tohoto principu je daná implementace velmi jednoduše škálovatelná a při případném budoucím nasazení nových webových aplikací je snadné tyto aplikace chránit.

V tomto konceptuálním návrhu je Apache server nastavený jako reverzní proxy, která pouze odkazuje na samotný Apache server. Jedná se totiž o pouhou demonstraci tohoto principu. V následující kapitole 5 je popsáno kompletní nasazení reverzní proxy, jež všechnu komunikaci přesměrovává na další reálné webové servery využívané v komunikační síti daného SOC.

Před nasazením reverzní proxy bylo nejdříve potřeba povolit potřebné moduly Apache serveru. Jedná se o moduly *proxy* a *proxy_http*, jež slouží k vytvoření reverzní proxy, a moduly *ssl* a *default-ssl*, které zajišťují bezpečnou ssl komunikaci.

Dále byl upraven konfigurační soubor `000-default.conf` nacházející se v adresáři `/etc/apache2/sites-available`. V tomto souboru byla přidána direktiva *ProxyPass* a *ProxyPassReverse*, jež se nastavila tak, aby byly všechny požadavky na kořenovou adresu
přesměrovány na DVWA aplikaci na stejném Apache serveru.

Výpis 4.3: Ukázka změn konfiguračního souboru 000-default.conf.

```
...
ProxyPass / http://192.168.100.10/index.php
ProxyPassReverse / http://192.168.100.10/index.php
...
```

Po aplikaci těchto změn byl Apache server restartován, aby došlo k aktivaci nové konfigurace. Dále bylo nezbytné ověřit, zda je reverzní proxy správně nastavena a zda funguje. Pro tento účel byl proveden přístup na hlavní IP adresu Apache serveru z klienta. Očekávaným výsledkem bylo zobrazení obsahu DVWA aplikace, což signalizuje úspěšné přesměrování požadavků skrze Apache server.

Demonstrace SSL pomocí Self-Signed Certifikátů

Z důvodu zvýšení bezpečnosti a ochrany přenášených dat byl implementován SSL šifrování pomocí certifikátů. Pro demonstraci tohoto postupu byly použity self-signed SSL certifikáty, pro jejichž vygenerování byl využit nástroj OpenSSL. Certifikát X.509 s platností jednoho roku byl vytvořen 2048bitovým RSA klíčem pomocí příkazu `openssl req -x509 -nodes -days 365`. Certifikát a privátní klíč byly přejmenovány a přesunuty do složek `/etc/ssl/certs` a `/etc/ssl/private`. Následně

byl Apache server konfigurován pro použití tohoto certifikátu. V konfiguračním souboru Apache `default-ssl.conf` byly ve složce `/etc/apache2/sites-available` nastaveny cesty k souborům certifikátu a privátního klíče [33].

Výpis 4.4: Ukázka změn konfiguračního souboru default-ssl.conf.

```
...
SSLCertificateFile /etc/ssl/certs/apache-ss.crt
SSLCertificateKeyFile /etc/ssl/private/apache-ss.key
...
```

V posledním kroku bylo provedeno restartování Apache serveru a následné ověření funkčnosti a nastavení DVWA aplikace. Ověření proběhlo na URL adrese `http://192.168.100.10/setup.php`. Jedná se o inicializační stránku DVWA aplikace, na níž byla vizuálně zkontrolovaná všechna potřebná nastavení a odhalena jedna chybějící konfigurace. Jde o konfiguraci reCAPTCHA klíče, který chrání před spamem způsobeným zneužitím automatických nástrojů. Pro tuto práci není dané nastavení důležité, a proto nebylo implementováno. Mimo kontroly nastavení stránka slouží také k vytvoření a resetování databáze, čímž bylo nasazení zranitelné webové aplikace DVWA dokončeno. Pro otestování správné funkčnosti aplikace lze navštívit URL adresu `https://192.168.100.10` a přihlásit se pomocí předem vytvořeného účtu *Admin* s heslem *password*.

4.1.2 Nasazení webového aplikačního firewallu

Výběr nejlépe vyhovujícího webového aplikačního firewallu a jeho kritéria byla popsána v kapitole 3.3. Nainstalování vybraného WAF ModSecurity je velmi jednoduché, protože se nachází v nástroji pro správu balíčků *apt-get*. ModSecurity byl tedy bez problémů nainstalován pomocí příkazu `apt-get install libapache2-mod-security2`. Jelikož se jedná o modul Apache HTTP serveru, pro následnou aktivaci daného modulu byl server pouze restartován.

Druhým důležitým krokem nasazení vybraného WAF byla jeho konfigurace, k níž byla opět využita základní doporučená konfigurace, tak jako v případě nasazení zranitelné aplikace. Nachází se ve složce `/etc/modsecurity` v souboru `modsecurity.conf-recommended`. Pouhým přejmenováním tohoto souboru na `modsecurity.conf` se konfigurace při následujícím znovuspuštění WAF použije. V této základní konfiguraci byla provedena pouze jedna změna. Jedná se o povolení ModSecurity upravením nastavení `SecRuleEngine DetectionOnly` na `SecRuleEngine On`. V předchozím nastavení by firewall škodlivou komunikaci detekoval, ale neprováděl by žádná aktivní opatření, jako je například zablokování komunikace [34].

Třetím a posledním krokem je nasazení pravidel, která bude WAF používat pro své správné fungování. ModSecurity obsahuje pár základních pravidel již od samotné instalace, s těmito základními pravidly ale není příliš užitečný a jeho fungování je značně omezené. Proto je osvědčeným postupem tato pravidla rozšířit alespoň o takzvaný Core Rule Set (CRS), jenž značně pomůže s ochranou před základními typy útoků. Dále je důležité obohatit WAF vlastními pravidly, která budou sloužit k ochraně specifických potřeb dané firmy. Této problematice se věnuje následující kapitola 4.1.3.

Na základě výše uvedeného textu byl stažen OWASP ModSecurity CRS, který by měl poskytnout ochranu před velkým množstvím webových útoků, včetně všech vyskytujících se v OWASP Top Ten. Po stažení tohoto CRS byla jeho pravidla nacházející se ve složce `corerules/rules` přesunuta do složky `/etc/modsecurity/rules`. Posléze byla provedena změněna nastavení ModSecurity v konfiguračním souboru `security2.conf` ze složky `/etc/apache2/modsenabled/`. V tomto souboru se nachází cesty k jednotlivým pravidlům, která budou načtena a použita. Konfigurace souboru je zobrazena ve výpisu 4.5 [35].

Výpis 4.5: Ukázka změněného nastavení v konfiguračním souboru security2.conf.

```
<IfModule security2_module>
        SecDataDir /var/cache/modsecurity
        Include /etc/modsecurity/modsecurity.conf
        Include /etc/modsecurity/crs/crs-setup.conf
        Include /etc/modsecurity/crs/REQUEST-900- \
            EXCLUSION-RULES-BEFORE-CRS.conf
        Include /etc/modsecurity/crs/RESPONSE-999- \
            EXCLUSION-RULES-AFTER-CRS.conf
        Include /etc/modsecurity/rules/*.conf
</IfModule>
```

Po předchozích krocích byl Apache HTTP server znovu restartován a modul ModSecurity se všemi pravidly byl správně načten.

4.1.3 Tvorba vlastních pravidel

Pravidla ModSecurity hrají klíčovou roli při určování způsobu, jakým webový server zpracovává příchozí požadavky. Konkrétní pravidla vyžadovaná pro požadavky HTTP se mohou lišit v závislosti na jedinečných potřebách každého webového serveru. Správci mohou chtít například blokovat určité typy přenosů pro jednu aplikaci a povolit je pro jinou. Kromě toho lze pravidla nakonfigurovat pro správu odpovědí

odesílaných externím uživatelům, čímž se zvýší zabezpečení dat. Dalším důležitým aspektem je nastavení pravidel pro monitorování, jež správce informuje o probíhajících útocích zaměřených na jejich aplikace. Výběr a konfigurace pravidel závisí na aplikacích provozovaných na serveru a na požadavcích společnosti.

Aby bylo zajištěno komplexní pokrytí zabezpečení, jsou pravidla uspořádána do skupin známých jako sady pravidel. To zdůrazňuje kolaborativní povahu pravidel, protože musí být navržena tak, aby fungovala ve spojení s ostatními. Obecně se doporučuje minimalizovat úpravy souborů základních sad pravidel. Místo toho se doporučuje zaměřit se na úpravu souboru s vlastními pravidly specifickými pro daný web. Tento přístup zajišťuje plynulejší a lépe spravovatelný proces údržby konfigurací ModSecurity.

1. **Definování pravidla** – Pravidlo začíná slovem SecRule. Jedná se o konfigurační direktivu, která vytvoří pravidlo, jež bude analyzovat vybrané proměnné pomocí zvoleného operátoru.
2. **Výběr proměnné** – Pravidlo pokračuje proměnnými, které se budou analyzovat, například REQUEST_URI. Tato proměnná obsahuje úplnou adresu URL požadavku včetně údajů řetězce dotazu. Pomocí proměnných můžeme určit, jaká část dotazu se bude analyzovat.
3. **Definování operátoru** – Dále se definuje operátor, například `@streq /index.php`. Operátor `@streq` kontroluje, zda se hodnoty v řetězci rovnají hodnotě `/index.php`. Obecně lze říci, že se pomocí operátorů určuje, kdy se má dané pravidlo aktivovat.
4. **Definování akce a Transformace** – Nakonec se pravidlem definují transformace a akce, které budou provedeny. Transformace říkají, jakým způsobem se normalizují data v proměnných. Akce popisují, co se stane, pokud se pravidlo aktivuje, například `id:1,phase:1,t: lowercase,deny`. Tento příkaz udává, že pravidlo ve fázi 1 převede obsah proměnné na malá písmena a porovná ji s operátorem. Pokud se hodnoty budou rovnat, zahodí tento požadavek a žádné další pravidlo se již kontrolovat nebude. Důležité je si také uvědomit, že transformace nemění data uvnitř požadavku, ale vytvoří si kopii dat, nad kterou dané operace probíhají.

V rámci této práce bylo vytvořeno několik vlastních vzorových pravidel, na nichž bylo demonstrováno, jakým způsobem se vlastní pravidla tvoří a zároveň jakým způsobem lze WAF pomocí vlastních pravidel přizpůsobit potřebám a požadavkům dané společnosti a jejím webovým aplikacím. Jedno z vlastních pravidel je zobrazeno na výpisu 4.6.

Toto vlastnoručně vytvořené pravidlo slouží k detekci a blokování SQLi útoků, kde se v URL objevuje řetězec *ORDER BY*. Proměnná, již byla v tomto pravidle využita, je *ARGS*. Obsahuje všechny argumenty požadavku. Operátorem je v tomto

Výpis 4.6: Ukázka vlastního vytvořeného pravidla pro zamezení SQLi útoků využívajících řetězec `ORDER BY`.

```
SecRule ARGS "ORDER BY \d+" \
    "id:1002, \
    phase:2, \
    deny, \
    log, \
    msg:'SQL Injection Attack Detected -
        ORDER BY with Numeric Parameter', \
    tag:'SQLi', \
    severity:'CRITICAL', \
    capture, \
    t:none, \
    setvar:'tx.sqli_attack_score=
        +%{tx.critical_anomaly_score}', \
    setvar:'tx.sqli_attack_score_blocking=
        +%{tx.critical_anomaly_score}'"
```

případě pouze řetězec *ORDER BY \\d+*. Kombinace těchto parametrů říká, že pokud se v dané proměnné bude nacházet řetězec *ORDER BY* následovaný jakýmkoliv číslem, tak se pravidlo aktivuje. Poslední část pravidla, která slouží pro definování akcí a transformací, je trochu obsáhlejší, a to z důvodu, že by tato část měla obsahovat nějaké dodatečné základní informace o pravidle. Pro správné fungování je povinné doplnit unikátní identifikátor pravidla (`id:1001`), fázi, kdy se pravidlo použije (`phase:2`), a akci, jež se použije (`deny`). Dále byly doplněny určité nepovinné parametry, které slouží pro následnou lepší orientaci v pravidlech pomocí povolení generování záznamů a definice, jak budou dané záznamy vypadat. Jedná se o povolení zaznamenávání informací do záznamů (`log`) a definování obsahu zaznamenané zprávy pomocí (`msg:`), značky (`tag:`), severity (`severity:`) a skóre (`setvar:`). Došlo k upřesnění, aby se zachytávala data, jež vedla k detekci, a neprováděla se na analyzovaných datech transformace. K tomu byly využity příkazy `capture` a `t:none`.

Pravidlo bylo náležitě pojmenováno jako `99_PSN_custom.conf` a bylo vloženo do složky `/etc/modsecurity/rules`. Bylo využito toho, že v konfiguračním souboru `security2.conf` je popsáno, že se mají z této složky načítat veškerá ModSecurity pravidla, která obsahuje.

Další vlastnoručně vytvořená pravidla slouží pro detekci komunikace, v níž pole požadavku `User-Agent` obsahuje jakékoliv symboly, které nejsou čísla nebo písmena, a k zablokování komunikace, jež má hlavičku větší než 100 000 bytů. Jedná se o pří-

pady, které by se v rámci běžné komunikace neměly vyskytnout, a proto bude daná komunikace blokována nebo zaznamenána formou záznamů.

V rámci zachování správného postupu bylo pravidlo nejdříve nasazeno s klíčovým slovem `pass`. Díky němu dané pravidlo provoz neblokovalo, ale pouze zaznamenávalo do příslušného souboru se záznamy. Poté, co byla ověřena správnost pravidla, bylo klíčové slovo upraveno na `deny`. Po následném restartování webového serveru Apache HTTP bylo nasazení kompletní a pravidlo zablokovalo veškeré požadavky, které se pokusily o SQLi pomocí příkazu `ORDER BY`.

4.1.4 Generování záznamů integrace

Generování záznamů v této integraci je velmi jednoduché zejména z důvodu, že ModSecurity je modulem Apache HTTP serveru. Díky tomu mají obě tyto technologie sloučené záznamy a ukládají je do jednoho souboru. Tímto souborem je `error.log` nacházející se v základní konfiguraci ve složce `/var/log/apache2`. Tento soubor se záznamy je potřeba číst nástrojem určeným pro přenos záznamů, jako je například Beats nebo Syslog-ng. Tento nástroj poté dané záznamy přesune do specializovaného nástroje, jenž slouží k uložení záznamů a následné analýze. Při tomto procesu je nutné dané záznamy správně parsovat. Parsování je proces, při kterém se extrahují strukturované informace z nelidsky čitelných záznamů. Jedná se o velmi důležitý proces pro provedení analýzy nad získanými daty. Pro zjednodušení následného parsování je velmi důležité, aby byla vytvořená pravidla stejně strukturována a obsahovala stejná klíčová pole jako pravidla z CRS.

Při integraci WAF do reálného prostředí by bylo současně velmi vhodné, aby se sbíraly záznamy ze všech aplikací běžících na daném webovém serveru, jak již bylo řečeno v kapitole 3 zabývající se potřebami a požadavky SOC. V kontextu této práce je využívaná aplikace DVWA určena pro testování, ne pro reálné použití. Z toho důvodu je její možnost logování velmi omezená, a proto není sběr záznamů jako součást této práce více popsán.

