Jihočeská univerzita v Českých Budějovicích

Přírodovědecká fakulta

Analýza dat z automatických bezpečnostních scannerů

Bakalářská práce

Vodstrčil Pavel

Vedoucí práce: Ing. Břehovský Petr

České Budějovice 2019

Bibliografické údaje

Vodstrčil P., 2019: Analýza dat z automatických bezpečnostních scannerů [Data analysis from automatic vulnerability scanners Bc. Thesis, in Czech] - 39 p., Faculty of Science, The University of South Bohemia, České Budějovice, Czech Republic

Anotace

Tato bakalářská práce se zabývá zkoumáním a zpracováním reportů z automatických bezpečnostních skenerů. V začátku teoretické části je krátké seznámení se skenováním. Dále jsou rozebrány jednotlivé výstupy ze skenerů (reporty) a popis položek. V další části následuje seznámení s Common Vulnerability Scoring System, který je využíván v praktické části pro ohodnocování. Na konci první části jsou již uvedeny některé funkce vytvořené aplikace. Začátek praktické části je věnován návrhu databáze a vybranému frameworku pro tvorbu. Následuje seznámení s funkcemi aplikace a samotné možnosti zobrazení výsledků.

Klíčová slova

Bezpečnostní skener, PHP, Laravel, Bootstrap, OpenVAS, Nessus, databáze, PostgreSQL, CVSS, Common Vulnerability Scoring System

Annotation

This bachelor thesis deals with the examination and processing of reports from automatic vulnerability scanners. In the beginning of the theoretical part there is a brief introduction to scanning. Further are analyzed individual outputs from scanners (reports), description of items. The next part is followed by familiarization with Common Vulnerability Scoring System, which is used in the practical part for evaluation. At the end of the first part are listed some functions of the created application. The beginning of the practical part is devoted to database design and selected framework for creation. The following is an introduction to the functions of the application and the possibility of displaying the results.

Key words

Vulnerability scanner, PHP, Laravel, Bootstrap, OpenVAS, Nessus, databsee, PostgreSQL, CVSS, Common Vulnerability Scoring System

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury. Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne 11.12.2019 podpis autora .....

Poděkování

Rád bych poděkoval panu Ing. Petru Břehovskému za rady a konečné korekce a za čas věnovaný vedení této práce. Také děkuji své rodině za podporu při studiu.

1 Obsah

Úvod

V současné době, kdy počítačové systémy tvoří velkou část našich životů, je potřeba tyto systémy chránit před napadnutím, aby nedošlo k nedostupnosti služeb, či ztrátě cenných dat. Abychom měli přehled o zařízeních v síti, byly vytvořeny automatické bezpečnostní scannery (např. OpenVAS, Nessus Professional™ Vulnerability Scanner), které dělají automatické kontroly zařízení v síti. To znamená, že uživatel nemusí hledat, zda spuštěné verze služeb obsahují chyby a zranitelnosti. Pouze si připraví ve výše zmiňovaných skenerech úlohu, kterou stačí pouštět. V úloze může být například celá síť nebo jen několik zařízení. Uživatel si jen počká na výsledek (report). Tyto reporty jsou v některých případech velmi obsáhlé a nachází se v nich navíc i informace o verzích, otevřených portech a mnohé další.

Pro posuzování vážnosti zranitelnosti se používá Common Valnerability Scoring System, zkráceně CVSS. Jedná se o standardizovaný otevřený systém, který řeší problém toho, že každý systém může posuzovat zranitelnosti jinak než druhý. Výpočet se skládá z tří složek. Základní (base), dočasné (temporal) a prostředí (environmental). První základní skupina je povinná, ale ostatní ne. Většina scannerů nemá možnost nastavit tyto další skupiny. Proto mohou být výpočty nepřesné v dané síti.

Cíle práce

Analýza exportovaných dat (reportů) z automatických bezpečnostních scannerů. Navrhnout a implementovat databázi určenou pro import a zpracování reportů. Vytvořit jednoduché uživatelské rozhraní, pro úpravy dat v databázi a hodnocení dle „Common Vulnerability Scoring System“ (CVSS). Vytvořit systém pro grafickou reprezentaci zpracovaných dat.

2 Metodika

2.1 Použitý software

- Webový server Apache 2.4 s PHP verze 7.2
- Databázový server PostgreSQL 11 + pgAdmin 4
- PHP framework Laravel
    - Doplněk Laravel-Excel (od Maatwebsite)
- Knihovna Bootstrap 3.3.7
- Vývojové prostředí PhpStorm 2018.3 a 2019.11
- Skener OpenVAS 8
- Skener Nessus Home 9 – omezená verze Nessus Professional™ Vulnerability Scanner
- Operační systém Ubuntu 18.04

3 Seznámení se skenováním

3.1 Proč skenovat naši síť?

Záleží na úhlu pohledu. Pro běžného domácího uživatele nemá toto skenování příliš význam provádět. Důvodů je hned několik. Běžný uživatel neprovozuje doma žádný server, který je dostupný z internetu a všechny zařízení používá pouze pro domácí užití. Dle mého názoru by velká většina uživatelů reportům nerozuměla.

U velkých společností, které provozují desítky serverů a jiných zařízení, kde se uchovávají citlivá data, jsou tyto zařízení důležité pro provoz. Zde se tyto skeny opravdu vyplatí a jsou velmi důležitým zdrojem informací pro zaměstnance, kteří mají na starosti bezpečnost.

3.2 Dostupné nástroje pro skenování

Pro skenování existuje velmi velké množství scannerů. Některé jsou zdarma, jiné placené (popřípadě je zdarma trial verze).

- Nexpose – komerční nástroj, dostupná roční licence zdarma (komunitní), síťové skenování [1]
- Qualys – komerční nástroj, kompletní management, komunitní verze zdarma [2]
- OpenScap – Open Source [3]
- OpenVAS – General Public License – více účelový skener [4]
- Nessus – komerční víceúčelový nástroj, do 16 IP verze zdarma [5]

Většina dostupných skenerů cílí na velké společnosti a ceny se pohybují v řádech tisíců dolarů. Například Nessus ve verzi „Professional“ stojí 2190 dolarů za rok. [6]

3.3 Jak funguje skenování

Na rozdíl od penetračního testování zde nedochází k praktickému zneužívání zranitelností. Pouze se na tyto zranitelnosti poukáže, že tato provozovaná verze SW může obsahovat zranitelnost.

V případě skenování jde o detekci softwaru. Na základě detekované verze SW dojde k přiřazení známých zranitelností této verze. K reálnému zneužití možné zranitelnosti nedochází.

Tyto informace skener získává z pluginů, které má uložené. Jelikož nedochází k zneužití, musí zde být uživatel (operátor), který report projde a popřípadě může otestovat ručně zneužití. Na základě tohoto otestování může operátor dojít k závěru, že se jedná o planý poplach (falsePositive).

Osobně jsem byl velmi překvapen při analýze reportů z domácí sítě. Nečekal jsem, že report dokáže poskytnout tolik informací. Například to, že scanner zjistil verzi operačního systému s přesným sestavením daného OS.

Pluginy jsou psané v jazyku nasl – Nessus Attack Scripting Language. Je velmi podobný jazyku C. Pro OpenVAS se používá stejný jazyk. OpenVAS je odnož od Nessusu, od doby, kdy se Nessus stal komerčním nástrojem. V roce 2005 uzavřel své zdrojové kódy a přestal se šířit pod GNU licencí). [7]

Součástí pluginu je hlavička s autorem a licencí, dále informace o pluginu (například OID, CVSS vektor a pod) a výpis informací, který se má vložit do reportu.

Vždy je třeba mít na paměti, že konečné rozhodnutí má člověk. A člověk je nedílnou součástí provádění skenování. Musí rozhodnout, zda zachycené výsledky jsou opravdový problém nebo pouze planý poplach. Pro tuto situaci je mnou vytvořená aplikace vybavena položkou falsePositive, kde má uživatel (operátor) možnost tuto skutečnost zaznamenat a daný řádek pro další zpracování ignorovat.

```
if(!infos = get_app_version_and_location(cpe:CPE, port:hport, exit_no_version:TRUE)) exit(0);
vers = infos['version'];
path = infos['location'];

affected = make_list("2.4.17", "2.4.18", "2.4.20", "2.4.23", "2.4.25", "2.4.26", "2.4.27", "2.4.28", "2.4.29");

if(version_in_range(version:vers, test_version:"2.4.17", test_version2:"2.4.29"))
{
  foreach version (affected)
  {
    if(vers == version)
    {
      report = report_fixed_ver(installed_version:vers, fixed_version:"2.4.30" , install_path:path);
      security_message(port:hport, data:report);
      exit(0);
    }
  }
}
exit(0);
```

Obr. 1 - ukázka kousku kódu pluginu gb_apache_httpd_dos_vuln_apr18_lin.nasl [8]

Na úryvku pluginu výše je možné vidět list zranitelných verzí webového serveru Apache. Pokud je některá z uvedených verzí detekována, záznam s IP adresou zařízení a portu je přidán do reportu.

Délka skenování závisí na velkosti skenované sítě. Malá domácí síť byla skenována cca 30 minut, ale u velkých sítí se může čas vyšplhat až na jednotky hodin. Dále záleží na intenzitě skenování. Při zadávání úlohy lze tyto parametry nastavit.

Obecně ale tyto skenery pomáhají najít a opravit některé zranitelnosti. Ale nikdy nenahradí znalého testera, který má více zkušeností a větší nadhled než program.[9]

3.4 Nevýhody testování

Jednou z nevýhod je, že provedený směn může být již za pár hodin neaktuální. Stačí na některém ze zařízení provést změnu v nastavení, která by mohla spustit napadnutelnou službu. Dokonce i provedení aktualizace vede k neaktuálnosti reportu. Dále se mohou objevovat nové zranitelnosti (záleží na rychlosti aktualizace pluginů scannerů).

I samotný scanner může „zastarat“ pokud nebude aktualizován (například nebude připojen k internetu). Poté budou reporty sice aktuální, ale nebude v nich brán ohled na nové zranitelnosti, protože scanner „zná pouze to, co zná“.

3.5 Reporty

Reporty z obou scannerů obsahují velmi podobné sloupce (položky). Často se jinak jmenují, ale mohou nést tu samou informaci. Níže jsou popsané reporty v základní konfiguraci, bez žádných modifikací výstupu.

3.5.1 OpenVAS

- **IP** – IP adresa skenovaného hosta
- **Hostname** – Hostname skenovaného hosta, ve většině případech prázdné
- **Port** – Port, na kterém byla informace zjištěna, může být i prázdný u informativních hlášek
- **Port Protocol** – Rozšiřující informace, zda zjištěný port byl TCP nebo UDP
- **CVSS** – Hodnota CVSS, která byla dosazena z informací pluginu, jedná se o verzi 2

- **Severity** – Závažnost – je založena na CVSS, ale je aplikován i na CVSS neohodnocené problémy, tyto informace jsou dle klasifikace NVD – může nabývat hodnot none, low, medium a high
- **Solution Type** – Navrhované řešení pro danou zranitelnost – hodnoty: Workaround (je dostupné dočasné řešení), Mitigation (zmírnění následků – jsou dostupné informace pro zlepšení konfigurace, ale nedojde k úplnému odstranění), Velndor-Fix (oficiální řešení od výrobce je dostupné), None-Available (žádné řešení není k dispozici), WillNotFix (nepočítá se s tím, že by problém byl v budoucnosti vyřešen, například se jedná o služby/zařízení, které již nejsou podporovány)
- **NVT Name** – Stručný popis problému
- **Summary** – Rozšířenější popis problému
- **Specific Result** – Výstup pluginu, například detekované verze a další informace popisující problém
- **NVT OID** – Jednoznačné ID pluginu
- **CVEs** – Pokud se jedná o známý problém, který je v databázi CVE jsou zde vypsány jeho čísla
- **Task ID** a **Name** – Vygenerované ID a název úlohy ke které report patří
- **Timestamp** – Časové razítko nalezení problému
- **Result ID** – Jedinečné ID záznamu (řádku)
- **Impact** – Stručný popis možného dopadu při zneužití
- **Solution** – Stručný návod, jak odstranit problém (popřípadě doplněný odkazem)
- **Affected Software/OS** – Seznam ostatních verzí, kterých se týká stejný problém
- **Vulnerability Insight** – Stručný popis toho, jak zranitelnost funguje

