Skenování zranitelností pro výrobní informační systémy

Albert Šiller

Bakalářská práce
2024

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Albert Šiller**
Osobní číslo:	**A21075**
Studijní program:	**B0613A140020 Softwarové inženýrství**
Forma studia:	**Prezenční**
Téma práce:	**Skenování zranitelností pro výrobní informační systémy**
Téma práce anglicky:	**Vulnerability Scanning for Manufacturing Information Systems**

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Seznamte se s komerčními a open-source nástroji určenými k analýze zranitelností webových a desktopových aplikací.
3. Seznamte se s architekturou jednoho z výrobních informačních systémů ve firmě Continental.
4. Pro daný systém zvolte vhodný nástroj k analýze zranitelností a tento nástroj nakonfigurujte tak, aby byl schopen monitorovat/analyzovat běžící testovací instanci výrobního systému.
5. Zhodnoťte dosažené výsledky a přínosy práce pro firmu Continental.

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. HOFFMAN, Andrew. Web Application Security: Exploitation and Countermeasures for Modern Web Applications. O'Reilly Media, 2020. ISBN 1492053112.
2. BLOKDYK, Gerardus. Vulnerability Scanning A Complete Guide. The Art of Service – Vulnerability Scan Publishing, 2020. ISBN 1867446383.
3. BLOKDYK, Gerardus. Vulnerability scanner The Ultimate Step-By-Step Guide. 5STARCooks, 2021. ISBN 0655330429.
4. STUTTARD, Dafydd; PINTO, Marcus. The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws. Wiley, 2011. ISBN 1118026470.
5. SULLIVAN, Bryan; LIU, Vincent. Web Application Security, A Beginner's Guide. McGraw Hill, 2011. ISBN 0071776168.

Vedoucí bakalářské práce: **Ing. Petr Žáček, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce: **5. listopadu 2023**
Termín odevzdání bakalářské práce: **13. května 2024**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 5. ledna 2024

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 7.5.202

Albert Šiller v.r.
podpis studenta

ABSTRAKT

Tato bakalářská práce se zaměřuje na analýzu zranitelností výrobního informačního systému Continental Global Manufacturing System (CGMS) ve firmě Continental Barum s.r.o. sídlící v Otrokovicích. Pro identifikaci zranitelností byla použita kombinace dvou nástrojů. Prvním z nich je Zed Attack Proxy, specializující se na skenování webových aplikací, a druhým je Nessus, zaměřující se na skenování zranitelností na serveru a v síti.

Analýza odhalila několik bezpečnostních rizik v analyzovaném výrobním informačním systému.

Zed Attack Proxy identifikoval zranitelnost vysokého rizika General Padding Oracle, která je způsobena kryptografickým selháním. Nessus pak prokázal existenci dvou kritických zranitelností. Společným znakem těchto identifikovaných zranitelností je jejich zastaralost a nedostatečná aktualizace, což znamená, že jim chybí podpora proti novým hrozbám a nejsou dostatečně připraveny na současné bezpečnostní výzvy. Tato neaktuálnost a nedostatečná podpora představují závažný problém, neboť aplikace a systémy jsou tak vystaveny vyššímu riziku útoků a narušení.

Zjištění této práce poskytuje cenné informace pro firmu Continental Barum s.r.o., které mohou pomoci zlepšení celkové bezpečnosti výrobního informačního systému CGMS. Identifikace a odstranění zjištěných zranitelností je klíčovým krokem ke zvýšení celkové kyberbezpečnosti a ochrany citlivých dat celého firemního systému CGMS.

Klíčová slova: Výrobní informační systém, Zranitelnost, Skenování, Kybernetické bezpečnost, Riziko, Zed Attack Proxy, Nessus

ABSTRACT

This bachelor's thesis focuses on the vulnerability analysis of the manufacturing information system Continental Global Manufacturing System (CGMS) in Continental Barum s.r.o., based in Otrokovice. Two tools were used to identify vulnerabilities. The first is Zed Attack Proxy, which specializes in scanning web applications, and the second is Nessus, which focuses on scanning server and network vulnerabilities.

The analysis revealed several security risks in the analyzed production information system.

Zed Attack Proxy identified a high-risk General Padding Oracle vulnerability caused by a cryptographic failure. Nessus then demonstrated the existence of two critical vulnerabilities. The common feature of these identified vulnerabilities is their outdatedness and lack of updating, which means they lack support against new threats and must be sufficiently prepared for current security challenges. This lack of up-to-date support is a severe problem, exposing applications and systems to a higher risk of attacks and breaches.

The findings of this work provide valuable information for Continental Barum s.r.o.. that can help improve the overall security of the CGMS manufacturing information system. Identifying and remediating the identified vulnerabilities is a key step in improving the overall cybersecurity and protection of sensitive data of the entire corporate CGMS.

Keywords: Manufacturing Information System, Vulnerability, Scanning, Cybersecurity, Risk, Zed Attack Proxy, Nessus

Chtěl bych poděkovat svému vedoucímu práce Ing. Petrovi Žáčkovi, Ph.D., za jeho vedení a neustálou podporu během psaní této bakalářské práce. Jeho odborné znalosti a schopnost vést mě správným směrem byly pro mě neocenitelné.

Dále bych rád poděkoval Ing. Jaromíru Šánkovi za jeho cenné rady, které mi poskytoval během praktické části této bakalářské práce. Jeho odbornost a zkušenosti mi otevřely nové perspektivy a přinesly mi obrovský přínos.

Nesmím zapomenout ani na mou rodinu, která mi během psaní poskytla nezbytnou podporu.

OBSAH

ÚVOD

Bakalářská práce se zaměřuje na skenování zranitelností výrobních informačních systémů, což je téma, které jsem si vybral z několika důvodů. Za prvé, mám zkušenosti v testování softwaru, avšak nikdy jsem se nepustil do testování jeho bezpečnosti, a proto jsem touto formou chtěl získat nové poznatky a zkušenosti v této klíčové oblasti.

Dále jsem se rozhodl zaměřit na kyberbezpečnost, což je v souladu s mým budoucím magisterským studiem na UTB, kde máme možnost vybírat mezi kybernetickou bezpečností a mým stávajícím oborem, softwarovým inženýrstvím. Doufám, že tato práce mi pomůže lépe porozumět kybernetické bezpečnosti a pomůže mi rozhodnout se, který z oborů si vybrat pro své navazující studium.

Příležitost spolupracovat s firmou Continental, konkrétně s Ing. Jaromírem Šánkem, mě nadchla možností nahlédnout do reálné praxe. Těším se na možnost aplikovat své znalosti a dovednosti v pracovním prostředí a získat cenné zkušenosti, které mi pomohou v budoucí kariéře.

Důležitost této práce spočívá v rostoucím nebezpečí, které výrobní oblasti čelí vůči kybernetickým útokům. Kybernetické zločiny jsou stále častější a výrobní sektor je jedním z nejvíce ohrožených. Skenování zranitelností představuje účinný nástroj pro identifikaci bezpečnostních rizik. Je znepokojující, že v této oblasti není mnoho dostupných odborných zdrojů, což podtrhuje naléhavost a důležitost této práce.

Očekávám, že v průběhu práce budu schopen identifikovat a analyzovat zranitelnosti výrobního informačního systému. Pokud se nějaké naleznou, bude to ukázka potenciálních slabých míst, která je nutné zabezpečit. Pokud však neobjevím žádné zranitelnosti, bude to důkazem vynikajícího zabezpečení tohoto systému a skvělé práce týmu, což přinese firmě Continental větší klid a jistotu v provozu svých výrobních procesů.

I. TEORETICKÁ ČÁST

1 POZADÍ BAKALAŘSKÉ PRÁCE

V úvodní kapitole se zaměřím na vysvětlení základních pojmů, které souvisí s výrobními informačními systémy, základy kybernetické bezpečnosti a celkovým rozborem skenerů zranitelností. Vyzdvihnu důležitost kybernetické bezpečnosti v současné době jak v České republice, tak také ve světě.

V rozboru objasním pojmy jako skener zranitelnosti, skenování zranitelností, bezpečnostní rizika, konkrétně ze seznamu OWASP Top 10 2021, taktéž i jaké metody používají skenery k detekci zranitelností v systému.

1.1 Výrobní informační systém

Výrobní informační systémy jsou technologické platformy a aplikace pro podporu, dokumentaci a sledování přeměny surovin na hotové výrobky. Tyto systémy zahrnují širokou škálu funkcionalit, od detailního plánování výroby, správy výrobních zdrojů, řízení výroby, řízení kvality, sběru dat a sledování výkonu. Výrobní informační systém pracuje v reálném čase a pokrývá celou škálu výrobních prvků v procesu, včetně strojů, vstupů, personálu a dalších. Cílem těchto systémů je poskytovat aktuální informace pro rozhodování, optimalizaci výrobních procesů, zvýšit efektivitu, snížit náklady a maximalizovat produktivitu. [1][2]

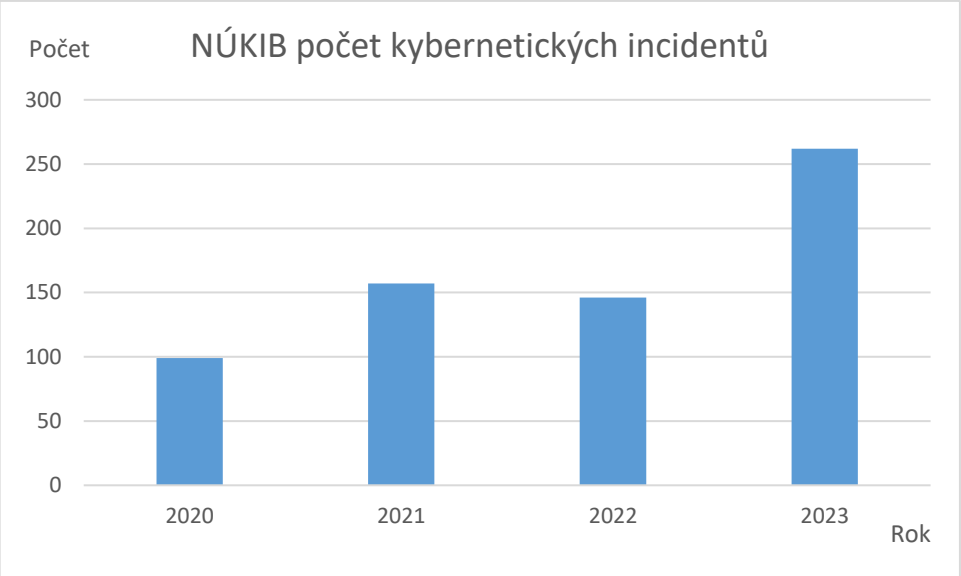
1.2 Kybernetická bezpečnost

„*Pojem „Kybernetická bezpečnost“ vycházející z anglického „Cyber Security“ označuje soubor nástrojů, technologií, technik, činností, zásad a procesů, jejichž cílem je prevence, ochrana a související správa zabezpečení počítačových systémů a technologií před kybernetickými útoky.*“ [3]

V této době je kybernetická bezpečnost klíčovým oborem, bez kterého bychom se neobešli. Ve světě s neustále rostoucím digitálním pokrokem, kde technologie proniká do všech aspektů našich životů, se kybernetická bezpečnost stává zásadním pilířem ochrany osobních a finančních informací. Kvůli novým technologií jako strojové učení a umělá inteligenci se objevují pro toto odvětví nové a více sofistikované bezpečnostní hrozby a zranitelnosti. V souvislosti s kybernetickou bezpečností je důležité si uvědomit také právní a regulační rámce, které hrají klíčovou roli v ochraně dat a prevenci kybernetických útoků.

Jedním z významných kroků v tomto směru je obecné nařízení o ochraně osobních údajů (GDPR) v Evropské unii. [4]

V roce 2023 evidoval Národní úřad pro kybernetickou a informační bezpečnost (NÚKIB) v České republice až dvojnásobek kybernetických incidentů ve srovnání s předchozím rokem 2022. [5]



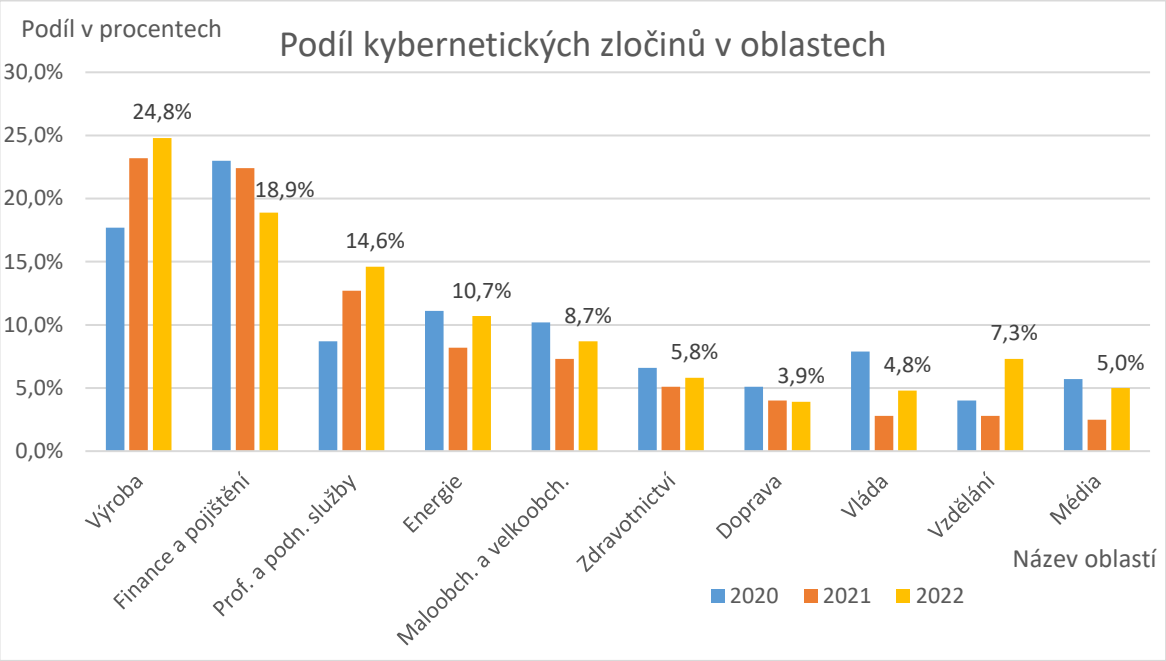
Obrázek 1. Graf počet kybernetických útoků v České republice [5][6]

1.3 Kyberbezpečnost ve výrobních systémech

V moderní době, kdy se většina procesů vyvíjí pomocí rostoucí zavádění robotiky a automatizace vytváří mnoho příležitostí pro hackery v odvětví, které v minulosti kybernetickou bezpečnost neupřednostňovalo. Výrobní systémy jsou již nyní hlavním cílem kyberzločinců. V roce 2022 bylo nejvíce kybernetických útoků zaznamenáno ve výrobním sektoru 25 %, následovaném odvětvím financí a pojištění 19 %. Výrobní systémy jsou klíčové pro integraci co nejkvalitnějších a nejefektivnějších výrobních procesů. [7]

Zabezpečení systémů výroby je kritické nejen z hlediska ochrany citlivých dat a informací, ale také pro udržení bezpečnosti a stability výrobního procesu jako takového. Kybernetické útoky mohou mít vážné důsledky pro fungování výrobních zařízení, od výpadků výroby až po manipulaci s procesy, což může vést k finančním ztrátám a ohrožení lidských životů v případě průmyslových havárií.

Z grafu je zřejmé, že kybernetické zločiny v oblasti výroby každoročně rostou a stávají se hlavním cílem útoků. Proto by měla být kyberbezpečnost jedním z klíčových prvků na které by se organizace měly zaměřit. [8][9]



Obrázek 2. Graf podíl kybernetických zločinů podle oblastí [8][9]

1.4 Zranitelnost

Zranitelnost v kontextu informačních technologií a bezpečnosti informací označuje slabé místo nebo nedostatek v systému, softwaru, síti nebo procesu, které může být zneužito k úniku, poškození nebo neoprávněnému přístupu k informacím či samotného systému. Tyto zranitelnosti mohou být způsobeny chybami v návrhu, nedostatečnými bezpečnostními opatřeními, zastaralým softwarem nebo lidským faktorem. [10]

1.5 Sken zranitelnosti

Skener zranitelnosti je nástroj, který dokáže analyzovat počítačové systémy, sítě nebo aplikace za účelem identifikace potencionálních bezpečnostních hrozeb. Skener zranitelností prohledává systémy na základě známých typů chyb, slabých míst nebo chybějících aktualizací, které by mohly být využity k proniknutí do systému. Když skener zranitelností identifikuje bezpečnostní hrozbu, dokáže vygenerovat podrobný report, který umožňuje správcům systémů nebo sítě opravit nebo odstranit nalezenou zranitelnost a tím vylepšit celkovou bezpečnost systému. Skener zranitelnosti je důležitým nástrojem v rámci kybernetické

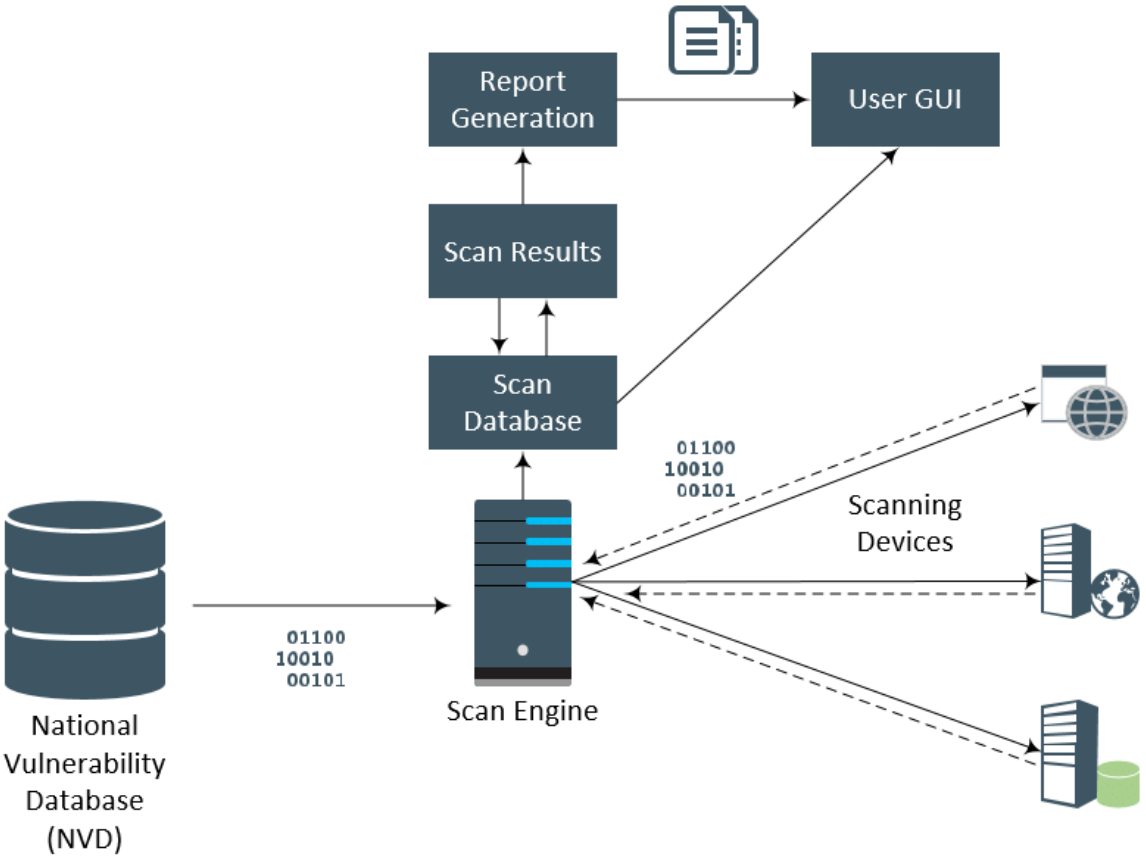
bezpečnosti a pomáhá organizacím identifikovat potenciální rizika spojená se zabezpečením jejich IT infrastruktury. [11][12]

1.5.1 Architektura skeneru zranitelnosti

Architektura skeneru zranitelnosti se skládá z mnoha komponent, které mezi sebou spolupracuji k identifikaci zranitelností.

