VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

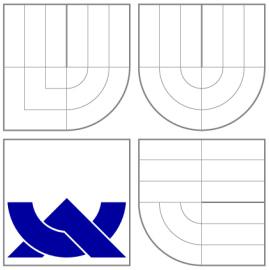
GUI NÁSTROJ NA MĚŘENÍ ZRANITELNOSTÍ SYSTÉMŮ POMOCÍ KNIHOVNY OPENSCAP

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

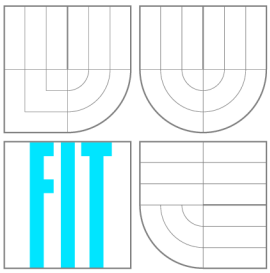
AUTOR PRÁCE
AUTHOR

Bc. VLADIMÍR OBERREITER

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

GUI NÁSTROJ NA MĚŘENÍ ZRANITELNOSTÍ SYSTÉMŮ POMOCÍ KNIHOVNY OPENSCAP

GUI TOOL FOR VULNERABILITY MEASUREMENT BASED ON OPENSCAP LIBRARY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. VLADIMÍR OBERREITER
AUTHOR

VEDOUCÍ PRÁCE — Ing. MAROŠ BARABAS
SUPERVISOR

BRNO 2011

Abstrakt

Tato diplomová práce popisuje standardy SCAP (Security Content Automation Protocol) pro zjišťování úrovně zabezpečení systémů a knihovnu OpenSCAP, poskytující rozhraní ke standardům SCAP. Dále se zabývá návrhem a implementací bezpečnostního nástroje, který využívá knihovnu OpenSCAP. Tento nástroj umožňuje vyhledávat známá, potencionální zranitelná místa systému a zkontrolovat nastavení systému dle předem určených kritérií.

Abstract

This work describes the SCAP standards (Security Content Automation Protocol) determining the level of computer security and the OpenSCAP library providing a framework to the SCAP standards. It also describes the way of designing and creating security tool using the OpenSCAP library. This tool enables to search for known, potential system vulnerabilities and check the system configuration according to previously set criteria.

Klíčová slova

bezpečnost, systém, diagnostika, správa, zranitelnost, knihovna OpenSCAP, OVAL, CCE, CVSS, databáze CVE, XCCDF

Keywords

security, system, analysis, administration, vulnerability, library OpenSCAP, OVAL, CCE, CVSS, database CVE, XCCDF

Citace

Vladimír Oberreiter: GUI nástroj na měření zranitelností systémů pomocí knihovny OpenSCAP, diplomová práce, Brno, FIT VUT v Brně, 2011

GUI nástroj na měření zranitelností systémů pomocí knihovny OpenSCAP

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Maroša Barabasa. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....

Vladimír Oberreiter

10. května 2011

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu Ing. Marošovi Barabasovi za odbornou pomoc a vedení této diplomové práce. A v neposlední řadě bych chtěl poděkovat firmě Red Hat za poskytnuté zadání a pomoc při řešení.

Obsah

Kapitola 1

Úvod

V dnešní době, kdy je skoro každý počítač připojený k internetu nebo je dokonce serverem poskytujícím služby na internetu, se bezpečnost stává jednou z hlavních otázek všech systémů. Nezabezpečený systém je nejen rizikem pro uživatele, kteří jej používají (např. ztráta dat, podvržení změněných dat, zneužití získaných dat útočníkem a mnoho dalších typů rizik), ale může také sloužit útočníkovi k provádění jeho podvodných praktik. Nezabezpečeným systémem pomůžete útočníkovi nevědomky v jeho činnosti.

Pokud uživatel systému používá bezpečnostní prvky, jako například antimalware či firewall, a v neposlední řadě provádí pravidelné aktualizace systému, je jeho systém schopen odolávat velké části pokusů o napadení malwarem či útočníkem.