- **Vulnerability Detection Method** – Jak byla zranitelnost zjištěna – může se stát, že pro zjištění byl využit i jiný plugin než ten, který dělá přímý výstup, zde je jeho jméno a OID
- **Product Detection Result** – Informace o tom, jak byla zjištěna verze SW, jako u předchozí položky je zde OID a metoda zjištění
- **BIDs** – K této položce se nepodařilo dohledat žádné informace
- **CERTs** – K této položce se nepodařilo dohledat žádné informace
- **Other References** – Odkazy na stránky s danou problematikou zranitelností nebo možností náprav

3.5.2 Nessus

- **Plugin ID** – Unikátní ID pluginu, který byl použit
- **CVE** – Pokud se jedná o známý problém, který je v databázi CVE, je zde vypsáno jeho číslo. Vždy jen jedna hodnota, nebo prázdné pole.
- **CVSS** – Hodnota CVSS dosazená z pluginu, dle provedené analýzy se jedná o verzi 2
- **Risk** – Podobně jako u Severity v reportu OpenVAS se jedná o hodnoty z NVD pro CVSS 2, ale v Nessusu jsou trochu upravené a všechny problémy s CVSS hodnotou 10.0 jsou označeny jako „Critical“. Při analýze byla zjištěna skutečnost, že se používá pouze základní část CVSS.
- **Host** – IP adresa vzdáleného hosta
- **Protocol** a **Port** – Stejné jako u OpenVAS
- **Plugin Output –** Výstup pluginu obsahuje informace o vyhodnocení, popřípadě výstup informací, který skener získal. Například banner z SSH.
- **Name, Synopsis, Description, Solution, See Also –** Obsahují podobné položky jako u reportu z OpenVAS, jen mají jiné názvy

Jak je možné vidět, reporty jsou si velmi podobné. Osobně si myslím, že report z OpenVAS je až moc podrobný a obsahuje zbytečné sloupce, které nenesou žádné užitečné informace pro

uživatele. Například Task Name je naprosto zbytečný sloupec. Oproti tomu Nessus má krásné řešení, kdy je název úlohy zakomponován do názvu exportovaného souboru.

4 Common Vulnerability Scoring System

4.1 Co to je a proč existuje CVSS

CVSS je otevřený koncept ke zjišťování charakteru a závažnosti zranitelností. Skládá se ze tří metrik (base, temporal, environmental). Metrika Base reprezentuje vnitřní vlastnosti zranitelnosti. Metrika temporal reprezentuje vlastnosti, které se mohou časem měnit, například jestli je pro danou zranitelnost dostupný exploit. Poslední environmental metrika odráží prostředí uživatele.

Výhodou CVSS je jeho otevřenost (je naprosto transparentní) a standardizovanost. Díky možnosti zasáhnout do výpočtu dočasnou a enviromentální metrikou, nám může pomoci upřednostnit rizika (změnit pohled na dané riziko).

CVSS může být velmi užitečný ukazatel rizik, aby byla účinnost co nejlepší, je nutné přizpůsobit ohodnocení přímo nám nebo naší organizaci. Tedy počítat s enviromentálním hodnocením. Bohužel toto mnoho organizací nedělá. Je nutné prioritizovat kritická aktiva, které jsou základem organizace. Kontext je vždy důležitý, i zranitelnosti, které nejsou na první pohled ohodnoceny jako kritické či vysoké mohou být zneužity k ničivému dopadu na kritická aktiva. Z druhé strany některé výsledky mohou být označeny jako kritické, ale mohou vyžadovat vektor útoku, který v daném prostředí není možný. [10]

Výsledné skóre je v rozmezí od 0.0 do 10.0 Lze převést do textové podoby závažnosti dle tabulky uvedené v dokumentaci.

CVSS 2 ohodnocení	CVSS 3 ohodnocení	Slovní ohodnocení
	0.0	None
0.0–3.9	0.1–3.9	Low
4.0–6.9	4.0–6.9	Medium
7.0–10.0	7.0–8.9	High
	9.0–10.0	Critical

Tabulka 1 - převod CVSS skóre na textové hodnocení závažnosti (pro verzi 3 i 2) [11]

4.1.1 Verze CVSS

CVSS je v tuto chvíli ve verzi 3.0, ale je možné se setkat se starší verzí 2.0. Jedna ze změn je změna položek v metrikách. Rozdíly jsou, jak v některých položkách, tak i v samotném výpočtu. Podrobnější informace s přesným popisem jsou dostupné v dokumentaci a uživatelském návodu. Zde je dostupný i přesný vzorec včetně hodnot pro výpočet.

2.0	3.0
Vulnerabilities are scored relative to the overall impact to the host platform.	Vulnerabilities now scored relative to the impact to the impacted component.
No awareness of situations in which a vulnerability in one application impacted other applications on the same system	A new metric, Scope, now accommodates vulnerabilities where the thing suffering the impact (the impacted component) is different from the thing that is vulnerable (the vulnerable component).
Access Vector may conflate attacks that require local system access and physical hardware attacks.	Local and Physical values are now separated in the Attack Vector metric.
In some cases, Access Complexity conflated system configuration and user interaction.	This metric has been separated into Attack Complexity (accounting for system complexity), and User Interaction (accounting for user involvement in a successful attack).
In practice, the Authentication metric scores were biased toward two of three possible outcomes, and not effectively capturing the intended aspect of a vulnerability.	A new metric, Privileges Required, replaces Authentication, and now reflects the greatest privileges required by an attacker, rather than the number of times the attacker must authenticate.
Impact metrics reflected percentage of impact caused to a vulnerable application.	Impact metric values now reflect the degree of impact, and are renamed to None, Low and High.
The Environmental metrics of Target Distribution and Collateral Damage potential were not found to be useful.	Target Distribution and Collateral Damage potential have been replaced with Mitigating Factors.
CVSS v2.0 could not accommodate scoring multiple vulnerabilities used in the same attack.	While not a formal metric, guidance on scoring multiple vulnerabilities is provided with Vulnerability Chaining.
No formal qualitative scoring guidelines were provided.	Numerical ranges have been mapped to a 5-point qualitative rating scale

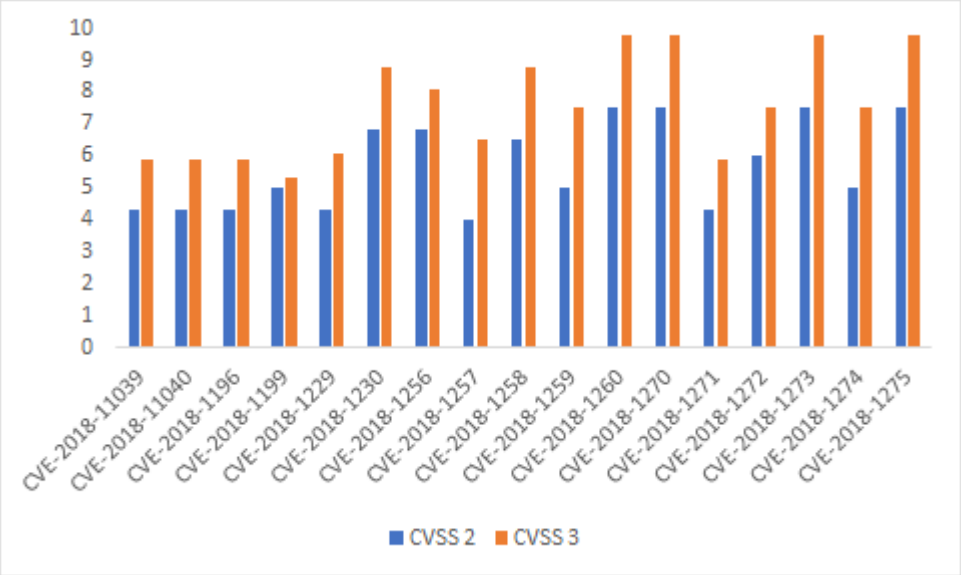
Tabulka 2 - seznam změn mezi verzí 3 a 2 [12]

V této tabulce je většina změn popsána a také vysvětlena. Z důvodu možnosti špatného překladu je zde v originálním znění. [12]

Některé metriky byly pouze přejmenovány, ale některé byly nahrazeny jinými, aby lépe popisovali prostředí nebo zranitelnost.

Dále bylo vytvořeno nové slovní ohodnocení (převod). Tato tabulka je v předchozí kapitole.

Největší změnou bylo zvýšení ohodnocení. To zapříčiňuje vyšší skóre než předtím. V jednoduchosti skóre 7.6 při použití CVSS 2 má při použití CVSS 3 skóre 9.8.



Obr. 2 - porovnání CVSS 2 a CVSS 3 [13]

Na obrázku lze vidět porovnání ohodnocení z obou verzí. Obecně verze 3 vede k rychlejšímu a znatelnějšímu poukázání na problém, než u verze 2. [13]

Vždy je nutné mít na paměti, že bodové ohodnocení CVSS 2 a 3 nejsou ve většině případů porovnatelné.

Dle mého názoru největší změnu zaznamenala část enviromentální. Metriky Target Distribution a Collateeal Damage potential byly nahrazeny novými metrikami, které dokážou lépe popsat prostředí.

Protože ve verzi 2 nejsou v enviromentální části položky „modifikované“ má uživatel možnost měnit tyto položky přímo v základní části (Base). Tento problém byl vyřešen ve verzi 3.

Zde bych rád vysvětlil, proč aplikace u skeneru Nessus počítá CVSS verzi 3 i verzi 2. CVSS 2 je již relativně zastaralé. CVSS 3 bylo uvolněno v červnu 2015. Z pohledu vývojáře není rozumné používat přes 10 let starý koncept, když mohu použít novější a aktuálnější. V aplikaci jsou implementované kalkulačky pro obě verze. V průběhu vytváření aplikace došlo ke zjištění, že pokud budeme získávat z reportů Nessus pouze CVSS 3, přijdeme o cenná data. Více na začátku kapitoly 6.2.2.

4.2 Vektor CVSS

Vektor je složen ze zkratky položky, dvojtečky a hodnoty. Tyto dvojice jsou odděleny lomítkem. Pokud je některá z hodnot nevyplněná, je z vektoru vypuštěna. Pokud by ve vektoru chyběla některá položka ze základní metriky, nebylo by možné provést výpočet. Vzorec pro výpočet není na tuto situaci připraven. U dočasné a enviromentální metriky to nevadí. Vzorec je připravený tak, aby bylo možné výpočet provést. Toto platí pro obě verze CVSS. [14]

Poznatku, jak je vektor zapisován, je využíváno v aplikaci, při získávání hodnot z pluginů.

AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H je příklad vektoru verze 3 se skórem 7.8.

4.3 Hodnoty položek pro výpočet CVSS

Položek pro výpočet je velmi mnoho. Nabízí se otázka: „Kdo vyplňuje, jakou položku?“. Ano, tato otázka je velmi opodstatněná, ale bohužel odpověď není universální a nenalezneme ji ani v oficiální dokumentaci.

Ve verzi 2 a spojení s OpenVAS se dají získat pouze hodnoty pro výpočet základní metriky. Hodnoty dočasných metrik nelze získat z dostupných pluginů. Je tedy na operátorovi (uživateli), aby tyto položky doplnil. Samozřejmě tyto hodnoty nemusí vůbec vyplňovat a výsledek se nezmění. Tato metrika popisuje, jak je momentálně daná zranitelnost využitelná a existuje-li oprava od výrobce apod. Více informací s přesným popisem lze nalézt v oficiální dokumentaci.

Položka	Metrika	Vyplňující
Acces Vector (AV)	Base	Plugin
Acces Complexity (AV)	Base	Plugin
Authentication (Au)	Base	Plugin
Confidentiality Impact (C)	Base	Plugin
Integrity Impact (I)	Base	Plugin
Availability Impact (A)	Base	Plugin
Exploitability (E)	Temporal	Uživatel – operátor
Remediation Level (RL)	Temporal	Uživatel – operátor
Report Confidence (RC)	Temporal	Uživatel – operátor
Collateral Damage Potential (CDP)	Environmental	Uživatel – vlastník aktiva
Target Distribution (TD)	Environmental	Uživatel – vlastník aktiva
Security Requirements – Integrity (IR)	Environmental	Uživatel – vlastník aktiva
Security Requirements – Confidentiality (CR)	Environmental	Uživatel – vlastník aktiva
Security Requirements – Availability (AR)	Environmental	Uživatel – vlastník aktiva

Tabulka 3 - výpis metrik CVSS 2 a vyplňující (převzato a doplněno o poslední sloupec) [15]

V tabulce je možné vidět teoretické vyplňování metrik. Základní metrika je doplněna z pluginu, dočasnou metriku musí doplnit operátor (uživatel) a enviromentální by měl určit vlastník aktiva (uživatel).