Centrálním prvkem je jádro systému, které provádí skenování zranitelností. Tento komponent komunikuje s databází, která obsahuje známé zranitelnosti. Data z databáze jsou používána skenovacím enginem k identifikaci možných zranitelností ve skenovaných zařízení. Skenovací zařízení jsou napojena na skenovací engine přes síť.

Po dokončení skenování jsou výsledky odeslány do skenovací databáze, která slouží k ukládání výsledků skenování. Následně generovaní reportu je komponenta, jenž zpracovává data ze skenovací databáze k vytvoření reportů, které jsou prezentovány uživateli pomocí uživatelského GUI (Graphical User Interface), tedy grafického uživatelského rozhraní. Uživatelské rozhraní slouží jako prostředek pro interakci uživatele se systémem, umožňuje mu zobrazit reporty, ale i zadávat parametry pro skenování. [13]



Obrázek 3. Základní architektura skeneru zranitelnosti [14]

1.6 Skenování zranitelností

Skenování zranitelností je proces identifikace a prioritizace bezpečnostních zranitelností a úniků v počítačových systémech a aplikacích. Jedná se o významný prvek v kybernetické bezpečnosti, který pomáhá organizacím předejít útokům, únikům citlivých dat a jejich krádeži, tím že pomáhá identifikovat slabá místa v systému. Sken zranitelností je automatizovaný test, který pomocí prohledávacích metod jako crawler, nachází a reportuje bezpečnostní rizika. Existují skenery zranitelností, které dokážou v jednom testu prověřit přes 50 000 unikátních zranitelností, jak vnějších, tak i vnitřních. [15]

1.6.1 Důležitost skenování zranitelností

Ve výrobních informačních systémech se často provádí operace, kde může dojít k významným finančním a reputačním ztrátám v důsledku kybernetických útoků, a proto by mělo skenování zranitelností být standardem. Napomáhá systému vůči kybernetickým hrozbám tím, že aktivně identifikuje potencionální bezpečnostní slabiny v systému předtím, než by mohly být zneužity. Kvůli digitalizaci a propojení systému se zvyšuje riziko kybernetických útoků, a proto skenování zranitelností by měl být jeden z hlavních aktérů v kybernetické bezpečnosti ve výrobních informačních systémech.

1.7 Bezpečnostní rizika OWASP Top 10 2021

OWASP Top 10 je seznam deseti nejčastějších bezpečnostních rizik pro webové aplikace, který sestavuje organizace OWASP (Open Web Application Security Project). Jedná se o neziskovou organizaci, která se zaměřuje na kybernetickou bezpečnost v softwaru. Seznam se aktualizuje každé 3–4 roky na základě dat z výzkumů, průmyslu a odborníků na kybernetickou bezpečnost. OWASP Top 10 je široce uznávaný standard pro identifikaci a řešení bezpečnostních rizik ve webových aplikacích. [16]

1.7.1 Broken Access Control

Chyba zabezpečení řízení přístupu (Broken Access Control) je bezpečnostní slabina aplikace, která umožňuje uživatelům přistupovat k datům a funkcionalitám, ke kterým by neměli mít přístup. Útočník se může dostat k prostředkům, které by měly být chráněny anebo by měly být dostupné pouze pro určitou skupinu, která by měla mít oprávnění. Existuje mnoho druhů útoků na tuto bezpečnostní slabinu například: URL manipulace, zneužití endpointů a zvýšení uživatelských oprávnění. Toto bezpečnostní riziko se z OWASP Top

10 2017, kde bylo na 5. místě, posunulo na nejčastější bezpečnostní slabinu ve světě a to na 1. místo o 4 roky později v OWASP Top 10 2021. [17]

1.7.2 Kryptografická selhání

Kryptografické selhaní (Cryptographic Failures) nastává při odesílání dat, které nejsou zašifrovány anebo mají špatně nastavenou konfiguraci. Přenos citlivých dat je v této době kritický a je důležité ho zabezpečit tak, aby útočník žádná data neodchytil nebo nepřečetl. Například: hesla, čísla kreditních karet a osobní údaje. Zejména pokud data spadají pod zákony na ochranu soukromí, jako třeba obecné nařízení EU o ochraně osobních údajů (GDPR). Nejčastější důvody, proč k selhání dochází jsou slabé šifrovací algoritmy, nedostatečné klíčové managementy, chyby v konfiguraci TLS/SSL a nezabezpečené ukládání hesel. [18]

1.7.3 Injection

Injektování je typ útoku, při kterém útočník vkládá neboli “injektuje” škodlivý kód do aplikace. Tento kód dokáže narušit anebo změnit chod dané aplikace. Pomocí škodlivých dotazů může útočník získat soukromá data z databáze, ke kterým by neměl mít přístup, může s nimi manipulovat i vymazávat. V tomto útoků se zneužívá nedostatečná validace dat ze vstupů anebo výstupů. Nejznámější metody tohoto útoku jsou SQL injection, Cross-Site Scripting a LDAP injection. [19]

1.7.3.1 SQL injection

Útok SQL injection spočívá ve vložení neboli "injektování" dotazu SQL prostřednictvím vstupních dat z klienta do aplikace. Úspěšný útok SQL injection může číst citlivá data z databáze (Select), modifikovat data databáze (Insert/Update/Delete) a provádět administrátorské operace nad databází. Útoky s příkazy SQL mají za cíl ovlivnit provádění předem definovaných příkazů SQL. [20][21]

Příklad SQL injection: Máme firemní databázi, kde se nachází tabulka zaměstnanců.

Základní příkaz na zobrazení zaměstnance se jménem Jan je:

SELECT * FROM zamestnanci WHERE jmeno = 'Jan'.

Pokud tento příkaz modifikujeme boolean podmínku přesněji 1=1:

SELECT * FROM zamestnanci WHERE jmeno = 'Jan' OR 1=1,

tato podmínka vrací všechny řádky z tabulky zamestnanci, protože výraz 1=1 je vždy pravdivý, což umožňuje získat přístup k všem záznamům v tabulce, aniž by útočník potřeboval znát platné jméno zaměstnance nebo mít dostatečná oprávnění, aby se k těmto datům dostal. V této tabulce se mohou nacházet citlivá data jako osobní údaje či bankovní účty.

1.7.3.2 Cross-Site Scripting

Útok XSS (Cross-Site Scripting) představuje injekční útok, při kterém útočník vkládá škodlivé skripty do webových stránek, které jsou považovány za bezpečné a důvěryhodné. Tyto chyby jsou bohužel běžné a vznikají v situacích, kdy webové aplikace zpracovávají uživatelský vstup a bez dostatečného ověření či zpracování jej vkládají do výstupu, který je zobrazen uživatelům.

XSS útoky umožňují útočníkům posílat škodlivé skripty nic netušícím uživatelům. Tyto skripty jsou poté ve webovém prohlížeči obětí spuštěny bez jakékoliv varovné signalizace, jelikož prohlížeč nemá prostředky k rozpoznání. V důsledku toho, že prohlížeč předpokládá, že skript pochází z důvěryhodného zdroje, může škodlivý kód neoprávněně získat přístup k citlivým údajům uživatele, jako jsou cookies, tokeny relací a další informace uložené v prohlížeči. [22]

Příklad XSS: Předpokládejme webovou stránku, která má formulář pro komentáře, který zobrazuje uživatelské vstupy bez jakékoliv ochrany či kontroly, může uživatel do pole vložit libovolný text.

Například, útočník by mohl vložit do pole následující kód:

```
<script>
  fetch('http://hack.com/?cookie=' + encodeURIComponent(document.cookie));
</script>
```

Když běžný uživatel zobrazí stránku s tímto textem, jeho prohlížeč spustí JavaScriptový kód a odešle obsah cookies na server útočníka bez vědomí uživatele. Tímto způsobem může útočník získat citlivé informace.

1.7.4 Nezabezpečený design

Nezabezpečený design spočívá v nedostatečném důrazu na bezpečnost při návrhu designu anebo v průběhu vývojové fáze dané služby. Špatná rozhodnutí během počáteční fáze projektu mohou mít za následek trvalé, potenciálně i vážné důsledky, vedoucí k funkčním selháním, kompromisům a dalším. Bezpečný návrh může mít stále chyby implementace vedoucí ke zranitelnostem. Jako příklad může nastat při ověření identity, kdy daný uživatel má odpovědět na osobní otázku například: „Jaká je vaše oblíbená barva?“. Tento design je považován za nebezpečný, protože více než jedna osoba může na tuto otázku znát odpověď. [23]

1.7.5 Nezabezpečená konfigurace

S nezabezpečenou konfigurací se můžeme setkat u všech možných druhů serverů aplikačních, webových, databázových a dalších. K této bezpečnostní hrozbě může dojít při zanechání výchozích hodnot nebo nedostatečné aktualizaci softwaru. Většinou k těmto chybám dochází při lidském pochybení. To je důležité v případě organizací, které spoléhají na pronájem serverů od poskytovatelů cloudových služeb. V takových případech může být složité nebo náročné provést nezbytné úpravy v konfiguraci serveru, protože organizace nemusí mít přímý přístup k hardwaru nebo ke správě konfigurace. Tato chyba nemusí na první pohled vypadat zásadně, avšak ponecháním výchozích hodnot může útočník využít vzniklých zranitelností a dosáhnout úplného vyřazení serverů z provozu. [24]

1.7.6 Zranitelné a zastaralé komponenty

Toto bezpečnostní riziko je často referované na knihovny z frameworků z třetí ruky, které povětšinou případů bývají open-source. Ke zranitelnosti nedochází v okamžiku, kdy knihovnu začneme používat, ale až poté, co knihovna není dále vyvíjená svými autory. Z toho vyplývá, že jakmile přestane být podporována, nedochází k její aktualizaci vůči novým zranitelnostem a měla by být nahrazena. [25]

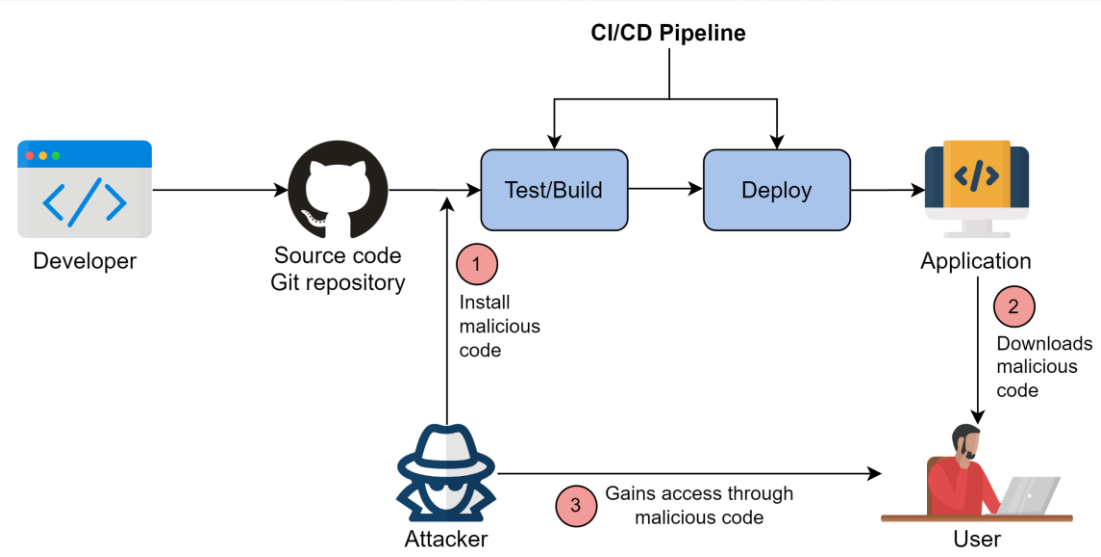
1.7.7 Selhání Identifikace a Autentizace

K selhání identifikace nebo autentizace dochází již při registraci nebo přihlášení do systému. Kvůli nedostatečné logice zmíněných operací ve vývoji systému, umožňuje útočníkům její zneužití. Systém by neměl umožňovat používání slabých nebo častých hesel jako "admin", neomezený počet pokusů o přihlášení (využití Brute Force útoku) nebo nezavedení

více faktorového ověření. K tomuto selhání nedávno došlo ve vládní sféře, kdy německé ministerstvo obrany zabezpečilo svůj komentář na webu heslem: 1234. [26][27]

1.7.8 Selhání integrity softwaru a dat

Selhaní integrity softwaru je stav, kdy dochází k narušení nebo poškození v systému. Integrita je známkou toho, jestli jsou data anebo software chráněny a nenarušeny. Selhání integrity může mít za následky narušení důvěryhodnosti a spolehlivosti systému. Jako příklad: útočník zneužívá slabiny CI/CD systému, který automatizuje vývoj aplikací a distribuci. Útočník nainstaluje do systému škodlivý kód, který je poté nevědomky nainstalován do prostředí zákazníka. Tímto způsobem získá útočník přístup k síti uživatele a může provádět škodlivé činnosti pod uživatelovou identitou. Tento proces můžeme vidět na následujícím obrázku. [28]



Obrázek 4. Ukázka procesu software and data integrity failure [28]

1.7.8.1 CI/CD systém

Kontinuální integrace (CI) a kontinuální doručování (CD) jsou koncepty v moderním softwarovém vývoji. CI je proces, kde vývojáři slučují svůj kód do sdíleného depozitáře a provádějí automatické sestavení a testování kódu. Cílem CI je identifikovat problémy v kódu co nejdříve a umožnit rychlé opravy. Dále, CD rozšiřuje CI tím, že automatizuje proces doručování softwaru do testovacího a produkčního prostředí. Po úspěšném sloučení kódu jsou změny automaticky nasazeny do testovacího prostředí, kde mohou být prověřeny. Pokud jsou testy úspěšné, software může být automaticky nasazen do produkčního prostředí. Proces je vede k rychlejšímu vývoji a zlepšení kvality softwaru. [29]

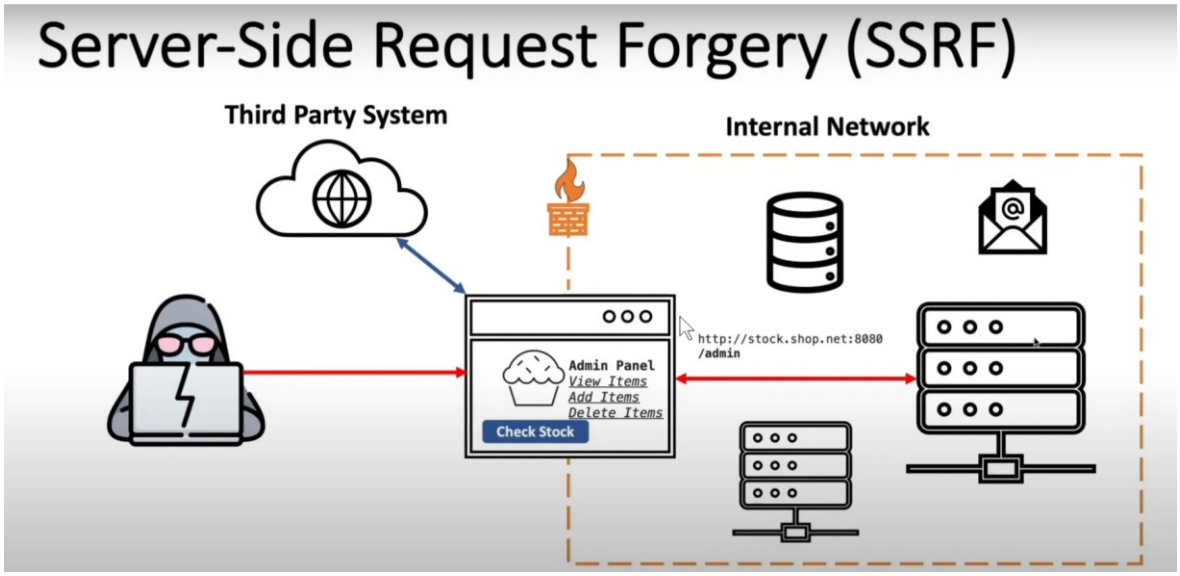
1.7.9 Nedostatečné bezpečnostní logování a monitorování

Nedostatečné bezpečnostní logování a monitorování představuje zranitelnost v systému nebo aplikaci při nedostatečném nebo žádném zaznamenávání nebo monitorování událostí či aktivity. Pro útočníka je tato zranitelnost ideální prostředí, jelikož je velice náročné zjistit, zda se útočník naboural do systému nebo jestli odcizil citlivá data.