Nemalá část, která přispívá k bezpečnosti systému, je jeho aktuální nastavení. K čemu by byl systém, který by byl naprosto bezpečný? V dnešní době neexistuje z teoretického hlediska stoprocentně bezpečný systém. Může existovat jen systém, u kterého nebylo doposud zjištěno zranitelné místo či chyba, která by se mohla využít k útoku. Kdyby takto bezpečný systém neměl nastaveno například heslo, bylo by zřejmé, že veškerá snaha návrhářů a programátorů, kteří daný systém navrhli a implementovali tak, aby byl co nejbezpečnější, byla úplně zbytečná.

Proto je pro správce systému dobré znát potencionální zranitelná místa v systému, který spravují, a to, jak spravovaný systém nastavit, aby eliminovali jeho potencionální zranitelnosti a tím v ideálním případě ztížili útoky na daný systém do takové míry, že se útočníkovi nevyplatí daný systém napadat. Čas a zdroje, které by útočníci investovali do pokusu napadnout daný systém, by je stály více než užitek, který by měli, pokud by se jim podařilo systém úspěšně napadnout. Bohužel toto platí pro útočníky, kteří chtějí z útoku na systém dosáhnout nějakého zisku. Je tu i skupina „nadšenců“, kteří napadají systémy pro zábavu nebo jen ze zajímavosti, a těm zpravidla na času a zdrojích nezáleží.

1.1 Cíle a motivace této práce

Úkolem této diplomové práce je seznámit se s knihovnou OpenSCAP a teorií automatického managementu zranitelností SCAP. Dalším cílem je navrhnout nástroj na zjišťování potencionální

zranitelnosti systémů a získávání informací o nastavení systému pomocí knihovny OpenSCAP. Úkolem bude umožnit nastavování různých úrovní pro kontrolu bezpečnosti systému, které by zohlednily využívání kontrolovaného systému, a umožnit získané informace uložit pro pozdější použití, například pro porovnání výsledku v čase. V neposlední řadě je cílem provést testování vytvořeného nástroje pro zjišťování potencionální zranitelností ve spolupráci s firmou Red Hat a toto testování zdokumentovat.

Zadání tohoto projektu poskytla firma Red Hat. Projekt bude vytvářen primárně pro distribuci Fedora, do které bude začleněn. Vytvářený nástroj bude možno stáhnout a nainstalovat do systému, což je pro mě výzvou a zároveň závazkem, abych projekt uskutečnil v co možná nejlepší kvalitě. Bude to můj první projekt, který vytvářím, a najde uplatnění pro širokou veřejnost, což je pro mě velkou motivací.

1.2 Předpokládaný přínos práce

Předpokládám, že tato práce bude mít celou řadu přínosů, jak pro mě samotného, tak pro firmu Red Hat, a v neposlední řadě pro uživatele, kteří budou tento nástroj na měření potencionálních zranitelností využívat ke kontrole nastavení svého počítače.

Pro mě bude přínosem hlubší seznámení s problematikou bezpečnosti, kontrolou nastavení systému před útočníky a zabezpečování systému různými nastaveními, dále seznámení s jazykem Python, ve kterém jsem dosud neprogramoval, a prohloubení znalostí s grafickou knihovnou Gtk. Nástroj bude vyvíjen pod linuxovým systémem Fedora, prohloubím si tedy znalosti i ohledně linuxového systému, protože zatím jsem převážně využíval systémy Windows, což byl také jeden z důvodů, proč jsem si vybral toto téma diplomové práce. Dalším velkým přínosem pro mě bude spolupráce s firmou Red Hat, kde si vyzkouším spolupráci ve skutečném týmu, a tak zjistím, zda bych mohl být v budoucnu platným členem programátorského týmu v podobné společnosti.

1.3 Struktura práce a návaznost na semestrální projekt

Práce se člení na sedm kapitol, ve kterých jsou popsány základní informace k vypracování projektu a popis samotného vypracování:

Kapitola 2: Stručný úvod do základů bezpečnosti a seznámení s metodami pro zjišťování zranitelností a chyb systému.

Kapitola 3: Tato kapitola popisuje standardy SCAP a související standardy, jejichž znalost je důležitá k vypracování tohoto projektu.