Posledním typem záznamů, na něž je v této práci brán ohled, jsou záznamy nacházející se v souboru `access.log` ve složce `/var/log/apache2`. Tento soubor obsahuje záznamy o přístupech (access logs) k webovým stránkám hostovaným na serveru. Dané záznamy obsahují informace o každém požadavku na server včetně IP adresy klienta, data, času, požadavku, odpovědi serveru a dalších informací. Pro potřeby SOC je tento soubor taktéž velmi důležitý, proto je vhodné tyto záznamy sbírat.

4.1.5 Napojení na ELK Stack

Jedním z nejdůležitějších kroků integrace WAF do SOC je napojení WAF na ostatní technologie nacházející se v prostředí SOC. Tato podkapitola se zabývá tím nejdůležitějším propojením, a to s technologií SIEM, jež byla detailně popsána v podkapitole 1.2.1. K této integraci byl využit takzvaný ELK Stack. Jedná se o soubor tří open-source projektů: Elasticsearch, Logstash a Kibana, které společně poskytují silný nástroj pro vyhledávání, analýzu a vizualizaci dat záznamů. Elasticsearch je vyhledávací a analytický engine založený na Apache Lucene. Umožňuje rychlé vyhledávání, agregaci a analýzu velkých objemů textových dat. Logstash umožňuje přijímat a ukládat data z různých zdrojů, transformovat je a posílat do různých cílů, jako je Elasticsearch. Kibana představuje webové rozhraní pro Elasticsearch, jež umožňuje vizualizaci dat v něm uložených. Toto řešení bylo vybráno jako vhodné pro demonstraci a vyzkoušení napojení WAF na SIEM, a to zejména z důvodu, že se jedná o open-source řešení, které je snadné na integraci.

Instalace a konfigurace Elasticsearch

Při integraci bylo postupováno následujícím způsobem. Nejdříve bylo nutné připojení oficiálního repozitáře Elasticsearch. Provedlo se stažení a přidání GPG klíče Elastic a následné přidání repozitáře do systémového seznamu repozitářů. Dále bylo již možné Elasticsearch instalovat pomocí odpovídajícího balíčku. Potřebný klíč a balíček byly staženy z adresy *https://artifacts.elastic.co/*. Elasticsearch vyžaduje úpravu konfiguračního souboru `elasticsearch.yml` ve složce `/etc/elasticsearch`. Jelikož se jedná o nasazení v testovacím prostředí, v konfiguračním souboru se pouze nastaví direktiva *network.host* na *localhost*. Jeho další základní nastavení bude dostačující. Následně bylo důležité Elasticsearch povolit, spustit a ověřit funkčnost pomocí příkazu `curl -X GET localhost:9200` [36].

Instalace a konfigurace Logstash

Pomocí integrace Logstash se záznamy z WAF stávají součástí širšího bezpečnostního kontextu, což umožňuje komplexnější analýzy a přesnější odhalování hrozeb. Logstash byl pomocí odpovídajícího repozitáře *logstash* opět nainstalován. Následně bylo nutné v adresáři `/etc/logstash/conf.d` vytvořit konfigurační soubor `wafLog.conf`. Tento soubor určuje, jakým způsobem Logstash zpracovává záznamy z WAF. Typická konfigurace obsahuje sekce *input* a *output*. V sekci *input* je definován zdroj záznamů, v tomto případě cesta k souborům, do nichž Apache server s nasazeným ModSecurity záznamy ukládá, a v sekci *output* je určeno, kam mají být data odeslána, v tomto případě do Elasticsearch. Konkrétní nastavení, které

Výpis 4.7: Ukázka nastavení konfiguračního souboru wafLog.conf.

```
input {
  file {
    path => "/var/log/apache2/error.log"
    start_position => "beginning"
    sincedb_path => "/dev/null"
    ignore_older => 0
  }
}
filter {
  # Zde mohou být přidány filtry pro transformaci logů
}
output {
  elasticsearch {
    hosts => ["localhost:9200"]
    index => "waf-logs-%{+YYYY.MM.dd}"
  }
}
```

bylo využito ve vytvořeném souboru `wafLog.conf`, je zobrazeno ve výpisu 4.7. Následně bylo opět nezbytné Logstash povolit, spustit a ověřit jeho funkčnost. Ověření, že Logstash správně zpracovává a odesílá záznamy do Elasticsearch, lze provést kontrolou záznamů z Logstash v Elasticsearch, k čemuž lze využít příkaz [37]:

```
curl -XGET 'http://localhost:9200/_search?q=source:logstash&pretty'.
```

Instalace a konfigurace Kibana

Kibana slouží v této testovací integraci jako vizualizační nástroj pro data uložená v Elasticsearch a je nezbytná pro efektivní práci se záznamy v SIEM systému. Její instalace byla opět provedena pomocí odpovídajícího repozitáře *kibana*. Po nainstalování byl vytvořen konfigurační soubor `kibana.yml` v adresáři `/etc/kibana`. V konfiguračním souboru je možné nastavit například URL pro připojení k Elasticsearch, hostname, na kterém Kibana naslouchá, anebo vlastní název indexu pro ukládání dat z Kibana. V rámci tohoto testovacího nastavení bylo pouze nutné nastavit directivu *server.host* na *"0.0.0.0"*, a to z důvodu, aby byla Kibana přístupná z jiného zařízení ve stejné síti a bylo možné ji konfigurovat pomocí webového rozhraní. Následně bylo opět důležité kibanu povolit a spustit [38].

Obr. 4.2: Ukázka napojení záznamů z ModSecuity do Kibana.

Ověření proběhlo pomocí přistoupení na adresu *http://192.168.100.10:5601*, na níž je Kibana dostupná. Po přístupu ke Kibana byl vytvořen nový Index pattern *apache-logs-** tak, aby odpovídal indexům vytvořeným nástrojem Logstash, což umožnilo začít záznamy analyzovat a vyhodnocovat. Napojení záznamů z ModSecurity do Kibana je zobrazeno na obrázku 4.2.

4.2 Testování a validace návrhu

Tato kapitola je věnována testování a validaci navrženého řešení integrace webového aplikačního firewallu do architektury dohledového bezpečnostního centra. Testování a validace návrhu představuje kritický krok, jehož cílem je zajistit, že navržené řešení je efektivní a zcela naplňuje očekávání a požadavky SOC. Pro tyto účely byly vytvořeny tři testovací scénáře využívající volně dostupnou zranitelnou aplikaci s otevřeným zdrojovým kódem a nejčastěji užívané typy webových útoků. Výsledky těchto testovacích scénářů byly vyhodnoceny čímž byla zhodnocena efektivita a bezpečnost celého navrženého řešení implementace.

4.2.1 Testování ochrany pomocí WAF

Hlavní scénář pro testování ochrany před běžnými webovými útoky byl vytvořen pomocí aplikace DVWA, jež umožnila otestovat správné fungování WAF a jeho pravidel.

SQL Injection

Testování této webové zranitelnosti bylo rozděleno na pět částí, a to z důvodu, že existuje velké množství typů této zranitelnosti. Pro samotné testování byl využit modul SQL Injection v aplikaci DVWA. Do formuláře v tomto modulu byly pomocí skriptu 4.8 vkládány škodlivé dotazy, které jsou k nalezení v elektronické příloze v souboru *sqli_queries.txt*. Tímto způsobem byly otestovány webové SQL Injection útoky zaměřené na autentifikaci, Union Select, časovač a chyby. Pomocí skriptu byl určen také počet zablokovaných a povolených pokusů o SQLi útok. Pro každý typ SQLi útoku bylo použito několik škodlivých dotazů na zranitelnou aplikaci. Celkem bylo otestováno 169 škodlivých dotazů.

Výpis 4.8: Ukázka použitého skriptu pro otestování SQL Injection.

```
import requests
base_url = 'http://192.168.100.10/vulnerabilities/sqli/'
vulnerable_param = 'id'
with open('payloads.txt', 'r') as file:
    payloads = file.read().splitlines()
i=0
for payload in payloads:
    url = f'{base_url}?{vulnerable_param}={payload}'
    response = requests.get(url)
    if '403 Forbidden' in response.text:
        i = i + 1
        print(f"SQL Injection Blocked: {payload}")
    else:
        print(f"SQL Injection Passed: {payload}")
print(f'{i}/{len(payloads)}')
```

Command Injection

K testování této webové zranitelnosti byl opět využit stejný skript jako v případě testování SQLi. Byly aplikovány škodlivé příkazy, jež jsou k nalezení v příloze v souboru `commandi_code.txt`. Celkem bylo použito 6 škodlivých příkazů, které obsahují nejčastěji zneužívané příkazy při webových útocích. Všechny byly zablokovány firewallem.

Reflected a Stored XSS

Tato zranitelnost byla totožně testována pomocí python skriptu 4.8, jenž byl lehce upraven. Změnily se hodnoty `base_url` a `vulnerable_param`. Škodlivé skripty byly načítány opět z textového souboru a jsou k nalezení v elektronické příloze jako `reflected_XSS.txt`. Nasazený WAF zablokoval všech 100 pokusů o XSS.

S deseti vybranými škodlivými skripty byl proveden i Stored XSS. Výsledkem bylo sto procent zablokovaných škodlivých dotazů. Forma provedení byla obdobná jako u Reflected XSS.

4.2.2 Zátěžové testování WAF

Výkonnostní testování integrace proběhlo pomocí nástroje Siege. Jedná se o nástroj pro regresní testování a benchmarking HTTP/HTTPS, určený k měření výkonu webového serveru při zátěži. Simuluje souběžný zásah více uživatelů na webový server a poskytuje několik metrik, které lze využít pro pochopení výkonu serveru.

Siege byl nainstalován přímo manažerem balíčků pomocí příkazu `sudo apt-get install siege`. Následně byla upravena jeho základní konfigurace, a to zejména z důvodu, že všechny části zranitelné aplikace DVWA vyžadují přihlášení uživatele. Úpravu konfigurace můžeme vidět ve výpisu 4.9. První úprava `show-logfile` na

Výpis 4.9: Ukázka změn konfiguračního souboru siege.conf.

```
...
show-logfile = true
...
logfile = /root/.siege/logs
...
login-url = http://192.168.100.10/login.php \
    POST name=admin&pass=password
...
```

hodnotu `true` a `logfile` na hodnotu `/root/.siege/logs` nástroji umožňuje výsledky testování zapisovat do samostatného zvoleného souboru, což velmi usnadní závěrečné zpracování výsledků. Druhá úprava tohoto souboru je velmi stěžejní, a to z již zmíněného důvodu potřeby autentizace uživatele využívajícího DVWA aplikaci. Parametr `login-url` říká, na jaké stránce se nachází přihlašovací formulář, který se následně využije pro ověření testovacího uživatele. Tímto způsobem se každý testovací uživatel nejprve autentizuje a následně může bez omezení otestovat aplikaci.

Samotné výkonnostní testování pomocí nástroje Siege bylo spuštěno příkazem `siege -c 150 -b -t 1M -f siege_URL.txt`. Tento příkaz nástroji udává počet

uživatelů použitých při testování (-c), spuštění testování v takzvaném *Benchmark* módu, jenž zajistí, aby byly výsledky při testování zobrazeny až v souhrnu na konci, ne v průběhu testů (-b), délku testování aplikace (-t) a využití speciálního souboru pro čtení jednotlivých URL adres, které budou testovány (-f). Tento soubor byl vytvořen z jednotlivých částí aplikace DVWA, jež jsou běžně dostupné. Obsahuje tedy 22 unikátních URL adres, na které bylo v rámci testování přistupováno. Tento soubor je k nalezení v elektronické příloze jako `siege_URL.txt` [39].

Celkem bylo provedeno 20 různých testů s postupně se zvyšující zátěží ve formě počtu uživatelů. Nejdříve 10 testů bez nasazeného ModSecurity a následně dalších 10 testů s nasazeným ModSecurity. Počet uživatelů při testování se pohyboval v rozmezí od 1 do 255 uživatelů. Horní hranice byla částečně určena omezením nástroje Siege, který je základní konfigurací limitován tímto počtem, jenž lze navýšit až na 1 000 uživatelů. Tato možnost byla s ohledem na výkonnostní testování použité v této práci zbytečná, a proto byla horní hranice ponechána na hodnotě 255 uživatelů.

Výpis 4.10: Ukázka všech měřených parametrů při jednom testování nástroje Siege.

```
{    "transactions":                  87357,
     "availability":                  100.00,
     "elapsed_time":                  59.52,
     "data_transferred":              553.35,
     "response_time":                 0.26,
     "transaction_rate":              1467.69,
     "throughput":                    9.30,
     "concurrency":                   142.03,
     "successful_transactions":       65361,
     "failed_transactions":           0,
     "longest_transaction":           1.27,
     "shortest_transaction":          0.00}
```

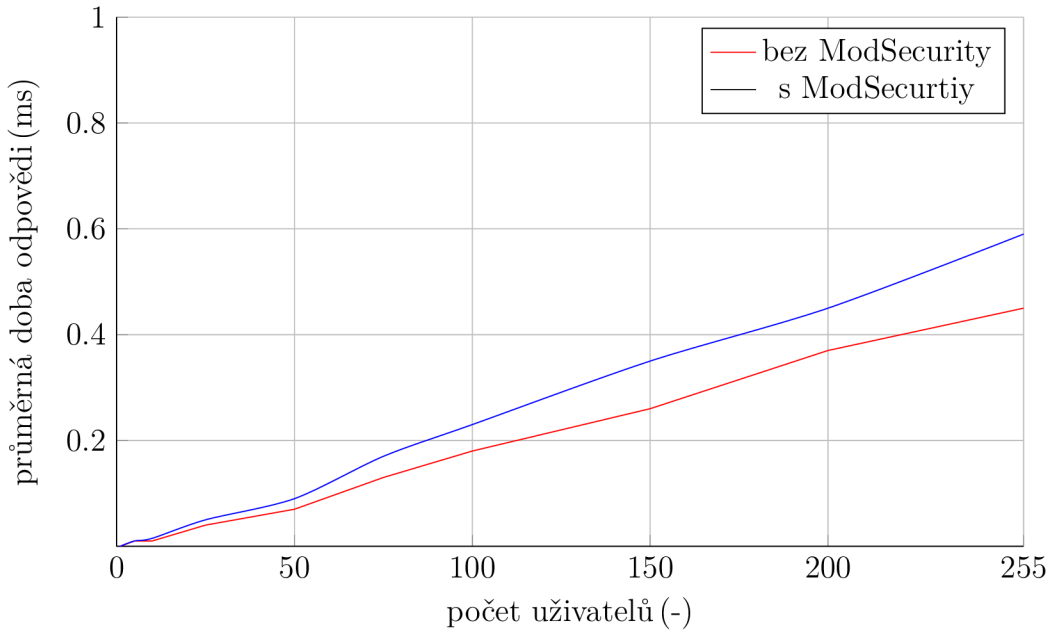
Jak je patrné z výpisu 4.10, výstupem testování nástroje Siege je velké množství různých údajů o testování. Pro potřeby této práce byly vybrány dva, konkrétně *response_time* (průměrná doba odezvy na jeden požadavek) a *longest_transacion* (nejdelší doba vyřízení jednoho požadavku). Pomocí těchto hodnot byly vytvořeny grafy 4.3 a 4.4.