Správné logování a monitorování aktivity umožnuje identifikovat, ale i reagovat na neoprávněný přístup. Data získaná z těchto operací mohou posloužit pro identifikaci bezpečnostních slabých míst, detekci neobvyklých vzorů anebo patentů v chování a k dalším účelům. [30]

1.7.10 Server-Side Request Forgery

Server-Side Request Forgery nebo česky podvržení požadavku mezi různými stránkami je bezpečnostní riziko, při kterém útočník využívá zranitelnost aplikace na serveru. Útočník manipuluje s požadavky, pomocí kterých může oklamat server, aby zadával požadavky na jiné systémy a tím může útočník získat přístup k citlivým informacím a manipulovat s interními službami serveru. [31]



Obrázek 5. Server-Side Request Forgery [31]

1.8 Metody pro skenování

Metody pro skenování zranitelností jsou různorodé techniky a procesy používané k identifikaci a prioritizaci zranitelností v systémech, sítích a aplikacích. Tyto metody jsou

zásadní pro odhalování slabých míst, která mohou být zneužita útočníky. Metody umožnují organizacím reagovat a posílit své obranné strategie.

1.8.1 Statické testování bezpečnosti aplikací

Statické testování bezpečnosti aplikací (SAST), analyzuje zdrojový kód s cílem identifikovat zranitelnosti, které by mohly způsobit, že by aplikace byly otevřené pro škodlivé útoky. SAST používá techniky skenování zranitelností, které se zaměřují na zdrojový kód a bajtový kód a odhalují bezpečnostní problémy, jako jsou útoky typu injektováním nebo problémy se správou paměti. Metoda se řadí do white-box testing, protože vyžaduje přístup k interní struktuře aplikace. Použitím nástrojů SAST jsou aplikace lépe chráněny před potenciálními bezpečnostními hrozbami. [21][32]

1.8.2 Dynamické testování bezpečnosti aplikací

Dynamické testování bezpečnosti aplikací (DAST) se zaměřuje na inspekci aplikací z pohledu externího útočníka v reálném čase, zatímco aplikace je spuštěná. DAST testuje aplikaci dynamicky z vnějšku bez nutnosti přístupu k jejímu internímu kódu, proto je označován jako black-box testing. DAST nástroje automaticky procházejí aplikaci, simulují akce koncových uživatelů a monitorují aplikaci na bezpečnostní zranitelnosti, jako jsou problémy s autentizací, konfigurační chyby, zranitelné skripty na straně klienta (XSS), nebezpečné využití externích zdrojů (SQL injection) a další běžné bezpečnostní hrozby, které mohou být zneužity bez přímého přístupu ke zdrojovému kódu. [33]

1.8.3 Aktivní skenování

Aktivní skenování je interaktivní metoda, při které skenovací nástroj aktivně komunikuje se systémem či sítí, aby identifikoval potenciální zranitelnosti. Skenovací nástroj může provádět různé operace, jako jsou pokusy o překonání autentizačních mechanismů, vytváření žádostí o neplatné vstupy nebo zasílání specifických síťových paketů pro testování reakcí systému. Aktivní skenování může být důležité pro odhalení skrytých zranitelností, které nejsou snadno identifikovatelné pouhým sledováním provozu. Často se stává, že aktivní skenování má hojný počet falešně pozitivních výsledků. [34]

1.8.4 Pasivní skenovaní

Pasivní skenování je méně invazivní a nepřímá metoda, která sleduje provoz v síti nebo aplikaci, aniž by s nimi aktivně reagovala. Tento typ skenování sleduje provoz a analyzuje

metadata, aby identifikoval potenciální zranitelnosti a hrozby. Pasivní skenování pouze pozoruje a zaznamenává provoz, který již probíhá. To může zahrnovat sledování síťových paketů, monitorování síťového provozu nebo analýzu logů událostí. Pasivní skenování je často používáno pro monitorování síťového provozu a detekci anomálií, což umožňuje organizacím identifikovat neobvyklé nebo podezřelé aktivity bez přímého zásahu. Pasivní skenování je velice spolehlivá metoda, jelikož zde je minimální počet falešně pozitivních výsledků. [34]

1.8.5 Skenování portů

Skenování portů je základní metoda zjišťování pro síťové administrátory i útočníky, protože odhaluje porty, které jsou v síti otevřené. Tyto porty by potencionálně mohly přijímat nebo odesílat citlivá data. Je to také proces odesílání paketů na konkrétní porty hostitele a analyzování odpovědí k identifikaci zranitelností. Metoda může odhalit nejen otevřené porty, ale také zranitelné služby, proto tato metoda dokáže poskytnout důležité informace pro plánování obranných strategií, jako posílení firewallů, aktualizaci zastaralých služeb a opravu konfigurací, které mohou být zneužity. [35]

1.8.5.1 TCP port

TCP (Transmission Control Protocol) je spojově orientovaný protokol, který vytváří spojení mezi odesílatelem a příjemcem před přenosem dat. Poskytuje spolehlivou a řízenou komunikaci, zahrnující doručení dat v pořadí, kontrolu chyb a znovu odesílání ztracených dat. Je vhodný pro aplikace, které vyžadují spolehlivý přenos dat. [36]

1.8.5.2 UDP port

UDP (User Datagram Protocol) je bez spojový protokol, který nevyžaduje vytvoření spojení před přenosem dat. Poskytuje jednoduchý mechanismus pro přenos dat bez zajištění spolehlivosti doručení nebo kontroly chyb. Je vhodný pro aplikace, které nepotřebují spolehlivost přenosu a preferují nižší latenci a menší režii síťového provozu. [36]

1.8.6 Web crawler

Web crawler, také nazývaný jako web spider, je skript, který systematicky prozkoumává World Wide Web (WWW). Jeho hlavní účel je indexování obsahu webových stránek. Toto indexování se používá ve vyhledávačích jako Google a dalších, umožnuje jim poskytovat relevantní výsledky vyhledávání. Při skenování zranitelností je web crawler klíčový pro

sběr informací, konkrétně pro identifikaci potencionálních slabých míst. Napomáhá k detekci například SQL injection, Cross-Site Scripting, expirovaných certifikátů a mnoha dalších.

Web crawler začíná pomocí samostatné url adresy anebo jejich seznamem. Navštíví webovou stránku URL adresy a stáhne její obsah, většinou v HTML kódu. Následně se ze staženého obsahu extrahují odkazy, které putují do seznamu či fronty, které má crawler navštívit. Tento proces se poté rekurzivně opakuje. [37]

1.8.7 Crawljax

Crawljax je metoda pro systematické procházení Ajax based webových aplikací či stránek. Je navržený tak, aby dokázal interagovat s dynamickými prvky webových stránek, které jsou z JavaScriptu a dalších moderních webových technologií. Jeho hlavním účelem je testování funkcionalit, použitelnosti a bezpečnosti. Při skenování zranitelností dokáže detekovat Cross-Site Scripting, SQL injection nebo neautorizovaný přístup. [38]

1.8.7.1 Ajax

AJAX, což je zkratka pro Asynchronous JavaScript and XML, je soubor webových technologií používaných k vytváření interaktivních webových aplikací. Umožňuje webovým stránkám asynchronně komunikovat s webovým serverem, což znamená, že může načítat a aktualizovat data, aniž by se znovu načítala celá stránka. To zlepšuje uživatelskou zkušenost tím, že umožňuje rychlejší a plynulejší interakce. [39]

1.8.8 Skenovaní s přihlašovacími údaji a bez nich

Autentizované skeny zranitelností jsou prováděny s přihlašovacími údaji, což skenovacím nástrojům umožňuje získat hlubší přístup do systémů a aplikací. Tento typ skenování poskytuje komplexnější a přesnější analýzu bezpečnostního stavu, protože může identifikovat zranitelnosti, které nejsou zjevné z vnějšku systému. Přístupové údaje umožňují skenovacím nástrojům odhalovat zranitelnosti spojené s nesprávným nastavením, zastaralým softwarem a dalšími interními problémy.

Skeny bez přihlašovacích údajů zranitelností jsou prováděny bez přihlašovacích údajů, to znamená, že skenovací nástroje mají omezený přístup a zkoumají systémy a aplikace z vnějšku. Tyto skeny se zaměřují na identifikaci zranitelností, které jsou viditelné bez inter-

ního přístupu. Neautentizované skeny jsou užitečné pro zjištění zjevných rizik a povrchové úrovně zabezpečení. [40]

1.8.9 Skenování zranitelností závislostí

Skenování závislostí je proces identifikace a analýzy knihoven, balíčků nebo modulů, na kterých software závisí, s cílem zjistit potenciální zranitelnosti, zastaralé verze nebo nesoulady v licencích. Tento proces umožňuje proaktivně řešit bezpečnostní rizika a udržovat softwarové projekty aktuální a v souladu s nejlepšími bezpečnostními praktikami. Skenovací nástroje automaticky porovnávají použité závislosti s databázemi známých zranitelností a poskytují doporučení pro jejich aktualizaci nebo náhradu. [41]

2 NÁSTROJE PRO SKENOVÁNÍ ZRANITELNOSTÍ

V této kapitole představuji přehled komerčních a open-source nástrojů na bezpečnostní testování webových aplikací. Nejprve představím komerční nástroje pro webové aplikace jako jsou Invicti, Nessus a Burp Suite. Poté popíšu open-source nástroje. Mezi tyto nástroje patří ZAP, Nmap a OpenVAS.

Momentálně není k dispozici žádný skener zranitelností specifický pro desktopové aplikace, a proto se zaměřím na několik alternativních metod, s konkrétními nástroji jako SonarQube, Wireshark a Metasploit.

2.1 Komerční nástroje pro webové aplikace

Komerční software neumožnuje uživatelům nahlédnout do kódu a upravit si ho podle svých potřeb. Tento software je spravován pouze autory anebo jeho vlastníkem, ať už se jedná o organizace nebo jednotlivce. Software je určen pro prodej nebo je poskytnut na určitou dobu přes různé druhy licencí. [42]

2.1.1 Invicti

Invicti, dříve známý jako Netsparker, je nástroj pro skenování zranitelností, který vyniká díky své schopnosti přesného detekování a ověření bez falešných pozitiv. Tento nástroj exceluje v pokročilém skenování různých typů zranitelností, včetně SQL injection, Cross-Site Scripting a dalších, v souladu s OWASP 10 standardy.

Jedinečností Invicti je jeho schopnost ověřovat a následně eliminovat falešně pozitivní výsledky. To je klíčový faktor, neboť falešné pozitivní výsledky mohou vést ke ztrátě drahocenného času a zdrojů při vyhodnocování bezpečnostních rizik.[43]

Zákazníci Invicti jsou s tímto produktem nadmíru spokojení. Na webu Gartner.com získal 4,4 bodů z 5 z 311 recenzí, na g2.com pak 4,6 bodů z 57 recenzí. Uživatelům je tak poskytována jistota kvality a spolehlivosti tohoto nástroje. [44][45]

Invicti nabízí uživatelům možnost vyzkoušet si produkt pomocí bezplatného trialu, který umožňuje skenování až pěti cílů po dobu dvou týdnů. Pro získání trial verze je třeba kontaktovat společnost prostřednictvím e-mailu, přičemž každá žádost je řešena individuálně. Osoba přidělená k vyřízení žádosti o trial verzi se snaží získat informace od budoucího uživatele ohledně použití produktu, na co chce produkt konkrétně použít, kolikrát chce

sken spustit a jaký je bezpečnostní cíl webové aplikace. Firma neposkytuje bezplatný trial pro výzkumné účely.

Invicti poskytuje placené licence ve třech úrovních: Standard, Team a Enterprise. Každá z těchto licencí má své vlastní specifické výhody a ceny, které se liší v závislosti na potřebách a velikosti organizace. [46]

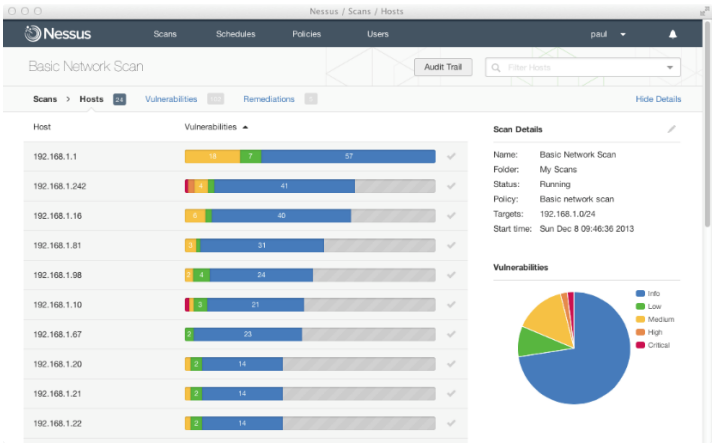
2.1.2 Nessus

Nessus je nástroj od firmy Tenable, který si získal uživatelskou přízeň díky své precizní a komplexní schopnosti skenování zranitelností. Využívá rozsáhlou knihovnu zranitelností a nabízí více než 205 879 pluginů pro identifikaci a validaci zranitelností. Nessus je multiplatformní nástroj, který umožňuje používání na operačních systémech Windows, Linux a macOS. [47]

Nessus je oblíbený mezi uživateli díky své snadné použitelnosti a konfigurovatelnosti. Jeho unikátnost spočívá v možnosti využití rozšířených šablon, které umožňují provádět různé druhy skenů zaměřených na webové aplikace, síťovou infrastrukturu, viry a další hrozby.

Hodnocení v recenzích tohoto produktu jsou nadprůměrné. Na webu Gartner.com získal 4,5 bodů z 5 z celkového počtu 572 recenzí, stejně tak na webu g2.com získal stejné hodnocení z 274 recenzí. [48][49]

Firma Tenable nabízí tři druhy licencí pro produkt Nessus: Tenable Nessus Essentials, Tenable Nessus Professional a Tenable Nessus Expert. Licence Essentials je bezplatná zkušební verze, která umožňuje skenování až 16 unikátních cílů. Licence Professional stojí organizace 124 208 Kč ročně a licence Expert, která nabízí rozšířené funkce jako skenování cloudové infrastruktury, externí útoky a další, vyjde na 186 468 Kč ročně. [50]



Obrázek 6. UI nástroje Nessus [51]

2.1.3 Burp Suite

Burp Suite je nástroj firmy PortSwigger a jedná se o komerční nástroj pro skenování zranitelností, který si získal pověst díky své rozsáhlé funkcionalitě a schopnosti detekovat širokou škálu bezpečnostních problémů. Burp Suite je navržen tak, aby odpovídal standardům OWASP 10 a detekci zranitelností jako SQL injection, Cross-Site Scripting a mnoha dalších.

Jednou z klíčových vlastností Burp Suite je jeho široké spektrum nástrojů, které umožňuje uživatelům provádět pokročilé analýzy a testování zranitelností. To zahrnuje nejen automatické skenování, ale také ruční testování, úpravy HTTP požadavků a odpovědí, a sledování komunikace mezi klientem a serverem. [52]

Burp Suite má skvělé hodnocení od svých uživatelů. Na platformě Gartner.com získal 4,6 bodů z 5 z 223 recenzí, zatímco na g2.com dosáhl hodnocení 4,8 bodů z 114 recenzí. Uživatelé oceňují především jeho všestrannost. [53][54]

Portswigger nabízí Burp Suite Community Edition, která je značně omezena svými funkcionalitami, ale umožnuje uživatelům také vyzkoušet si Burp Suit Professional pomocí bezplatné trial verze, která umožňuje testovat zranitelnosti po dobu 30 dnů.

Pro organizace, které potřebují rozsáhlejší funkcionality a podporu, firma nabízí placené licence ve třech úrovních: Community, Professional a Enterprise. Každá z těchto licencí má své vlastní specifické výhody a ceny.

Professional licence stojí 11225 Kč za uživatele. Za Enterprise licenci organizace zaplatí na rok od 90000 Kč až do 6 249 975 Kč, podle plánu, který jim více vyhovuje.