Kapitola 4: Zpracování vstupních požadavků projektu s následnou analýzou.

Kapitola 5: Popisuje návrh programu, který vychází z analýzy v kapitole 4.

Kapitola 6: Uvádí postup implementace a implementační problémy projektu a jejich řešení.

Kapitola 7: Závěrečná kapitola, kde je popsáno zhodnocení celého projektu, přínosy a možné další rozšíření projektu.

Diplomová práce navazuje na semestrální projekt, který se týkal studie knihovny OpenSCAP a popisu jednotlivých standardů, které tato knihovna využívá. Výsledky semestrálního projektu jsou využity v kapitole 3 a na základě těchto výsledků se také prováděla analýza a návrh vytvářeného nástroje.

Kapitola 2

Bezpečnost a její zajištění

Počítačová bezpečnost je široká a důležitá oblast, kterou se zabývá velké množství skupin, organizací a jednotlivců. S neustálým rozmachem počítačů ve světě, zejména počítačů připojených do sítí, narůstá velkou měrou riziko napadení těchto počítačů a tím i poškození, ztráta či zneužití dat. Tento problém si nejpravděpodobněji většina běžných uživatelů uvědomuje při elektronickém bankovnictví, kdy se může ztráta cenných přístupových informací nebo napadení systému uživateli značně prodražit. Z tohoto důvodu někteří lidé (zpravidla ti starší) raději tuto možnost nevyužívají.

Abychom se mohli detailněji zabývat bezpečností a zjišťováním její úrovně, musíme se nejprve v následujícím textu seznámit se základními pojmy a principy bezpečnosti. Bez těchto základů by nemělo cenu pouštět se do hlubšího zkoumání zranitelností.

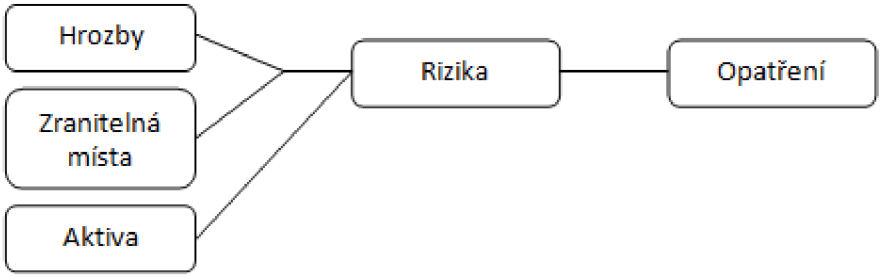
2.1 Úvod do bezpečnosti

Co je to vlastně bezpečnost? Bezpečnost je ochrana věcí a majetku proti poškození, ochrana zdraví nebo samotného života. Je to ochrana movitých i nemovitých cenností, které mají pro někoho takovou hodnotu, že pro něj má cenu je chránit. Tyto hodnoty v bezpečnosti nazýváme aktiva. Jedna z nejčastějších ochran, se kterou se setkává skoro každý dnes a denně, je například obyčejný bezpečnostní zámek na dveřích, kterým zamezujeme v přístupu zlodějům a nežádoucím lidem do domu, aby zde nemohli napáchat žádnou škodu. Ale bezpečnost neznamená ochranu jen před útoky zapříčiněnými jinými lidmi, ale například i před živelnými katastrofami, jako je povodeň, požár, zemětřesení atd. V bezpečnosti nazýváme všechny příčiny, které mohou narušit bezpečnost, hrozby.

Výše jsme uvedli některé základní pojmy. Nyní si tyto pojmy rozebereme podrobněji a uvedeme další [1]:

Zranitelné místo je slabina nebo chyba v systému, která může být využita k narušení bezpečnosti. Zpravidla vzniká selháním analýzy, v návrhu nebo v implementaci.