4.3 Výsledky testů a jejich evaluace

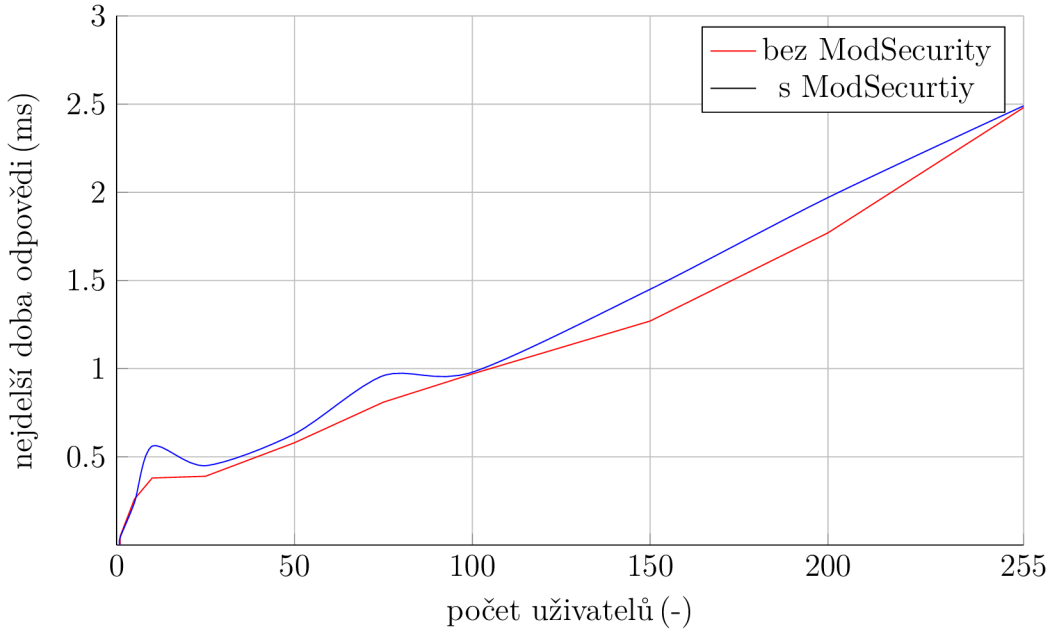
Výsledky testování ochrany WAF před webovými útoky jsou velmi dobré. Při všech scénářích simulujících nejčastější webové útoky byla míra odhalení a zablokování těchto útoků 100%. Celkem bylo při tomto testování použito 275 škodlivých dotazů směřujících na zranitelnou aplikaci. Na druhou stranu byly v průběhu testování objeveny dva případy, kdy byly zablokovány požadavky, které by nemusely být zablokovány. Konkrétně se jedná o přístup na stránku *instructions.php* a *phpinfo.php* v rámci aplikace DVWA. Jedná se o stránky, jež jsou běžně blokovány již na úrovni samotné aplikace, a to z důvodu ochrany důležitých informací, které by mohly pomoci útočníkům při útoku na danou aplikaci. V tomto případě jsou stránky na úrovni aplikace povoleny, protože obsahují důležité informace pro správné nastavení a ovládání aplikace. Tento případ je tedy v této části práce zmíněn zejména z důvodu, že se jedná o velmi individuální požadavek této aplikace, který je v obecných pravidlech CRS blokován. Z tohoto důvodu je tedy nutné tato pravidla pro korektní fungování WAF upravit podle specifických potřeb společnosti a jejich aplikací.

Výsledky výkonnostního testování jsou dobré. Při zapnutí modulu ModSecurity a využití všech pravidel, jež jsou v této práci popsána, bylo navýšení zpoždění při komunikaci s webovou aplikací zanedbatelné. Ze samotných výsledků měření nebo z vytvořených grafů 4.3 a 4.4 lze vyčíst, že při nasazení WAF se průměrná doba odezvy aplikace navýšila o 24 % a průměrná doba nejdelší odpovědi se zvýšila o 8 %. Je přirozené, že při přidání dalšího článku do řetězce zpracování požadavků se navýší průměrné zpoždění. Avšak při integraci v rámci této práce bylo navýšení relativně nízké, což je velmi pozitivní.

Celkově lze tvrdit, že integrace navržená a popsaná v této práci je funkční a dosahuje velmi uspokojivých výsledků. Dalším směrem, kterým se samotná práce v následujících kapitolách ubírá, je nasazení této integrace v reálném prostředí, úprava pravidel pro nejoptimálnější fungování v daném prostředí a vyhodnocení vlivu nasazení dané integrace na detekci hrozeb a bezpečnost v daném prostředí. Dále bude popsán také vliv integrace na viditelnost v chráněné síti a s tím spojenou změnu počtu falešně pozitivních a skutečně pozitivních detekcí.



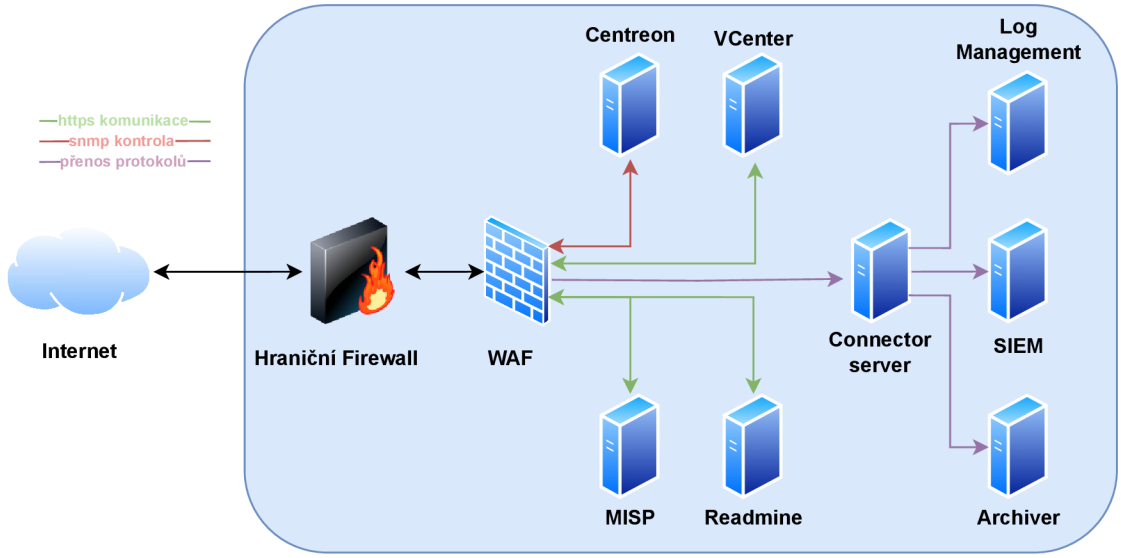
Obr. 4.3: Průběh metriky response_time v čase.



Obr. 4.4: Průběh metriky longest_transaction v čase.

5 Nasazení integrace do reálného prostředí

V této kapitole je popsáno nasazení WAF jako reverzní proxy do reálného prostředí SOC. Následně je kapitola věnována i samotné integraci do architektury SOC. Z důvodu, že je nutné veškerou komunikaci, již chceme pomocí WAF chránit, směrovat přes reverzní proxy, je implementace WAF do prostředí SOC velmi invazivní. Proto byla v rámci implementace zvolena možnost, kdy se přes reverzní proxy přesměrovala pouze komunikace s webovou aplikací Easy Redmine a platformou MISP. Easy Redmine je technologie, ke které je v rámci interní sítě a internetu přistupováno velmi intenzivně a pravidelně, a zároveň jde o velmi kritickou část infrastruktury SOC. K platformě MISP je přistupováno pouze v rámci interní sítě, ale výhodou je její open-source charakter, díky němuž se v ní častěji objevují zranitelnosti a vydání příslušných záplat, které by je opravily, trvá delší dobu. Tímto způsobem tedy v práci bude provedena integrace do reálného prostředí a posléze z dostatku sesbíraných dat integrace analyzována. Následně po dokončení diplomové práce může být postupně na WAF napojen zbytek technologií v SOC, které mají webové rozhraní.



Obr. 5.1: Ukázka části SOC infrastruktury důležité pro integraci WAF.

Na obrázku 5.1 je znázorněné zjednodušené schéma infrastruktury SOC, do něhož bude WAF integrován. V rámci této diplomové práce bude tedy přes reverzní proxy přesměrována veškerá komunikace směřující na technologii Easy Redmine a MISP. Dále bude reverzní proxy s WAF napojená na server s provozním monitoringem, který zajišťuje technologie Centreon. A posledním krokem integrace do reálného prostředí bylo napojení všech důležitých nově generovaných záznamů na connector server, jenž záznamy nejprve odešle do archiveru, aby se zajistilo jejich bezpečné uložení, a poté se data odešlou do provozovaného Log Managementu a SIEM. Jedná

se o běžný postup v rámci SOC. Z obrázku integrace je patrné, že napojení dalších technologií SOC nebylo náročné a jednalo se o pouhé přesměrování komunikace přes reverzní proxy a zanesení potřebných informací do interní dokumentace a s tím spojenou minimální úpravu interních procesů.

V této diplomové práci je kladen velký důraz na bezpečnost a ochranu citlivých informací. Z tohoto důvodu pro síťové prvky a servery nejsou použity jejich skutečné IP adresy. Toto opatření chrání citlivé informace a zajišťuje, aby práce sloužila pouze pro vzdělávací a demonstrační účely, aniž by byly zveřejněny detaily, které by mohly ohrozit bezpečnost v SOC.

5.1 Nasazení webového aplikačního firewallu

Prvním krokem nasazení WAF do produkčního prostředí bylo vytvoření virtuálního serveru. Nejdříve byly na hostovské platformě ESX vyčleněny požadované hardwarové požadavky. Jedná se o 4 CPU, 8 GB RAM a 80 GB HDD. To jsou parametry, které byly shledány jako dostačující v rámci testovacího nasazení. Následně byl s danými požadavky vytvořen virtuální server, na nějž byl nainstalován operační systém Ubuntu Server 22.04. Virtuálnímu serveru byla přiřazena IP adresa 192.168.10.10.

Následně bylo na serveru provedeno nasazení reverzní proxy a WAF obdobným způsobem, jako bylo popsáno v podkapitole 4.1. Do reálného prostředí zranitelná DVWA aplikace, která byla pouze pro testovací účely a v produkčním prostředí není potřebná, již nebyla instalována. Zároveň by nebylo bezpečné ji do produkčního prostředí nasazovat. Další změnou bylo nastavení konfiguračního souboru `waf-proxy.conf`, jenž slouží pro vytvoření virtuálního hosta. V tomto souboru byla zadána doménová jména, na kterých reverzní proxy naslouchá, a adresy technologií Easy Redmine a MISP, na něž se komunikace na WAF následně přesměruje. Stěžejní část konfigurace tohoto souboru je zobrazena ve výpisu 5.1 a celý konfigurační soubor je přiložený v elektronické příloze.

Easy Redmine je pokročilý software pro řízení projektů, který vychází z open-source řešení Redmine. V rámci SOC, jejž je zmiňovaný v této práci, se Easy Redmine používá zejména pro správu projektů, plánování zdrojů, sledování úkolů, reportování a spolupráci. Je využíván také všemi zaměstnanci SOC k denní činnosti, takže je důležité, aby byl náležitě zabezpečen. Tato fakta dělají napojení Easy Redmine na WAF důležitější a vhodnější.

V rámci zajištění bezpečné komunikace s Apache serverem, na němž je nasazený ModSeciruty, bylo nutné vygenerovat příslušný SSL certifikát. Ve společnosti provozující SOC, do kterého je WAF integrován, je požadováno využívání kryptografických klíčů, jež mají alespoň 128bitovou bezpečnost. Jedná se o klíče, které by podle NIST měly zajišťovat dostatečnou bezpečnost i po roce 2030. Nabízí se tedy

Výpis 5.1: Ukázka potřebného nastavení v souboru waf-proxy.conf.

```
...
ServerName redmine.axenta.cz
ProxyRequests Off
ProxyPreserveHost On
ProxyPass / https://192.168.50.20/
ProxyPassReverse / https://192.168.50.20/
...
ServerName misp.soc.local
ProxyPreserveHost On
ProxyPass / http://192.168.20.199/
ProxyPassReverse / http://192.168.20.199/
```

využití algoritmu RSA s délkou klíče 3072 bitů nebo algoritmu EdDSA (Edwards-curve Digital Signature Algorithm) s délkou klíče 255 bitů. V rámci této práce byl zvolen algoritmus ed25519 z rodiny podpisových schémat EdDSA. Jedná se o schéma zajišťující stejnou úroveň bezpečnosti s využitím výrazně kratších klíčů a menších zdrojů pro autentizaci [40].

Privátní klíč pro certifikát byl vygenerován pomocí nástroje openssl a následným příkazem `openssl genpkey -algorithm ed25519 -out reProxy.key`. Pomocí tohoto klíče byla vytvořena takzvaná žádost o podepsání certifikátu CRS (Certificate signing request). Po použití příkazu `openssl req -new -key reProxy.key -out reProxy.csr` na vytvoření žádosti je nutné zadat ještě základní informace jako například název společnosti, lokalitu, nebo zemi. Jedná se o informace, které budou následně součástí SSL certifikátu.

Posledním krokem bylo odeslání CSR žádosti certifikační autoritě provozované v rámci SOC. Po úspěšném obdržení SSL certifikátu byl certifikát a privátní klíč přesunut do příslušných složek `/etc/ssl/certs` a `/etc/ssl/private`. Dále byl také příslušným způsobem upraven konfigurační soubor `default-ssl.conf` tak, aby cesty v souboru směřovaly k příslušným souborům. Změna konfigurace byla provedena stejným způsobem, jako bylo popsáno v podkapitole 4.1.

5.2 Integrace do infrastruktury SOC

Klíčovým krokem integrace je napojení na již využívané technologie SOC. Čím více bude WAF na dané technologie napojena, tím lepších výsledků integrace je možné dosáhnout. V této kapitole jsou popsána čtyři hlavní oblasti integrace: parsování

nových záznamů, napojení na SIEM, napojení na provozní monitoring a napojení na platformu Threat Intelligence. S postupným vývojem samotného SOC je možné, že se integrace rozšíří například o napojení na SOAR nebo nějaké jiné další technologie.

5.2.1 Napojení na technologii Log Management

První základní integrací je napojení na technologii Log Managementu, kterou je v tomto případě ArcSight Logger. Jedná se o nástroj specializovaný na správu záznamů. ArcSight Logger poskytuje rozsáhlé možnosti pro analýzu a uchovávání záznamů, včetně pokročilých vyhledávacích, filtrujících a reportovacích nástrojů. Tento nástroj tedy umožní efektivně pracovat se záznamy shromážděnými z ModSecurity a poskytne hluboký vhled do bezpečnostních událostí [41].

Výpis 5.2: Ukázka potřebného nastavení v souboru modsecurity.conf.

```
...
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \
    \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %O \
\"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
...
```

Nejdříve bylo nutné definovat si, jaké typy záznamů budou do Log Managementu zasílány a v jakém formátu budou. Generování záznamů integrace je podrobně popsáno v podkapitole 4.1.4. Pro napojení byl zvolen soubor `error.log` v adresáři `/var/log/apache2`. Jedná se sice pouze o zkrácený záznam, než který se ukládá do souboru `modsec_audit.log`, ale pro potřeby této integrace je dostačující a obsahuje všechny potřebné informace. Malou výhodou je rovněž fakt, že soubor `error.log` obsahuje i jiné záznamy popisující chybové stavy na daném serveru. Dále bylo nutné upravit konfiguraci Apache serveru tak, aby měly ostatní chybové záznamy požadovaný formát a bylo možné je na Connector serveru zpracovat bez úpravy parseru. Provedla se tedy úprava v souboru `apache2.conf` ve složce `/etc/apache2`. Změněné nastavení je zobrazeno ve výpisu 5.2 [42].