2.2 Open-source nástroje pro webové aplikace

Open-source software je typ softwaru, jehož zdrojový kód je veřejně přístupný. To znamená, že kdokoliv může tento kód prohlížet, modifikovat a distribuovat, za předpokladu, že dodržuje podmínky licence OSS (Open source software). [55]

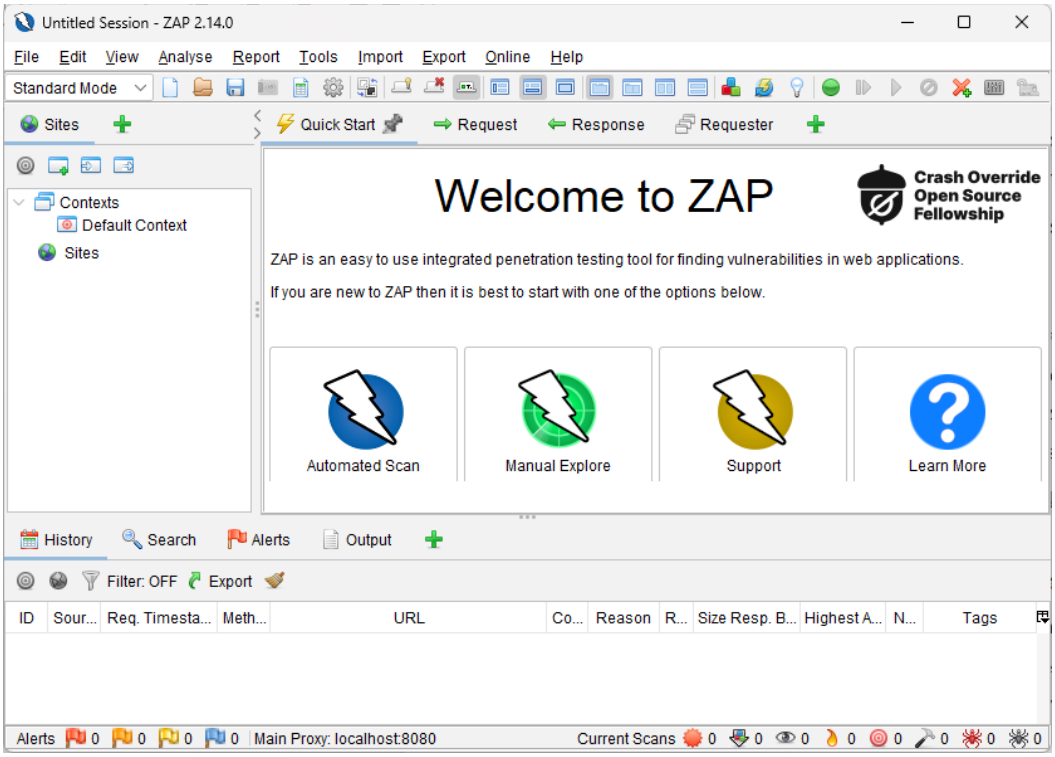
2.2.1 Zed Attack Proxy

ZAP (Zed Attack Proxy) je open source nástroj, vyvíjený a udržovaný organizací OWASP (Open Web Application Security Project), a implementovaný v programovacím jazyce Java. Díky své flexibilní konfigurovatelnosti a rozsáhlým možnostem skenování se stal ZAP široce využívaným v oblasti bezpečnostního testování webových aplikací.

Tento nástroj je známý svou schopností provádět jak aktivní, tak pasivní bezpečnostní testování webových aplikací. To zahrnuje odhalování různých typů zranitelností, jako jsou SQL injection, Cross-Site Scripting nebo broken authentication.

Díky své open source povaze má ZAP širokou komunitu uživatelů a vývojářů, což umožňuje uživatelům přizpůsobit nástroj podle specifických potřeb jejich projektů. Uživatelé mohou vytvářet vlastní rozšíření, add-ony a pluginy nebo přizpůsobit již existující funkcionalitu.

ZAP je k dispozici zdarma ke stažení a použití na různých operačních systémech, včetně Windows, Linuxu a macOS, což přispívá k jeho širokému rozšíření a popularitě po celém světě. [56][57]



Obrázek 7. UI nástroje Zed Attack Proxy

2.2.2 Nmap

Nmap (Network Mapper) je open source nástroj určený k síťovému průzkumu a analýze. Jeho hlavním účelem je skenovat a mapovat sítě, identifikovat připojená zařízení, sbírat informace o síťových službách a zjišťovat zranitelnosti v síťových zařízeních. Nmap je jedním z nejznámějších a nejčastěji používaných nástrojů pro síťové průzkumy a bezpeč-

nostní auditování. Nmap je implementován v jazyce C a je dostupný pro různé operační systémy, včetně Windows, Linuxu a macOS.

Tento nástroj nabízí širokou škálu funkcí, včetně skenování síťových zařízení, identifikaci služeb a operačních systémů, mapování topologie sítě a analýzu zranitelností. [58]

2.2.3 OpenVAS

OpenVAS (Open Vulnerability Assessment System) je open-source nástroj pro skenování zranitelností v síťových zařízeních a webových aplikacích.

Historie OpenVAS je úzce spojena s již řečeným projektem Nessus. Nessus byl původně open-source nástroj pro skenování zranitelností, ale byl převeden na komerční nástroj. Tato změna vedla k vytvoření odvozených verzí Nessusu, které zahrnovaly modifikace a úpravy původního kódu, aby zůstal open-source. V roce 2005 vznikla odvozená verze Nessusu nazvaná OpenVAS. Tento projekt byl vytvořen kvůli obavám z uzavřeného zdrojového kódu a změn v licencování původního Nessusu. OpenVAS si zachoval důležité vlastnosti původního nástroje, ale s cílem poskytnout uživatelům otevřenou a zdarma dostupnou alternativu pro skenování zranitelností. [59]

OpenVAS poskytuje jak aktivní, tak i pasivní skenování zranitelností. To zahrnuje odhalování širokého spektra potenciálních rizik, jako jsou zranitelnosti v operačním systému, slabá autentizace, SQL injection, Cross-Site Scripting a mnoho dalších. OpenVAS je flexibilní a umožňuje uživatelům vytvářet vlastní pluginy a rozšíření nebo upravovat stávající funkcionalitu podle potřeb svých projektů.

Nástroj je primárně navržen pro Linuxové operační systémy. Je možné ho však používat i na Windows, ale jedná se o složitější proces. [60]

2.3 Nástroje pro desktopové aplikace

V současné době není dostupný žádný skener zranitelností pro desktopové aplikace, nicméně existuje řada alternativních metod pro testování zranitelností. Mezi tyto metody patří např. statická analýza kódu, která se zaměřuje na identifikaci potenciálních chyb a zranitelností přímo ve zdrojovém kódu aplikace, aniž by byla spuštěna. Další metodou je dynamická analýza síťového provozu, která sleduje komunikaci generovanou aplikací a odhaluje potenciální zranitelnosti, jako jsou nezabezpečené komunikace nebo přenos citli-

vých dat. Dále můžeme využít penetrační testování, které pak aktivně zkoumá aplikaci s cílem najít chyby v implementaci, neoprávněný přístup nebo další zranitelnosti.

Firma Continental již využívá výše uvedené nástroje v rámci svých bezpečnostních opatření a bylo by pro mě zbytečné je aplikovat. Z tohoto důvodu se nástrojům v praktické části nevěnuji.

2.3.1 SonarQube

SonarQube je open-source a zároveň komerční nástroj podle dané licence, konkrétně komunitní licence je jako jediná open-source. Nástroj je navržen pro automatickou kontrolu kvality kódu, což je statická analýza kódu. Jejím hlavním účelem je identifikovat problémy v kódu a poskytnout zpětnou vazbu vývojářům, aby mohli vytvářet lepší a efektivnější software.

SonarQube provádí pečlivou statickou analýzu kódu a detekuje různé typy chyb, jako jsou nedostatky v kvalitě kódu, zranitelnosti, duplicity a další potenciálně problematické prvky. Tím pomáhá vývojářům odhalit a odstranit problémy ještě před tím, než se dostanou do produkčního prostředí.

Jednou z výhod SonarQube je jeho podpora různých programovacích jazyků. Platforma podporuje širokou škálu jazyků, včetně Java, C#, JavaScript, Python, Ruby a mnoho dalších.

Integrace s vývojovými nástroji je další významnou funkcí SonarQube. Tento nástroj lze snadno integrovat s různými vývojovými prostředími a nástroji pro kontinuální integraci a nasazení (CI/CD).

Další klíčová vlastnost je rozsáhla knihovna pravidel kontroly kvality. Tato knihovna obsahuje mnoho předdefinovaných pravidel, která umožňují konfigurovat a přizpůsobit kontroly kódu podle svých potřeb v konkrétním projektu. [61]

2.3.2 Wireshark

Wireshark je open-source nástroj pro analýzu síťového provozu. Jeho hlavním účelem je sledovat a analyzovat data, která cestují po síti, a poskytovat podrobné informace o komunikaci mezi různými zařízeními.

Jednou z hlavních funkcí Wiresharku je schopnost zachytávat pakety dat, které putují po síti. To znamená, že může odposlouchávat a zaznamenávat veškerý provoz, který prochází určitým síťovým rozhraním.

Wireshark umožňuje uživatelům podrobně prozkoumat každý zachycený paket. Poskytuje informace o zdroji a cíli komunikace, použitých síťových protokolech, velikosti datových paketů a dalších metadat. Díky tomu mohou uživatelé porozumět, jak síťová komunikace probíhá, a identifikovat případné problémy.

Další funkcí Wiresharku je možnost filtrování dat. Uživatelé mohou použít různé filtry a vyhledávací nástroje k omezení zobrazení zachycených paketů pouze na ty, které jsou relevantní pro jejich konkrétní potřeby nebo analytické účely.

Wireshark podporuje mnoho různých síťových protokolů, včetně běžných protokolů jako TCP/IP, UDP, HTTP a dalších. [62]

2.3.3 Metasploit

Metasploit je jedním z nejpopulárnějších nástrojů pro penetrační testování, což je proces provádění kontrol bezpečnosti v systémech nebo sítích.

Jednou z klíčových funkcí Metasploit Frameworku je průzkum, který umožňuje skenování síťových portů, sběr informací o systému a identifikaci služeb a jejich verzí. Tímto způsobem je možné nalézt potenciální zranitelnosti, na které lze následně útočit.

Důležitou funkcí je využití zranitelností, kdy Metasploit nabízí rozsáhlou databázi exploitů. Tyto exploitové moduly mohou být použity k automatickému využití nalezených zranitelností.

Po úspěšném využití zranitelnosti Metasploit poskytuje možnosti post-exploitačních aktivit. To umožňuje penetračním testerům provádět další testy a získávat další informace o cílovém systému. To umožňuje provádět hlubší analýzu a identifikovat další zranitelnosti.

Metasploit rovněž umožňuje generování a doručování různých typů payloadů, což jsou malé kusy kódu nebo skripty, které provádějí specifické akce na cílovém systému. Tímto způsobem lze například získat přístup k souborům. [63]

2.3.3.1 Penetrační testování

Penetrační testování je proces aktivního testování bezpečnosti systému, sítě nebo aplikace s cílem identifikovat a vyhodnotit možné zranitelnosti a slabiny, které by mohly být zneužity

neoprávněnými útočníky. Penetrační testování simuluje útoky, které by mohly být prováděny reálnými útočníky, s cílem poskytnout organizaci informace o jejich bezpečnostních rizicích a pomoci jim přijímat informovaná rozhodnutí ohledně zlepšení bezpečnosti. [64]

II. PRAKTICKÁ ČÁST

3 CONTINENTAL

V této části práce se zaměřím na klíčový prvek fungování společnosti Continental AG, a to její výrobní informační systém Continental Global Manufacturing System (CGMS). Nejprve uvede firmu Continental AG a poté její pobočku Continental Barum s.r.o. Představím strukturu výrobního informačního systému a jeho technologické základy, které umožňují účinné řízení výrobních procesů a sledování kvality výrobků. Dále analyzuji, jak je systém navržen a jaké programovací jazyky a služby využívá. Tento informační systém představuje páteřní prvek ve fungování společnosti Continental, zejména v automobilovém průmyslu, kde je kvalita a efektivita výroby zásadní pro úspěch a konkurenceschopnost firmy.

3.1 Continental AG

Continental AG, často jednoduše nazývaná Continental, je přední německá společnost specializující se na výrobu pneumatik, systémů pro automobilový průmysl, brzdových systémů a dalších komponentů pro dopravní a automobilový průmysl. Založena byla v roce 1871 v Hannoveru v Dolním Sasku, Německo, a od té doby se rozrostla do jedné z největších dodavatelských společností v automobilovém průmyslu na světě. [65]

3.1.1 Continental Barum s.r.o.

Continental Barum je česká pobočka společnosti Continental AG, která má hluboké kořeny v českém průmyslu pneumatik. Historie společnosti Barum sahá do roku 1945, kdy byla v Československu založena kombinací tří firem: Baťa, Rubena a Matador. Které se zaměřovali na výrobu gumových výrobků, a to včetně pneumatik. V roce 1993 se společnost Barum stala součástí skupiny Continental.

Závod Continental Barum v Otrokovicích patří k největším a technologicky nejvyspělejším výrobním zařízením pneumatik v České republice a je významným hráčem v evropském kontextu. Vyrábí širokou škálu pneumatik pod značkami Continental, Barum, Semperit, Matador a další, pokrývající osobní vozidla, nákladní vozidla a mnoho dalších. [66]

3.2 CGMS systém

Continental Global Manufacturing System je výrobní informační systém, který představuje páteřní technologii pro celou nadnárodní firmu Continental a který jí umožňuje efektivní správu a kontrolu výrobních operací v reálném čase, což je zásadní pro zajištění vysoké

kvality výrobků a optimalizaci výrobních procesů. CGMS je centralizované shromaždiště dat z různých částí výrobního procesu napříč celou firmou.

Jedním z hlavních přínosů CGMS je jeho schopnost integrace s různými dalšími systémy a technologiemi v rámci výrobního podniku. Systém CGMS je navržen tak, aby poskytoval plynulý tok informací mezi výrobními linkami a dalšími odděleními, jako jsou plánování, nákup, skladování a distribuce. Systém je klíčový pro optimalizaci zásob, snižování výrobních cyklů a zlepšení celkové efektivity operací.

CGMS umožňuje nepřetržité sledování a analýzu výrobních procesů a shromažďování dat v reálném čase. Díky tomu může organizace rychle identifikovat a řešit potenciální problémy, ještě než dojde k významným výpadkům nebo i ztrátám. Systém shromažďuje a analyzuje data z různých částí výrobního procesu, což umožňuje rychlou reakci na aktuální situace. Například systém dokáže monitorovat výkonnost strojů, spotřebu surovin nebo kvalitu výroby v reálném čase.

CGMS hraje významnou roli při získávání informací ohledně kvality a dodržování předpisů. Systém systematicky sleduje průběh výrobních procesů a kontroluje výrobky v souladu s předem stanovenými normami. Jakékoli odchylky od těchto norem jsou okamžitě hlášeny, což umožňuje rychlou reakci. To je zvláště důležité v automobilovém průmyslu, kde jsou požadavky na kvalitu a bezpečnost mimořádně vysoké.

Implementací CGMS dokáže Continental výrazně zlepšit produktivitu a efektivitu svých výrobních operací. Systém pomáhá snižovat výrobní ztráty, optimalizovat využití zdrojů a zkracovat dobu potřebnou k uvedení produktů na trh. Tato zlepšení mají dopad na konkurenceschopnost společnosti.

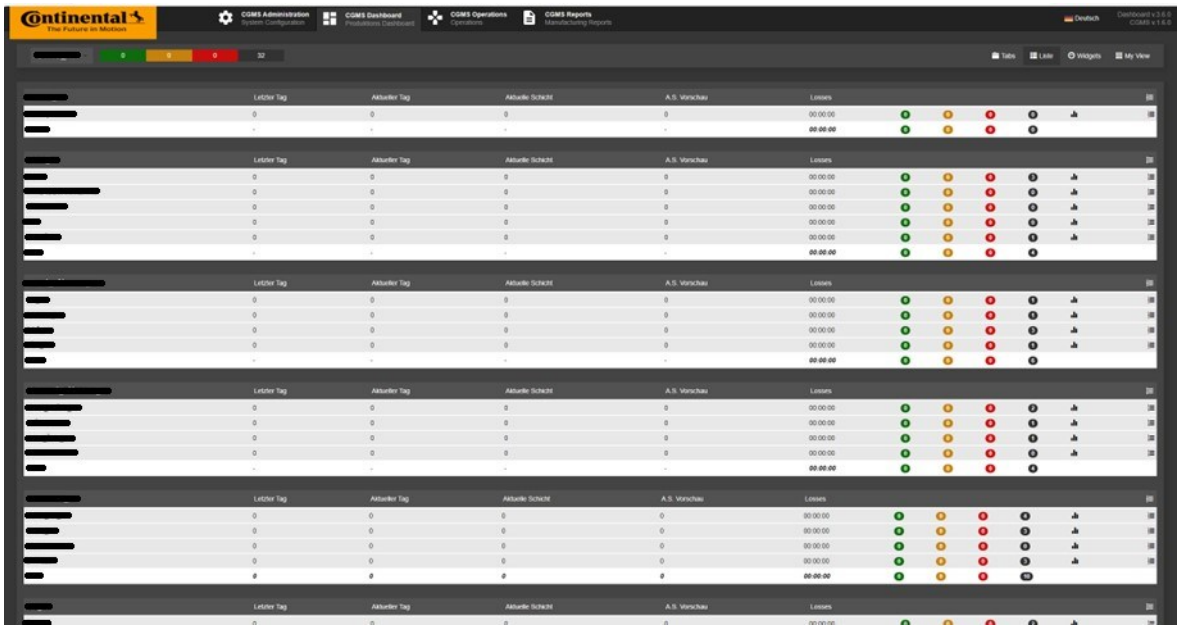
3.2.1 CGMS webová aplikace

Webová aplikace je založená na architektuře MVC a implementovaná pomocí technologií ASP.NET a .NET Framework. Architektura MVC (Model-View-Controller) poskytuje strukturovaný přístup k oddělení datové logiky, uživatelského rozhraní a řízení aplikační logiky, což usnadňuje údržbu a rozšíření aplikace. Frameworky ASP.NET a .NET nabízejí bohatou sadu nástrojů a knihoven, které zjednodušují vývoj, zvyšují produktivitu a zlepšují výkon aplikace.

Tato aplikace využívá serverů Internet Information Services (IIS), které jsou od firmy Microsoft a jako úložiště dat využívá relační databázi typu Microsoft SQL. Díky integraci

s IIS je možné aplikaci snadno nasadit a spravovat na webovém serveru. Využití databáze Microsoft SQL zajišťuje spolehlivé a výkonné ukládání dat.

Tato kombinace technologií vytváří robustní a spolehlivou platformu pro webovou aplikaci, která splňuje moderní standardy v oblasti vývoje webových aplikací.