Hrozba je vlastnost prostředí, která může ve spojení se zranitelným místem systému vést k narušení bezpečnosti, pokud dostane příležitost. Hrozby dělíme na:



Obrázek 2.1: Vztahy mezi základními pojmy (převzato z literatury [2])

- Objektivní:
  - přírodní, fyzické: živelné katastrofy (např. zemětřesení); u takovýchto hrozeb je prevence obtížná, spíše je potřeba řešit snížení dopadů;
  - fyzikální: např. elektromagnetické vyzařování;
  - technické nebo logické: např. poškození pamětí;
- Subjektivní:
  - neúmyslné: způsobené např. nedbalostí či neznalostí uživatele;
  - úmyslné: jedná se zpravidla o pokus útočníka o získání, poškození nebo podvržení dat, poškození HW zařízení, krádež SW.

Aktiva jsou něco, co má hodnotu, má tedy smysl to chránit.

Riziko znamená zranitelné místo zkombinované s bezpečnostní hrozbou.

Útok označuje úmyslné využití zranitelného místa za účelem způsobení škody a ztráty na aktivech nebo neúmyslné využití zranitelného místa, jehož výsledkem je škoda nebo ztráta na aktivech. Útok můžeme též nazvat bezpečnostním incidentem.

Útočník je osoba nebo více osob, které provádí útok. Útočníci mohou být amatéři, náhodní útočníci, hackeři i organizované zločinecké skupiny.

Bezpečnostní opatření vytváří bariéru mezi hrozbami a aktivy. Cílem bezpečnostních opatření je co nejvíce omezit počet zranitelných míst a zabezpečit případná zranitelná místa, která nejsme schopni odstranit. Bezpečnostní opatření můžeme kategorizovat do dvou základních skupin:

- omezující: minimalizují ztráty a maximalizují zotavitelnost po útoku;
- preventivní: snižují pravděpodobnost útoku a zvyšují případné „náklady“ útočníka.

2.2 Počítačová bezpečnost

V dnešní době, kdy má velké procento lidí počítač a připojení k internetu, je počítačová bezpečnost velmi důležitá. Dokud nebylo připojení na internet běžnou věcí, šíření virů a útok na data v počítači nebyly tak jednoduché jako nyní. V dnešní době se stačí chvíli pohybovat po internetu na nedůvěryhodných stránkách a už podléháte riziku, že váš počítač bude napaden malwarem (bez antiviru a firewallu je napadení téměř jisté). Velkou roli hraje také nastavení operačního systému v počítači a programů, které jsou využívány. Odpojení počítače od sítě není řešením, dokonce si už dnes málokdo dokáže představit, že by nevyužíval internet. Počítač připojený k internetu využíváme k platbám, komunikaci, nákupům, je velkou studnicí informací a mnozí z nás jej využívají i v práci. Vidíme, že počítač a síť jsou úzce spojeny, a z toho důvodu rozdělujeme počítačovou bezpečnost na [1]:

Bezpečnost systému: integrita systému a dat, dostupnost a důvěryhodnost.

Síťová bezpečnost: bezpečnost přístupu k systému, komunikace a přenos dat.

Bezpečnost informačních systémů si klade za úkol plnit tyto *bezpečnostní cíle*:

1. Důvěrnost (confidentiality) – ochrana proti neoprávněnému prozrazení informace;
2. Integrita (integrity) – ochrana proti neoprávněné modifikaci informace;
3. Dostupnost (availability) – ochrana proti neoprávněnému odepření přístupu k datům nebo k službám.

Bezpečnost můžeme zvýšit třemi základními přístupy:

1. Prevencí – ochranou proti hrozbám;
2. Detekcí – odhalením slabého místa v systému a odhalením neoprávněné činnosti;
3. Nápravou – odstraněním slabého místa v systému.

Bezpečnost je tvořena:

1. Bezpečnostními cíli – proč něco chráníme?
2. Bezpečnostními funkcemi – čím je bezpečnost zajištěna?
3. Bezpečnostními mechanismy – jakým konkrétním způsobem provádíme zajištění bezpečnosti?