V tomto duchu je vytvořená aplikace. S tím rozdílem, že je zanechána operátorovi (uživateli), možnost změnit jakoukoliv hodnotu v daném řádku. V základu se enviromentální hodnoty překopírují z aktiv do tabulky k ostatním hodnotám, které se využívají pro výpočet.

U verze 3 je rozhodování trochu těžší. Je zde více položek a dovoluji si říci, že také hodně záleží na úhlu pohledu. Odpověď na tuto otázku není ani přímo v dokumentaci. Skener Nessus na rozdíl od OpenVAS má u některých pluginů vyplněnou dočasnou metriku.

Položka	Metrika	Vyplňující
Attack Vector (AV)	Base – Exploitability Metrics	Plugin
Attack Complexity (AC)	Base – Exploitability Metrics	Plugin
Privileges Required (PR)	Base – Exploitability Metrics	Plugin
User Interaction (UI)	Base – Exploitability Metrics	Plugin
Scope (S)	Base – Exploitability Metrics	Plugin
Confidentiality Impact (C)	Base – Impact Metrics	Plugin
Integrity Impact (I)	Base – Impact Metrics	Plugin
Availability Impact (A)	Base – Impact Metrics	Plugin
Exploit Code Maturity (E)	Temporal	Uživatel – operátor
Remediation Level (RL)	Temporal	Uživatel – operátor
Report Confidence (RC)	Temporal	Uživatel – operátor
Security Requirements – Integrity (IR)	Environmental	Uživatel – vlastník aktiv
Security Requirements – Confidentiality (CR)	Environmental	Uživatel – vlastník aktiv
Security Requirements – Availability (AR)	Environmental	Uživatel – vlastník aktiv
Modified Base Metric (MAV, MAC, MPR, MUI, MS, MC, MI, MA)	Environmental	Uživatel – operátor -> úpravy

Tabulka 4 - výpis metrik CVSS 3 a vyplňující (převzato a doplněno o poslední sloupec) [16]

Protože se položky dočasné metriky mohou časem měnit, má operátor (uživatel) stále možnost tyto položky také měnit. Položky Security Requirements by měl vyplňovat vlastník aktiv, proto jsou jako v předchozím případě kopírovány z tabulky zařízení (jsou přednastavené pro celé zařízení). Položky Modified Base Metric jsou specifické v tom, že pokud nejsou vyplněné, tak se výsledná hodnota CVSS nezmění. Spíše jsou to položky pro upřesnění prostředí

Například: V základní metrice je Attack Vector (AV) uvedený Network (N), protože test probíhal přes síť. Operátor ale ví, že dané zařízení je mimo internet a je dostupné pouze

z místní sítě – změní zde hodnotu na Adjecent (A). Výsledek pro enviromentální skóre se změní, protože hodnota spadne, Modifed Attack Vector se změní z 0,85 na 0,62.

Podle této tabulky je vytvořena i možnost měnit položky v aplikaci. Pro zjednodušení a umožnění „jiného pohledu“ je ponechána operátorovi (uživateli) možnost měnit již zkopírované enviromentální položky. Další alternativa je tuto možnost zakázat a při výpočtu se pokaždé dotázat databáze na aktuální enviromentální hodnoty daného zařízení.

V reálném nasazení by část zařízení byla napojená na management aktiv, ze kterého by byla data převzata. Zde by se popřípadě jen přidali hodnoty pro výpočet enviromentálního CVSS.

V aplikaci byla část aktiv vytvořena velmi zjednodušeně, pouze pro demonstraci.

5 Návrh databáze a importování dat z reportů

5.1 Výběr databázového serveru a druhu databáze

První otázkou bylo, zda využít relační nebo NoSQL databázi. Velmi dlouho byla zvažována dokumentově orientovaná NoSQL databáze, ale ta se spíše hodí pro větší objem dat, který není strukturovaný. V případě dat z reportů není možno říci, že jsou nestrukturovaná. Množství ukládaných dat není takové množství, aby tento nápor relační databáze nezvládala zpracovat. Také se spíše přikláním k položkám filozofie ACID u relačních databází než k BASE u NoSQL.

Dostupných databázových systémů relačních databází je více. Například MySQL, PostgreSQL, MS SQL Server, Oracle Database a mnoho dalších. Například využití SQLite by bylo asi nejlepší. SQLite je lokální databáze pro malé programy (využívaná hojně v mobilních aplikacích). Ve své aplikaci jsem chtěl mít možnost provozovat databázi mimo lokální PC, například na vzdáleném serveru. Proto toto řešení nebylo zvoleno.

Byl vybrán databázový server PostgreSQL, protože se jedná o databázový server, který je šířený pod licencí PostgreSQL Licence, která je velmi podobná licencím BSD nebo MIT.[17]

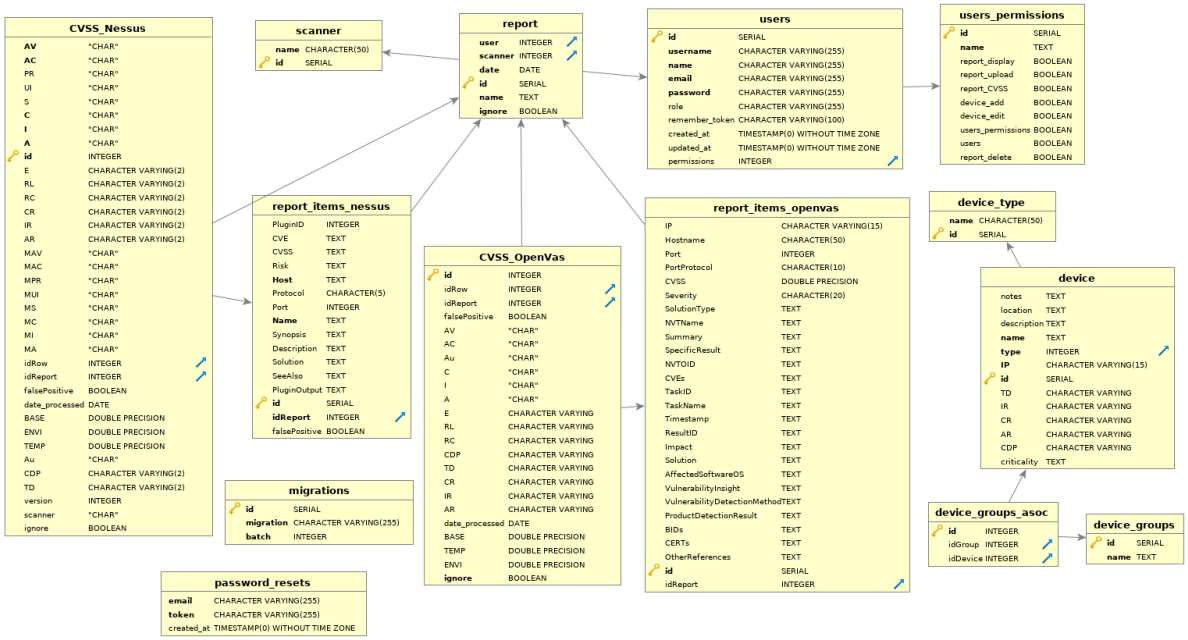
Z praktického hlediska by nebyl problém využít i MySQL, ale tento projekt je vlastněn firmou Oracle. Je zde teoretická možnost, že by se firma Oracle mohla rozhodnout MySQL nevydávat pod open-source licencí. PostgreSQL poskytuje více možností, ale je složitější.

Nasazení aplikace s MySQL serverem by měl být bezproblémový. Ze zkušeností se liší jen ve vyhledávání. Muselo by být upraveno několik dotazů do databáze.

5.2 Návrh databáze a práce s PostgreSQL

Návrh struktury databáze provázelo několik problémů a také dodatečných změn. Například změna typu nebo povinnost.

Prvotní nastavení je trochu složité. V základu je zde jen uživatel postgres bez hesla (tento uživatel je superuživatel pro DB). Z bezpečnostního hlediska je nutné tomuto uživateli vytvořit heslo. Je doporučeno vytvořit ještě jednoho záložního uživatele, který bude mít stejná práva jako uživatel postgres.



Obr. 3 - návrh struktury databáze

Struktura databáze byla logicky rozdělena podle skenerů. Každý skener má 2 tabulky. První tabulka obsahuje importované řádky z reportu a druhá získané hodnoty pro výpočet CVSS. Tyto dvě tabulky nebylo nutné dělit, ale bylo to uděláno z důvodu větší přehlednosti a původně zamýšlených počítaných sloupců. Na použití počítaných sloupců nedošlo a bylo by složité předělávat logiku aplikace. Proto tabulky zůstali rozdělené. Datové typy byly vybrány podle možného obsahu, aby nedošlo k nemožnosti uložení a zároveň vybraný typ nebyl zbytečně velký. Tyto tabulky spojuje společná tabulka „report“, která udržuje informace o názvu nahraného souboru, použitém skeneru, datumu a uživateli, který report nahrál.

Tabulky pro správu aktiv a skupin nejsou přímo spojeny s reporty. Je to z důvodu, že může být spojení nalezeno mezi IP adresy a také nemusí. Tato vazba se zjišťuje až při vyhodnocení v PHP.

Automaticky byly vygenerovány tabulky pro správu uživatelů. Byla změněna pouze tabulka users a vymyšlené jednoduché oprávnění– více v kapitole 7.3 a 7.3.1.

5.3 Využití triggerů a funkcí v databázi

Aplikace umožňuje ignorovat při vyhodnocení celé reporty. Pro tuto funkcionalitu jsou využívané takzvané triggery a funkce.

Po označení reportu, aby byl ignorován, se spustí trigger funkce, která automaticky změní u všech záznamů CVSS z daného reportu položku „ignore“.

Na výběr byly dvě možnosti, první změnu provést v PHP pomocí smyčky, toto řešení by bylo jednodušší na vytvoření, ale méně spolehlivé. Varianta s triggerem a funkcí je spolehlivější, o vše se stará databázový server.

Trigger se automaticky spustí, pokud se provede změna v tabulce s reporty (sloupec ignore). Hodnotu ze sloupce ignore upraví u samotných řádků, kde jsou hodnoty CVSS. Jedná se o redundantní hodnoty, ale aplikace používá více metod vyhodnocení, je tedy nutné změnit tento údaj v reportu, tak u řádků s hodnotami pro výpočet.

5.4 Indexace

Aplikace obsahuje velmi mnoho dat, se kterými se často pracuje. Za předpokladu správného použití indexace je možné dotazy velmi urychlit.

Aplikace indexuje sloupce s IP adresami, jelikož je některé funkce využívají ke spárování zařízení. Jedná se o velmi mnoho dotazů a je potřeba tyto dotazy rychle obsloužit.

6 Import do databáze a získávání vektorů

6.1 Problematika reportů a jejich import

Reporty se dají ze scannerů stahovat ve velkém množství formátů. Například XML, csv, HTML, PDF a další. V případě OpenVAS lze přidat další.

Nejjednodušší pro další zpracování je soubor ve formátu .csv, který lze otevřít například v tabulkovém editoru a popřípadě jej upravit před nahráním do aplikace. Je to jeden z důvodů, proč byl vybrán tento formát.

Samotné importování není moc složité. Soubor se přes grafické rozhraní nahraje a uloží. Tento uložený soubor se přečte doplňkem Excel od Maatwebsite. Tento doplněk je šířen pod MIT licencí, umožňuje import i export). [18] Přes smyčku foreach se postupně řádek po řádku dosadí do předpřipravené šablony a uloží do databáze.

```
$request->cvs_file->storeAs('cvs_openvas', $request->cvs_file-
>getClientOriginalName());

$id =  DB::table('report')->insertGetId(['name' => $filename, 'date' => $date,
'scanner' => "1", 'user' => $user]);

$data = \Excel::load('storage/app/cvs_openvas/'.$filename)->get();

if ($data->count()) {
foreach ($data as $key => $value) {
$arraydata[] = [
'IP' => $value->ip,
'Hostname' => $value->hostname,
.
.
.
'idReport' => $id];
}
if (!empty($arraydata)) {
DB::table('report_items_openvas')->insert($arraydata);
app('\App\Http\Controllers\cvss_openvasController')->getCVSS($id);
$message = "Report ".$filename. " byl nahrán OK";
return view('message')->with('message', $message);
}
```

Ukázka 1 - ukázka zdrojového kódu importování reportu OpenVAS (zkráceno)

Pro každý scanner se musí používat jiná šablona, protože položky reportu nejsou identické, viz kapitola 3.5.