Obrázek 8. UI webové aplikace CGMS

3.2.2 Architektura CGMS

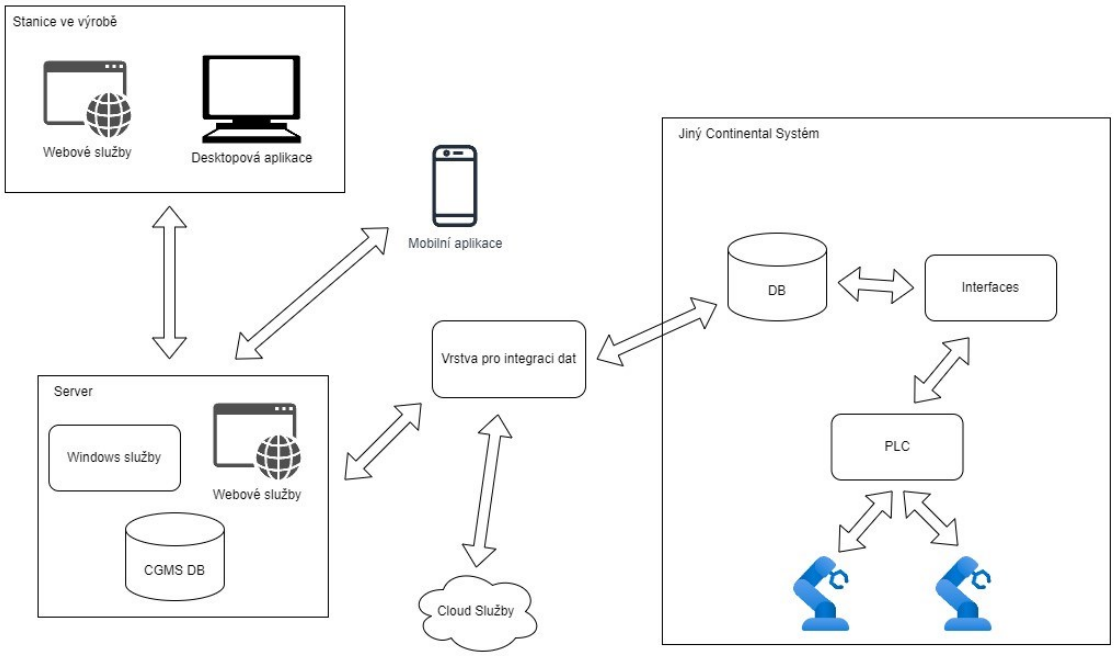
Architektura výrobního informačního systému CGMS se skládá z několika prvků, které efektivně integrují a spravují data z výrobního prostředí

Klientská část systému představuje rozhraní pro uživatele přímo v prostředí výroby. Zahrnuje webové a desktopové aplikace nasazené na pracovních stanicích v továrně a mobilní aplikace, které poskytují mobilní funkcionalitu pro pracovníky a techniky. Tato část komunikuje se serverem, umožňující sběr dat z výroby a poskytování zpětné vazby do klientských aplikací.

Hlavní infrastruktura systému je umístěna na serveru, který hostuje operační systém Windows a webové služby. Zároveň obsahuje hlavní databázi systému, konkrétně relační Microsoft SQL databázi. Tato centrální databáze uchovává data týkající se výrobních operací a umožňuje zpracování a shromažďování informací z různých klientských aplikací.

Cloudové služby slouží jako centrální bod pro integraci, shromažďování a distribuci dat. Vrstva pro integraci dat zajišťuje propojení serveru, cloudových služeb a dalších systémů od firmy Continental.

Pro příklad jsem použil zjednodušený systém, který pracuje přímo s výrobními stroji. Tento systém je propojen se CGMS pomocí databáze, která se nachází na serveru, což umožňuje synchronizaci dat. V systému je PLC (Programmable Logic Controller) pro ovládání a monitorování strojů ve výrobních procesech. Tento systém může zahrnovat nástroje pro správu, řízení výroby a další aplikace spojené s výrobními procesy.



Obrázek 9. Diagram architektury CGMS

4 APLIKOVÁNÍ SKENERŮ DO CGMS

V této kapitole se zaměřím na dva klíčové nástroje, ZAP a Nessus, které budou použity k provedení bezpečnostní analýzy výrobního informačního systému. Nejdříve vysvětlím důvody, proč jsem se rozhodl pro právě jejich použití, a poté se zaměřím na implementaci a konfiguraci těchto nástrojů. Také odůvodním, proč jsem nevybral dříve zmíněné nástroje z 2. kapitoly.

Ve druhé části této kapitoly se zaměřím na aplikování nástrojů ZAP a Nessus do výrobního informačního systému CGMS. Nejdříve provedu skenování serveru s webovou aplikací a poté skenování otevřeného portu 8082. Následně popíšu postupy konfigurace nástrojů pro tato skenování. Shrnu nalezené zranitelnosti a navrhnu, jak ke zranitelnostem přistupovat při jejich odstranění. To stejné provedu i se samostatnou výrobní stanicí CGMS a její otevřeným portem 61745.

4.1 Výběr skenerů zranitelnosti

Rozhodl jsem se využít kombinaci skenerů zranitelností k maximalizaci efektivity a pokrytí při analýze bezpečnosti výrobního informačního systému CGMS. Po vyzkoušení a testování skenerů zranitelnosti z druhé teoretické části jsem se rozhodl pro nástroje ZAP a Nessus, které mají své jedinečné schopnosti a konkrétní oblasti využití, které se vzájemně doplňují a umožňují mi získat komplexní pohled na bezpečnostní situaci v systému.

Nástroj **ZAP** využiji kvůli jeho specializaci na webové aplikace a chování aplikací na úrovni HTTP/HTTPS. Ve své práci často využívám jako cíle skenů URL adresy, a proto je ZAP ideální volbou. Také má k dispozici širokou škálu užitečných add-onů, jak od komunity, tak od jeho vývojářů. Těmito add-ony lze rozšířit funkcionality nástroje, přizpůsobit pro konkrétní potřeby a vylepšit vzhled reportů tak, aby byly více uživatelsky přívětivé.

Nástroj **Nessus** využívá mnoho druhů skenů pomocí předem definovaných šablon, které je možné dále konfigurovat podle potřeb. Tyto šablony umožňují provádět širokou škálu skenů, včetně skenování portů, služeb, protokolů a konfigurací. Díky této rozmanitosti je schopen identifikovat různé typy zranitelností, které by nástroj ZAP nedokázal pokrýt. Nessus je kvůli jeho flexibilitě a širokému zaměření ideální spolupracovník s nástrojem ZAP.

Kombinace těchto nástrojů mi umožní pokrýt širokou škálu možných bezpečnostních hrozeb a zranitelností. Díky tomu dosáhnu maximalizace efektivity, účinnosti, pokrytí při analýze bezpečnosti a ochraně dat v systému CGMS.

4.1.1 Nevyužité nástroje

U skenerů zranitelností, které jsem zmínil ve 2. teoretické části jsem nevybral a nepoužil nástroje Invicti, Burp Suite, Nmap a OpenVAS.

- Společnost **Invicti** je velmi restriktivní ohledně propůjčení licencí pro výzkumné účely a z tohoto důvodu mi licence k bakalářské práci nebyla poskytnuta.
- **Burp Suite** má ve free verzi omezenou funkcionalitu a nepřinášel by významnou přidanou hodnotu do této práce, například nebylo možné využít automatický sken, který je k dispozici od vyšší placené verze.
- **Nmap** je sice vysoce kvalitním nástrojem pro celkové zmapování sítě, ale jeho používání mi bylo zakázáno bezpečnostním oddělením společnosti Continental. Často se stává, že Nmap analyzuje nejen svůj cíl, ale i zbytek sítě.
- Ohledně skeneru **OpenVAS** jsem limitován, protože provádím skeny ve virtuálním prostředí s operačním systémem Windows. OpenVAS je primárně určen pro Linux.

4.2 Instalace využitých nástrojů

Nástroj ZAP jsem si stáhnul z oficiálního webu https://www.zaproxy.org/download/ a nainstaloval jsem ho podle instrukcí. ZAP je desktopová aplikace pro bezpečnostní testování webových aplikací. K ní jsem přidal tři add-ony (doplňky) ze ZAP Marketplace:

1. Advanced SQL Injection Scanner – tento doplněk vylepšuje schopnost aplikace provádět útoky typu SQL injection. To znamená, že rozšiřuje možnosti základního útoku SQL injection, což je jedna z technik kybernetických útoků, která zneužívá nedostatky v databázových systémech prostřednictvím SQL dotazů.
2. Port Scanner – základní verze aplikace nezahrnuje funkci skenování portů. Tento doplněk přidává schopnost skenovat porty na cílovém serveru. Skenování portů je užitečné pro identifikaci otevřených portů a možných bezpečnostních zranitelností.
3. Report Generation – tento doplněk zlepšuje přehlednost a strukturu vytvářených zpráv o bezpečnostních testech provedených aplikací. Poskytuje také další šablony pro tvorbu reportů, což usnadňuje dokumentaci nalezených zranitelností.

Nástroj Nessus jsem nainstaloval z oficiálního webu společnosti Tenable pro Windows https://www.tenable.com/downloads/nessus?loginAttempted=true a postupoval jsem podle instrukcí. Jelikož se jedná o komerční nástroj, bylo nutné získat licenci. Po registraci jsem vyplnil formulář pro její zapůjčení, a licenci mi byla poskytnuta. Po aktivaci licence jsem neměl žádné potíže. Nessus se spouští ve webovém prohlížeči na localhostu na konkrétním portu.

4.3 Skenování webové aplikace a jejího serveru

CGMS je webová aplikace složena ze čtyř částí a to Dashboard, Administration, Operation a Reports. V částích Administration, Operation a Reports je potřeba se autentizovat přihlášením, jinak není možné se do těchto částí dostat. Nejdříve provedu skenování bez přihlašovacích údajů a následně autentizační skenování.

4.3.1 Skenování bez přihlašovacích údajů nástrojem ZAP

Jako první krok v procesu provádění bezpečnostního skenování zvolím použití ZAP skeneru. Toto skenování bude zaměřené na konkrétní cílovou URL adresu, a to: http://xxxxxxxx/CGMS.Dashboard . Důvodem tohoto výběru je skutečnost, že na této webové stránce není vyžadováno přihlašování. Tím pádem jsou crawlery schopny projít dál než pouze na úvodní stránku s přihlašovacím formulářem.

4.3.1.1 Konfigurace a spuštění metod

Provedení skenování pomocí ZAP vyžaduje specifický postup. Nejprve musím spustit automatické skenování nebo provést manuální prohlížení s cílovou URL adresou. Poté se webová stránka automaticky přidá do bočního panelu aplikace. V tomto panelu mám možnost nastavit různé parametry, jako jsou typy útoků, jejich síla a používané metody, které chci aplikovat na cílovou URL adresu.

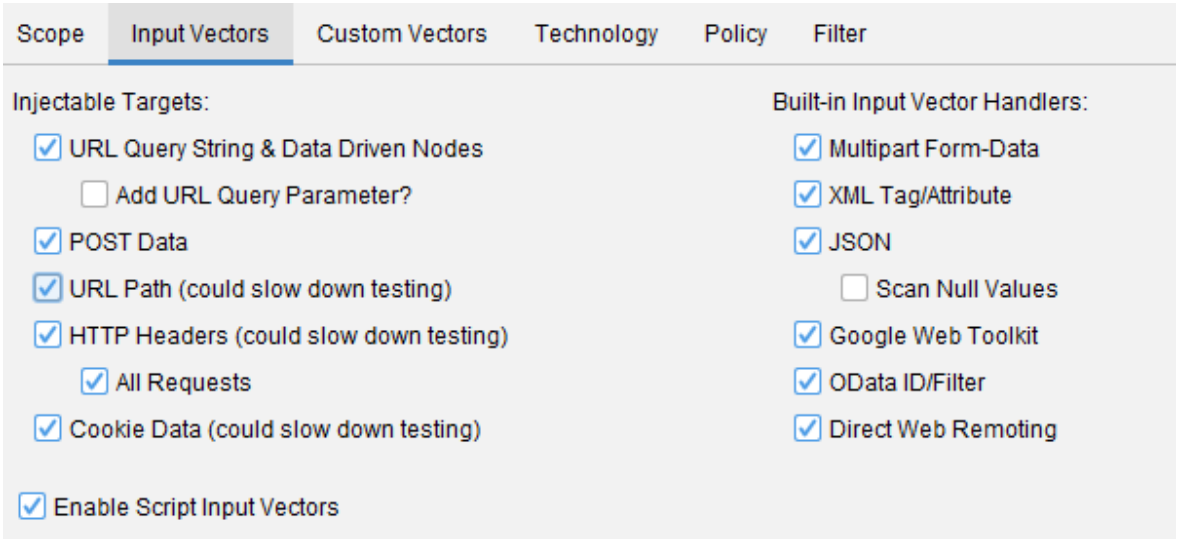
Před spuštěním prvního skenování pomocí ZAP je důležité využít jednu z klíčových vlastností skeneru, a to jsou módy. Výchozí nastavení módu je standardní, který je poměrně vyvážený a není příliš agresivní ani pasivní z bezpečnostního hlediska.

Nicméně, abych využil plný potenciál skeneru a aktivněji odhalil bezpečnostní zranitelnosti, rozhodl jsem se použít útočný mód. Tento mód je charakterizován svou agresivitou a aktivitou při provádění skenování. To znamená, že skener bude provádět intenzivnější a komplexnější testy, což zvyšuje šanci na odhalení širší škály potenciálních zranitelností.

Pro identifikaci dostupných stránek, funkcí a zdrojů webové aplikace jsem jako první metodu použil web spider, známý též jako crawler. Tento nástroj prochází aplikaci a vytváří mapu stránek. Následně jsem spustil AJAX spider, také známý jako crawljax, který identifikuje dynamicky generovaný obsah a interakce na webové stránce. Tyto crawlery vytvořily podrobnou mapu webové aplikace a poskytly seznam dostupných stránek a funkcí. Aktivní a pasivní skenování následně mapu využije, aby testovalo tyto zmapované prvky na možné zranitelnosti. Před spuštěním crawlerů jsem provedl úpravy v jejich nastavení, protože výchozí konfigurace byla nedostačující. Například jsem zvýšil dosah crawlerů pomocí nastavení depth neboli hloubky.

Při spuštění crawlerů se paralelně aktivuje pasivní skenování. Pasivní skenování ve výchozím nastavení prohledává HTTP požadavky a HTTP odpovědi. Tyto požadavky jsou odesílány webovou aplikací, která je skenována.

Pro aktivní skenování jsem zvolil následující postup. Nejprve jsem přes mapu sítě, konkrétně výchozí URL adresu, zahájil nastavení skenu pomocí funkce Attack a zde jsem vybral Active Scan. V novém otevřeném okně aktivního skenu jsem nastavil počáteční bod skenování. Dále jsem přešel do pokročilého nastavení, kde jsem upravil vstupní vektory pro skenování zahrnující URL cestu, HTTP záhlaví a další.



Obrázek 10. Použité vektorové vstupy

Pro zvýšení účinnosti skenování jsem upravil sílu útoku na vysokou hodnotu, konkrétně jsem zvolil režim High. Tento postup umožní vyšší i agresivnější počet útoků a tím se zvýší šance na identifikaci a nalezení zranitelností. Po provedení těchto nastavení jsem spustil aktivní skenování.

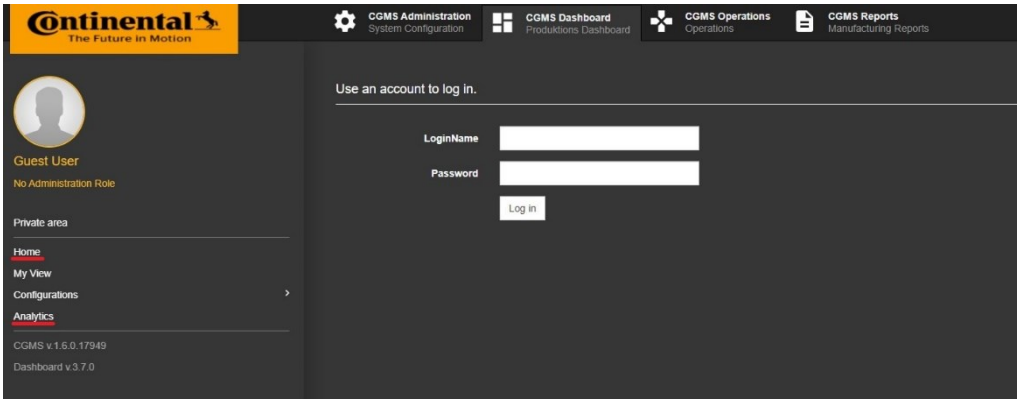
4.3.1.2 Výsledky ZAP skenování

Aktivní a pasivní skenování celkem identifikovalo 32 druhů zranitelností. Mezi nimi bylo zjištěno:

- 14 vysoce rizikových (z toho 11 typu SQL injection)
- 4 středně rizikové
- 7 nízko rizikových
- 7 informačních

Aktivní sken, identifikoval všech 14 vysoce rizikových zranitelností. Je obecně známý tím, že má vysoký počet falešně pozitivních výsledků. V tomto skenování se to jednoznačně potvrdilo. Všech těchto 14 zranitelností jsem musel manuálně ověřit tím, že jsem všechny útoky, které identifikovaly tyto zranitelnosti, manuálně zopakoval. Po všech 14 útocích se 13 z nich potvrdilo jako falešně pozitivní. Falešně pozitivní výsledky jsou následující:

1. Path Traversal – tento útok umožňuje útočníkovi přístup k souborům, adresářům a příkazům, které potenciálně existují mimo kořenový adresář webového dokumentu. V mém případě se jednalo o webovou stránku s přihlášením. Sken vyhodnotil tuto zranitelnost, jelikož při přihlášení se v bočním panelu dalo kliknout na odkazy, které odkazovaly na stránku. Jedná se o falešně pozitivní zranitelnost, protože odkazovaná stránka byla stejná a pouze se obnovila.



Obrázek 11. Ukázka přihlašovací stránky s označenými odkazy

2. External Redirect – tato zranitelnost umožňuje útočníkovi přesměrovat uživatele na externí a zároveň i nebezpečné webové stránky. Může být využita k provádění phishingových útoků nebo k přesměrování uživatelů na stránky obsahující škodlivý kód.