Následně byla využita technologie rsyslog, jež je v rámci SOC využívána k flexibilnímu a spolehlivému sběru a k přeposílání záznamů z různých zdrojů na Connector server. Jedná se o open-source implementaci protokolu syslog, jež rozšiřuje

tradiční protokol Syslogd. Její instalace proběhla pomocí příkazu `apt-get install rsyslog`.

Výpis 5.3: Ukázka potřebného nastavení v souboru rsyslog.conf.

```
module(load="imfile" PollingInterval="10")

input(type="imfile"
      File="/var/log/apache2/error.log"
      Tag="modsec"
      Severity="info"
      Facility="local7")

template(name="ModsecFormat" type="string" \
    string="<%PRI%>%TIMESTAMP% \
    %HOSTNAME% %syslogtag% %msg%\n")

action(type="omfwd"
       Target="192.168.13.127"
       Port="6515"
       Protocol="tcp"
       template="ModsecFormat")
```

Poté proběhla konfigurace pomocí konfiguračního souboru `rsyslog-waf.conf`, který se nachází v adresáři `/etc/rsyslog.d`. Ukázka celého konfiguračního souboru je zobrazena ve výpisu 5.7. První část definuje načtení modulu `imfile`, jenž umožňuje sledovat soubor na disku a zpracovávat v něm nové záznamy. Parametr `PollingInterval` nastavuje, že se má soubor kontrolovat každých 10 sekund. Konfigurace zdroje záznamů probíhá pomocí funkce `input`, ve které se definuje, o jaký typ zdroje se jedná (type), cesta k sledovanému souboru (File), specifický tag záznamů (Tag), úroveň závažnosti (Severity) a kategorie zařízení (Facility). Správné nastavení parametrů `type` a `File` je velmi důležité, ostatní parametry slouží k lepší identifikaci a kategorizaci jednotlivých záznamů. Funkce `template` je určena k definici šablony přenášených záznamů. Tato šablona formátuje záznamy do specifického výstupního formátu obsahujícího prioritu zprávy, časové razítko, název hostitele, tag systému syslog a samotnou zprávu. Poslední částí je funkce `action`, která slouží k odeslání záznamů na Connector server. Bylo důležité nastavit, že se jedná o předání zprávy (type), IP adresu cílového serveru (Target), síťový port a protokol pro komunikaci (Port a Protocol) a použitou šablonu pro formátování (template). Tato

konfigurace umožňuje efektivně sledovat specifické záznamy, formátovat je a odesílat na Connector server pro další zpracování nebo archivaci [43].

Výpis 5.4: Ukázka záznamu vygenerovaného ModSecurity před normalizací.

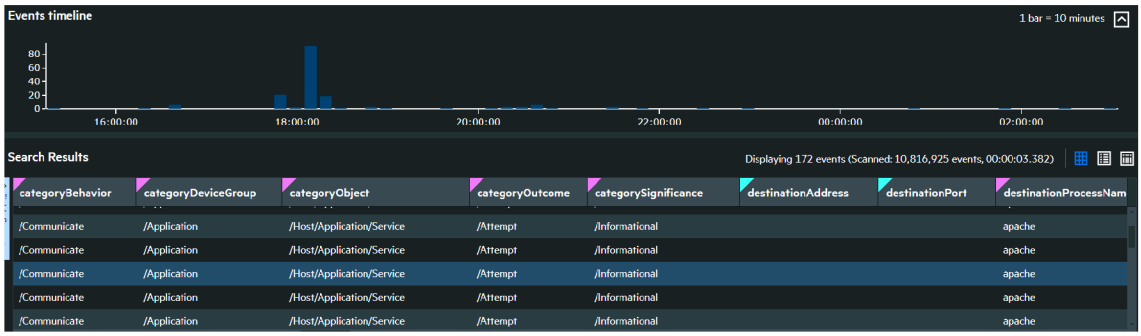
```
[Sat Mar 16 09:41:43.248296 2024] [:error] [pid 270242]
[client 192.168.29.15:50147] [client 192.168.29.15]
ModSecurity: Warning. Unconditional match in SecAction.
[file "/etc/modsecurity/rules/RESPONSE-980.conf"]
[line "96"] [id "980170"] [msg "Anomaly Scores:
(Inbound Scores: blocking=3,detection=3, per_pl=3-0-0-0,
threshold=5) - (Outbound Scores: blocking=18,
detection=18, per_pl=18-0-0-0, threshold=4) -
(SQLI=10, XSS=0, RFI=0, LFI=0, RCE=0, PHPI=0, HTTP=0,
SESS=0, COMBINED_SCORE=21)"] [ver "OWASP_CRS/4.0.0-rc2"]
[tag "reporting"] [hostname "192.168.10.10"]
[uri "/instructions.php"] [unique_id
"ZfVpV5BW_VyoTpP4KpomgwAAAAk"], referer:
http://192.168.10.10/index.php
```

Connector server umožňuje normalizaci záznamů pomocí parserů. Díky tomu se z textových nestrukturovaných záznamů extrahují klíčové informace, jako je například čas vzniku záznamu, zdrojové zařízení, které záznam vygenerovalo, anebo typ události. Příklad záznamu před normalizací vygenerovaného ModSecurity je zobrazen ve výpisu 5.4. Na Connector server se zasílají dva druhy záznamů, jedná se o chybové záznamy Apache serveru a záznamy generované samotným ModSecurity. Formát záznamů z Apache serveru byl změněn tak, aby je Connector server dokázal parsovat automaticky pomocí SmartConnector. Pro záznamy generované ModSecurity byl vytvořen vlastní parser. Pro příjem záznamů na Connector serveru byla přidána konfigurace do nástroje Syslog-ng. Ve výpisu 5.5 je tato konfigurace zobrazena. Tato konfigurace se skládá z tří hlavních částí, jež definují zdroj, cíl a cestu záznamů. Jako zdroj záznamů byla nastavena libovolná IP adresa s portem 6515 a protokol TCP. Cíl byl nakonfigurován jako localhost s portem 8514 a to z důvodu, že na tomto portu Connector server poslouchá a provádí následné přesměrování do technologií Archiver, SIEM a Log Management. Poslední částí konfigurace je vytvoření cesty záznamů, což znamená, že záznamy ze zdroje `s_apache_waf` budou zasílány do cíle `d_apache_waf`.

Stěžejní část vytvořeného parseru je regulární výraz, který slouží k extrahování specifických informací ze záznamů generovaných ModSecurity. Každý segment regulárního výrazu odpovídá specifickému datovému bodu, který je ze záznamu třeba

Výpis 5.5: Ukázka přidané konfigurace do souboru syslog-ng.conf.

```
...
source s_apache_waf {
    network(ip(0.0.0.0) port(6515) transport("tcp"));
};

destination d_apache_waf {
    tcp("127.0.0.1" port(8514));
};

log {
    source(s_apache_waf);
    destination(d_apache_waf);
};
...
```



Obr. 5.2: Ukázka parsovaných záznamů v technologii ArcSight Logger.

získat. Tyto datové bloky jsou vymezeny závorkami, které zachycují specifické údaje, viz následující výčet.

- Závažnost incidentu (severity)
- ID procesu (pid)
- Zdrojová IP adresa (srcip)
- Zdrojový port (srcport)
- Název pravidla (rule)
- Číslo řádku pravidla (rule_line)
- ID pravidla (id)
- Zpráva (msg)
- Verze (version)
- Tag (tag)

- IP adresa zařízení (dvcip)
- URI (uri)
- Unikátní ID (unique_id)
- Referer (referer) - pouze pokud existuje

Dále parser obsahuje tokeny, jejichž účelem je vytvoření kontejneru pro uložení hodnot extrahovaných pomocí regulárních výrazů. Tokeny se předem definují s očekávaným datovým typem a názvem. Slouží jako stavební bloky pro sestavení strukturovaných záznamů. Využívá se mapování jednotlivých tokenů na dané události. Například `event.deviceCustomString2=unique_id` znamená, že hodnota `unique_id` bude uložena jako custom string pod značkou `Unique ID` ve finálním strukturovaném záznamu.

Výpis 5.6: Ukázka regulárního výrazu parseru ModSecurity záznamů.

```
regex=modsec\\s\\[[^\\]]+\\]\\s\\[\\:(\\w+)\\]
\\s\\[pid\\s(\\d+)]\\s\\[client\\s(\\d{1,3}\\.\\
d{1,3}\\.\\d{1,3}\\.\\d{1,3})\\\:(\\d+)\\]\\s
[^\\]]+\\]\\s.*\\s\\[file\\s\\"([^\\"]+)\\"\\]
\\s\\[line\\s\\"(\\d+)\\"\\]\\s\\[id\\s\\"
(\\d+)\\"\\]\\s\\[msg\\s\\"([^\\"]+)\\"\\]
.*\\s\\[ver\\s\\"([^\\"]+)\\"\\]\\s\\[tag\\s\\
"([^\\"]+)\\"\\].*\\s\\[hostname\\s\\"([^\\"]+)
\\"\\]\\s\\[uri\\s\\"([^\\"]+)\\"\\]\\s\\[unique_id
\\s\\"([^\\"]+)\\"\\](?\:\\,\\sreferer\:\\s(.*))?
```

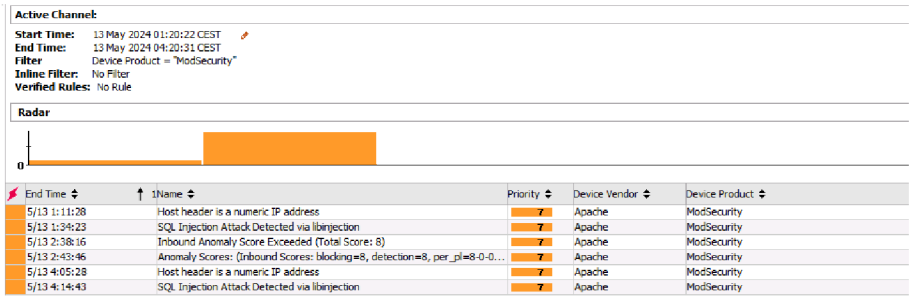
Výstupem parseru je série normalizovaných záznamů, které jsou konzistentní, což umožňuje jejich efektivní uložení a zpracování. Ukázku využitého regulárního výrazu ve vytvořeném parseru zobrazuje výpis 5.6. Celý parser je k dispozici v elektronické příloze jako `apache_modsec.properties`. Obrázek 5.2 znázorňuje ukázku normalizovaných záznamů z WAF v nástroji pro správu záznamů. Kterými bylo ověřeno, že výše popsaná konfigurace je funkční, záznamy z WAF se v pořádku normalizují a dostávají se až do technologie pro správu záznamů.

5.2.2 Napojení do SIEM

Po úspěšné normalizaci záznamů na Connector serveru následuje jejich napojení také do SIEM systému, konkrétně do ArcSight Enterprise Security Manager (ESM). ArcSight ESM je pokročilý SIEM systém, jenž umožňuje efektivní správu bezpečnostních informací a událostí. Jeho klíčovou vlastností je schopnost shromažďovat, analyzovat a vizualizovat bezpečnostní data z mnoha různých zdrojů v reálném čase.

Podrobnější popis SIEM systémů obsahuje podkapitola 1.2.1. Po přenosu záznamů na Connector server se záznamy automaticky přepošlou přímo do ESM a z pohledu napojení není nutné již žádných dalších kroků.

Jedním z nejdůležitějších aspektů napojení na SIEM je vytvoření a implementace korelačních pravidel. Korelační pravidla umožňují identifikovat složité vzory a vztahy mezi různými bezpečnostními událostmi, což vede k rychlejší a přesnější detekci hrozeb. V rámci této práce bylo vytvoření pravidel zjednodušeno tím, že z ModSecurity již přicházejí záznamy, které obsahují informace o případné uskutečněné bezpečnostní události. Díky čemuž není potřebné vytvářet složitá korelační pravidla, ale stačí pouze vytvořit pravidlo, které z normalizovaných záznamů vytvoří odpovídající výstrahy v technologii SIEM. Vygenerované výstrahy na základě záznamů z ModSecurity jsou zobrazeny na obrázku 5.3.



Obr. 5.3: Ukázka výstrah v ESM vytvořených na základě záznamů z ModSecurity.

Pro vytvoření takového pravidla je důležité definovat podmínky, jejich splněním se vytvoří daná výstraha. Jedná se především o podmínku, kdy přijatý záznam musí obsahovat v poli `Device Vendor` hodnotu `Apache` a v poli `Device Product` hodnotu `ModSecurity`. Jelikož se v dané síti nenachází žádné jiné zařízení s ModSecurity, jsou tyto podmínky dostačující. Dále byla již jen přidána podmínka, že se v poli `Type` nachází hodnota `Base, Aggregated`. Jež zajistí, že nevznikne zacyklení, při kterém by docházelo k vytváření nové výstrahy z každé již vytvořené.

Následně byla nastavena agregace, která říká, že pokud by byl vytvořen větší počet výstrah za jednu minutu, které by sdílely určité klíčové parametry, jako jsou například `Device Address`, `Source Address` nebo `Name`, tak by došlo k agregaci těchto výstrah do jedné. S největší pravděpodobností by k takové agregaci dojít nemělo, ale jedná se o pojistku proti zbytečnému zatížení SOC. Na obrázku 5.4 je zobrazeno nastavení provedené v nástroji ESM [44].

5.2.3 Napojení do provozního monitoringu

Provozní monitoring je v SOC nezbytný pro efektivní řízení bezpečnosti a dohled nad IT infrastrukturou. Jeho důležitost spočívá v neustálém sledování, analýze a re-

Obr. 5.4: Ukázka vytvoření korelačního pravidla v technologii ESM.

akci na rozmanité provozní události, které mohou mít vliv na bezpečnostní stav organizace. V prostředí SOC, do něhož je WAF integrován, se pro potřeby provozního monitoringu používá technologie Centreon. Je to komplexní IT monitorovací nástroj, který organizacím umožňuje sledovat jejich celou IT infrastrukturu a operační služby. Vychází z open-source projektu a nabízí širokou škálu funkcionalit pro monitorování síťových zařízení, serverů, aplikací a služeb.

Pro zařazení serveru, na kterém běží WAF, do provozního monitoringu je nejprve nutné na tento server nainstalovat SNMP agenta. Instalace byla provedena pomocí příkazu `apt-get isntall snmpd`. Následně je nutné tohoto agenta správně nakonfigurovat, včetně vytvoření SNMPv3 klienta, který nám zajistí bezpečnou komunikaci mezi serverem a Centreonem. Nastavení agenta proběhlo konfigurací souboru `snmpd.conf` ve složce `/etc/snmp/`. Zde byl pomocí direktivy `createUser` vytvořen nový SNMPv3 uživatel s názvem `Centreon`, s autentizačním heslem *[authPass]* a s šifrovacím heslem *[encPass]*. Posledním krokem úpravy bylo nastavení práv nově vytvořenému uživateli pomocí direktivy `rouser`, jejíž hodnota byla nastavena na *priv*. To znamená, že uživatel bude mít plná privátní práva na čtení, což zahrnuje autentizaci a šifrování. Celý konfigurační soubor `snmpd.conf` je k nalezení v elektronické příloze. Po konfiguraci byl snmpd agent restartován pomocí příkazu `systemctl restart` a na Centreon serveru byla provedena kontrola SNMP agenta příkazem `snmpwalk -v3 -u Centreon -l authPriv -a SHA -A [authPass] -x AES -X [encPass] 192.168.10.10`, který umožňuje procházet strom MIB (Management Information Base) s využitím protokolu SNMPv3 [45].