6.2 Získávání hodnot vektorů

6.2.1 OpenVAS

U scanneru OpenVAS lze položky vektoru získat přímo z pluginů, které se používají pro testování. Nejdou získat oficiálních webových stránek OpenVAS, jako to je možné u Nessus (neoficiálně dostupné například na vulners.com).

Při analýze pluginů bylo zjištěno, že neobsahují CVSS vektor verze 3, ale pouze verze 2 (aktuální pluginy z března 2019). Tato informace je velmi důležitá, protože CVSS 2 má jiné položky a jiný výpočet. Nebylo možné využít kalkulátor, který byl již vytvořený pro scanner Nessus (zde byl vektor pro CVSS 2 i 3).

Pluginů je opravdu obrovské množství. Například jen složka za rok 2018 obsahovala 5524 souborů. Při analýze za použití programu grep jsem čekal na výsledky i několik vteřin.

Zde se naskytla otázka, jak samotné získávání vektoru řešit. První varianta je vytvořit další tabulku v DB a do té nahrát informace z pluginů, datum změny pluginu, OID a další potřebné informace. Tabulka by se musela aktualizovat optimálně před nahráním nového reportu, aby data byly aktualizované. Případný skript pro aktualizaci DB by vyžadoval přítomnost scanneru OpenVAS na stroji, kde by aplikace fungovala. Tato metoda by byla rychlejší při vyhledávání a při získávání dat (za použití indexace apod.), ale dle mého názoru by tento krok jen zvýšil režii a vyžadoval by vytvořit další skript pro aktualizaci. V tomto případě by bylo nutné, aby se skript spouštěl periodicky.

Finální rozhodnutí padlo na prostší variantu, která při každém získávání projde všechny pluginy v adresáři.

```
$path = '/var/lib/openvas/plugins';

$hledane = "'script_oid(\"$oid\")'";

$return = exec('grep  -R -F  '.$hledane.' '. $path);

$pathfile = preg_split('/:/', $return, -1, PREG_SPLIT_NO_EMPTY);
```

Ukázka 2 - hledání názvu pluginu s použitím funkce exec() a programu grep

Jak je možné vidět z ukázky zdrojového kódu výše, přes funkci exec() v PHP se na pozadí spustí program grep, který vrátí cestu k souboru, kde byla nalezena shoda. Zde byl nalezen problém, kdy při hledání OID program grep vracel i podobné výsledky. Proto zde muselo být

použito hledání celého řetězce včetně začátku a uvozovek na konci, aby byla nalezena pouze jedna shoda. Pokud není nalezena žádná, řádek reportu je přeskočen.

```
pavel@MalyLenin:/var/lib/openvas/plugins$ grep -R -F  'script_oid("1.3.
6.1.4.1.25623.1.0.80091")'
2008/tcp_timestamps.nasl:  script_oid("1.3.6.1.4.1.25623.1.0.80091");
pavel@MalyLenin:/var/lib/openvas/plugins$ grep -R 1.3.6.1.4.1.25623.1.0
.800
gb_bournal_detect.nasl:  script_oid("1.3.6.1.4.1.25623.1.0.800300");
gb_wireshark_detect_win.nasl:  script_oid("1.3.6.1.4.1.25623.1.0.800038
");
gb_winftp_serv_detect.nasl:  script_oid("1.3.6.1.4.1.25623.1.0.800345")
;
```

Obr. 4 - ukázka správného a špatného použití grep v terminálovém okně

Zde na obrázku výše je ukázka přesné shody OID (za použití přesného řetězce) a také ukázka špatného použití, kdy bylo nalezeno více výsledků.

Po získání cesty k pluginu se obsah pluginu načte přes PHP funkci file_get_contents().

```
$findme = 'cvss_base_vector';
$pos = strpos($filecontent, $findme);
$stringpart = array();
for ($i = 26; $i < 52; $i++) {
    array_push($stringpart, $filecontent[$pos + $i]);
}
$string = implode("", $stringpart);
$vectoritems = preg_split('/\//', $string, -1, PREG_SPLIT_NO_EMPTY);
$size = sizeof($vectoritems);
for ($i = 0; $i <= $size - 1; $i++) {
    $vectoritemtest = array();
    foreach ($vectoritems as $item) {
        array_push($vectoritemtest, preg_split('/\:/', $item, -1,
PREG_SPLIT_NO_EMPTY));
    }
}
```

Ukázka 3 - nalezení vektoru z obsahu pluginu

Za pomocí funkce implode() z tohoto pole vytvoříme řetězec, který za pomocí preg_split() a substr() přesně zkrátíme na vektor, který byl v pluginu. Funkce preg_split() je využita několikrát po sobě v celém kódu.

```
foreach ($vectoritemtest as $key => $value) {
    $polozka = (string)$value[0];

    if (!isset($value[1])) {
        return 0;
    } else {
        $vector->$polozka = $value[1];
    }
}
```

Ukázka 4 - vytvoření vlastností objektu ze získaných položek

Na konci stačí řetězec rozdělit na dvojice (položka + hodnota), přidat ID řádku (ke kterému záznamu z reportu vektor patří) a ID reportu.

Pokud se jedná o známou IP adresu (je uložená v aktivech), jsou zkopírovány hodnoty pro výpočet enviromentální části CVSS.

Výsledný vektor (objekt) se pak už jen uloží do předpřipravené tabulky v databázi.

Tento celý postup je nutné opakovat pro celý report (všechny řádky v reportu). Tyto uložené hodnoty je možné ručně upravit přes uživatelské rozhraní. Respektive jejich modifikované položky, které se používají pro výpočet enviromentálního CVSS skóre.

Tato metoda má jednu nevýhodu a tou je časová náročnost a zbytečné procházení všech pluginů. Při každém hledání grep projde všechny pluginy. Tato smyčka se provádí pro každý řádek reportu.

6.2.2 Nessus

V průběhu programování bylo zjištěno, že některé pluginy neobsahují vzorec CVSS verze 3, ale pouze verze 2. Pokud by aplikace pouze počítala a vyhledala pouze vzorec CVSS 3, přišli bychom o velmi cenná data, z tohoto důvodu byla aplikace předělána tak, že pokud nebude nalezen vzorec CVSS 3, uloží se vzorec CVSS 2 a bude se dále pracovat pouze s ním.

Dle mého názoru je naprosto nepřijatelné, abychom o tyto záznamy přišli.

Report generovaný scannerem Nessus obsahuje velmi mnoho informativních hlášek, které pouze nesou informace a nenesou žádné informace o zranitelnostech. Tyto záznamy se dají rozeznat tím, že mají v položce Risk hodnotu „None“. Tyto řádky jsou hned na začátku zpracování ignorovány, aby zbytečně nezdržovaly prohledávání.

Získání hodnot vektorů je možné dvěma způsoby. Jeden ze způsobů je z pluginů, jako to je v předchozím případě, ale také je zde možnost získávat CVSS vektor z oficiálních stránek.

Druhé řešení není o moc lepší, ale nevyžaduje přítomnost scanneru Nessus na stroji, kde běží aplikace. Vyžaduje pouze funkční internetové připojení. Problém by mohl nastat, pokud se změní stránka s pluginy. V tomto případě by se musel předělat zdrojový kód. Jedná se o elegantní řešení problému, z důvodu toho, že jsou vždy stahovány aktuální informace. Na rozdíl od procházení všech pluginů je tato metoda rychlejší. Bohužel vyžaduje připojení k internetu.

Oficiální stránka s pluginy Nessus umožňuje zobrazit informace o pluginu po dosazení ID pluginu do URL.

Toto řešení není v aplikaci používáno. Požívá se postupné procházení pluginů. Zde je tato metoda uvedena jako další možné řešení.

Na začátku se stáhne celá webová stránka pomocí funkce htmlentities(), vrátí veliké pole znaků (celé stránky). V tomto poli dále pomocí strpos() najdeme hledaný řetězec (v tomto případě „CVSS:3.0“. Protože není zapotřebí pracovat s celou stránkou, v dalším kroku do předem vytvořeného pole uložíme jen část stánky (požadovaná délka + rezerva).

```
$url = "https://www.tenable.com/plugins/nessus/";

$path = $url . $PluginID;
$contents = htmlentities(file_get_contents($path));
$findme = 'CVSS:3.0';
$pos = strpos($contents, $findme);
$stringpart = array();

for ($i = 8; $i < 200; $i++) {
    array_push($stringpart, $contents[$pos + $i]);
}
```

Ukázka 5 - demonstrace možnosti stahovat vektor z oficiálních stránek

Tento postup byl otestován, ale ve finální verzi nebyl využit z důvodu větší časové náročnosti a nutnosti mít funkční internetové připojení. Tento způsob je zde uveden jako demonstrace další možné cesty získávání hodnot vektoru.

6.3 Časová náročnost

Nahrání nového reportu a jeho celkové zpracování může být časově náročnější operace. Vždy záleží na počtu řádků v reportu.

Vybraný způsob získávání není nejrychlejší, protože program grep musí projít obrovské množství souborů. Nechtěl jsem aplikaci dělit, spoléhat na více součástí, a mít další program, například v C++, který by tento krok zpracovával na pozadí za využití vláken. Bohužel PHP pracovat s vlákny neumí. Tato skutečnost je největší kámen úrazu. Při zpracování každého řádku se musí čekat na jeden proces a nelze jich spustit více najednou. Více o této problematice v kapitole 9.

Po úspěšném nahrání reportu automaticky začne vyhledávání hodnot pro výpočet CVSS a poté ihned samotný výpočet.

Příklad: Testovací report OpenVAS

- 1072 řádků
- Velikost 1,2MB
- Doba nahrání + celkové zpracování: 17 minut
- Doba skenování OpenVAS: 2 h
    - Rozsah 172.16.1.0/24
    - Počet unikátních hostů: 46

7 Tvorba uživatelského rozhraní

Vytvořená aplikace zahrnuje část, přes kterou se nahrávají reporty, část, přes kterou se dají upravovat položky pro vyhodnocení CVSS. Dále obsahuje jednoduchou správu zařízení (aktiv) a uživatelů. Po nahrání a automatickém zpracování reportů má uživatel možnost nahlížet na samotný report a také na ohodnocení CVSS nahraného repotu jednotlivě nebo dalšími způsoby. Tyto způsoby jsou popsané níže v kapitole 7.6.

Jednoduchý návod je přiložen v příloze číslo 2.

Rozhodnutí o vytvoření webové aplikace bylo uděláno z několika důvodů. V případě, že aplikace bude běžet na lokálním počítači, potřebuje k běhu pouze webový a databázový server a přítomnost pluginů z OpenVAS, popřípadě Nessus. V případě, že bude aplikace na vzdáleném PC, nevyžaduje od uživatelského PC vůbec nic. Vše běží na vzdáleném zařízení a uživatel potřebuje jen své přihlašovací jméno a heslo.

V dnešní době se již jednoduché PHP „nenosí“, programátoři jsou zhýčkaní a nechtějí pokaždé vymýšlet vše od začátku. Proto existují frameworky, které usnadňují programování. Výběr frameworků je opravdu veliký. Byl vybrán framework Laravel, který je šířen pod svobodnou licencí MIT.[19] Dlouhou dobu je tento framework považován za nejlepší PHP framework.[20]

Jediná nevýhoda, která byla zaznamenána, byla složitost zprovoznění celého frameworku a pochopení jeho struktury. Z druhé strany tento framework poskytuje nepřeberné množství již hotových funkcí a celý projekt je vyvíjen v architektuře MVC (Model-View-Controller).

Pro vyvíjení aplikace za pomocí PHP a Laravelu je zapotřebí mít v počítači předpřipravené prostředí. Pokud bereme v potaz programování na lokálním počítači, je nutné mít webový server s nainstalovaným PHP a dále mít nainstalované a aktivované správné doplňky. Také již zmíněný databázový server (PostgreSQL). Pro Laravel je potřeba správce knihoven a dalších zdrojů v PHP jménem Composer.[21] Vývojové prostředí je na zvážení každého.