3. SQL injection – ostatních 11 falešně pozitivních zranitelností byly tohoto typu. Tato zranitelnost je řazena do kategorie OWASP Top 10 A03 2021.

   V první pokročilé SQL injection došlo k úspěšné manipulaci stránky pomocí boolean podmínky, po vyzkoušení daného útoku na webovou stránku se tato injection ukázala jako falešně pozitivní, jelikož vrátila úplně stejná data jako bez boolean podmínky.

   Další 2 SQL injection manipulovali s user-agentem. Do user-agenta přidaly boolean podmínky konkrétně:

   1. `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 OPR/102.0.0.0 AND 1=1`
   2. `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 OPR/102.0.0.0' AND 8174=2100 AND 'VAOO' LIKE 'VAOO`

   Útoky se jevily jako nejvíce nadějné ze všech SQL injection, ale po osobním otestování se mi nepodařilo získat jiná data. Získaná data byla naprosto identická jako s výchozím user-agentem.

   Zbylé SQL injection, které byly od různých databázových systémů jako MsSQL, MySQL a mnoho dalších, dokázali manipulovat s funkcí SLEEP(). Funkce umožňuje zpomalení provedení SQL dotazu. Při prvním testování dotazu obsahujícího funkci SLEEP() se zdálo, že výsledky jsou přesné, ale po opakovaném použití dotazu se odezva jevila rychlejší, což naznačuje, že dotazy byly uloženy do cache paměti databáze. Tento proces naznačuje, že první provedení dotazu mohlo trvat déle kvůli novému požadavku, který nikdy předtím nebyl zadán, zatímco následné opakování dotazu využilo vyrovnávací paměť cache, což vysvětluje rychlejší zpracování požadavku. Během těchto testů jsem se pokoušel manipulovat i s parametrem funkce SLEEP(), ale nezaznamenal jsem žádný vliv na dobu trvání dotazu v souladu s nastaveným parametrem. Tento fakt naznačuje, že změny v dotazu neovlivňovaly čas zpracování dotazu, a proto jsou tyto time-based SQL injection útoky falešně pozitivní.

   Překvapující bylo, že žádný z útoků nedokázal zjistit o jaký typ databáze se jedná.

Nejrizikovější zranitelnost v celém systému CGMS, kterou ZAP skener dokázal identifikovat a byla potvrzená, je:

1. Generic Padding Oracle – zranitelnost představuje kryptografické selhání, které spadá do OWASP Top 10, konkrétně do kategorie A02. Tato zranitelnost využívá nedostatky v implementaci paddingu (doplňkových dat) v šifrovacím algoritmu. Šifry šifrují data pouze ve určitých blocích. Data, jejichž velikost neodpovídá násobku velikosti bloku použité šifry, musí být do určité míry doplněna tak, aby dešifrovač byl schopen odstranit padding. Při útoku útočník využívá službu, která pošle zašifrovaná data. Služba následně vrátí chybovou zprávu, indikující, že se jedná o neplatný padding. Útočník může tuto informaci postupně využít k odhalení obsahu šifrované zprávy. Tento proces odhalování může trvat značnou dobu.
   Tato zranitelnost představuje vysoké riziko pro zabezpečení systémů. Její využití může vést k odhalení citlivých dat a ohrožení integrity a důvěrnosti informací. Proto je důležité tuto zranitelnost brát vážně.
   Aby bylo možné předejít této zranitelnosti, je nezbytné aktualizovat šifrovací algoritmy tak, aby nedávaly útočníkovi žádné informace o stavu paddingu. Implementace bezpečnějších kryptografických protokolů může významně snížit riziko úspěšného využití této zranitelnosti. [67]

Během pasivního skenování bylo identifikováno celkem 17 druhů zranitelností středního a nízkého rizika.

Zranitelnosti středního rizika:

1. Absence of Anti-CSRF Tokens (Chybějící Anti-CSRF tokeny) – zranitelnost znamená, že aplikace nepoužívá žádné tokeny proti Cross-Site Request Forgery (CSRF) útokům. CSRF útoky mohou vést k provedení neautorizovaných akcí v rámci účtu uživatel.
   Aplikace by měla implementovat rozšíření proti CSRF útokům, jako jsou CSRF tokeny nebo použití speciálního hlavičkového pole, které se ověřuje při každém odeslání formuláře.
2. Content Security Policy (CSP) Header Not Set – zranitelnost označuje nedostatek nastavení politiky zabezpečení obsahu (CSP) pro webovou stránku. CSP umožňuje správci webové stránky určit, odkud může být na stránce načítán obsah.

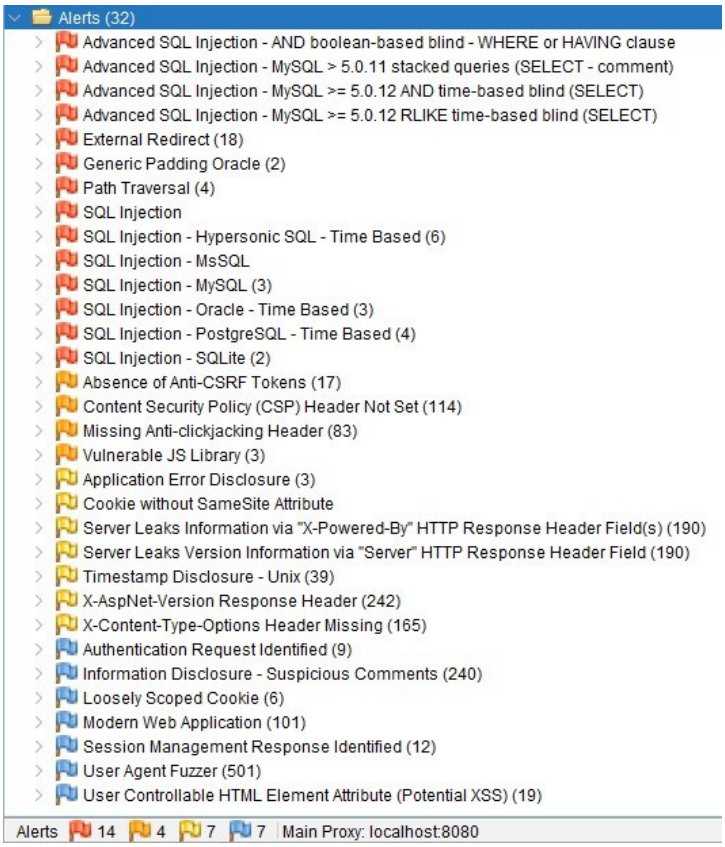
Webová aplikace by měla nastavit správnou politiku CSP pomocí HTTP hlavičky nebo meta značky v HTML kódu.

3. Missing Anti-clickjacking Header (Chybějící hlavička proti clickjackingu) – zranitelnost označuje chybějící ochranu proti útoku clickjacking. Clickjacking umožňuje útočníkovi zobrazit obsah jiného webu v iframe nebo frame (používá pro zobrazeni videi například z Youtube) na stránce útoku.

   Aplikace by měla nastavit správnou HTTP hlavičku X-Frame-Options s hodnotou "DENY" nebo "SAMEORIGIN", toto by mělo zabránit těmto útokům.
4. Vulnerable JS Library (Zranitelná JS knihovna) – zranitelnost označuje použití zranitelné JavaScriptové knihovny.

   Zranitelnost lze opravit nahrazením nebo aktualizací na novější verze.

Většina dalších chyb jsou spíše informační povahy a nejsou zpravidla považovány za závažné riziko.



Obrázek 12. Výsledky skenování v nástroji ZAP

4.3.2 Skenování autentizačním skenem nástrojem ZAP

Při použiti autentizačního skenu ZAP se neidentifikovali žádné jiné zranitelnosti, které by již nenalezl ZAP sken bez přihlašovacích údajů.

4.3.3 Skenování bez přihlašovacích údajů nástrojem Nessus

Nessus je komerční nástroj, tudíž je jeho nastavení velmi omezené, ale také snadné. Skener využívá šablony skenů, přičemž každá šablona je jedinečná a zaměřuje se na různé druhy skenů. Ze šablon jsem vybral Web Application Tests, jelikož nyní primárně cílím na webovou aplikaci.

Nastavení pro tuto šablonu bylo snadné a intuitivní. Po zadání cílové URL adresy: http://xxxxxxxx/, jsem v záložce se jménem Discovery vybral skenování běžných portů. Dále, v záložce hodnocení, jsem upravil sken tak, aby použil komplexní sken pro všechny webové zranitelnosti. Po těchto úpravách jsem skenování spustil.

4.3.3.1 Výsledky Nessus skenování

Sken identifikoval pouze 4 druhy informačních zranitelností. Jedna z těchto zranitelností je Nessus SYN scanner, zde nalezneme výsledky ze skenování portů. Skener objevil 4 standardní otevřené porty a 10 neznámých otevřených portů, konkrétně: 3389, 5985, 8080, 8081, 8082, 8083, 8085, 8090, 9593, 9594 a 9595, následně v ZAP pomocí add-onu skenování portů se výsledek potvrdil.

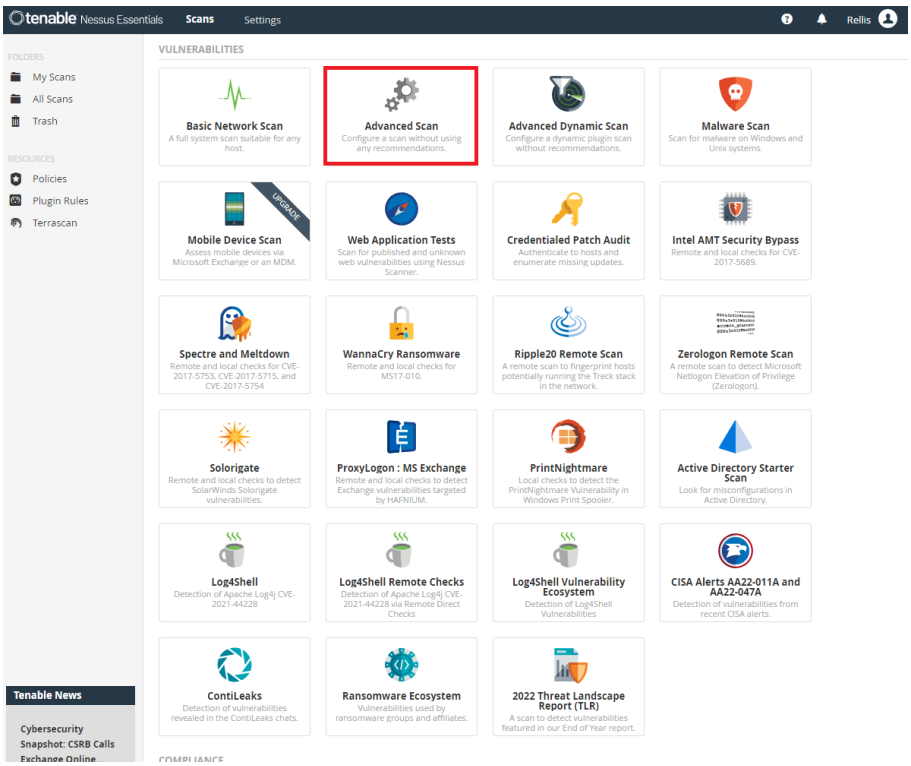
Mezi 4 standardní otevřené porty patřily:

- TCP port 80 - World Wide Web HTTP: Tento port je standardním portem pro webový provoz pomocí protokolu HTTP.
- TCP port 135 - DCE endpoint resolution: Port slouží k rozlišení koncových bodů distribuovaných v systému Windows a umožňuje komunikaci mezi aplikacemi v různých počítačových systémech.
- TCP port 139 a TCP port 445 - NETBIOS Session Service, Microsoft-DS: Tyto porty umožňují přístup k souborům a tiskárnám na síťovém zařízení.

Zbývající 3 identifikované druhy informačních zranitelností se zaměřily na odhalení informací o používaném serveru a technologii webové aplikace.

4.3.4 Skenování pokročilým autentizačním skenem nástrojem Nessus

V rámci nabídky šablon v Nessus jsem vybral Advanced Scan, která umožňuje provést hlubší a důkladnější skenování sítě a serveru. Tato šablona nabízí širší škálu funkcí a možností nastavení, a proto je nejvhodnější pro skenování celé sítě a serveru.



Obrázek 13. Nessus šablony s označenou šablonou Advanced Scan

Cílová URL adresa je základní: xxxxxxxx.

V panelu s názvem Discovery jsem aktivoval funkci Ping the remote host, která slouží k pingování vzdálených hostitelů za účelem zjištění, zda jsou aktivní (tzv. alive). V podrobnějším nastavení jsem vybral metody jako ARP (Adress Resolution Protocol), která ověřuje dostupnost a konektivitu zařízení v lokální síti a další metody jako TCP, UDP a ICMP (Internet Control Message Protocol). Tyto metody používají ICMP žádosti s příkazem "echo" k cílovému hostiteli a čekají na odpověď. Pro systémy Windows nebo unixové systémy, jako jsou Linux nebo macOS, se používá příkaz "ping".

Následně jsem vybral Service Discovery, kde jsem povolil funkci vyzkoušet všechny porty pro nalezení služeb. Tyto služby zahrnují SSL (Secure Sockets Layer) a TLS (Transport Layer Security), což jsou protokoly pro zabezpečení komunikace přes internet.

V panelu Hodnocení jsem vypnul skenování webové aplikace, jelikož jsem tuto funkci již využil v předchozím skeneru. Následně jsem zapnul skenování malwaru. Poté jsem zapnul

metody enumerace uživatelů, konkrétně Brute Force, Sam Registry, ADSI Query a WMI Query. Tyto techniky jsou používané v kybernetických útocích k získání informací o uživatelských účtech. Všechny zmíněné techniky jsou určené pro operační systém Windows, přičemž Sam Registry slouží k přístupu do registrační databáze, ADSI Query ke správě identit v Active Directory a WMI Query ke správě systémů.

Jelikož se jedná o autentizační skenování je klíčové zajistit, aby Nessus měl správné přihlašovací údaje. Pro tento účel slouží sekce Credentials. Zde jsem vyplnil vlastní přihlašovací údaje do sekce "Windows", které používám k přihlášení do počítače a zahájil skenování.

4.3.4.1 Výsledky Nessus autentizačního skenování

Autentizační sken identifikoval zranitelnosti:

- 1 vysoko rizikovou
- 5 středně rizikových
- 1 nízko rizikovou
- 46 informačních

Jako vysoce rizikovou zranitelnost vyhodnotil chybu:

1. SSL Certificate Signed Using Weak Hashing Algorithm – zranitelnost spočívá v použití slabého hashovacího algoritmu ve SSL certifikátu. Konkrétně se jedná o použití hashovacího algoritmu SHA-1 s RSA šifrováním. Tato kombinace je považována za nebezpečnou, protože SHA-1 má známé slabiny a není již považována za bezpečnou volbu pro kryptografické operace, jako je podepisování certifikátů. Nessus, hodnotící bezpečnostní zranitelnosti, automaticky označuje všechny certifikáty používající SHA-1 za slabé, přestože to tak nemusí být.
   Tento problém má vysoký index rizika 7,5 ve škále CVSS (Common Vulnerability Score System), což znamená, že má potenciál způsobit vážné problémy a měl by být co nejdříve řešen.

Sken vyhodnotil 5 středně rizikových zranitelností.

1. Remote Desktop Protocol Server Man-in-the-Middle Weakness – tato zranitelnost představuje potencionální bezpečnostní zranitelnost spojenou s protokolem vzdálené plochy (RDP). Man-in-the-middle je útok u kterého útočník zasahuje do komunikace mezi dvěma stranami. Komunikací může číst nebo manipulovat, pomocí

úprav dat nebo vložení falešných dat. V tomto kontextu může útočník odposlouchávat či manipulovat s komunikací mezi klientem a serverem RDP.

S touto zranitelností se dá zabránit více způsoby například: zabezpečit spojení šifrováním SSL/TLS (server již používá SSL certifikát, ale Nessus označil certifikát za nedůvěryhodný) nebo přidání bezpečnostní vrstvy skrze autentizaci – přístup k RDP server by měli pouze určití uživatelé nebo zařízení.

2. SSL Certificate Cannot Be Trusted (SSL certifikát nelze důvěřovat) – certifikát je považován za nedůvěryhodný kvůli tomu, že je podepsán podepsaný neznámou certifikační autoritou.

   Náprava této chyby se je jednoduchá, organizace by měla používat certifikát, který je vydán důvěryhodnou certifikační autoritou.
3. SSL Self-Signed Certificate (Certifikát SSL s vlastním podpisem) – tato chyba souvisí s chybou číslo 2. Akorát v tomto případě Nessus detekoval, že uživatel, který certifikát vytvořil, tak ho i podepsal.
4. Terminal Services Doesn't Use Network Level Authentication (NLA) Only – terminálové služby nevyužívají ověření na úrovni sítě (NLA). Ověřování poskytuje dodatečnou vrstvu zabezpečení tím, že vyžaduje ověření uživatele před samotným navázáním spojení s RDP serverem. Tímto způsobem může být zamezeno možným útokům jako Man-in-the-Middle.
5. Terminal Services Encryption Level is Medium or Low – Nessus konkrétně detekoval střední šifrování u terminálových služeb.

   Chyba je rázu středního rizika s indexem 4,3 CVSS. Šifrování stačí nastavit na silnější jako high, záleží na organizacích, zda jim střední úroveň šifrování postačí.

Většina dalších chyb jsou spíše informační povahy a nejsou zpravidla považovány za závažné riziko.

Obrázek 14. Výsledky Nessus skenování Serveru

4.3.5 Skenování otevřeného portu 8082

V této části práce jsem měl za úkol skenovat server webové aplikace, konkrétně port 8082, u kterého jsem dříve zjistil, že se jedná o otevřený port. Tudíž cílové adresy se odvíjí od URL adresy http://xxxxxxxx:8082.

Při použití Nessus skeneru cílová adresa nebyla povolena a byla označena jako “restricted“ neboli že přístup k adrese je omezen bezpečnostním opatřením.