2.2.1 Bezpečnostní opatření

Bezpečnostní opatření systému můžeme rozdělit do několika kategorií:

Fyzická zabývá se fyzickou kontrolou přístupu k systému (zdi, bezpečnostní zámky, vrátný) a ochranou proti vlivům zvenčí (detektory požárů, hasicí systémy, záložní zdroje napájení atd.).

Administrativní zabývá se bezpečnostními procedurami prováděnými lidmi (evidence přístupu, dokumentace, zálohovací procedury, přihlašování atd.).

Personální lidé jsou nejméně spolehlivou částí IS, proto se zavádějí opatření a omezení proti hrozbě od uživatelů (školení a osvěta zaměstnanců, důvěryhodnost a spolehlivost pracovníka atd.).

Technická tato opatření jsou implementována v HW, SW IS a zajišťují, že uživatel je zodpovědný za akce, které provedl (identifikace, autentizace, protokolování, šifrování atd.).

Aby bezpečnostní opatření měla požadovaný účinek, musí být periodicky kontrolováno, zda se jejich implementace už nestala časem neefektivní či zda je ještě relevantní. Bezpečnostní opatření jsou omezena cenou, která by měla být nižší než cena dat, která by byla ztracena, pokud by bezpečnostní opatření nebylo implementováno. Cena opatření je složena z jednorázových nákladů na nákup HW a SW a jejich instalace, plus provozní náklady (např. energie, údržba atd.). Účelem bezpečnostních opatření je zvednout cenu útoku natolik, že případný zisk útočníka z něj bude menší než cena útoku.

2.3 Zjišťování zranitelností systémů

V této sekci budou popsány některé metody, které se používají pro zjišťování zranitelností systému (bezpečnosti systému) [3, 4]. Poslední z nich je metoda, která je založena na standardech SCAP. Tato metoda bude podle mého názoru do budoucna velmi rozšířená, protože umožňuje porovnávat míru bezpečnosti odlišných systémů, je založena na standardech a akceptuje aktuální nastavení systému.

Zjišťování zranitelností, které bere v úvahu nastavení systému, může být užitečné jak pro administrátory, tak pro uživatele, a pokud metoda zjišťování zranitelností umožní použít nějaké předdefinované nastavení profesionála, které jen uživatel spustí, bude tato metoda ideální i pro nezkušené uživatele, kteří si tak budou moci jednoduchým způsobem otestovat počítač a při zjištěné zranitelnosti si vyhledat řešení.

2.3.1 Metoda na úrovni kódu

Mnoho úsilí bylo věnováno metodě založené na hledání chyb na úrovni kódu, která se běžně používá [4]. Uvedeme si její základní nevýhody.

- Detekce chyb není stoprocentní, mnoho chyb nemusí být nalezeno, díky čemuž se může zdát systém bezpečnější.

- Stejná důležitost je přisuzována všem chybám i přesto, že některé chyby jsou méně důležité než jiné a naopak.
- Tato metoda nebere v úvahu aktuální nastavení systému, které bezpečnost systému ovlivňuje velkou měrou.

2.3.2 Metoda založená na Attack surface

Tato metoda je založena na *Attack surface* (dále oblast útoku) [4], zasahuje do úrovně návrhu (nad úroveň kódu, ale pod úroveň systému jako celku).

Oblast útoku je tvořena systémovými akcemi, které jsou viditelné pro systémové uživatele společně se systémovými zdroji, které jsou zpřístupněny nebo modifikovány těmito akcemi. Čím více systémových akcí, které může využívat uživatel, nebo více zdrojů, které jsou zpřístupněny pomocí těchto akcí, tím větší je oblast útoku. Pokud má systém špatně chráněnou oblast útoku, tím je větší pravděpodobnost, že systém bude úspěšně napadán. Nejjednodušším krokem, jak vytvořit systém bezpečnější, je tedy zmenšit oblast útoku systému.