Po nastavení SNMP agenta na serveru s WAF byly pro konfiguraci SNMP kontrol v Centreon provedeny následující kroky. Nejdříve je nutné přidat dané zařízení, to je možné provést pomocí webového rozhraní a v něm tlačítka `Add`, jež se nachází v podsekci `Hosts` v sekci `Configuration`. Zde je nutné vyplnit potřebné informace jako *Name*, *Alias*, *IP Address* a *SNMP version*. Ještě je možné vytvořit první kontrolu daného zařízení. V tomto případě byla zvolena kontrola *base_host_alive*, která

Výpis 5.7: Ukázka potřebného nastavení v souboru snmpd.conf.

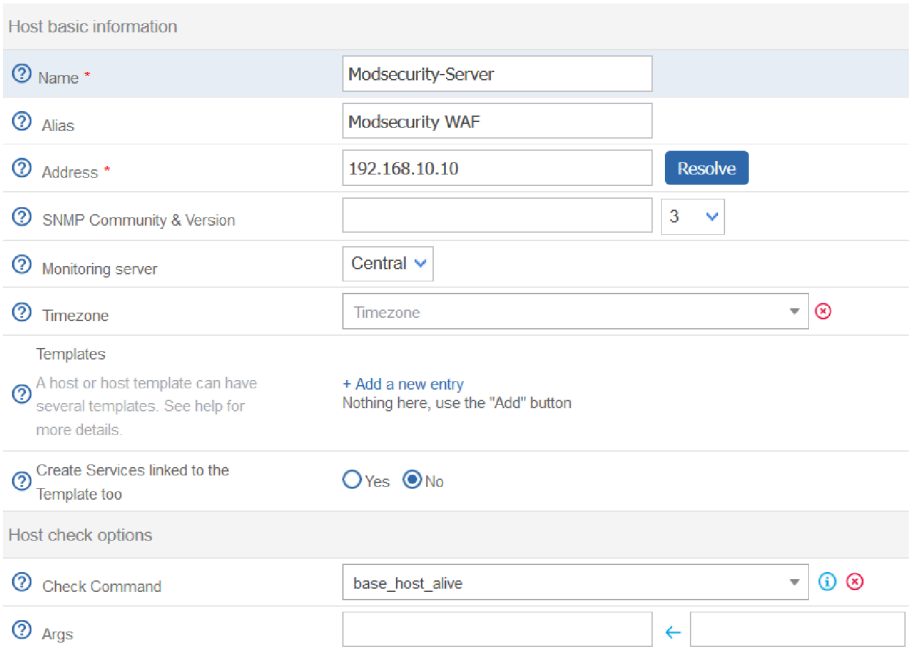
```
...
# Základní konfigurace
agentAddress udp:161              # Naslouchání na
sysLocation ESX2                  # Umístění serveru
sysContact tech@axenta.cz         # Kontaktní informace

# Vytvoření uživatele SNMPv3
createUser Centreon SHA password AES password
rouser Centreon priv

# Definice sítě pro SNMP přístup
com2sec network 192.168.40.0/24 Centreon
group AccessGroup v3 network
view all included .1 80
access AccessGroup "" any noauth exact all none none
...
```

hlídá dostupnost daného zařízení. Na obrázku 5.5 je zobrazeno vytvoření hosta pro *ModSecurity* v Centreon.

V rámci bezpečnostního centra je pro provozní monitoring již využíváno mnoho šablon, které mohly být pro účely integrace WAF použity. Jedná se o šablony na kontrolování různých typů zařízení a jejich parametrů. Pro potřeby monitorování bylo použito osm kontrol základních parametrů, jako je například vytížení procesoru, disků nebo pamětí serveru, na němž běží WAF ModSecurity. Jednotlivé šablony, které byly využity pro vytvoření příkazů a kontrol v Centreon, jsou k dispozici v elektronické příloze. Příkazy sloužící pro vytvoření kontrol byly zhotoveny pomocí tlačítka `Add` v podsekci `Commands` v sekci `Configuration`. Poté pouze stačilo zadat název a daný příklad ze šablony.

Příkazy byly následně použity pro vytvoření konkrétních kontrol, a to pomocí podsekce `Services` v sekci `Configuration`. V této podsekci slouží k přidání nové služby tlačítko `Add`. Následně stačilo vyplnit pole `Name`, `Hosts` a `Check Command`. Poté bylo důležité vyplnit všechna povinná pole v `Custom macros`. Konkrétní vyplněné hodnoty jsou zobrazeny na obrázku 5.6.

Posledním důležitým krokem bylo exportování konfigurace na používaný poller, jenž spravuje konkrétního hosta. Tím se kontrola aktivuje a lze provést její kontrolu v GUI Centreonu [46].



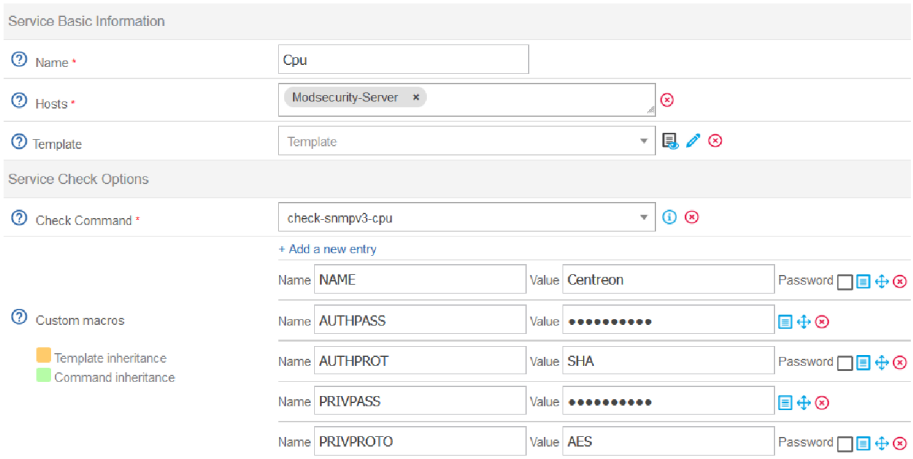
Obr. 5.5: Ukázka vytvoření nového hosta pro kontrolu v Centreon.

5.2.4 Napojení na Threat Inteligence platformu

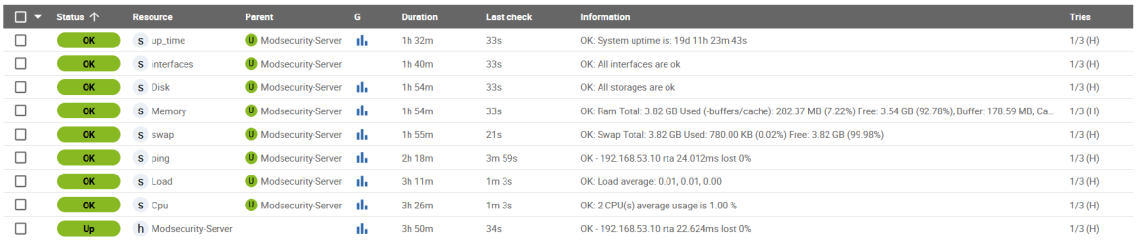
Další technologií v SOC, se kterou proběhla integrace, je platforma pro Threat Inteligence MISP (Malware Information Sharing Platform & Threat Sharing). Jedná se o otevřenou a bezplatnou platformu, jež slouží pro sdílení, ukládání a korelaci indikátorů kompromitace (IoC) a hrozeb. Tato platforma umožňuje SOC efektivně sdílet a spravovat informace o bezpečnostních hrozbách a jejich atributech, což představuje významný zdroj informací pro zvýšení kybernetické bezpečnosti. Rovněž ji lze použít i jako zdroj informací nejen pro IDS (Intrusion Detection System), ale rovněž i pro IPS (Intrusion Prevention System). MISP umožňuje pracovat s těmito daty odděleně, a zamezit tak zbytečně velkému množství falešně pozitivních případů.

Využití dat z MISP v ModSecurity přináší několik klíčových výhod. Nejdůležitější je, že MISP poskytuje bohatý zdroj aktuálních a relevantních informací o hrozbách. Tato data mohou zahrnovat specifické IP adresy, domény či škodlivé dotazy, které byly identifikovány jako součást škodlivých kampaní a útoků. Exportem těchto dat z MISP a jejich následnou integrací do ModSecurity je možné efektivně blokovat potenciální škodlivý provoz směřující k webovým aplikacím.

MISP je navržen tak, aby podporoval různé formáty výměny dat, což s pomocí svého API (application programming interface) umožňuje snadnou integraci do různých bezpečnostních nástrojů, včetně webových aplikačních firewallů, jako je ModSecurity. Tímto způsobem může MISP sloužit jako dynamický zdroj pro aktualizaci bezpečnostních pravidel a zásad reagující na neustále se měnící zdroje kybernetic-



Obr. 5.6: Ukázka vytvoření nové kontroly v Centreon.



Obr. 5.7: Ukázka aktivních kontrol pro ModSecurity v Centreon.

kých hrozeb [47].

V rámci tohoto napojení byl vytvořen Python skript, jenž z MISP exportuje data o SQLi a XSS dotazech, která jsou aktuálně zneužívána. Tato data jsou ukládána primárně ve dvou feedech `XSS Payloads for ModSecurity` a `SQLi Pay- loads for ModSecurity`. Dále jsou v nich jsou dále ukládány jednotlivé atributy, které nesou informaci o samotném škodlivém dotazu, krátkém popisu a určení místa v HTTP požadavku, kde se má daný škodlivý kód nacházet. Formát uložení jednotlivých atributů je zobrazený na obrázku 5.8. Hlavní úlohou Python scriptu je poslat příslušný dotaz na API rozhraní platformy MISP. Ukázka API dotazu ze souboru `payloads_MISP.py` je zobrazena ve výpisu 5.8, případně celý script je k nalezení v elektronické příloze. Následně dojde ke stažení odpovídajících eventů a uložení důležitých dat. Uložení probíhá do jednotlivých souborů, jež jsou staženy na server, na kterém běží ModSecurity. Poté jsou pomocí těchto adres tvořena dynamicky se měnící pravidla. Tvorba těchto pravidel je popsána v následující podkapitole 5.2.4.

Obr. 5.8: Ukázka uložení jednotlivých škodlivých SQLi dotazů v MISP.

Výpis 5.8: Ukázka MISP API dotazu ze souboru `payloads_MISP.py`

```
...
misp_url = [URL]
api_key = [API Key]
verify_ssl = False
headers = {
    'Authorization': api_key,
    'Content-Type': 'application/json',
    'Accept': 'application/json'
}
...
url = f'{misp_url}/events/view/{event_id}'
response = requests.get(url, headers=headers, \
verify=verify_ssl)
...
```

Vytvoření vlastních dynamických pravidel

V rámci této integrace s platformou MISP byly vytořeny dva Python scripty, které slouží k vytváření pravidel pro ModSecurity z dat, jež byly získány z platformy MISP. Jedná se o dva skripty `dynamic_rules_sqli.py` a `dynamic_rules_xss.py`, které slouží k tvorbě dynamických pravidel chránících před SQLi a XSS útoky.

Vytvořené skripty tvoří základní pravidla pomocí třech hlavních informací. Jedná se o místa v HTTP požadavku, ve kterých se bude hledat specifický obsah nebo vzory. Následně o samotné vzory, kterými jsou konkrétní škodlivé dotazy anebo regulární výrazy popisující celou skupinu škodlivých dotazů. Poslední informací je krátký popis sloužící k identifikaci účelu daného pravidla a k jeho pojmenování. Při vytváření pravidel se také zkontroluje, zda již není stejné pravidlo vytvořeno, a pokud ne, zapíše se do odpovídající složky `generated_sqli_rules` nebo `generated_XSS_rules`. Oba skripty jsou k nalezení v elektronické příloze.

Výpis 5.9: Využívaná konfigurace souboru crontab

```
# Spuštění skriptu pro generování payloadů
0 6,18 * * * python3 /etc/modsecurity/misp_rules \
    /payloads_MISP.py

# Spuštění skriptů pro generování pravidel SQLi a XSS
0 7,19 * * * python3 /etc/modsecurity/misp_rules \
    /dynamic_rules_sqli.py
0 7,19 * * * python3 /etc/modsecurity/misp_rules \
    /dynamic_rules_xss.py

# Reload Apache serveru po dokončení předchozích úloh
5 5,17 * * * systemctl reload apache2
```

Aby bylo pravidelné spouštění skriptů zajištěno, je nastaven na serveru s ModSecurity nástroj Crontab. Ten využívá službu na pozadí operačního systému Cron, jež pravidelně kontroluje soubor `crontab`, zda neobsahuje úlohy, které mají být v daný čas spuštěny. Využívaná konfigurace souboru `crontab` je zobrazena ve výpisu 5.9. Tato konfigurace zajišťuje, aby se skript pro export škodlivých SQLi a XSS dotazů do ModSecurity prováděl každý den dvakrát, a to v 6 hodin ráno a večer. Následně se spustí skripty pro vytvoření příslušných pravidel. Ty se spustí v 7 hodin ráno a večer. Z toho vyplývá, že každých 12 hodin dojde k aktualizování pravidel. Posledním krokem je spuštění příkazu pro načtení nové konfigurace Apache serveru, která zajistí fungování nových pravidel. K tomu dojde 5 minut po spuštění skriptů na tvorbu pravidel [48].

5.3 Implementace do fungování SOC

Integrace WAF má nemalý vliv na fungování SOC. V předchozí kapitole bylo popsáno, jakým způsobem se integrace provedla, ale nebylo zmíněno jakým způsobem to ovlivní fungování SOC jako celku. Ať už se jedná o jednotlivé zaměstnance, anebo celé procesy. Vliv na tyto aspekty je popsán v následujících podkapitolách.

5.3.1 Vliv na zaměstnance SOC

Největší dopad má integrace WAF na technické pracovníky SOC, kteří při integraci asistovali, a v případě, kdy by integrace nebyla prováděna v rámci samotné diplomové práce, by měli celou integraci na starost. Protože integrace WAF není příliš

rozdílná od integrace jiných technologií, neměla by být časově náročná a technické pracovníky velmi vytížit.

Další dopad integrace je na autory obsahu do SIEM, kteří musejí vytvořit nové korelace pro přenesení výstrah z WAF do SIEM. V rámci této diplomové práce bylo korelační pravidlo vytvořeno a je detailně popsána v podkapitole 5.2.2. Následně by měl autor obsahu SIEM taktéž vytvořit odpovídající dashboardy, které by prezentovaly relevantní informace, jako jsou statistiky hrozeb, upozornění na bezpečnostní incidenty nebo informace o provozu, jenž prošel přes WAF. Tím se usnadní monitorování a rychlá reakce na potenciální hrozby. V rámci práce bylo vytvořeno několik základních dashboardů, které zobrazují objem komunikace procházející přes WAF. Posledním úkolem, jenž integrací pro autora obsahu do SIEM vzniká, je vytvoření detekce proti výpadku WAF, který by mohl práci SOC značně ohrozit. Tento úkol byl v rámci této práce v plném rozsahu realizován a je popsán v podkapitole 5.2.3.