4.3.5.1 Nastavení ZAP skeneru a konkrétních cílů

Port 8082 nemá výchozí webovou stránku, to samo o sobě není problém pro ZAP. Ale znamená to, že není možné provést automatický sken celého portu pomocí ZAP. Proto budu cílit na API požadavky a na konkrétní GET a POST metody na tomto portu, které se dají testovat.

4.3.5.1.1 GET metody

Nejprve jsem prověřil metody GET, konkrétně tři:

1. http://xxxxxxxx:8082/ServDA/api/GetList?param=xx
2. http://xxxxxxxx:8082/ServDA/health
3. http://xxxxxxxx:8082/ServMT/health.

U těchto cílů jsem posílil útoky SQL injection a Path Traversal (manipulace s URL adresou). Tyto útoky by mohly nejvíce využít těchto metod pro získání neoprávněného přístupu nebo pro manipulaci s daty. Zbývající útoky jsem nechal v defaultním nastavení. Tímto přístupem jsem měl za cíl zjistit, zda jsou tyto specifické metody a cesty odolný vůči známým bezpečnostním hrozbám.

4.3.5.1.2 POST metoda

ZAP skener nemá defaultně nastavený cíl pro metodu POST. V ZAP toto přenastavím v sekci Tools a po otevření možností vyberu Manual Request Editor. V novém otevřeném okně přepíšu požadavek z metody GET na POST a upravím cíl a hosta:

```
POST http://xxxxxxxx:8082/ServMT/api/SendInformation HTTP/1.1
host: xxxxxxxx:8082
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
pragma: no-cache
cache-control: no-cache
```

V metodě POST jsem opět podpořil útok SQL injection a nově Cross-Site Scripting. SQL injection je zásadní pro získání neoprávněného přístupu k datům, zatímco XSS může být využit k manipulaci s daty, obzvlášť pokud jsou zobrazena uživateli. Ostatní útoky jsem ponechal v defaultním nastavení.

4.3.5.2 Výsledky skenování otevřeného portu 8082

ZAP skenování nalezlo opět time-based SQL injection, konkrétně 5 druhů. Po testování těchto zranitelností znovu nastalo, že první provedení dotazu trvalo déle kvůli novému požadavku, který nikdy předtím nebyl zadán, zatímco následné opakování dotazu využilo vyrovnávací paměť cache a to vysvětluje rychlejší odezvu. Tento fakt naznačuje, že změny v dotazu neovlivňovaly čas požadavku, a proto jsou tyto time-based SQL injection útoky falešně pozitivní.

Také skenování nalezlo opět zranitelnost středního rázu Content Security Policy (CSP) Header Not Set, která je podrobně popsána v odstavci 4.3.1.2.

Dále identifikovalo 3 nízko rizikové zranitelnosti a to:

1. Server Leaks Information via "X-Powered-By" HTTP Response Header Field (s) – zranitelnost spočívá v tom, že server posílá HTTP hlavičku "X-Powered-By", která může odhalit informace o používaných technologiích a jejich verzích. To může pomoci útočníkovi identifikovat potenciální slabiny a cíleně napadnout server. Oprava spočívá v konfiguraci serveru tak, aby neposílal tuto konkrétní hlavičku.
2. Server Leaks Version Information via "Server" HTTP Response Header Field – podobně jako v předchozím případě, tento problém spočívá v odhalování informací o serveru a jeho verzi pomocí HTTP hlavičky "Server". Oprava je obdobná jako u předchozí zranitelnosti, a to v konfiguraci serveru tak, aby tato informace nebyla poskytována.
3. X-Content-Type-Options Header Missing – HTTP hlavička "X-Content-Type-Options" chybí v odpovědích serveru, která umožňuje webovým prohlížečům určit, zda mají respektovat deklarovaný MIME (Multipurpose Internet Mail Extensions) typ obsahu. Bez této hlavičky mohou prohlížeče být náchylnější k útokům typu MIME-sniffing. Oprava spočívá v nastavení této hlavičky s hodnotou "nosniff" v HTTP odpovědích serveru.

4.4 Skenování stanice CGMS

Nyní budu identifikovat zranitelnosti ve výrobní stanici CGMS. Nejdříve popíšu konfiguraci a následné výsledky pokročilého skenování nástrojem Nessus. Poté provedu skenování s nástrojem ZAP na otevřený port 61745 dané stanice.

4.4.1 Konfigurace skenu

Stanice nemá jednotné webové rozhrání, a proto je zde zbytečné aplikovat skener ZAP. Jeho zjišťovací metody, crawlery, by zde nebyly efektivní, a proto by aktivní skener by neměl co otestovat.

Nessus nedokázal najít cílovou URL adresu http://xxxxxxxx. Tuto chybu se obešel pomocí zjištění IP adresy, a to pomocí příkazu: **ping xxxxxxxx**. Nessus následně IP adresu xx.xxx.xx.xx úspěšně identifikoval jako cíl pro skenování.

Konfiguraci skenování jsem využil stejného nastavení jako u konfigurace Nessus autentizačního skenování pro cíl serveru webové aplikace.

4.4.2 Výsledky Nessus skenování stanice

Nessus pokročilý sken identifikoval 68 různých druhů zranitelností.

Nejrizikovější zranitelnosti v celém systému CGMS, které Nessus skener dokázal identifikovat ve stanici jsou dvě kritické zranitelnosti, těmi jsou:

1. SSL Version 2 and 3 Protocol Detection – SSL verze 2 a 3 jsou zastaralé kryptografické protokoly používané pro zabezpečenou komunikaci mezi klientem a serverem. Tyto verze SSL jsou již známé svými zranitelnostmi a umožňují útočníkovi odposlech komunikace anebo odhalení citlivých dat.
   Nejjednodušším řešením je vypnutí podpory SSL verze 2 a 3 a nahrazení modernějšími a bezpečnějšími protokoly, jako je TLS verze 1.2 a vyšší. Je také důležité zajistit pravidelné aktualizace softwaru, který používá SSL/TLS, aby byly zajištěny nejnovější zabezpečení.
2. Microsoft SQL Server Unsupported Version Detection (remote check) – zranitelnost znamená, že cílový Microsoft SQL Server používá nepodporovanou verzi softwaru, což může představovat riziko z hlediska bezpečnosti a stability. Nedostatek podpory znamená, že výrobce nevydává žádné nové bezpečnostní vylepšení pro produkt.

Oprava této zranitelnosti zahrnuje aktualizaci Microsoft SQL Serveru na podporovanou verzi. To zajišťuje, že systém využívá nejnovější opravy a zabezpečení poskytované Microsoftem.

Dále skener objevil i vysoko rizikové zranitelnosti, a to opět slabý hashovací algoritmus jako u serveru webové aplikace a jednu novou:

1. SSL Certificate Signed Using Weak Hashing Algorithm
2. SSL Medium Strength Cipher Suites Supported (SWEET32) – zranitelnost SWEET32 spočívá v tom, že webový server podporuje šifrovací algoritmy s 64bitovým šifrováním, což je považováno za střední úroveň síly. Útočník může využít tuto zranitelnost k prolomení šifrování. Nessus považuje za středně silné šifrování každé šifrování, které používá klíče o délce alespoň 64 bitů a méně než 112 bitů.

   Nejlepším způsobem, jak opravit zranitelnost, je zakázat používání šifrovacích algoritmů se střední sílou a nahradit je modernějšími a bezpečnějšími šifrovacími algoritmy. To lze provést konfigurací webového serveru tak, aby podporoval pouze silné šifrovací algoritmy s dostatečnou délkou klíče.

Nessus identifikoval 11 středně rizikových zranitelností a těmi jsou:

1. HSTS Missing From HTTPS Server (RFC 6797) – HSTS (HTTP Strict Transport Security) je bezpečnostní mechanismus, který umožňuje webovému serveru informovat prohlížeč, že by měl vždy používat HTTPS pro komunikaci se serverem. Pokud je HSTS nastavena, prohlížeč si tuto informaci pamatuje a automaticky přesměruje nezabezpečené HTTP požadavky na HTTPS. Mechanismus minimalizuje riziko útoků typu Man-in-the-Middle.

   Pro opravu této zranitelnosti je nutné implementovat HSTS na HTTPS serveru. To lze provést přidáním speciálního hlavičkového pole "Strict-Transport-Security" do odpovědi serveru.
2. Remote Desktop Protocol Server Man-in-the-Middle Weakness
3. SSL Certificate Cannot Be Trusted
4. SSL Self-Signed Certificate
5. TLS Version 1.0 Protocol Detection – TLS verze 1.0 je zastaralý šifrovací protokol. Používání tohoto protokolu může představovat riziko pro bezpečnost komunikace mezi klienty a serverem.

Verze stačí nahradit za novější verzi protokolu, a to je považována verze TLS 1.2 nebo TLS 1.3.

6. TLS Version 1.1 Protocol Deprecated – TLS verze 1.1 je zastaralý šifrovací protokol. Používání tohoto protokolu může představovat riziko pro bezpečnost komunikace mezi klienty a serverem.
   Verze stačí nahradit za novější verzi protokolu, a to je považována verze TLS 1.2 nebo TLS 1.3.
7. SSL RC4 Cipher Suites Supported (Bar Mitzvah) – šifra RC4 má chybu v generování pseudonáhodného proudu bajtů, takže je možné generovat nejrůznější typy malých zkreslení, což snižuje jeho náhodnost. Pokud je otevřený text šifrován opakovaně a útočník je schopen získat mnoho (mnoho se myslí jako desítky milionů) šifrovaných textů, je schopen odvodit otevřený text.
   Zde stačí nahradit RC4 šifru za spolehlivější jako AES-GCM (Advanced Encryption Standard s provozním režimem Galois/Counter Mode).
8. SMB Signing not required – u protokolu by mělo být povoleno podepisování SMB, aby se zajistila autenticita a integrita dat přenášených prostřednictvím tohoto protokolu. Útočník může využít zranitelnost pomocí útoku Man-in-the-Middle dokáže získat data a přepisovat je na podvržená dat.
9. SSL Certificate with Wrong Hostname – SSL certifikát použitý na serveru má nesprávný hostname. To indikuje možný problém s konfigurací serveru. To lze provést vygenerováním nového certifikátu se správným hostname, aby odpovídal doméně serveru.
10. Terminal Services Doesn't Use Network Level Authentication (NLA) Only
11. Terminal Services Encryption Level is Medium or Low

Nalezené 4 nízko rizikové zranitelnosti:

1. SSLv3 Padding Oracle On Downgraded Legacy Encryption – Vulnerability (POODLE) – hostitel je zranitelný proti útoku typu man-in-the-middle, která je známá jako POODLE. Zranitelnost je způsobena tím, jakým protokol SSL 3.0 zpracovává výplňové bajty při dešifrování zpráv. Dešifrovat vybraný bajt šifrového textu jde během 256 pokusů, ale pouze pokud útočník donutí aplikaci opakovaně odesílat stejná data. Opět se SSL protokol stačí nahradit modernějším protokolem TLS.
2. SSL Certificate Chain Contains RSA Keys Less Than 2048 bits

3. Terminal Services Encryption Level is not FIPS-140 Compliant – FIPS 140 je federální standard pro šifrování, který je často vyžadován pro ochranu citlivých informací. Konfiguraci stačí nastavit, aby splňovala požadavky FIPS 140.
4. Web Server Allows Password Auto-Completion

Zbylých 49 zranitelností byly informačního rázu.

Obrázek 15. Výsledky Nessus skenování Stanice

4.4.3 Skenování otevřeného portu 61745 nástrojem ZAP

Na výchozí webové stránce portu 61745 jsem zjistil přítomnost Swaggeru, což je nástroj využívaný k dokumentaci API. Vzhledem k tomu, že na této stránce existuje webové rozhraní, je možné využít crawlery ZAP skeneru k prohledávání jejího obsahu. Tento postup jsem realizoval a ZAP skenerem provedl analýzu tohoto portu.

Nessus nedokázal zacílit na URL adresu otevřeného portu a následně ani na IP adresu, u obou možností Nessus neměl pří-stup, proto jsem ho nemohl využít.

Po provedení ZAP skenování s výchozím nastavením na adrese http://xxxxxxxx:61745 jsem neidentifikoval žádné nové zranitelnosti, které by nebyly již zahrnuty v předchozích skenováních. Výsledky tohoto skenování se shodovaly s výsledky skenování portu 8082, které jsem popsal dříve v odstavci 4.3.5.2 (s výjimkou SQL injection).

5 VYHODNOCENÍ

Výrobní informační systém CGMS je klíčovým prvkem výrobního procesu firmy Continental, poskytujícím nezbytné informace pro řízení výroby a logistiky. Zajištění bezpečnosti tohoto systému je tedy kritické pro ochranu důvěrnosti, integrity a dostupnosti dat.

Považuji tuto mojí bakalářskou práci za prospěšnou a to proto, že jsem pomocí aplikovaných nástrojů pro analýzu bezpečnosti, zejména nástroje Nessus, odhalil několik závažných zranitelností ve výrobním informačním systému CGMS. Tato identifikace zranitelností je pro ochranu systémů klíčová. Díky mé práci byla firma Continental schopna provést nezbytné kroky k odstranění identifikovaných zranitelností a zabezpečit tak své prostředí proti potenciálním hrozbám.

Nástrojem ZAP jsem ve výrobním závodě v Otrokovicích, kde jsem celou praktickou část mé práce prováděl, jsem identifikoval jak pár falešně pozitivních výsledků, tak i zranitelnost vysokého rizika známou jako Generic Padding Oracle. Ta představuje závažnou hrozbu, která může vést k odhalení citlivých informací.

Pomocí nástroje Nessus se mi povedlo identifikovat mnoho zranitelností, z nichž jsou nejdůležitější následující.

Kriticky rizikové zranitelnosti jako je využívání zastaralých protokolů SSL verze 2 a 3 a detekce nepodporovaných verzí SQL Serveru, představují naléhavé bezpečnostní hrozby. Tyto zranitelnosti poskytují útočníkům přímou cestu k provádění útoků typu man-in-the-middle nebo získání neoprávněného přístupu k důležitým datům.

Vysoce rizikové zranitelnosti, jako je použití slabých hashovacích algoritmů při podepisování SSL certifikátů a podpora šifrovacích algoritmů s nižší sílou, představují další vážné hrozby. Tyto nedostatky mohou umožnit útočníkům dešifrovat komunikaci a získat citlivé informace.

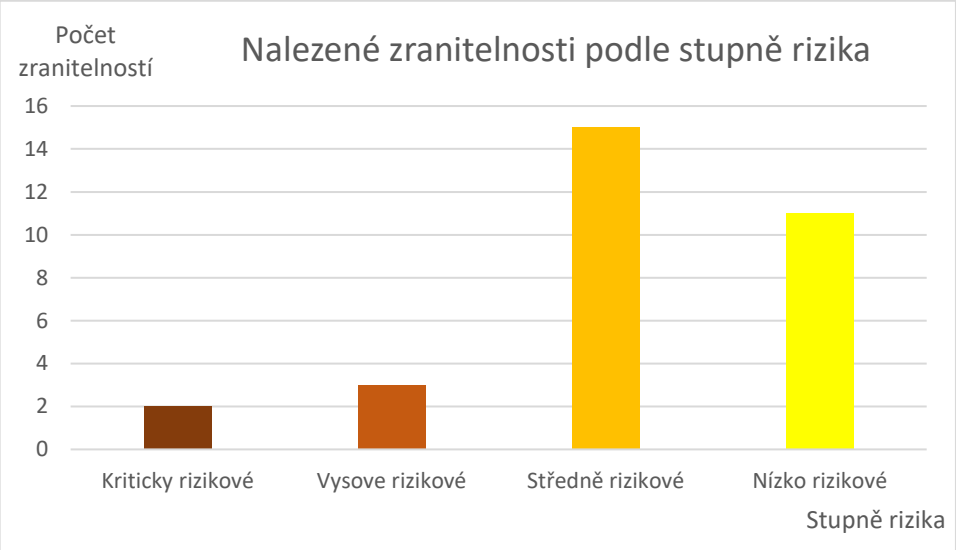
Firma Continental by měla reagovat na tyto zranitelnosti. To zahrnuje aktualizaci softwaru na podporované verze, vypnutí zastaralých protokolů a implementaci bezpečnostních opatření odolných proti novým typům útoků.

Je důležité, aby firma Continental uplatňovala bezpečnostní opatření jak v testovacím, tak v produkčním prostředí. Zranitelnosti objevené při práci na mé bakalářské práci v testovacím prostředí, mohou indikovat potenciální rizika i v produkčním prostředí.

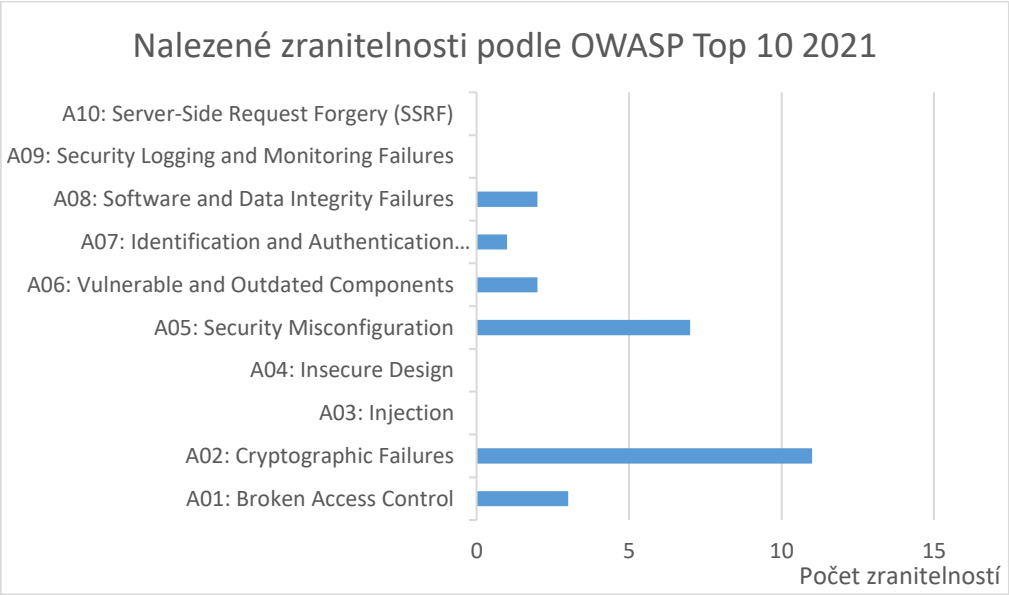
Naštěstí firma Continental pro monitorování a správu bezpečnosti systému CGMS již využívá pokročilé nástroje jako např. Nessus Professional. Tento proaktivní přístup je důležitý pro včasnou identifikaci a řešení bezpečnostních problémů.