Během let, kdy se útočníci pokoušeli o útoky na systém, se ukázalo, že systémové zdroje poskytují více příležitostí pro útok než ostatní možnosti. Například služby v Unixu běžící pod uživatelem root, který má podstatně větší oprávnění než obyčejný uživatel, budou mnohem vhodnější jako cíl útoku, než právě obyčejný uživatelský účet. Stejně tak soubory, které mají plné oprávnění, jsou vhodnější pro útok než soubory s nízkým oprávněním. Pokud je povoleno používání maker např. v Microsoft Office, tak se možnost pravděpodobného útoku zvyšuje. Podobných případů bychom našli určitě spousty. Tato metoda rozeznává zdroje, které jsou příležitostí pro útok, nastavuje jim vlastnosti související se zdroji a rozděluje je do tzv. *Attack class* (dále třídy útoku).

Dle vytvořených tříd útoku, které se vytvoří na základě verzí porovnávaných systémů, pak můžeme porovnat, jestli je jeden systém bezpečnější než druhý s ohledem na tyto třídy útoku. Existují dva způsoby, jak můžeme systémy porovnat.

1. Porovnat počty ve třídách útoku jednotlivých systémů, které získáme tak, že pro každou verzi spočítáme počet běžících instancí v každé třídě útoku (např. počet otevřených soketů a služeb běžících jako root).
2. Druhý způsob je o něco přesnější, protože nastavuje váhy jednotlivým instancím v třídách útoku. Tyto váhy slouží pro určování pravděpodobnosti útoku související s touto instancí. Například funkce, která může posloužit k útoku, bude mít jinou pravděpodobnost, pokud bude spuštěna jako root nebo bude spuštěna jako obyčejný uživatel.

Tato metoda je vhodnější pro porovnávání dvou podobných systémů (např. dvou různých verzí systému Linuxu Fedora) než pro porovnávání dvou plně odlišných systémů (např. Linux a Windows), protože různé systémy mají rozdílné třídy útoku.

Tato metoda využívá k modelování systému hrozeb, pokusů útoků na systém a uživatelů v systému stavový automat. Využití stavového automatu v bezpečnosti není ničím novým, na-

příklad jsou používány pro detekci instrukcí [5, 6] nebo jako model bezpečnostní politiky [7]. Automat se u této metody liší oproti automatům uvedeným v literatuře (např. tím, že systém je reprezentován jako oddělená entita).

Základní postup zjišťování zranitelností Tento postup může být použit pro identifikaci tříd útoku v systému a zjištění oblasti útoku systému. Je aplikovatelný na různé druhy systémů. Postup popisuje porovnání dvou systémů.

1. Najít a identifikovat potenciální zdroje pro útoky na zkoumaném systému a přiřadit těmto zdrojům typ.

2. Určit množinu vlastností, které nás budou zajímat vůdčí zdrojům a na základě těchto vlastností (typů zdrojů) vytvořit hierarchii. Každý list v této hierarchii je jedna třída útoku systému. Při tomto kroku je velmi důležitá znalost daného systému, protože musíme vybrat relevantní množinu vlastností, kterou použijeme. Množina, kterou použijeme, se liší systém od systému a může se měnit i v čase. Pro každý systém bude tedy tato množina jiná a nová verze systému může mít jinou množinu těchto vlastností než předešlá verze.

3. Nyní musíme určit pravděpodobnost útoku. Třídy útoku s větším pravděpodobnostním ohodnocením útoku na zdroje mají větší možnost, že budou vybrány za cíl útoku, než třídy s nižším pravděpodobnostním ohodnocením. Pravděpodobnost útoku můžeme nastavit více způsoby. Jedna z možností je pomocí získaných reportů o útocích a zranitelných místech, která třída má větší počet zranitelných míst té dát větší pravděpodobnost. Mnohem sofistikovanější přístup je vypočítat pravděpodobnost na základě škod, které by byly způsobeny, pokud by se útok podařil.

4. V praxi nejsou potřeba všechny třídy útoku, proto vybereme jen některé z nich. Měli bychom je vybrat podle pravděpodobnostního ohodnocení, které jsme přiřadili v kroku 3.