Při zakládání projektu Composer zařídí stažení a vytvoření Laravel projektu za nás. Může nastat problém s právy k souborům, ale to se dá vyřešit změnou vlastníka a změnou práv k složce, kde se projekt nachází.

7.1 Připojení k databázi z Laravelu

Framework Laravel obsahuje předpřipravené šablony pro práci s různými databázemi, aby programátor nemusel zasahovat přímo do cizích zdrojových kódů, v tomto případě konektoru k databázi, je v Laravel projektu vytvořen soubor „.env“. V tomto souboru se nachází proměnné, které se poté berou do ostatních částí aplikace. Programátor není nucen dělat tyto změny ručně, pouze si změní/přidá záznamy v tomto souboru.

```
APP_NAME=Analýza reportů
APP_ENV=local
APP_KEY=base64:S9gUKv7V+9uWcFnNV3S3HLPMQth010W2fLMz9I0poDY=
APP_DEBUG=true
APP_LOG_LEVEL=debug
APP_URL=http://localhost/BP-program1/public

DB_CONNECTION=pgsql
DB_HOST=127.0.0.1
DB_PORT=5433
DB_DATABASE=BP_program
DB_USERNAME=postgres
DB_PASSWORD=StrasneDlouhoHeslo
```

Ukázka 6 - část konfiguračního souboru .env

Po změně tohoto souboru se doporučuje provést z terminálu příkaz „php artisan config:clean“, který znovu načte konfigurace projektu, včetně souboru „.env“. V případě chyby upozorní na problém. Například v ukázce výše chybí uvozovky okolo názvu projektu. Celý projekt přestane fungovat, ale po provedení příkazu, se zobrazí v terminálu hláška, která popíše chybu.

7.2 Layout a Bootstrap

Vytvořit layout lze několika způsoby. Například ručně nebo využít již hotové styly. Rozhodnutí bylo jasné – využití již předpřipravených CSS stylů v Laravelu, které vychází z Bootstrap. Důvodů je hned několik. Časová náročnost vytváření svých vlastních a rozdíly v kvalitě. Mnoho velkých projektů využívá Bootstrap knihovnu, například Best Buy (prodejce elektroniky v USA) a MBA.com (basketbalová liga USA). Zjištěno náhledem do zdrojového kódu v přihlížeči.

Bootstrap je open-source knihovna pro vytváření projektů v HTML, CSS a dalších. Obrovskou výhodou je, že programátor teoreticky nemusí nic vytvářet, jen použije hotové styly a při správném použití dostane pěkně vypadající stránku, která je i responzivní. Projekt Bootstrap je také vydáván pod licencí MIT.[22]

V projektu Laravel je již obsažen. Pokud ne, lze přidat.

7.2.1 Problémy s Bootstrap

Jako veliký problém se ukázal rozdíl ve verzích. V současné době je verze 4.1.0. V projektu jsem se rozhodl používat starší verzi 3.3.7. Dokumentace pro tuto verzi je přehlednější a problémy jsou na internetových diskuzích více rozebrány.

Některé styly jsou stejné, ale některé jsou naprosto jiné. Popřípadě ve starší verzi vůbec neexistují. Dále může nastat problém s JavaScript knihovnami, ale je možnost je ručně přidat.

K této problematice je nejlepší zdroj oficiální dokumentace, ve které jsou příklady využití včetně zdrojových kódů a pokud něco nefunguje, komunita okolo projektu Bootstrap je opravdu velká a lze dohledat řešení snad ke každému problému.

7.3 Uživatelé

Aplikace obsahuje velmi citlivá data, které je nutné chránit. Nechceme, aby si kdokoli mohl přečíst jakými zranitelnostmi trpí naše síť, jaké rozsahy IP adres se používají apod. Tato aplikace by se mohla stát velmi užitečnou studnicí informací pro případného útočníka.

Framework Laravel disponuje součástí, která vše zařídí za programátora. Vytvoří tabulku s uživateli, vytvoří cesty pro přihlašovací a registrační formulář, prostě vše potřebné. Není potřeba vymýšlet celý mechanismus. Hesla nejsou ukládána do databáze z čistém textu.

Autentizace v Laravelu je dělaná tak, aby přihlašování probíhalo pomocí emailu a hesla. To není v tomto případu nutné a byla zde provedena změna. Tabulka byla vygenerována, bylo do ní přidáno několik sloupců. Například přihlašovací jméno a oprávnění.

Oproti základu bylo ještě změněno několik zásad, ale mechanismus zůstal stejný. Například, že registraci nového uživatele může provést jen přihlášený uživatel s příslušným oprávněním, po registraci automaticky neproběhne přihlášení na nového uživatele.

Ručně byla dodělána jen jednoduchá správa uživatelů, formulář přihlášení a registrace uživatelů byl jen upraven.

7.3.1 Oprávnění

Laravel paradoxně oprávnění a role neumí. Existuje mnoho doplňků, které Laravel o tuto funkcionalitu rozšíří. [23] Tyto doplňky jsou zbytečně rozsáhlé, proto žádný z těchto doplňků nebyl využit. Byla vytvořena alternativní cesta, ve speciální tabulce aplikace udržuje skupiny

a jejich oprávnění. Při načítání stránky zkontroluje, o jakého jde uživatele, do jaké skupiny patří a na základě odpovědi od databáze (typ boolean) se rozhodne, zda má, či nemá oprávnění.

Další možnost by byla mít oprávnění u každého uživatele a ručně mu je přidávat, popřípadě odebírat. Dovoluji si říct, že je jednodušší správa pro celé skupiny než pro jednotlivce.

Již výše bylo popisováno několik typů uživatelů, pro které je aplikace připravená. Pro tyto typy jsou vymyšlené oprávnění dle předchozího textu.

Typ uživatele	Vlastník aktiv	Operátor	Nahlížeč	FULL
Zobrazení reportů	X	X	X	X
Nahrávání reportů		X		X
Úprava CVSS v reportech		X		X
Přidávání zařízení	X			X
Úprava zařízení vč. CVSS	X			X
Mazání reportů		X		X
Změna oprávnění				X
Správa uživatelů				X

Tabulka 5 -návrh skupin oprávnění

Grafické rozhraní pro správu oprávnění nebylo vytvořeno. Změny je potřeba provést ručně přímo v databázi, respektive v tabulce users_permissions. Například přes grafické administrační rozhraní pgAdmin.

7.4 Funkce označení řádku falsePositive

Skenování není bezchybné a je potřeba rozlišit řádky, které jsou planý poplach a nejsou pro nás hrozbou. Například pokud operátor (uživatel) ověřil, že se daná zranitelnost na zařízení nevyskytuje, popřípadě již byly postoupeny kroky k nápravě. Takový řádek by zbytečně ovlivňoval výsledky reportů.

Pro tento případ je každý řádek s hodnoty CVSS opatřen dalším atributem „falsePositive“, který při zobrazení výsledků řádek zneviditelní při vyhodnocení nejhorších hodnot.

7.5 Ignorování vybraných nahraných reportů

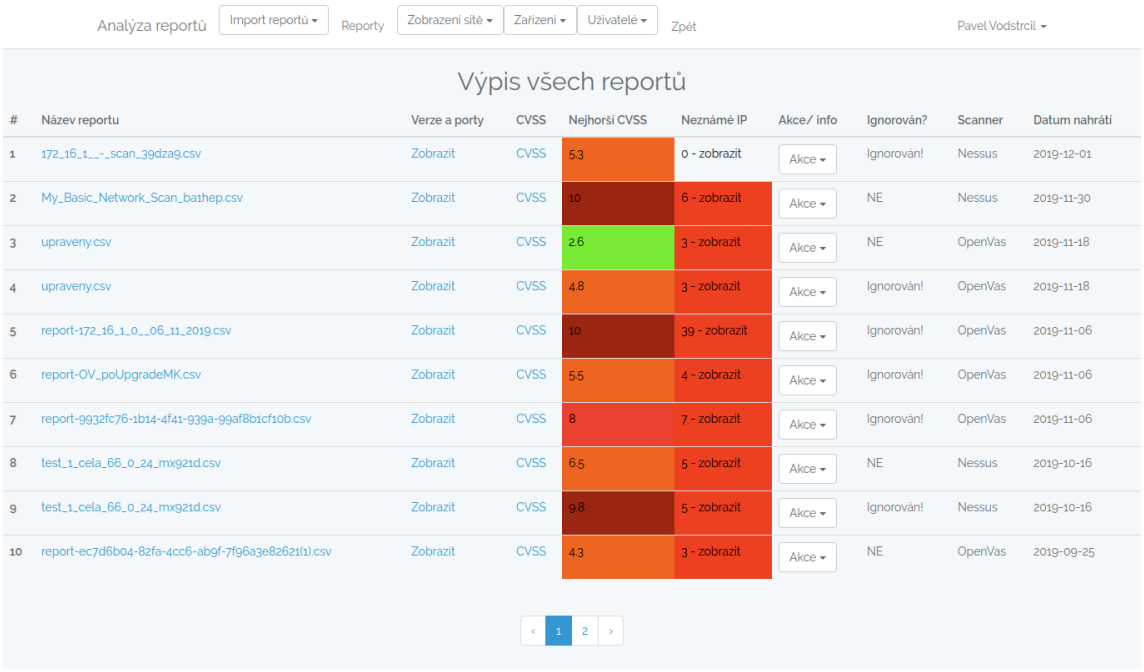
Tato funkce umožňuje uživateli (operátorovi) vynechat při vyhodnocení vybrané reporty bez toho, aby musely být smazány.

Výhodou této funkce je uchování starších reportů, které jsou neaktuální, ale mohou být zpětně využity pro dohledání starších informací.

Technicky funguje tato funkce jako označení řádku falsePositive, ale pro celý report.

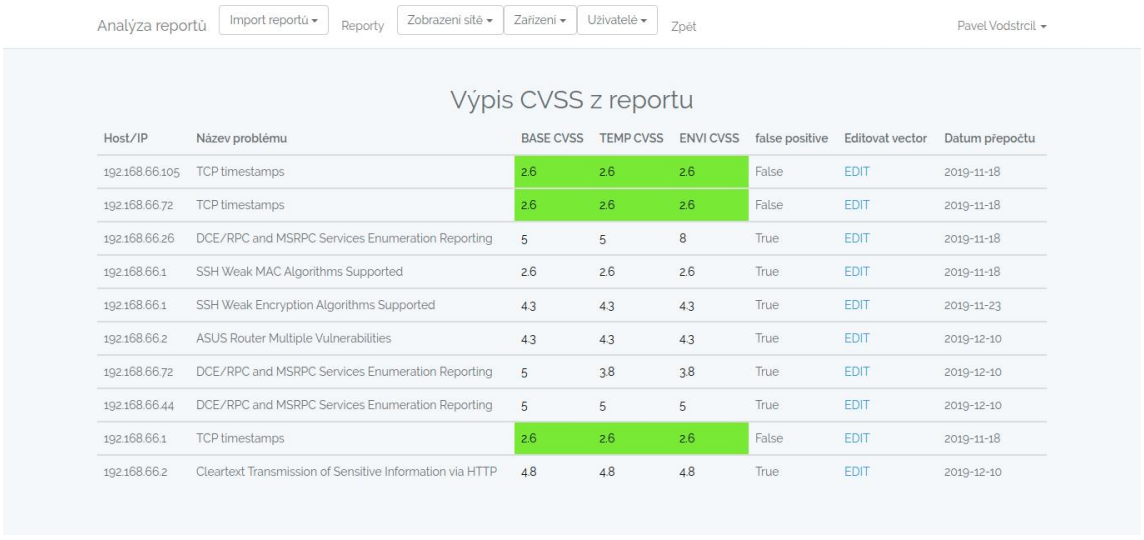
7.6 Zobrazení nahraných reportů

Po nahrání má uživatel možnost přes rozhraní aplikace vidět všechny nahrané reporty, jejich stav a nejhorší CVSS ohodnocení. Po rozkliknutí je možné prohlížet jejich obsah.



Obr. 5 - výpis nahraných reportů v aplikaci

Dále je uživatel upozorněn na neuložené IP adresy. Po kliknutí na CVSS jsou uživateli zobrazeny výsledky CVSS za daný report.