Povinnosti SOC analytiků na všech úrovních (L1, L2 a L3) se příliš nemění. Základní změnou je, že musejí upravit svůj způsob přistupování ke všem technologiím, které jsou chráněné pomocí WAF, protože veškerá komunikace probíhá přes proxy, na níž je WAF umístěn. V případě prostředí SOC, jež je popsáno v této diplomové práci, se jedná například o změnu z `https://192.168.50.20/login` na `https://192.168.10.10/easyredmine`. Práce L1 analytiků zůstane téměř stejná. Je pouze třeba provést školení, v němž bude popsáno, jakým způsobem je možné WAF využít k analýze alertů a seznámení s novými alerty, které budou pomocí informací z WAF generovány. Analytici druhé a třetí úrovně L2 a L3 budou muset být s fungováním WAF detailněji seznámeni, protože budou muset konfiguraci a obsah WAF spravovat tak, aby co nejlépe splňovaly požadavky SOC a jejich zákazníků.

5.3.2 Vliv na procesy v SOC

V rámci diplomové práce byl veškerý postup zaznamenáván a zdokumentován v průběhu řešení integrace. Díky tomu je velká část dokumentace již vytvořena a dokumentaci z této práce stačí pouze přenést do dokumentace SOC. Co se týče změn, tak se jedná pouze o malé úpravy, které berou v potaz nově integrovaný WAF a změnu průchodu některé komunikace sítí, jež nově prochází skrz reverzní proxy server. Následně je také potřeba upravit postupy a runbooky, kterých se integrace týkají. Tato úprava je poměrně rozsáhlá, a proto nebyla v rámci této práce provedena. Je tedy důležité, aby ji odpovědní pracovníci SOC ještě co nejdříve provedli.

Jedná se o zdokumentování integrace a vytvoření návodů, díky nimž bude možné tento postup jednoduše a rychle zopakovat při napojování dalších technologií na reverzní proxy, případně nasazování WAF u dalších zákazníků. Je to dokumentace popisující instalaci a konfiguraci WAF, jejíž vytvoření mají na starosti techničtí

pracovníci a z velké části byla již v rámci diplomové práce vytvořena. Následně se jedná o zdokumentování vytvořených korelačních pravidel, dashboardů a detekcí výpadků, za které je zodpovědný pracovník spravující obsah SIEM. Opět byla dokumentace z velké části vypracována, je nutné pouze vytvořit příslušné dashboardy. Dále je potřeba upravit a doplnit jsou postupy a runbooky, které analytici při své každodenní práci využívají. Analytici první úrovně upraví runbooky, jež využívají, a vytvoří nové potřebné runbooky, které budou popisovat nově vznikající incidenty. Analytici druhé a třetí úrovně pouze upraví postupy, jež využívají při analýze případných incidentů.

Poslední částí úpravy procesů je vytvoření zcela nového procesu, který bude popisovat udržování a správu samotného WAF. Jedná se zejména o aktualizaci využívaných ModSecurity pravidel tak, aby odpovídala aktuálním hrozbám v síti SOC. Součástí tohoto procesu bude i vytváření dynamických pravidel pomocí vytvořené integrace s platformou MISP. Tento proces by měli vytvořit analytici třetí úrovně.

6 Analýza vlivu WAF na prevenci a detekci kybernetických hrozeb

Tato kapitola se zabývá přínosem WAF pro SOC ve všech ohledech. Jedná se především o vliv na prevenci a detekci kybernetických hrozeb, zvýšení viditelnosti v síti a vliv na tvorbu alertů. Ať už se jedná o změny v generování stávajících alertů, nebo generování nových. V rámci rozsahu této diplomové práce je kladen důraz spíše na změnu viditelnosti v síti než na generování alertů. Toto je dáno především tím, že WAF nebyl nasazen v rozsahu celého SOC.

6.1 Zlepšení viditelnosti v síti

Jedním z hlavních přínosů integrace WAF do prostředí SOC je zlepšení viditelnosti v monitorované síti, ať už se jedná o samotnou síť v SOC, nebo o síť zákazníka využívající služby SOC. Jak bylo již popsáno v kapitole 3, viditelnost monitorované sítě je jedním z nejdůležitějších požadavků na dobrou práci SOC v oblasti zabezpečení prevence a detekce proti hrozbám.

V aktuální architektuře SOC bez integrovaného WAF dochází k omezené viditelnosti komunikace na aplikační úrovni sítě. Viditelnost je určena převážně zařízeními, z nichž se záznamy v dohledovém centru sbírají, a zároveň využívanými technologiemi. Jedná se primárně o technologii Flowmon, která využívá sondy k získávání a analýze síťového provozu pomocí technologie zvané NetFlow nebo IPFIX. Sondy sbírají metadata o síťových tocích, což centru umožňuje získávat hluboký pohled na provoz v daných sítích. Jedná se hlavně o detailní monitorování provozu, objemovou analýzu a detekci anomálií v síti, jako jsou například pokusy o DDOS útoky nebo pokusy o exfiltraci. Další technologií je ESET Inspect and Protect, jež se zaměřuje převážně na koncové stanice, jako jsou osobní počítače nebo telefony. EDR (Endpoint detection and response) neustále monitoruje a zaznamenává všechny procesy a síťové komunikace na zařízeních a zároveň používá různé metody, včetně heuristik, behaviorální analýzy a reputačních databází, pro identifikaci malware, ransomware a dalších hrozeb. Lze tvrdit, že Flowmon zajišťuje viditelnost sítě z makroskopického pohledu a EDR z pohledu mikroskopického.

Viditelnost komunikace webových aplikací je částečně řešená pomocí sběru záznamů z různých síťových zařízení a aplikací vyskytujících se v monitorované síti. Díky těmto záznamům lze zčásti odhalit například pokus o XSS útok nebo SQL útok na webovou aplikaci, případně pokus o nadměrné využívání HTTPS odpovědí. Jedná se ale pouze o velmi omezenou viditelnost, která je značně závislá na možnostech protokolování jednotlivých zařízení a aplikací.

ModSecurity tedy rozšiřuje viditelnost webové komunikace, a to tím, že veškerá komunikace musí procházet skrz webový firewall. V případě této integrace se jedná o viditelnost, jež se týká všech webových aplikací dostupných přes reverzní proxy, na které je ModSecurity umístěn. Následně jsou viditelnost a detekce velmi ovlivněné pravidly, jež ModSecurity využívá.

6.2 Demonstrace aktivního využití WAF

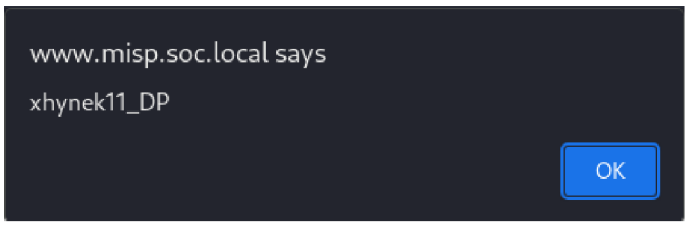
V rámci této diplomové práce byly zkoumány možnosti aktivního využití WAF na specifické a aktuální kybernetické hrozby v síti. Důraz byl kladen na schopnost flexibilní reakce ModSecurity na měnící se bezpečnostní prostředí tím, že umožňuje rychlé implementace pravidel zaměřených na nově identifikované techniky útoků a zranitelnosti. Z tohoto důvodu byl popsán konkrétní případ použití na zranitelnosti s označením CVE-2024-25674, který se nachází v platformě MISP, jež je pomocí WAF v SOC chráněna.

Kritická zranitelnost CVE-2024-25674 se nachází ve verzích platformy MISP, které jsou starší než verze 2.4.184. Při průběhu psaní této práce se tato zranitelnost objevila a trvalo několik dní, než na ni vývojáři MISP zareagovali a vydali aktualizaci s danou záplatou. Jedná se o poměrně běžný scénář u projektů s otevřeným zdrojovým kódem, jež jsou celé nebo z velké části vyvíjeny komunitou. Následně není u některých systémů vhodné aktualizaci okamžitě provést, zejména v případech, kdy se jedná o větší aktualizace, v rámci kterých jsou potřebné i bezpečnostní záplaty a zároveň se jedná o kritické systémy. Osvědčeným postupem je provést aktualizaci nejdříve v testovacím prostředí a všechny důležité funkcionality následně vyzkoušet a ověřit jejich funkčnost. Tímto může vzniknout až několikadenní prodleva, kdy bude systém zranitelný. Díky ModSecurity je ale možné napsáním vlastního pravidla, které tuto zranitelnost mitiguje, rychle a efektivně reagovat. Tento postup byl zvolen i v tomto případě a v rámci této práce popsán [49].

Výpis 6.1: Jednoduchý XSS payload sloužící pro demonstraci zranitelnosti v platformě MISP.

```
<svg xmlns="http://www.w3.org/2000/svg" \
    viewBox="0 0 500 500">
    <script>//<![CDATA[
        alert("xhynek11_DP")
    //]]>
    </script>
</svg>
```

Aktuální nastavení ModSecurity využívá základní sadu pravidel CRS doplněnou o dynamicky vytvořená pravidla pomocí platformy MISP a python scriptů. Detailní popis nastavení ModSecurity a využívaných pravidel se nachází v podkapitolách 4.1.2 a 4.1.3. Při tomto nastavení je možné zneužít výše zmíněnou zranitelnost pro nahrání souboru ve formátu Scalable Vector Graphics (SVG), jenž bude obsahovat škodlivý payload umožnující XSS útok. Obsah škodlivého souboru je zobrazen ve výpisu 6.1. Ve výpisu je uveden obsah SVG souboru s vloženým JavaScriptem, který při načtení ve webovém prohlížeči vyvolá funkci alert() zobrazující řetězec `"xhynek11_DP"`. Celý kód využívá XML namespace specifikace pro SVG a CDATA sekci k izolaci JavaScriptu od XML parseru, což umožňuje spustit skript bez interpretace jako XML značky. Skript slouží pro demonstraci samotné zranitelnosti, kterou je špatná MIME validace souborů, a je díky ní možné nahrát škodlivý soubor na platformu MISP a následně ho spustit. K nahrání souborů může dojít v sekci `Administration` v podsekci `Server Settings & Maintence`, kde lze pomocí nastavení v `Manage files` nahrát logo organizace ve formátu `.png` nebo další grafické soubory ve formátech `.png` a `.svg`. Obě tyto možnosti nahrání grafických souborů by měly být zranitelné, ale v rámci této práce byl demonstrován pouze způsob zneužití SVG souborů. Po nahrání škodlivého souboru s názvem `logo.svg` do složky `/var/www/MISP/app/webroot/img/custom` na platformu MISP bylo přistoupeno na stránku `https://www.misp.soc.local/img/custom/logo.svg`. Po přistoupení byl spuštěn JavaScript příkaz, jenž vyvolal akci zobrazenou na obrázku 6.1.



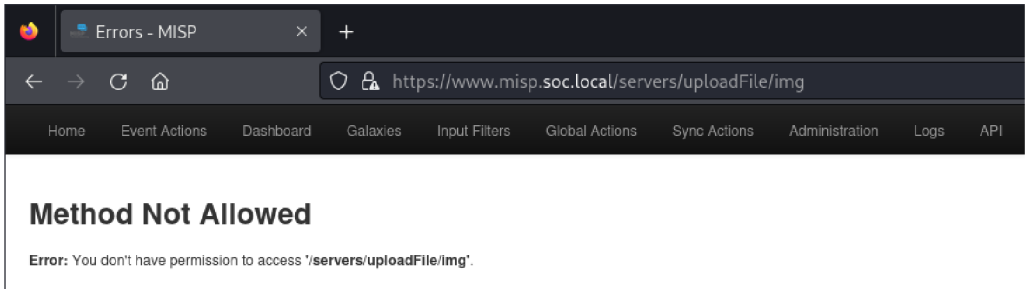
Obr. 6.1: Ukázka spuštění JavaScript skriptu na platformě MISP.

V rámci mitigace této zranitelnosti bylo vytvořeno vlastní komplexní pravidlo zamezující nahrání SVG souboru s různými typy škodlivých kódů. Celé pravidlo je zobrazeno ve výpisu 6.2 a v elektronické příloze jako `96_PSN_cusotm_SVG.conf`. Hlavní zaměření pravidla je nejdříve detekce pokusu o nahrání SVG souboru a následně jeho zablokování, pokud se v obsahu souboru nachází nějaký potenciálně škodlivý kód, který by mohl vést k jeho zneužití. Jeho detekce probíhá pomocí hlídání klíčových slov v těle dotazu. Příklad těchto klíčových slov je `<script`, `<link` nebo `@import`. Přidání tohoto pravidla do ModSecurity proběhlo stejným způsobem, jaký byl popsán v podkapitole 4.1.3. Po přidání pravidla znovu proběhlo načtení konfigurace Apache serveru. Následně byla znovu otestována výše zmíněná zranitelnost.

Výpis 6.2: ModSecurity pravidlo vytvořené pro zamezení nahrání SVG souborů obsahujících potenciálně zneužitelný kód.

```
ySecRule REQUEST_FILENAME "@endsWith \
    /servers/uploadFile/img" \
    "id:'1000002',\
    phase:2,\
    block,\
    log,\
    msg:'Blocking potentially malicious SVG uploads.',\
    chain"
    SecRule REQUEST_BODY "@rx <svg[^>]*> \
        (?:.*?<script.*?>.*?</script>|.*?on[a-z]+=['\"] \
        javascript:.*?['\"]|.*?href=['\"] \
        (http|https|ftp)://.*?['\"])" \
        "t:none,\
        ctl:auditLogParts=+E"
```

Vlastní pravidlo slouží k zablokování škodlivého požadavku, takže při pokusu o nahrání škodlivého souboru byl dotaz zablokován a k jeho nahrání na platformu MISP nedošlo. Zablokování požadavku je zobrazeno na obrázku 6.2.



Obr. 6.2: Ukázka zablokování škodlivého dotazu na platformu MISP.

Další dobrý příklad poukazující na přínos ModSecurity je zranitelnost CVE-2023-37307. Jedná se o Stored XSS zranitelnost nacházející se v platformě MISP verze 1.7.1 a starších. Přihlášený uživatel s nízkými oprávněními může pomocí podsekce `Add Cluster` v sekci `Galaxies` vložit XSS škodlivý kód, jejž nejdříve uloží a následně při zobrazení relevantního clusteru spustí vložený kód. Podobné typy zranitelností jsou v open-source systémech běžné. Nasazený ModSecurity s konfigurací, již je popsána v této práci, dokáže těmto typům zranitelností zabránit, a to bez jakéhokoliv zásahu člena SOC týmu [50].

6.3 Vyhodnocení vlivu na detekci hrozeb

Tato podkapitola se zaměřuje na posouzení integrace WAF do SOC v kontextu vzniku detekcí a identifikace hrozeb. Cílem podkapitoly je analyzovat, do jaké míry WAF změnil charakter a frekvenci detekovaných hrozeb a jaké nové typy výstrah byly zavedeny po jeho implementaci. Zároveň se podkapitola věnuje evaluaci, zda implementace WAF vedla k redukci falešně pozitivních výstrah, což je klíčové pro efektivní využití zdrojů SOC a zlepšení jeho reakční schopnosti. V neposlední řadě výsledky poskytují cenné empirické údaje pro SOC a mohou sloužit jako vodítko pří budoucím rozšiřování WAF na další webové aplikace, případně v integraci do prostředí zákazníků.