Odstranění identifikovaných zranitelností a udržení vysoké úrovně bezpečnosti jsou klíčové pro ochranu výrobního informačního systému CGMS a zabránění potenciálním bezpečnostním hrozbám. Použití pokročilých nástrojů pro analýzu bezpečnosti a proaktivní přístup k bezpečnosti jsou nezbytné pro dosažení tohoto cíle.

5.1 Grafické vyhodnocení



Obrázek 16. Graf znázorňující počet nalezených zranitelností podle stupně rizika



Obrázek 17. Graf znázorňující počet zranitelností podle standardu OWASP Top 10 2021

ZÁVĚR

Cílem mé bakalářské práce bylo analyzovat výrobní systém CGMS společnosti Continental pomocí nástrojů na skenování zranitelností Nessus a Zed Attack Proxy. Při této analýze jsem využil již zmíněné volně dostupné nástroje, které mi umožnily efektivně identifikovat zranitelnosti. Praktická část mé práce probíhala v prostorách závodu Continental Barum v Otrokovicích, kde jsem strávil mnoho hodin při skenování a hodnocení zranitelností.

Výsledky mé bakalářské práce jasně demonstrovaly úspěch v dosažení stanoveného cíle. Identifikoval jsem řadu zranitelností ve výrobním informačním systému CGMS, včetně kritických a vysoce rizikových, a detailně je popsal, jak fungují a jaké problémy mohou způsobit. Následné jsem navrhl také doporučení pro jejich opravu a zlepšení bezpečnosti systému.

Firma Continental reagovala na výsledky mé bakalářské práce a mé doporučení ohledně zlepšení bezpečnosti výrobního informačního systému CGMS. Moje reporty s nalezenými zranitelnostmi byly přijaty s otevřeností a vstřícností, což svědčí o ochotě společnosti přijímat zpětnou vazbu a pracovat na zajištění bezpečnosti svých systémů.

Zasílal jsem reporty s nalezenými zranitelnostmi zástupcům společnosti Continental a doufám, že moje práce přispěla k jejich zlepšení.

Je povzbudivé vidět, že i při použití volně dostupných nástrojů jsem byl schopen identifikovat zranitelnosti, a to navzdory tomu, že firma Continental již využívá profesionální skenery zranitelností.

Závěrem je jasné, že pravidelné skenování zranitelností má klíčový význam pro udržení bezpečnosti systémů. Věřím, že tato práce poskytuje hodnotný příspěvek k diskusi o ochraně dat a infrastruktury a že může sloužit jako inspirace pro další výzkum v této oblasti.

Také je důležité zdůraznit, že i přes úspěch této práce je nezbytné většinu nalezených zranitelností manuálně přetestovat, převážně ty, co byly nalezeny aktivním skenováním. Některé z nich mohou být označeny jako falešně pozitivní a spoléhat se výhradně na výsledky zvolených nástrojů by mohlo být riskantní.

Nakonec bych chtěl zdůraznit, že úspěch této práce byl možný díky podpoře a povzbuzení od mého akademického prostředí. Velmi si vážím příležitosti a podpory, kterou jsem během zpracování této bakalářské práce a mého studia obdržel.

SEZNAM POUŽITÉ LITERATURY

[1] Manufacturing Information System. *Matics.live* [online]. 2023 [cit. 2024-04-29]. Dostupné z: https://matics.live/glossary/manufacturing-information-system

[2] Executive information system. *Wikipedia.org* [online]. 2023 [cit. 2024-04-29]. Dostupné z: https://en.wikipedia.org/wiki/Executive_information_system

[3] Kybernetická bezpečnost (kyberbezpečnost). Definice, význam, řízení, povinnosti a legislativa. *Legislativa.cz* [online]. 2022 [cit. 2024-04-29]. Dostupné z: https://legislativa.cz/zdroje/kyberneticka-bezpecnost/kyberbezpecnost

[4] GDPR. *Acsoffice.cz* [online]. 2016 [cit. 2024-04-29]. Dostupné z: https://acsoffice.cz/kyberneticka-bezpecnost/gdpr/

[5] NÚKIB v roce 2023 zaznamenal rekordní počet kybernetických incidentů. *Nukib.gov.cz* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://nukib.gov.cz/cs/infoservis/aktuality/2073-nukib-v-roce-2023-zaznamenal-rekordni-pocet-kybernetickych-incidentu/

[6] ZPRÁVA O STAVU KYBERNETICKÉ BEZPEČNOSTI ČESKÉ REPUBLIKY ZA ROK 2021. *Nukib.gov.cz* [online]. 2022 [cit. 2024-04-29]. Dostupné z: https://nukib.gov.cz/download/publikace/zpravy_o_stavu/Zprava_o_stavu_kybernetick_bezpenosti_2021.pdf

[7] CHIN, Kyle. Cybersecurity in the Manufacturing Industry. *Upguard.com* [online]. 2023 [cit. 2024-04-29]. Dostupné z: https://www.upguard.com/blog/cybersecurity-in-the-manufacturing-industry#toc-0

[8] Distribution of cyberattacks across worldwide industries in 2023. *Statista.com* [online]. 2024 [cit. 2024-04-29]. Dostupné z:

https://www.statista.com/statistics/1315805/cyber-attacks-top-industries-worldwide/

[9] Cybersecurity Statistics: The Definitive List [2023]. FEDOR, Octav. *Antivirusguide.com* [online]. 2023 [cit. 2024-04-29]. Dostupné z: https://www.antivirusguide.com/cybersecurity/cybersecurity-statistics/

[10] Co je to zranitelnost. *Aptien.com* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://aptien.com/cs/kb/articles/what-is-vulnerability

[11] Vulnerability Scanners and Scanning Tools: What To Know. *Balbix.com* [online]. 2020 [cit. 2024-04-29]. Dostupné z: https://www.balbix.com/insights/what-to-know-about-vulnerability-scanning-and-tools/

[12] Vulnerability scanner. *Wikipedia.org* [online]. 2021 [cit. 2024-04-29]. Dostupné z: https://en.wikipedia.org/wiki/Vulnerability_scanner

[13] HARRELL, Christopher R., Mark PATTON, Hsinchun CHEN a Sagar SAMTANI. Vulnerability Assessment, Remediation, and Automated Reporting: Case Studies of Higher Education Institutions. *Ieeexplore.ieee.org* [online]. 2018, **2018**(1), 153 [cit. 2024-04-29]. Dostupné z: doi:10.1109/ISI.2018.8587380

[14] SAMTANI, Sagar. Vulnerability Scanner System Diagram. In: *Researchgate.net* [online]. 2018, s. 153 [cit. 2024-04-29]. Dostupné z: https://www.researchgate.net/figure/Vulnerability-Scanner-System-Diagram_fig1_329958789

[15] Vulnerability Scanning 101. *Securitymetrics.com* [online]. 2020 [cit. 2024-04-29]. Dostupné z: https://www.securitymetrics.com/learn/vulnerability-scanning-101

[16] Welcome to the OWASP Top 10 - 2021. *Owasp.org* [online]. 2021 [cit. 2024-04-29]. Dostupné z: https://owasp.org/Top10/

[17] What is Broken Access Control Vulnerability And How to Prevent it. *Authgear.com* [online]. 2023 [cit. 2024-04-29]. Dostupné z: https://www.authgear.com/post/what-is-broken-access-control-vulnerability-and-how-to-prevent-it

[18] Co zjišťujeme. *Skenerwebu.cz* [online]. 2021 [cit. 2024-04-29]. Dostupné z: https://www.skenerwebu.cz/cs/co-zjistujeme/

[19] CODE INJECTION. *Contrastsecurity.com* [online]. 2022 [cit. 2024-04-29]. Dostupné z: https://www.contrastsecurity.com/glossary/code-injection

[20] KINGTHORIN. SQL Injection. *Owasp.org* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://owasp.org/www-community/attacks/SQL_Injection

[21] HOFFMAN, Andrew. *Web Application Security: Exploitation and Countermeasures for Modern Web Applications*. O'Reilly Media, 2020. ISBN 1492053112.

[22] KIRSTENS. Cross Site Scripting (XSS). *Owasp.org* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://owasp.org/www-community/attacks/xss/

[23] A04:2021 – Insecure Design. *Owasp.org* [online]. 2021 [cit. 2024-04-29]. Dostupné z: https://owasp.org/Top10/A04_2021-Insecure_Design/

[24] Security Misconfiguration: Impact, Examples and Prevention. *Balbix.com* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://www.balbix.com/insights/security-misconfiguration-impact-examples-and-prevention/

[25] VALEZQUEZ, Alex. OWASP Top 10: #6 Vulnerable and Outdated Components. *Foresite.com* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://foresite.com/blog/owasp-top-10-vulnerable-and-outdated-components/

[26] VALEZQUEZ, Alex. OWASP Top 10 Identification and Authentication Failures. *Securityjourney.com* [online]. 2023 [cit. 2024-04-29]. Dostupné z: https://www.securityjourney.com/post/owasp-top-10-identification-and-authentication-failures

[27] NAPRYS, Ernestas. Following Taurus leak, Bundeswehr issues a statement using 1234 as password. *Cybernews.com* [online]. 2024 [cit. 2024-04-29]. Dostupné z: https://cybernews.com/news/bundeswehr-issues-statement-using-1234-password/

[28] What are software and data integrity failures? *Educative.io* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.educative.io/answers/what-are-software-and-data-integrity-failures

[29] What is CI/CD? *Gitlab.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://about.gitlab.com/topics/ci-cd/

[30] What are security logging and monitoring failures? *Educative.io* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.educative.io/answers/what-are-security-logging-and-monitoring-failures

[31] V, Adithyakrishna. Server-side request forgery (SSRF). *Medium.com* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://medium.com/@adithyakrishnav001/server-side-request-forgery-ssrf-bf23802cfb12

[32] Static Application Security Testing. *Synopsys.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.synopsys.com/glossary/what-is-sast.html

[33] Dynamic Application Security Testing (DAST). *Synopsys.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.synopsys.com/glossary/what-is-dast.html

[34] RISKOPTICS. Vulnerability Scanners: Passive Scanning vs. Active Scanning. *Reciprocity.com* [online]. 2022 [cit. 2024-05-01]. Dostupné z: https://reciprocity.com/blog/vulnerability-scanners-passive-scanning-vs-active-scanning/

[35] What is port scanning? *Avast.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.avast.com/business/resources/what-is-port-scanning#pc

[36] GORMAN, Ben. TCP vs UDP: What's the Difference and Which Protocol Is Better? *Avast.com* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://www.avast.com/c-tcp-vs-udp-difference

[37] Web crawler. *Wikipedia.org* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://en.wikipedia.org/wiki/Web_crawler

[38] Crawljax. *Dpedia.org* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://dbpedia.org/page/Crawljax

[39] ŠTRÁFELDA, Jan. AJAX. *Strafelda.cz* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.strafelda.cz/ajax

[40] ALI, Neda. Authenticated vs unauthenticated scans. *Beaglesecurity.com* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://beaglesecurity.com/blog/article/authenticated-vs-unauthenticated-scans.html

[41] Dependency Scanning. *Gitlab.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://docs.gitlab.com/ee/user/application_security/dependency_scanning/

[42] ROUSE, Margaret. Commercial Software. *Techopedia.com* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://www.techopedia.com/definition/4245/commercial-software

[43] Invicti. *Invicti.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.invicti.com/

[44] Invicti Reviews. *Gartner.com* [online]. 2015 [cit. 2024-05-01]. Dostupné z: https://www.gartner.com/reviews/market/application-security-testing/vendor/invicti

[45] Invicti (formerly Netsparker). *G2.com* [online]. 2018 [cit. 2024-05-01]. Dostupné z: https://www.g2.com/products/invicti-formerly-netsparker/reviews

[46] Invicti Licensing. *Invicti.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.invicti.com/support/invicti-licensing/

[47] AWATI, Rahul. Nessus. *Techtarget.com* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://www.techtarget.com/searchnetworking/definition/Nessus

[48] Tenable Nessus Reviews. *Gartner.com* [online]. 2015 [cit. 2024-05-01]. Dostupné z: https://www.gartner.com/reviews/market/vulnerability-assessment/vendor/tenable/product/tenable-nessus

[49] Tenable Nessus. *G2.com* [online]. 2018 [cit. 2024-05-01]. Dostupné z: https://www.g2.com/products/tenable-nes-sus/reviews.html?focus_review=4463724&page=11&product_id=nessus

[50] Tenable Nessus Licensing. *Tenable.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://docs.tenable.com/quick-reference/licensing-guide/Content/tenable-nessus-licensing.htm

[51] ASADOORIAN, Paul. Nessus HTML5 UI 2.1 Provides Enhanced Usability. In: *Tenable.com* [online]. 2014 [cit. 2024-05-01]. Dostupné z: https://www.tenable.com/blog/nessus-html5-ui-21-provides-enhanced-usability

[52] Burp Suite's web vulnerability scanner. *Portswigger.net* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://portswigger.net/burp/vulnerability-scanner

[53] Burp Suite Professional Reviews. *Gartner.com* [online]. 2016 [cit. 2024-05-01]. Dostupné z: https://www.gartner.com/reviews/market/application-security-testing/vendor/portswigger/product/burp-suite-professional

[54] Burp Suite. *G2.com* [online]. 2018 [cit. 2024-05-01]. Dostupné z: https://www.g2.com/products/burp-suite/reviews

[55] What is open source? *Opensource.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://opensource.com/resources/what-open-source

[56] Zed Attack Proxy. *Sciencedirect.com* [online]. 2014 [cit. 2024-05-01]. Dostupné z: https://www.sciencedirect.com/topics/computer-science/zed-attack-proxy

[57] Getting Started. *Sciencedirect.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.zaproxy.org/getting-started/

[58] Nmap. *Nmap.org* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://nmap.org/

[59] Background. *Greenbone.github.io* [online]. 2021–2024 [cit. 2024-05-01]. Dostupné z: https://greenbone.github.io/docs/latest/background.html#

[60] OpenVAS. *Bugcrowd.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.bugcrowd.com/glossary/openvas-vulnerability-scanner/

[61] SonarQube. *Sonarsource.com* [online]. 2024, 2008-2024 [cit. 2024-05-01]. Dostupné z: https://www.sonarsource.com/products/sonarqube/

[62] What Is Wireshark and How Is It Used? *Comptia.org* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it

[63] Metasploit. *Imperva.com* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://www.imperva.com/learn/application-security/metasploit/

[64] Penetrační test. *Wikipedia.org* [online]. 2022 [cit. 2024-05-01]. Dostupné z: https://cs.wikipedia.org/wiki/Penetra%C4%8Dn%C3%AD_test

[65] Continental. *Wikipedia.org* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://cs.wikipedia.org/wiki/Continental

[66] Continental Barum. *Wikipedia.org* [online]. 2023 [cit. 2024-05-01]. Dostupné z: https://cs.wikipedia.org/wiki/Continental_Barum

[67] Testing for Padding Oracle. *Owasp.org* [online]. 2024 [cit. 2024-05-01]. Dostupné z: https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/09-Testing_for_Weak_Cryptography/02-Testing_for_Padding_Oracle

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CGMS	Continental Global Manufacturing System
UTB	Univerzita Tomáše Bati
OWASP	Open Web Application Security Project
GDPR	General Data Protection Regulation
NÚKIB	Národní úřad pro kybernetickou a informační bezpečnost
IT	Informační Technologie
GUI	Graphical User Interface
URL	Uniform Resource Locator
EU	Evropská unie
TLS	Transport Layer Security
SSL	Secure Sockets Layer
SQL	Structured Query Language
LDAP	Lightweight Directory Access Protocol
XSS	Cross-Site Scripting
CI/CD	Continuous Integration/Continuous Deployment
SAST	Static Application Security Testing
DAST	Dynamic Application Security Testing
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WWW	World Wide Web
HTML	Hypertext Markup Language
AJAX	Asynchronous JavaScript and XML
XML	Extensible Markup Language
ZAP	Zed Attack Proxy

HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
OSS	Open Source Software
Nmap	Network Mapper
OpenVAS	Open Vulnerability Assessment System
IP	Internet Protocol
AG	Aktiengesellschaft
MVC	Model-View-Controller
UI	User Interface
PLC	Programmable Logic Controller
CSRF	Cross-Site Request Forgery
CSP	Content Security Policy
JS	JavaScript
ARP	Address Resolution Protocol
ICMP	Internet Control Message Protocol
ADSI	Active Directory Service Interfaces
WMI	Windows Management Instrumentation
SHA-1	Secure Hash Algorithm 1
RSA	Rivest-Shamir-Adleman
CVSS	Common Vulnerability Scoring System
RDP	Remote Desktop Protocol
NLA	Network Level Authentication
API	Application Programming Interface
MIME	Multipurpose Internet Mail Extensions
HSTS	HTTP Strict Transport Security
SMB	Server Message Block

SEZNAM OBRÁZKŮ

SEZNAM PŘÍLOH

PŘÍLOHA P I: CD

CD obsahuje adresář “prilohy.zip“, který obsahuje podadresáře “Port serveru 8082“, “Port stanice 61745“, “Server“ a “Stanice“ v těchto složkách se nachází reporty ze skenování. Dále CD obsahuje elektronickou verzi této bakalářské práce, která má jméno “fulltext.pdf“