Obr. 6 - zobrazení ohodnocení CVSS pro jeden vybraný report

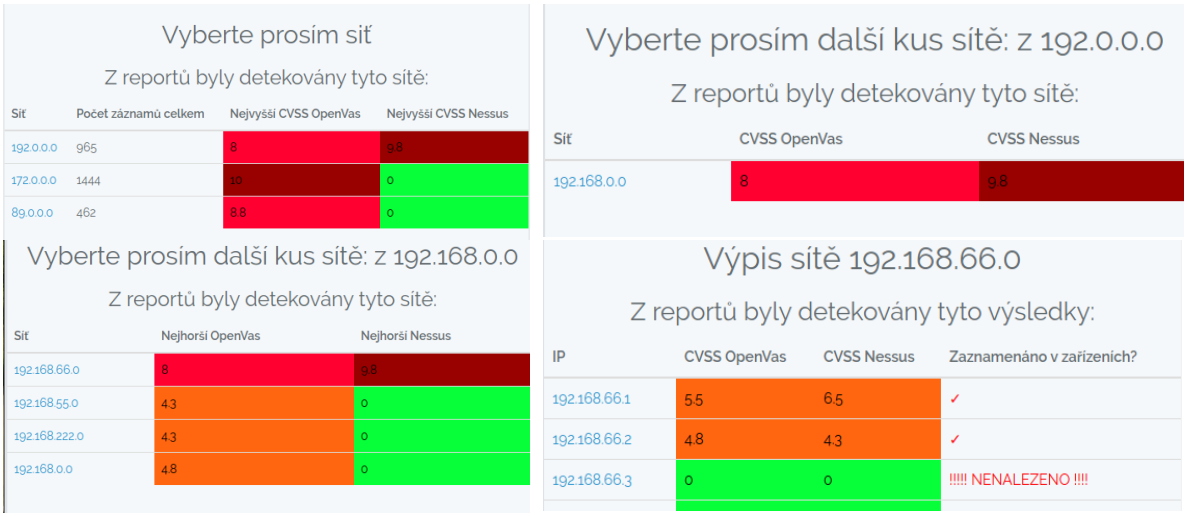
7.7 Grafické znázornění výsledků

Jak nahlížet na zobrazení výsledků? V jednom reportu můžou být výsledky celé sítě, ale i jen pár zařízení, pokud bychom znázornili jen jeden report, je to málo. Pokud všechny, je to zase moc informací. Proto bylo vytvořeno několik pohledů na výsledky.

Všechny pohledy mají společné to, že pokud je report označený jako ignorovaný, tak se nezobrazuje v žádných výsledcích.

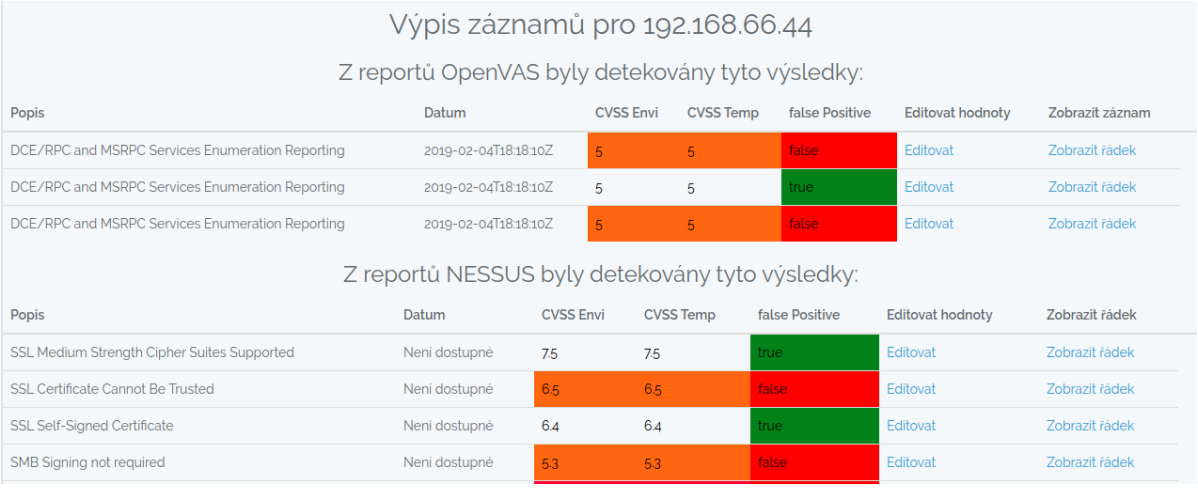
7.7.1 Zobrazení celé sítě – „po oktetech“

Tento pohled dovolí uživateli vidět celou síť po částech IP adresy. Od prvního oktetu IP adresy se prokliká až na jedno dané zařízení, vedle záznamu je vždy vidět nejvyšší CVSS ze skeneru OpenVAS i Nessus. Nejvyšší z toho důvodu, aby tento výsledek přitáhl pozornost uživatele a byl nucen klikat dále.



Obr. 7 - ukázka postupného prohlížení výsledků "po oktetech"

U předposledního kroku je uživatel upozorněn, zda je daná IP adresa uložená v zařízeních, aby mohl reagovat i na tuto skutečnost.



Obr. 8 - zobrazení výsledků pro danou IP adresu – poslední krok

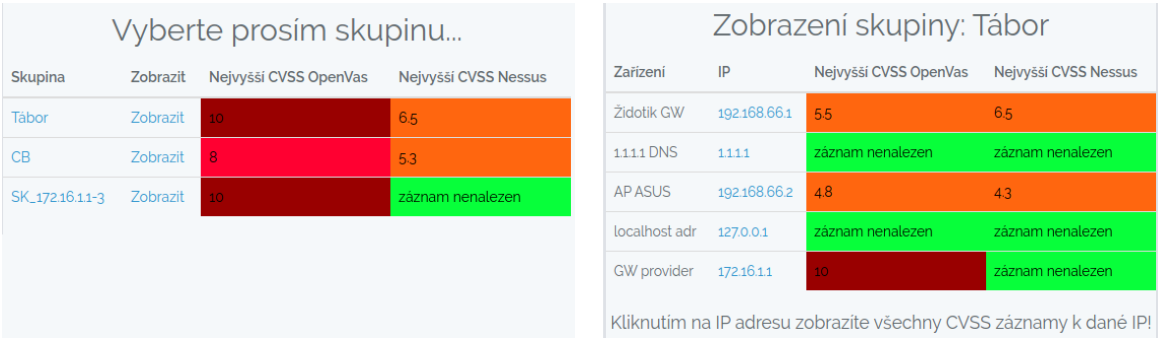
Jako poslední krok vidí uživatel výpis všech záznamů pro danou IP, může zde provést editaci záznamu (označení jako falsePositive, úprava CVSS), popřípadě si zobrazit celý obsah řádku z reportu.

Tento pohled umožňuje prozkoumat postupně všechny záznamy. Nevýhoda této metody je delší zpracování v prvních krocích, kdy se musí projít všechny záznamy.

7.7.2 Zobrazení dle skupin

I v dnešních domácnostech se najde velmi mnoho zařízení, které jsou připojená do sítě. V malé síti, jako je ta domácí, je jednoduché vědět jaká IP adresa náleží jakému zařízení, ve větší síti už ne. Z tohoto důvodu je aplikace vybavená možností zobrazovat data po skupinách.

Tato funkce umožňuje operátorovi rozdělit si zařízení do skupin, dle kterých lze poté zobrazovat profilované výsledky (CVSS) za celou skupinu. Pouze výpis za typ zařízení nestačí. Jako příklad typ zařízení „SERVER Linux“. Je možné si zobrazit všechny zařízení, které jsme označili jako Linuxový server, ale můžeme mít v síti server, který se stará o evidenci místní kantýny a není pro nás tak důležitý, jako databázový server s platbami zákazníků. Proto je lepší mít skupinu „SERVERY – prio 1“, kde budou pouze pro nás důležité systémy, aniž bychom museli dále rozlišovat.



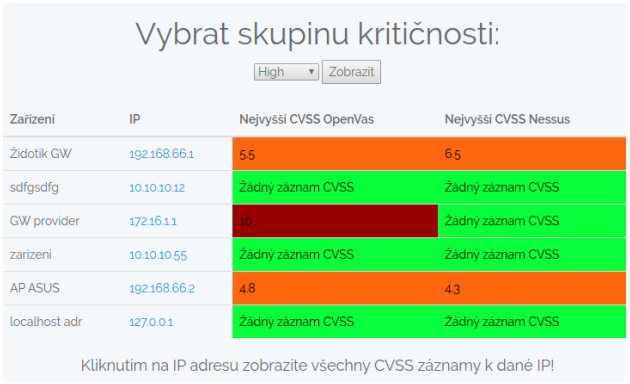
Obr. 9 - zobrazení skupin – krok 1 a 2

Pokud uživatel rozklikne IP adresu zařízení, bude přesměrován na obrazovku s výpisem všech CVSS hodnot pro danou IP, jako je možné vidět na obrázku 8.

Samozřejmě byla vytvořena i jednoduchá správa skupin, kde si uživatel může definovat nové skupiny, přidávat, odstraňovat zařízení apod.

7.7.3 Zobrazení dle kritičnosti

Podobná situace je s vyhodnocením kritičnosti. Jedná se o obdobu skupin zařízení, jen s tím rozdílem, že každé uložené zařízení má svou (jednu) kritičnost. Pro demonstraci bylo vytvořeno několik úrovní – Low, Medium a High. Zvolení správné kritičnosti je na vlastníkovi aktiva. Tato hodnota je zadávána u zařízení.



Obr. 10 - ukázka zobrazení skupin kritičnosti

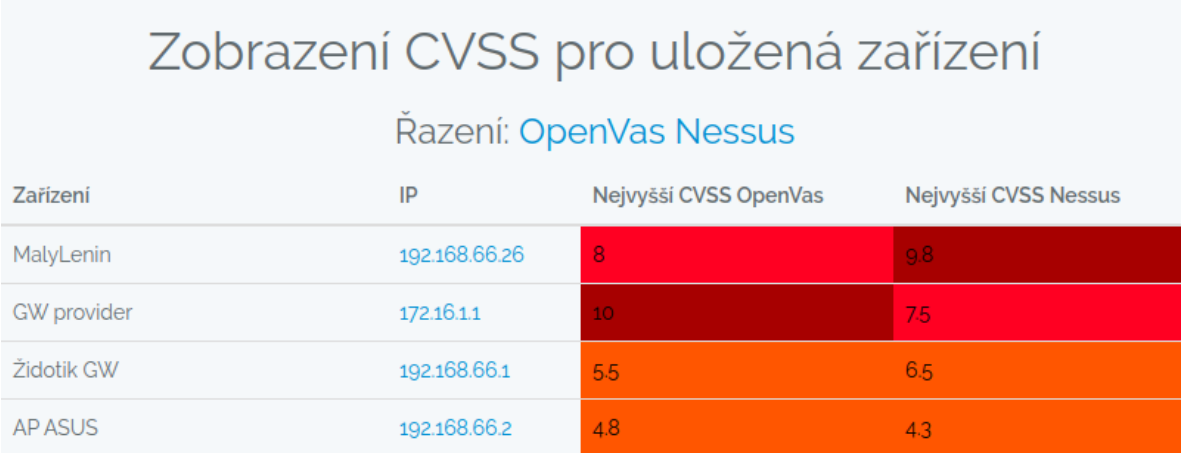
Rozdíl mezi skupinou a kritičností je, že jedno zařízení má jednu kritičnost, ale může být v neomezeně skupinách.

7.7.4 Zobrazení dle typu zařízení

Každé uložené zařízení obsahuje informaci o tom, o jaké zařízení se jedná (switch, PC, server, router a pod). Na základě těchto typů je možné si zobrazit jeden daný typ zařízení. Funkčně je velmi podobné zobrazení dle kritičnosti.