Datová sada, která byla pro analýzu použita, se skládá z výstrah, jež vznikly v rámci SOC za období jednoho měsíce. V prvních 14 dnech nebyl do SOC WAF integrován a následujících 14 dnů již byla integrace provedena. Data byla očištěna o několik sloupců, které byly prázdné, neúplné nebo pro tuto analýzu irelevantní. Jedná se například o sloupce `Agent Hostname`, `Device Custom String1` nebo `Generator ID`. Dále byla odstraněna data, jež se týkala chyby v DeviceDown listu, v němž se uchovávají informace o tom, jak často a kdy naposledy kontrolované zařízení zaslalo informace o záznamech. Tato chyba byla ojedinělá a vygenerovala velké množství událostí, které by mohly celou analýzu ovlivnit. Poslední a nejmenší úpravou bylo odstranění několika duplicit, jež se v datové sadě objevovaly.

End Time	Manager Receipt Time	Name	P	Attacker Address	Attacker Host Name
26 Apr 2024 05:15:33 CEST	26 Apr 2024 05:16:31 CEST	FlowMon ADS Alert - SSH attack	5	192.168.40.43	centreon
26 Apr 2024 11:55:54 CEST	26 Apr 2024 11:56:39 CEST	Multiple Inside Denies and an Allow from Same Source	7	192.168.50.20	easyredmine
26 Apr 2024 13:53:24 CEST	26 Apr 2024 13:53:51 CEST	Multiple Inside Denies and an Allow from Same Source	7	192.168.50.20	easyredmine
26 Apr 2024 15:55:53 CEST	26 Apr 2024 15:57:02 CEST	Multiple Inside Denies and an Allow from Same Source	7	192.168.50.20	easyredmine
27 Apr 2024 01:36:07 CEST	27 Apr 2024 01:37:22 CEST	Detected Linux File Inclusion Attack	7	192.168.20.199	misp
27 Apr 2024 01:36:07 CEST	27 Apr 2024 01:37:22 CEST	Detected Linux File Inclusion Attack	7	192.168.20.199	misp
20 Apr 2024 11:24:51 CEST	20 Apr 2024 11:26:07 CEST	Detected Null Byte or Other Potential Dangerous Character	7	192.168.20.199	misp
27 Apr 2024 01:38:30 CEST	27 Apr 2024 01:40:11 CEST	Detected Null Byte or Other Potential Dangerous Character	7	192.168.50.20	easyredmine
27 Apr 2024 01:38:30 CEST	27 Apr 2024 01:40:11 CEST	Detected Null Byte or Other Potential Dangerous Character	7	192.168.50.20	easyredmine
27 Apr 2024 01:38:30 CEST	27 Apr 2024 01:40:11 CEST	Detected SQL Concatenated Injection	7	192.168.20.199	misp
15 Apr 2024 10:35:17 CEST	15 Apr 2024 10:36:19 CEST	Detected XSS payload in SVG file	8	192.168.20.199	misp

Obr. 6.3: Ukázka části datové sady obsahující výstrahy SOC po integraci WAF.

Dále byla v rámci zachování bezpečnosti SOC a dodržování jeho interních pravidel datová sada anonymizována tak, aby při zveřejnění datasetu v elektronické příloze práce nebylo možné získat z dat jakékoliv citlivé údaje, například o jednotlivých uživatelích nebo dílčích částech segmentace sítě. Nepotřebná data byla odstraněna a citlivá data, která jsou pro následující analýzu stěžejní, byla maskována. Maskování probíhalo zejména pomocí substituce, kdy byly konkrétní IP adresy nahrazeny jinými. Zároveň na sloupec `Comments` byla využita metoda generalizace. Pomocí této metody byla z komentářů zachována pouze informace, zda se jedná o falešně nebo opravdově pozitivní výstrahu. Díky tomu může být následující analýza

v případě potřeby replikována nebo mohou být data zpracována jiným způsobem, bez ohrožení bezpečnosti daného SOC. Ukázka části anonymizované datové množiny je zobrazena na obrázku 6.3. Celé datové sady jsou k nalezení v elektronické datové příloze pod názvy `dataset_bez_waf.csv` a `dataset_s_waf.csv`.

Tab. 6.1: Tabulka zobrazující změny v datové sadě 1, která byla vytvořena před nasazením WAF a v datové sadě 2, která byla vytvořena po nasazení WAF.

-	Datová sada 1	Datová sada 2	Změna [ks]	Změna [%]
Počet výstrah	1 136	1 214	78	6,4 %
Typy výstrah	22	26	4	15,4 %
Míra FP výstrah	36,2 %	30 %	-	-14,5 %
Výstrahy MISP	13	13	0	0 %
Výstrahy ERM	11	16	5	31,25 %

Z analýzy je patrné, že ve sledovaných obdobích došlo po integraci WAF k nárůstu detekovaných výstrah. Konkrétně se jedná o 7% změnu z 1 130 výstrah na 1 215 výstrah. Takovýto nárůst je zcela odpovídající integraci nové technologie, jež generuje a zasílá záznamy do SIEM, kde jsou následně korelovány. Lze tedy předpokládat vyšší viditelnost sítě, a díky tomu také zlepšení detekčních schopností SOC. Změny v jednotlivých datových množinách jsou zobrazeny v tabulce 6.1. Další patrnou změnou ve zkoumaných datových množinách je nárůst typů výstrah. To je způsobeno nejen integrací WAF, ale také napsáním vlastních korelačních pravidel v SIEM. Z velké části došlo k detekci opravdově pozitivních výstrah, což svědčí o správně napsaných a optimalizovaných pravidlech ve WAF a SIEM. Zároveň dobrou optimalizaci pravidel a přínos WAF potvrzuje fakt, že došlo k detekci stejné události pomocí WAF i SIEM, jež odhalila pokus o SQLi útok pomocí sběru aplikačních záznamů. Důkazem správné integrace je rovněž stabilita v detekci známých hrozeb. Přes rozšíření detekčních schopností totiž zůstává počet výstrah z aplikací jako Easy Redmine a MISP stabilní. To značí, že implementace WAF nevedla k oslabení detekce stávajících hrozeb.

Výsledky naznačují, že WAF významně zvyšuje možnosti SOC detekovat kybernetické hrozby. Nasazení WAF nejenže zlepšilo viditelnost v dané síti společně s počtem a rozmanitostí výstrah, ale taktéž zmenšilo míru vzniku falešně pozitivních výstrah. Při interpretaci těchto výsledků je ale nutné brát v úvahu délku období, za které datové množiny pro analýzu vznikaly. Obě datové množiny pokrývají pouze období 14 dní, což je relativně krátký časový úsek pro definitivní závěry o dlouhodobých trendech a celkové efektivitě nasazení WAF.

Přestože má tato analýza určitá omezení z hlediska délky sledovaného období, předběžné výsledky naznačují, že nasazení WAF má pozitivní dopad nejen na zvý-

šení počtu a rozmanitosti detekovaných výstrah, ale i na míru generování falešně pozitivních výstrah, což představuje významný krok k posílení obranných mechanismů a služeb, jež SOC nabízí.

6.4 Doporučení a budoucí směry

Níže zmíněná doporučení a budoucí směry do této diplomové práce nebyly zahrnuty, a to z důvodu překročení rámce a rozsahu této práce. Zároveň jde o velmi časově náročné části, jichž by nebylo možné v časovém rozmezí vymezeném pro tuto práci dosáhnout.

Výsledky dosavadní navržené integrace a testování naznačují, že nasazení WAF na další webové aplikace je žádoucí krok v posílení kybernetické bezpečnosti v síti daného SOC. V souladu s tím se WAF rozšíří na všechny klíčové technologie používané v rámci SOC, které používají webové rozhraní. Jedná se o nižší desítky technologií s různou kritičností pro fungování celého SOC. Rozšíření WAF by tedy probíhalo postupně od nejvíce kritických systémů, jako je již chráněný Easy Redmine, až po méně kritické systémy. Tento proces nebude příliš náročný a podle zátěžových testů, které byly provedeny, by integrovaný WAF měl bez problémů napojení všech těchto systémů zvládnout. Zahrnuje pouze přidání nových záznamů o přesměrování provozu do reverzního proxy serveru a úpravu DNS záznamů tak, aby byla používaná doménová jména správně přiřazena k IP adrese proxy serverů.

Po rozšíření na další technologie SOC by měla opět proběhnout fáze přizpůsobení jednotlivých WAF pravidel potřebám přidaných technologií. Tato fáze již nebude příliš náročná, protože využívaná CRS pravidla jsou velmi dobře optimalizována a v rámci této práce přizpůsobena potřebám SOC. Co se týká korelačních pravidel v rámci technologie SIEM nebo například vytvořeného parseru v rámci integrace WAF, nebude potřeba provádět žádné změny. Obě části byly v rámci integrace nastaveny tak, aby mohlo bez zásahů pracovníků SOC dojít k rozšíření WAF na další technologie.

Pro udržení efektivity WAF je klíčové vybudovat pevné procesy pro jeho správu a udržování. Tyto procesy by měly zahrnovat pravidelné aktualizace pravidel, monitorování jejich výkonnosti a reagování na nově identifikované hrozby. S pravidelnou aktualizací pravidel by měla pomoci vytvořená integrace WAF s platformou MISP. Potřebné procesy byly blíže popsány v podkapitole 5.3.2. Všechny tyto činnosti zajistí, že WAF zůstane účinným nástrojem v boji proti kybernetickým hrozbám a adaptuje se na měnící se bezpečnostní povrch organizace.

Posledním směrem, jak by bylo možné integraci a všechny postupy s ní spojené využít, je nasazení WAF i v infrastrukturách zákazníků SOC. Jak již bylo v této práci zmíněno, popisovaný SOC, v němž byl WAF integrována, nabízí svoje služby

i dalším společnostem. Implementace WAF do jejich infrastruktur by výrazně rozšířil rozsah ochrany a umožnil SOC poskytovat pokročilejší bezpečnostní služby, které by dané společnosti chránily. Integrace WAF u dalších společností by zahrnovala analýzu specifických potřeb každého zákazníka, konfiguraci WAF a kontinuální podporu a správu systému. Všechny postupy popsané v této práci by takové nasazení ulehčily a zkrátily.

Závěr

Diplomová práce se zabývá integrací WAF do architektury SOC. V souvislosti s tím byly identifikovány současné možnosti integrace, jež by co nejlépe vyhovovaly požadavkům a potřebám SOC, které byly v rámci práce rovněž identifikovány.

Na základě podrobné analýzy všech možností integrace byl jako nejvhodnější WAF zvolen ModSecurity nasazený v módu reverzního proxy serveru. Tímto způsobem může ModSecurity chránit velké množství aplikací s webovým rozhraním, které jsou v rámci prostředí SOC využívány. Následně byl WAF nasazen a nakonfigurován v testovacím prostředí, včetně nasazení a upravení detekčních pravidel. Po tomto nasazení proběhlo důkladné testování fungování vybraného WAF jak z pohledu možností detekování webových útoků, tak z pohledu zátěžového testování. Výsledkem tohoto testování je skutečnost, že základní sada pravidel CRS je velmi dobře optimalizovaná a pro potřeby SOC je nutnost pouze minimálních úprav. Nasazený WAF dokáže eliminovat většinu základních webových útoků a v módu reverzní proxy je i velmi dobře škálovatelný. V testovacím prostředí byl také úspěšně demonstrován způsob napojení WAF na technologie SOC jako Log Management a SIEM.

Po úspěšné validaci testovacího nasazení ModSecurity bylo přistoupeno k integraci do reálného prostředí. V rámci SOC byly vybrány dvě technologie, na kterých bylo nasazení provedeno. Jednalo se o technologii pro správu projektů Easy Redmine a platformu pro sdílení IoC MISP. Tyto aplikace byly vybrány určitým kompromisem mezi mírou zásahu do infrastruktury, využitelností technologií, jež jsou pro SOC důležité a aktivně používané, a současně množstvím výstrah, které vznikne nasazením WAF na vybrané aplikace, jež budou následně podrobněji analyzovány. Přidání WAF do infrastruktury SOC bylo poměrně snadné, protože se jednalo o vložení serveru s reverzní proxy, přes niž byl požadovaný síťový provoz přesměrován.

Stěžejní částí integrace bylo napojení WAF na již používané technologie SOC, kterými jsou ArcSight Archiver, ArcSight Logger, ArcSight ESM, Centreon a MISP. K napojení záznamů byla použita technologie rsyslog, jež zasílá záznamy na Connector server, který je následně distribuuje do technologií ArcSight. K tomuto napojení musel být vytvořen vlastní parser, který před zpracováním záznamy z ModSecurity normalizuje. Napojení na provozní monitoring proběhlo pomocí využívaných šablon v rámci daného SOC. Poslední částí byla integrace s platformou MISP, jež musela být vytvořena pomocí jejího API a vlastních skriptů. Tato integrace nakonec umožňuje tvorbu dynamických ModSecurity pravidel za pomoci sdílených informací na platformě MISP.

Poslední část diplomové práce se zaměřila na analýzu vlivu integrace WAF na prevenci a detekci kybernetických hrozeb. V rámci této části byl vytvořen i scénář, jak lze ModSecurity použít pro zabezpečení proti aktivním hrozbám. Primárně

byl ale analyzován vliv integrace na viditelnost v chráněné síti a na vznik bezpečnostních výstrah v rámci SOC. Výsledky naznačují, že došlo k nezanedbatelnému zvýšení viditelnosti i detekci bezpečnostních hrozeb a zároveň ke snížení počtu falešně pozitivních výstrah. Je však důležité zmínit, že nasazení nebylo tak rozsáhlé a časové období, kdy probíhala analýza vznikajících výstrah, bylo relativně krátké. To znamená že, výsledky analýzy mohou být mírně zkreslené.

Hlavním přínosem práce bylo tedy navrhnutí a otestování integrace WAF do SOC, jež nejlépe odpovídá jeho potřebám a požadavkům. Zároveň byla současně provedena integrace do reálného prostředí, která byla následně analyzována a byl popsán její vliv na prevenci a detekci kybernetických hrozeb. Dílčími přínosy bylo optimalizování sady pravidel CRS, vytvoření vlastního parseru pro normalizaci záznamů a vytvoření vlastní integrace s platformu MISP umožňující včasnou detekci aktuálních webových útoků.

Výsledky této práce by mohly být využity k rozšíření integrace WAF na další technologie v rámci daného SOC anebo pro integraci do sítí jeho zákazníků. Případně lze jednotlivé části práce využít pro zvýšení bezpečnosti webových aplikací v libovolných sítích za pomoci WAF.

Literatura

[1] SHARP, R. *Introduction to Cybersecurity: A Multidisciplinary Challenge.* 2197-1781. Springer Cham, 2023. ISBN 978-3-031-41463-3. [online]. [cit. 25. 04. 2024]. Dostupné z: <https://doi.org/https://doi-org.ezproxy.lib.vutbr.cz/10.1007/978-3-031-41463-3>.

[2] Sabyasachi, P., DEBABRATA, S., VINAY, M., GUHA, A. *Cyber Security and Network Security: Advances in Cyber Security.* John Wiley & Sons, 2022. ISBN 1119812496, 9781119812494. [online]. [cit. 28. 04. 2024]. Dostupné z: <https://www.google.cz/books/edition/Cyber_Security_and_Network_Security/EulmEAAAQBAJ?hl=cs&gbpv=0>.