5. V posledním kroku nám zbývá porovnat jednotlivé třídy útoku. Jednou možností je jednoduše spočítat počet instancí ve třídách v obou verzích systému. Dalším způsoben je využít pravděpodobnostní ohodnocení z kroku 3. Toto není výčet všech způsobů porovnání, existuje jich více.

Výhody metody založené na Attack surface

- Princip zjišťování zranitelností je relativní (není jednoduché stanovit měřítko pro měření bezpečnosti). Relativnost umožňuje pohodlně srovnávat na základě tříd útoku různé systémy, jak ty nové, tak i ty staré.

- Umožňuje porovnávat systém v čase, zda různá nastavení prospěla bezpečnosti či nikoliv.

Nevýhody metody založené na Attack surface

- Vyžaduje velmi dobrou znalost systému pro vytvoření tříd útoků.
- Třídy útoků jsou u různých systémů odlišné, to komplikuje jejich porovnávání.

2.3.3 Metoda založená na protokolu SCAP (Security Content Automation Protocol)

V této části Vás seznámím jen zběžně s protokolem SCAP, protože se jím zabývá celá kapitola Knihovna OpenSCAP na straně 14, která standardy SCAP implementuje.

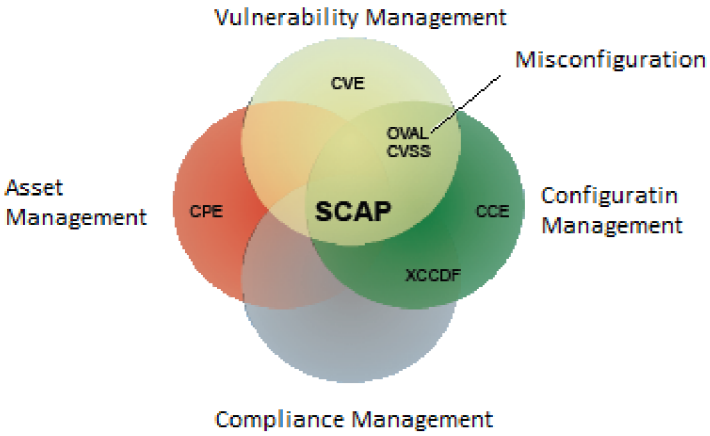
SCAP specifikuje celou řadu otevřených standardů, které se používají k výčtu zranitelností v systému, nastavení managementu bezpečnosti k automatickému zjišťování zranitelností, ohodnocení závažnosti zranitelnosti (určení skóre), automatickou správu těchto míst a dodržování zásad hodnocení. Dále standardizuje komunikaci mezi jednotlivými standardy a také informace, které jsou předávány [10, 12]. Souvislosti mezi jednotlivými standardy jsou zobrazeny na obrázku 2.2.

Seznam standardů, které SCAP zahrnuje:

- **Common Vulnerability Enumeration (CVE)** – veřejný seznam známých zranitelných míst a chyb;
- **Common Platform Enumeration (CPE)** – definuje schéma popisu pro identifikaci systému, platformy a programového balíku;
- **Common Configuration Enumeration (CCE)** – poskytuje unikátní identifikátory v seznamu známých konfigurací systému;
- **eXtensible Checklist Configuration Description Format (XCCDF)** – standard, který definuje XML formát specifikující bezpečnostní kontrolní seznamy, srovnávací testy a konfigurační dokumenty;
- **Open Vulnerability and Assessment Language (OVAL)** – standardizuje kroky na zjištění různých bezpečnostně relevantních informací z počítače;
- **Common Vulnerability Scoring System (CVSS)** – standard, který definuje systém na výpočet skóre závažnosti zranitelnosti na základě charakteristik systému a prostředí.

Výhody metody založené na SCAP

- Pokud je vytvořen nástroj a jsou specifikovány dokumenty, které se využívají pro kontrolu (XCCDF, OVAL), kontrola je velmi jednoduchá.
- Vhodné pro různé typy systému.



Obrázek 2.2: Závislost standardů SCAP (převzato z literatury [12])

Nevýhody metody založené na SCAP

- Zranitelnosti, které chceme kontrolovat, musí být již objeveny a specifikovány.