7.7.5 Zobrazení všech uložených zařízení

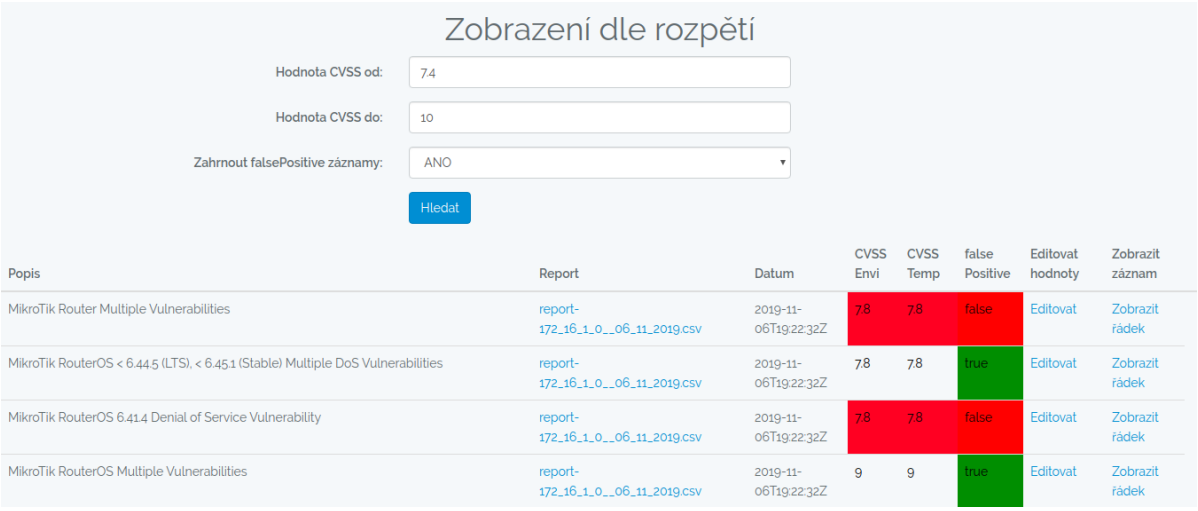
Jak název napovídá, jedná se o vypsání všech uložených zařízeních a jejich příslušných výsledků.



Obr. 11 - zobrazení výsledků pro uložená zařízení

7.7.6 Zobrazení dle rozpětí skóre

Zde má uživatel možnost hledat výsledky dle rozpětí skóre. Výsledky je možné zobrazit bez řádku falsePositive nebo včetně řádku falsePositive.

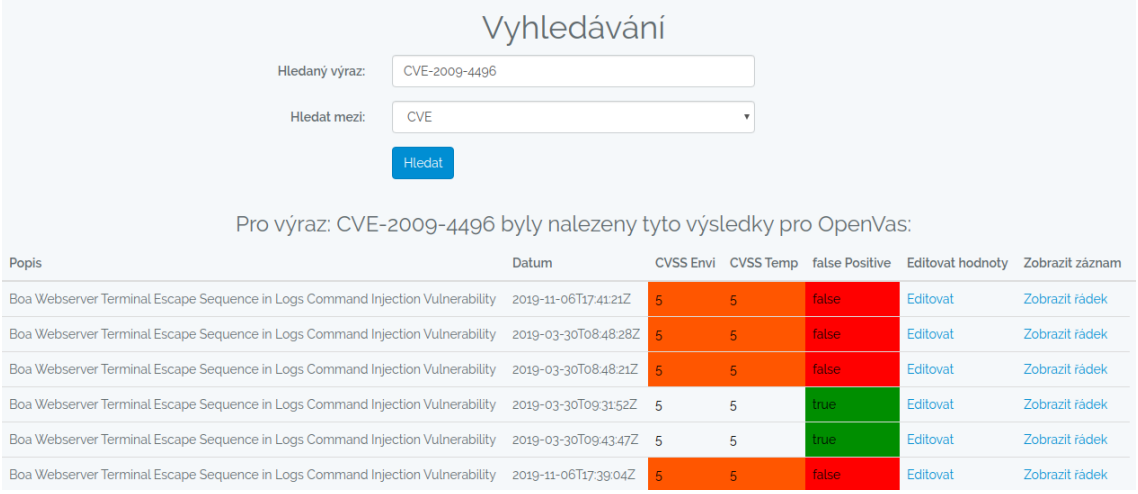


Obr. 12 - zobrazení výsledků dle zadaného rozpětí skóre

7.7.7 Hledání

Hledat se dá v několika polích. Jedná se spíše o demonstraci a je potřeba rozšířit dle potřeb

- CVE
- IP
- část popisu řádku v reportu

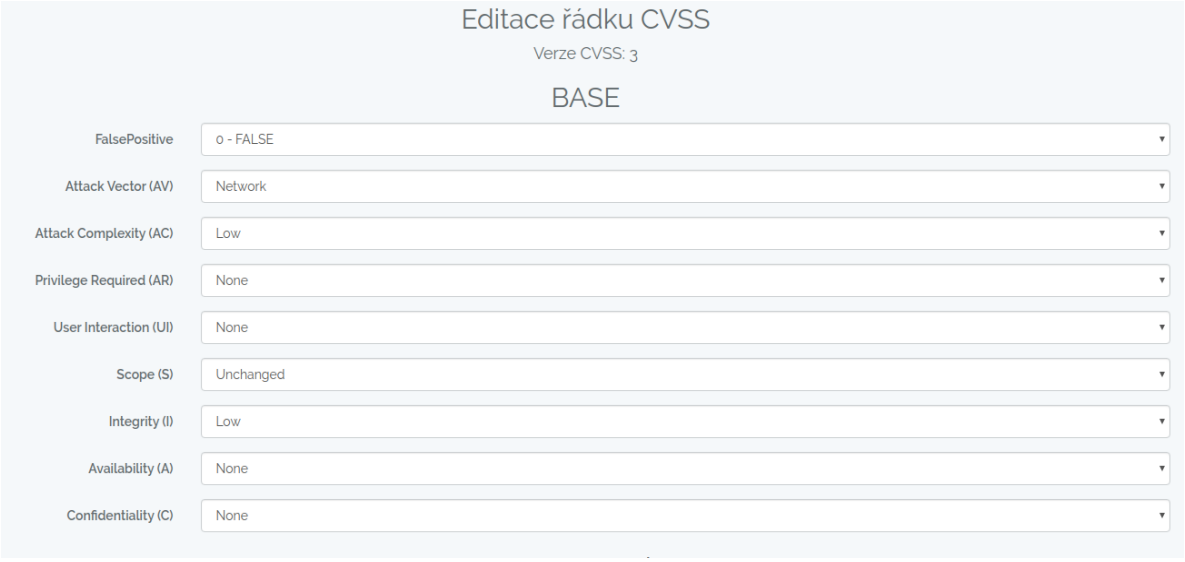


Obr. 13 - zobrazení hledání

7.8 Editace vektoru CVSS

Již výše je zmiňováno, že uživatel má možnost měnit hodnoty vektoru. Tyto změny probíhají přes jednoduchý formulář, na který se uživatel dostane ze všech zobrazení a také z výpisu ohodnocení CVSS reportu. Tyto odkazy je možné vidět na obrázcích výše. Například na obrázku 9.

V tomto formuláři probíhá také označení řádků jako falsePositive, viz 7.4.



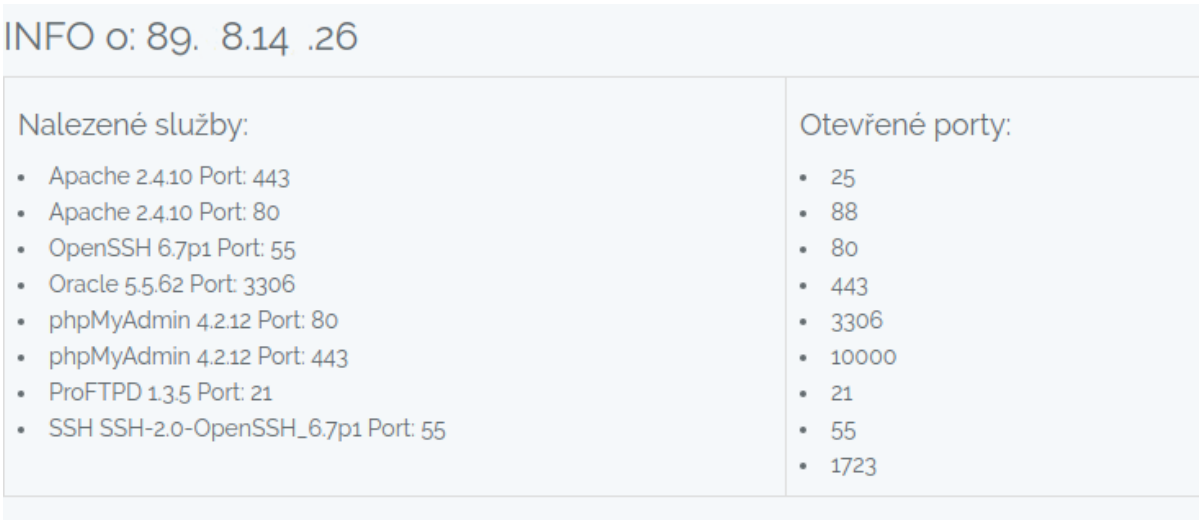
Obr. 14 - zobrazení editace CVSS – zkráceno

Uživatel musí znát položky, aby mohl efektivně a správně měnit hodnoty.

Po uložení hodnot do databáze proběhne automaticky přepočet skóre a nové skóre je uloženo.

7.9 Získávání verzí a otevřených portů z reportů

Reporty v sobě ukrývají informace o IP adrese a také o portu na jakém byla služba nalezena. Toho poznatku se je využíváno ve funkci, kde má uživatel možnost vidět všechny nalezené IP a popřípadě nalezené spuštěné služby (pokud je nalezeno a správně rozpoznáno aplikací) a samozřejmě je informace doplněna o port.



Obr. 15 - zobrazení nalezených spuštěných služeb a otevřených portů

V případě reportu z OpenVAS je získání informace o verzi jednoduché. Výstup z pluginu má stejný tvar a jednoduše jde získat.

```
$string = preg_split('/ version:/', $row->SpecificResult, -1, PREG_SPLIT_NO_EMPTY);
$name = preg_split('/[\s]+/', $row->NVTName, -1, PREG_SPLIT_NO_EMPTY);
  if (!empty($string[1])){
  $verze = preg_split('/[\s]+/', $string[1], -1, PREG_SPLIT_NO_EMPTY);
  $pair = $row->IP.' '. $row->Port. ' '. $verze[0]. ' '. $name[0];
```

Ukázka 7 - získání názvů služeb a portů z reportu

Bohužel u reportů ze skeneru Nessus není situace tak jednoduchá. Výstup není pokaždé stejný. Vytvořen byl pouze výstup pro několik pluginů. Výpis otevřených portů funguje bez problému. Jelikož tato funkce není hlavní funkcí práce, je zde pouze zmínka.

8 Integrování dalších programů do rozhraní

V PHP je možné spustit pomocí funkce exec() jakýkoliv program nebo skript. Již v předchozích částech byl využit například grep pro prohledávání pluginů. Nabízí se hned několik programů, které by mohly být spustitelné přímo z programu. Vždy je dobré mít na paměti, že daný program je spuštěn s právy uživatele, pod kterým je spuštěn webový server.

8.1 PING

Ping není potřeba složitě popisovat a vysvětlovat funkci pingu a jeho fungování. Pro základní ověření spojení mezi zařízeními bohatě dostačuje. To je hlavní důvod, proč bylo rozhodnuto o jeho integrování do rozhraní.

V první fázi byl vytvořen jen jednoduchý ping, ale po několika minutách jsem uznal, že takto v aplikaci nemůže zůstat. Teoreticky by dovoloval spustit ping na jakoukoliv IP a zároveň nebyl odolný proti útoku path traversal.[24] Umožnil by případnému útočníkovi spustit jakýkoliv program nebo skript s právy uživatele webového serveru (v nejhorším případě s právy superuživatele).

```
//verze 2
$IPsub = substr($IP, 0, 15);
//verze1
exec("ping -c 3 $IPsub", $output, $status);
```

Ukázka 8 - ranné verze integrace programu Ping

V případě, který je na obrázku by se IP adresa, kterou chceme testovat předávala přes URL. Útočník by mohl například zadat „URL/device/ping/10.0.0.1 | ls “ a vypsat si obsah složky. Mohlo by být napácháno více škody. Proto tato varianta nebyla zvolena.

Ve druhé verzi byla využita funkce substr(), která přijatý řetězec ořízla max na 15 znaků (max délka IPv4 adresy), ale i přes tuto ochranu by šlo zadat například „URL/device/ping/|pwd“.