[3] JACOBS, P., ARNAB, A., IRWIN, B. *Classification of Security Operation Centers.* Information Security for South Africa, Johannesburg, South Africa, 2013. DOI 10.1109/ISSA.2013.6641054. [online]. [cit. 12. 04. 2024]. Dostupné z: <https://ieeexplore.ieee.org/document/6641054>.

[4] MUGHAL, A. A. *Building and Securing the Modern Security Operations Center (SOC). International Journal of Business Intelligence and Big Data Analytics. 2022. 1-15.* [online]. [cit. 19. 04. 2024]. Dostupné z: <https://research.tensorgate.org/index.php/IJBIBDA/article/view/21/20>.

[5] KIIVERI, K. *Automation in Cyber Security.* Bakalářská práce. Turku, Finsko: Turku University of Applied Sciences, 2021. [online]. [cit. 18. 5. 2024]. Dostupné také z: <https://www.theseus.fi/bitstream/handle/10024/503899/Kiiveri_Konsta.pdf?sequence=1&isAllowed=y>.

[6] MUNIZ, J., MCINTYRE, G., ALFARDAN, N. *Security Operations Center: Building, Operating, and Maintaining your SOC.* Indianapolis, USA: Cisco Press, 2016. [cit. 14. 5. 2024]. ISBN 978-0-13-405201-4.

[7] SHAHJEE, D., WARE, N. *Integrated Network and Security Operation Center: A Systematic Analysis.* 2022, roč. 10, s. 27881-27898. ISSN 2169-3536. [online]. [cit. 20. 5. 2024]. Dostupné z: <https://doi.org/10.1109/ACCESS.2022.3157738>.

[8] UMESH, H. R., UMESHA, N. *Firewalls. Online. Berkeley, CA: Apress, 2014.* ISBN 978-1-4302-6383-8. [online]. [cit. 20. 4. 2024]. Dostupné z: <https://doi.org/978-1-4302-6383-8_10>.

[9] SIDABUTAR, J., PRIAMBODO, D. F., SEPTIANTY, N. F., GURNING, K. Y., JULIARTA, F. *Comparative Study of Open-source Firewall.* IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs). 2023, s. 165-172. ISBN 979-8-3503-3943-7. [online]. [cit. 17. 4. 2024]. Dostupné z: <https://doi.org/10.1109/ICoCICs58778.2023.10276707>.

[10] *Cloudflare: What is a Web Application Firewall (WAF)?.* 2024, Cloudflare, Inc. [online]. [cit. 20. 5. 2024]. Dostupné z: <https://www.cloudflare.com/en-gb/learning/ddos/glossary/web-application-firewall-waf/>.

[11] YASAR, K. *TechTarget Security: Web application firewall (WAF).* 2024, TechTarget [online]. [cit. 4. 4. 2024]. Dostupné z: <https://www.techtarget.com/searchsecurity/definition/Web-application-firewall-WAF>.

[12] *StealthLabs: Cyber Security Threats and AttacksAll (You Need to Know).* 2020, Stealth Labs, Inc. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://www.stealthlabs.com/blog/cyber-security-threats-all-you-need-to-know/>.

[13] *Proofpoint: Network-Delivered Threats.* 2024, Proofpoint LTD. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://www.proofpoint.com/uk/threat-reference/network-delivered-threats>.

[14] *OWASP: SQL Injection.* 2024, OWASP Foundation, Inc. [online]. [cit. 21. 4. 2024]. Dostupné z: <https://owasp.org/www-community/attacks/SQL_Injection#>.

[15] *OWASP: Cross Site Scripting (XSS).* 2024, OWASP Foundation, Inc. [online]. [cit. 21. 4. 2024]. Dostupné z: <https://owasp.org/www-community/attacks/xss/#>.

[16] *OWASP: Cross Site Request Forgery (CSRF).* 2024, OWASP Foundation, Inc. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://owasp.org/www-community/attacks/csrf#>.

[17] *OWASP: Command Injection.* 2024, OWASP Foundation, Inc. [online]. [cit. 22. 4. 2024]. Dostupné z: <https://owasp.org/www-community/attacks/Command_Injection>.

[18] SEGAL, E. *DZone: The SOC Technology Stack: XDR, SIEM, WAF, and More.* 2021, DZone. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://dzone.com/articles/the-soc-technology-stack-xdr-siem-waf-and-more>.

[19] RAZZAQ A., HUR A., SHAHBAZ S., MASOOD M., AHMAD H. F. *Critical analysis on web application firewall solutions.* 2013, IEEE. DOI: 10.1109/I-SADS.2013.6513431. [online]. [cit. 19. 4. 2024]. Dostupné z: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6513431>.

[20] *AWS: AWS Firewall Manager, and AWS Shield Advanced. Developer guide.* 2024, Amazon Web Services, Inc. [online]. [cit. 28. 4. 2024]. Dostupné z: <https://docs.aws.amazon.com/waf/latest/developerguide/waf-chapter.html>.

[21] *NGINX docs: NGINX App Protect WAF.* [online]. 2024, F5, Inc. [cit. 12. 5. 2024]. Dostupné z: <https://docs.nginx.com/nginx-app-protect-waf/>.

[22] *Cloudflare docs: Cloudflare Web Application Firewall.* 2024, Cloudflare, Inc. [online]. [cit. 12. 5. 2024]. Dostupné z: <https://developers.cloudflare.com/waf/>.

[23] PLESKY, E. *Plesk: ModSecurity Comprehensive Guide.* 2021, WebPros International GmbH. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://www.plesk.com/blog/various/modsecurity-comprehensive-guide/>.

[24] MARTINHSV. *SpiderLabs: ModSecurity.* 2024, GitHub repository, GitHub, Inc. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://github.com/SpiderLabs/ModSecurity>.

[25] *Open-Appsec: open-appsec Documentation.* 2024, Check Point Software Technologies Ltd. [online]. [cit. 10. 5. 2024]. Dostupné z: <https://docs.openappsec.io/>.

[26] *Microsoft: AQTRONiX WebKnight.* 2024, AQTRONiX. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://www.iis.net/downloads/community/2016/04/aqtronix-webknight>.

[27] *Zenarmor: The Best Open Source Web Application Firewalls.* 2024, Sunny Valley Cyber Security Inc. [online]. [cit. 21. 4. 2024]. Dostupné z: <https://www.zenarmor.com/docs/network-security-tutorials/best-open-source-web-application-firewalls>.

[28] *Miscrosoft: WAFBench.* GitHub repository, 2024, GitHub, Inc. [online]. [cit. 24. 4. 2024]. Dostupné z: <https://github.com/microsoft/WAFBench>.

[29] *DVWA: Damn Vulnerable Web Application (DVWA).* GitHub repository, 2024, GitHub, Inc. [online]. [cit. 10. 5. 2024]. Dostupné z: <https://github.com/digininja/DVWA>.

[30] VIELBERTH, M., BöHM, F., FICHTINGER, I., PERNUL, G. *Security operations center: A systematic study and open challenges.* IEEE Access. 2020, roč. 8, s. 227756-227779. ISSN 2169-3536. [online]. [cit. 11. 5. 2024]. Dostupné z: <https://doi.org/10.1109/ACCESS.2020.3045514>.

[31] SHAW, H., ELLIS, D. A., ZIEGLER, F. V. *The Technology Integration Model (TIM). Predicting the continued use of technology.* Computers in Human Behavior. 2018, roč. 83, s. 204-214. ISSN 07475632. [online]. [cit. 4. 5. 2024]. Dostupné z: <https://doi.org/10.1016/j.chb.2018.02.001>.

[32] *Apache: Apache HTTP Server Version 2.4 Documentation.* 2024, The Apache Software Foundation. [online]. [cit. 15. 5. 2024]. Dostupné z: <https://httpd.apache.org/docs/2.4/>.

[33] *OpenSSL: Cryptography and SSL/TLS Toolkit.* [online]. [cit. 2. 5. 2024]. Dostupné z: <https://www.openssl.org/docs/>.

[34] *Github ModSecurity: Open Source Web Application Firewall.* GitHub repository, 2024, GitHub, Inc. [online]. [cit. 14. 5. 2024]. Dostupné z: <https://github.com/owasp-modsecurity/ModSecurity>.

[35] *OWASP ModSecurity: Core Rule Set Documentation.* 2024, CRS Project. [online]. [cit. 7. 5. 2024]. Dostupné z: <https://coreruleset.org/docs/>.

[36] *Elastic: Elasticsearch Guide.* 2024, Elasticsearch B.V. [online]. [cit. 11. 5. 2024]. Dostupné z: <https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html>.

[37] *Elastic: Logstash Reference.* 2024, Elasticsearch B.V. [online]. [cit. 11. 5. 2024]. Dostupné z: <https://www.elastic.co/guide/en/logstash/current/index.html>.

[38] *Elastic: Kibana Guide.* 2024, Elasticsearch B.V. [online]. [cit. 11. 5. 2024]. Dostupné z: <https://www.elastic.co/guide/en/kibana/current/index.html>.

[39] *Github Siege: Siege is an http load tester and benchmarking utility.* GitHub repository, 2024, GitHub, Inc. [online]. [cit. 13. 5. 2024]. Dostupné z: <https://github.com/JoeDog/siege>.

[40] LENSTRA, Arjen K. a VERHEUL, Eric R. *Selecting Cryptographic Key Sizes. Journal of Cryptology.* 2001, roč. 14, č. 4, s. 255-293. ISSN 0933-2790. [online]. [cit. 4. 4. 2024]. Dostupné z: https://doi.org/10.1007/s00145-001-0009-4.

[41] *Micro Focus: ArcSight Logger Administration.* 2024, Micro Focus International plc. [online]. [cit. 13. 5. 2024]. Dostupné z: <https://www.microfocus.com/documentation/arcsight/logger-7.3/Logger_AdminGuide/>.

[42] *Opentext: SmartConnector for Apache HTTP Server Error File 8.4.3.* 2024, OpenText Corporation. [online]. [cit. 15. 5. 2024]. Dostupné z: <https://www.microfocus.com/documentation/arcsight/arcsight-smartconnectors-8.4/apache-http-error/index.html#ApacheHTTPError/03-configuring-apache-http-server-for-event-collection.htm?TocPath=_____3>.

[43] *rsyslog: The rocket-fast system for log processing.* 2020, Adiscon GmbH. [online]. [cit. 15. 5. 2024]. Dostupné z: <https://www.rsyslog.com/doc/index.html>.

[44] *Opentext: ArcSight Documentation.* 2024, Micro Focus International plc. [online]. [cit. 14. 5. 2024]. Dostupné z: <https://www.microfocus.com/documentation/arcsight/#gsc.tab=0>.

[45] *IBM Documentation: snmpd.conf File.* 2023, IBM Corporation. [online]. [cit. 15. 5. 2024]. Dostupné z: <https://www.ibm.com/docs/en/aix/7.1?topic=files-snmpdconf-file>.

[46] *Centreon: Setting up the monitoring.* 2024, Centreon. [online]. [cit. 15. 5. 2024]. Dostupné z: <https://docs.centreon.com/docs/category/setting-up-the-monitoring/>.

[47] *MISP Threat Sharing: MISP Documentation and Support.* 2024, MISP project. [online]. [cit. 15. 5. 2024]. Dostupné z: <https://www.misp-project.org/documentation/>.

[48] *IBM Documentation: crontab Command.* 2023, IBM Corporation. [online]. [cit. 18. 5. 2024]. Dostupné z: <https://www.ibm.com/docs/en/aix/7.1?topic=c-crontab-command>.

[49] *National Vulnerability Database: CVE-2024-25674 Detail.* 2024, NIST. [online]. [cit. 10. 5. 2024]. Dostupné z: <https://nvd.nist.gov/vuln/detail/CVE-2024-25674>.

[50] *National Vulnerability Database: CVE-2023-37307 Detail.* 2024, NIST [online]. [cit. 10. 5. 2024]. Dostupné z: <https://nvd.nist.gov/vuln/detail/CVE-2023-37307>.

Seznam symbolů a zkratek

API	Application Programming Interface – Rozhraní pro programování aplikací
CDN	Content delivery network – Síť pro doručování obsahu
CIA	Confidentality, Integrity, Availability – důvěrnost, integrita, dostupnost
CRS	Core Rule Set – Základní sada pravidel
CSRF	Cross-Site Request Forgery – Typ útoku na webové aplikace
DDoS	Distributed Denial of Service – Distribuované odepření služby
DNS	Domain Name System – Decentralizovaný systém doménových jmen
DoS	Denial of service – Odepření služby
DVWA	Damn Vulnerable Web Application – Zranitelná webová aplikace
ESM	Enterprise Security Manager – Manažer zabezpečení podniku
IIS	Internet Information Services – Softwarový webový server
IMS	Information management system – Manažerský informační systém
LM	Log management – Správa logů
MitM	Man in the middle – Typ síťového útoku
NIST	National Institute of Standards and Technology – Národní institut standardů a technologie
OWASP	Open Web Application Security Project – Projekt zabývající se bezpečností webových aplikací
RCE	Remote code execution – Typ útoku na webové aplikace
SE	Security Engineer – Bezpečnostní inženýr
SIEM	Security Information and Event Management – Management bezpečnostních informací a událostí
SOAR	Security orchestration, automation, and response – Systém pro orchestraci, automatizaci a reakci

SOC	Security Operation Center – Bezpečnostní dohledové centrum
SQLi	SQL injection – Typ útoku na webové aplikace
TI	Threat Intelligence – Informace o kybernetických hrozbách
WAF	Web Application Firewall – Brána firewall pro webové aplikace
XSS	Cross-site scripting – Typ útoku na webové aplikace

A Obsah elektronické přílohy

V elektronické příloze jsou k nalezení všechny soubory, které nebylo možné nebo vhodné vložit přímo do obsahu této práce. Obsahuje zejména všechny soubory využité při testování, a to jak použité škodlivé dotazy, tak i použité skripty. Taktéž jsou v příloze k nalezení výsledky zátěžového testování a vlastní napsané pravidlo.

```
/ ..... kořenový adresář přiloženého archivu
├── Analýza ..... adresář obsahující datové sady použité pro analýzu
│   ├── dataset_bez_waf.csv
│   └── dataset_bez_waf.csv
├── Centreon ..... adresář obsahující příkazy pro Centreon kontroly
│   ├── cpu.txt
│   ├── disk.txt
│   ├── load.txt
│   ├── memory.txt
│   └── snmpv3-any.txt
├── Konfigurační soubory ..... adresář obsahující všechny konfigurační soubory
│   ├── rsyslog.conf
│   ├── waf-proxy.conf
│   ├── snmpd.conf
│   ├── syslog-ng.conf
│   └── apache_modsec.properties
├── MISP ..... adresář obsahující soubory pro tvorbu dynamických pravidel
│   ├── dynamic_rules_sqli.py
│   ├── dynamic_rules_xss.py
│   └── payloads_MISP.py
├── Pravidla ..... adresář obsahující napsaná pravidla
│   ├── 96_PSN_cusotm_SVG.conf
│   ├── 97_PSN_custom.conf
│   ├── 98_PSN_custom.conf
│   └── 99_PSN_custom.conf
└── Test ..... adresář obsahující soubory využité u testování
    ├── Soubory
    │   ├── commandi_code.txt
    │   ├── reflected_XSS.txt
    │   ├── siege_URL.txt
    │   └── sqli_queries.txt
    ├── Skripty
    │   └── SQLi.py
    └── Výsledky
        ├── Bez_ModSecurity
        └── S_ModSecurity
```