Jako nejlepší řešení se ukázalo předělat myšlenku předávání IP adresy a umožnit testovat pouze IP adresy, které jsou uložené v databázi aktiv.

```
$IP = App\device::find($id)->IP;
$IPsub = substr($IP,0,15);
exec("ping -c 3 $IPsub", $output, $status);
```

Ukázka 9 - finální integrování programu Ping

V této verzi se přes URL předává ID zařízení a na příslušnou IP adresu se dotazujeme databáze. Není možné tedy přes URL předat nic navíc.

8.2 NMAP

Nmap neboli Network Maper je nástroj pro prozkoumávání sítě.[21] Šířen je pod licencí Open Source – GNU. Nmap je aplikace bez grafického prostředí (existuje i grafická varianta Zenmap). Ověří, zda je zařízení dostupné a pokud ano provede řadu testů. V základním nastavení bez žádného přepínače vypíše pouze zjištěné otevřené porty (1-1024) a předpokládanou službu, ale bez verzí běžících služeb.

Aby Nmap začal zjišťovat i verze, musí se spustit s přepínačem – sV. V tomto případě ověřil pouze známé porty, tedy pod 1024.

```
pavel@MalyLenin:~$ nmap -sV localhost

Starting Nmap 7.60 ( https://nmap.org ) at 2019-03-12 13:38 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000056s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE         VERSION
80/tcp   open  http            Apache httpd 2.4.29 ((Ubuntu))
631/tcp  open  ipp             CUPS 2.2
902/tcp  open  ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC, SOAP)
8080/tcp open  ssl/http-proxy  VMware Workstation SOAP API 15.0.2
Service Info: CPE: cpe:/o:vmware:Workstation:15.0.2

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 137.76 seconds
```

Obr. 16 - ukázka výstupu NMAP – terminálové okno

Protože Nmap docela dlouho skenuje, bylo provedeno několik měření, při kterých bylo hledáno nejrychlejší a zároveň nejpřesnější nastavení přepínačů. Tyto testy byly provedeny na mém testovacím serveru v internetu a také na lokálním počítači, ze kterého bylo testováno. Výsledky jsou v tabulce níže.

Přepínač	localhost localhost	localhost SRV v int.	SRV v int. SRV v int.	Popis přepínačů
žádný	0,05	112	2,45	
-sV	137,76	144,7	44,79	-sV -zjištění služeb a verzí
-sS -sV	139,37	305	44,66	-Ss -TCP SYN sken (není celý TCP handshake)
-sV -n	137,74	266,56	44,71	-n – nepřekládat DNS názvy
-sV -T5	139,89	224	43,84	-TX - přednastavená „časová“ šablona
-sV -T3	139,37	432	44,71	
-A	186,96	201,8	48,14	-A – detekce OS, verze, tracetoute

Tabulka 6 – naměřené výsledky Nmap v sekundách s popisem přepínačů

Při testování nebyla prolomena hranice pod 100 sekund na lokálním počítači, při testování z lokálního PC k serveru v internetu byly časy ještě horší. Za velmi paradoxní jev bych označil rychlost skenování PC v internetu, který skenoval sám sebe. Možná za to mohla starší verze programu Nmap. Ale ostatní výsledky z pohledu uživatele hodnotím, jako skoro dvouminutové čekání, které není moc příjemné.

Z výše uvedené tabulky a porovnání bylo rozhodnuto, že nejlepší nastavení je jen přepínač –sV, který vypíše verze. Čas potřebný ke skenování se nepodařilo snížit. Proto je možné z aplikace spustit Nmap s přepínačem –sV a také bez žádného přepínače, jen pro rychlé skenování.

Jako v předchozím případě lze Nmap spustit pouze na uložená zařízení.

9 Možnosti vylepšení a rozšíření

Osobně mám v plánu po dokončení práce přidat funkcionalitu, která bude spolupracovat se zařízeními Mikrotik, aby zařízení z aktiv měla automaticky propsaný záznam v DHCP serveru. Tj. vytvoření whitelistu na základě uložených dat v aplikaci.

Pro urychlení získávání vektorů by bylo vhodné napozadí využít programovacího jazyka, který podporuje vlákna, například C++. Celé zpracování reportů by mohlo být mnohonásobně urychleno.

Jak již bylo zmíněno výše, aplikace má jen jednoduchou správu aktiv, pro reálné nasazení by bylo nutné dodělat dle potřeb organizace, nebo ji připojit na již fungující databázi aktiv.

Obecně se vždy dá něco vylepšit.

10 Závěr

V této bakalářské práci jsme se seznámili s fungováním automatických bezpečnostních skenerů a jejich výstupy (reporty). Dále krátké seznámení s CVSS a jeho verzemi, kterého je dále využíváno v praktické části pro ohodnocení.

Začátek praktické části je věnován rychlému seznámení s využitými technologiemi a návrhem databázové struktury. Následuje krátký rozbor celé problematiky importování reportů z výstupů bezpečnostních skenerů.

Dlouhá kapitola byla věnována grafickému zobrazení výsledků a ostatním funkcím ve vytvořené aplikaci.

Je zde popsána cesta reportů, přes nesrozumitelný vektor až po číselné ohodnocení, které bere ohled na prostředí organizace. Tato funkcionalita chybí u samotných skenerů, proto jsou výsledky zkreslené a neodpovídají prostředí organizace.

Myslím si, že vytvořené uživatelské prostředí je jednoduché a srozumitelné. Je připravená pro více druhů uživatelů. Pro uživatele bylo připraveno několik možností zobrazení a také jednoduchá správa aktiv.

Uživatel má možnost měnit vektor CVSS a tím upravovat pohled na skenerem detekované problémy.

Mimo jiné byly do aplikace integrovány i jiné programy, které jsou užitečné pro uživatele.

Dovoluji si říct, že aplikace splňuje požadavky, které byly jako cíle práce a věřím, že aplikace může být užitečným nástrojem.

Seznam zdrojů a literatury

[1] *Top Rated Vulnerability Management Software* [online]. [cit. 2019-03-14]. Dostupné z: https://www.rapid7.com/products/nexpose/

[2] *Information Security and Compliance | Qualys, Inc.* [online], [cit. 2019-03-15]. Dostupné z: https://www.qualys.com

[3] *Open Source Community | OpenSCAP portal* [online], 2019. [cit. 2019-03-14]. Dostupné z: https://www.open-scap.org/features/open-source-community

[4] *OpenVAS - OpenVAS - Open Vulnerability Assessment Scanner* [online], [cit. 2019-12-04]. Dostupné z: http://openvas.org

[5] *Nessus Vulnerability Assessment | Tenable®* [online], 2019. [cit. 2019-12-04]. Dostupné z: https://www.tenable.com/products/nessus

[6] *Buy Tenable Solutions | Tenable®* [online], 2019. [cit. 2019-12-04]. Dostupné z: https://www.tenable.com/buy

[7] KERNER, Sean Michael, 2005. Is Open Source Nessus Closing Its Source? *InternetNews.com* [online]. 2005 [cit. 2019-11-13]. Dostupné z: http://www.internetnews.com/dev-news/article.php/3554781/Is+Open+Source+Nessus+Closing+Its+Source.htm

[8] MISHRA, Rajat, 2018. *Gb_apache_httpd_dos_vuln_apr18_lin.nasl: Apache HTTP Server Denial of Service Vulnerability Apr18 (Linux)* [plugin OpenVAS]. 2018.

[9] Hack I.T. - Security Through Penetration Testing, c2002. *Hack I.T.: security through penetration testing*. Boston: Addison-Wesley, 166 - 167. ISBN 0201719568.

[10] MORGENSTERN, Tal, 2019. The Problem with CVSS Scores and What It Means for Vulnerability Management Programs. *Vulcan.io* [online]. 05.2.2019 [cit. 2019-12-05]. Dostupné z: https://blog.vulcan.io/the-problem-with-cvss-scores-and-what-it-means-for-vulnerability-management-programs

[11] Vulnerability Metrics, *NATIONAL VULNERABILITY DATABASE* [online]. [cit. 2019-12-09]. Dostupné z: https://nvd.nist.gov/vuln-metrics/cvss

[12] FIRST.ORG, INC. (FIRST), *Common Vulnerability Scoring System v3.0: User Guide: Summary of Changes*, 2015. 21 s. [cit. 2019-05-15] Dostupné také z: https://www.first.org/cvss/v3.0/cvss-v30-user_guide_v1.6.pdf

[13] HABUSHA, DAVID, CVSS v3 Creates New Challenges For Developers. In: *White Source* [online]. 7.6.2018 [cit. 2019-10-19]. Dostupné z: https://resources.whitesourcesoftware.com/blog-whitesource/cvss-v3-creates-new-challenges-for-developers

[14] Common Vulnerability Scoring System v3.0: Specification Document: CVSS v3.0 Specification (v1.9), 2015. In: *Common Vulnerability Scoring System SIG* [online]. [cit. 2019-05-19]. Dostupné z: https://www.first.org/cvss/v3.0/cvss-v30-specification_v1.9.pdf

[15] *A Complete Guide to the Common Vulnerability Scoring System Version 2.0* [online], 2005. 23 s. [cit. 2019-03-17]. Dostupné z: https://www.first.org/cvss/v2/cvss-v2-guide.pdf

[16] *Common Vulnerability Scoring System v3.0: Specification Document* [online], 2015. 21 s. [cit. 2019-03-14]. Dostupné z: https://www.first.org/cvss/v3.0/cvss-v30-specification_v1.9.pdf

[17] PostgreSQL: License, *PostgreSQL* [online]. [cit. 2019-10-12]. Dostupné z: https://www.postgresql.org/about/licence/

[18] Laravel Excel: Supercharged Excel exports and imports in Laravel, *Supercharged Excel exports and imports in Laravel | Laravel Excel* [online]. [cit. 2019-12-04]. Dostupné z: https://laravel-excel.com/

[19] A PHP framework for web artisans: A PHP framework for web artisans, 2019. *GitHub - laravel/laravel:: A PHP framework for web artisans* [online]. [cit. 2019-11-28]. Dostupné z: https://github.com/laravel/laravel/

[20] REIGNS, Stephanie, 11 Best PHP Frameworks for Modern Web Developers in 2019. In: *Coders Eye* [online]. [cit. 2019-12-04]. Dostupné z: https://coderseye.com/best-php-frameworks-for-web-developers/

[21] Composer: Introduction Composer, *Introduction - Composer* [online]. [cit. 2019-12-04]. Dostupné z: https://getcomposer.org/doc/00-intro.md

[22] Twbs/bootstrap · GitHub: bootstrap/LICENSE at master, 2019. *Twbs/bootstrap · GitHub: bootstrap/LICENSE at master* [online]. [cit. 2019-12-04]. Dostupné z: https://github.com/twbs/bootstrap/blob/master/LICENSE

[23] POVILAS, Korop, 2017. Two Best Laravel Packages to Manage Roles/Permissions. *Laravel News: Two Best Laravel Packages to Manage Roles/Permissions* [online]. 20.06.2017 [cit. 2019-11-30]. Dostupné z: https://laravel-news.com/two-best-roles-permissions-packages

[24] Path Traversal - OWASP, 2015. *The Open Web Application Security Project: Path Traversal* [online]. [cit. 2019-12-04]. Dostupné z: https://www.owasp.org/index.php/Path_Traversal

Seznam obrázků

Seznam tabulek

Seznam ukázek zdrojových kódů

Seznam použitých zkratek a pojmů

CVSS	Common Vulnerability Scoring System
BASE	Čtyři vlastnosti NoSQL databází (Basic Availability, Soft-state a Eventual consistency)
ACID	Čtyři vlastnosti relačních databází (atomicity, consistency, isolation a durability)
NoSQL	Databázový koncept – nefungující jako klasické relační databáze
BSD licence	Licence pro svobodný SW umožňující volné šíření po uvedení autora
MIT licence	Licence, která podporuje užití SW i v aplikacích s uzavřeným zdrojovým kódem. Je nutné uvést text MIT licence
MVC	Architektura při vývoji SW, která rozděluje datový model, logiku a rozhraní (Model-view-controller)
Open source	Typ SW s otevřenými zdrojovými kódy, je možné je používat, upravovat apod.

Seznam příloh

Příloha 1 – Stručný instalační manuál

Příloha 2 – Stručný návod k ovládání aplikace

Příloha 3 – dostupná online – Nejaktuálnější zdrojové kódy a přílohy:
https://github.com/pavelvodstrcil/BP-program1