Kapitola 3

Knihovna OpenSCAP

V této kapitole si popíšeme knihovnu OpenSCAP[1], která poskytuje rozhraní k standardům SCAP, jež bude využívat vytvářený nástroj ke splnění jednotlivých požadavků. V první řadě si popíšeme knihovnu OpenSCAP jako celek a poté se zaměříme na její jednotlivé části, které se vážou k vytvářenému nástroji.

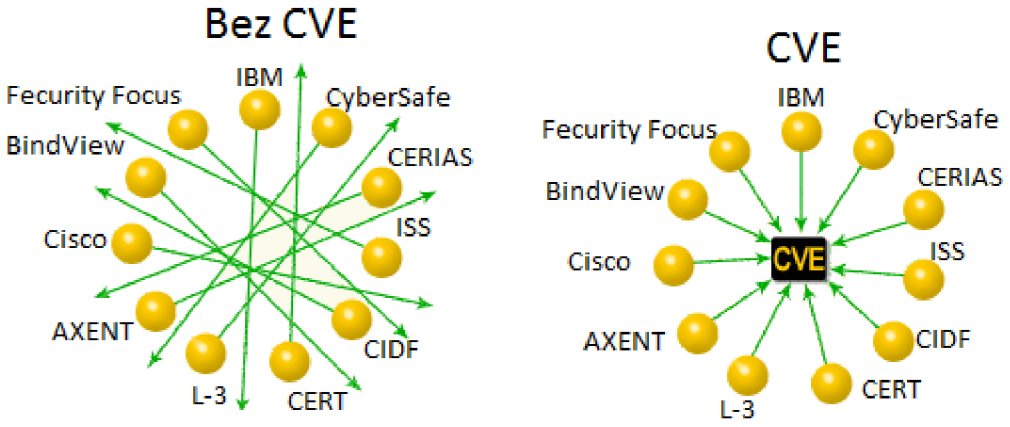
3.1 Knihovna OpenSCAP

Knihovna OpenSCAP byla vytvořena, aby poskytovala open-source rozhraní komunitě programátorů, kterým umožní využívat sadu standardů a schopností souvisejících s bezpečností nazvanou SCAP (Security Content Automation Protocol). Cílem OpenSCAP je tedy poskytnout jednoduché a snadno použitelné rozhraní k využívání standardů SCAP [13, 8, 14].

SCAP je řada standardů spravovaných organizací NIST. Ta byla vytvořena za účelem poskytnout standardizovaný přístup k zabezpečení systému, například automatické ověřování přítomnosti důležitých aktualizací, kontrolu nastavení systému z hlediska bezpečnosti a posouzení, zda systém neobsahuje zranitelná místa.

SCAP poskytuje velké množství formátů pro výměnu dat, například formát pro popis důležitých zranitelností, konfigurací systémů a mnoho dalších formátů pro přenos bezpečnostních informací. Dříve existovalo jen několik nástrojů, které poskytovaly možnost získat data v požadovaném formátu, ale tento nedostatek nástrojů byl překážkou vstupu na trh a odrazoval přijetí těchto protokolů komunitou. Proto vznikla knihovna OpenSCAP s cílem zlepšit dostupnost standardů SCAP a zlepšit využitelnost informací, které představuje, a tím poskytnout rozhraní a funkční příklady, které by umožnily ostatním komunitám, aby využívaly standardy SCSP a zároveň minimalizovali úsilí potřebné k práci s těmito standardy.

[1] www.open-scap.org



Obrázek 3.1: Výhoda CVE (převzato z [15])

3.2 Standardy knihovny OpenSCAP a související standardy

V této podkapitole si popíšeme nejdůležitější součásti knihovny OpenSCAP (standardy SCAP [10, 9]) a standardy, které knihovna využívá.

Standardy SCAP (The Security Content Automation Protocol) jsou metody na používání standardů, jež umožňují automatickou správu zranitelností, měření zranitelností a vyhodnocování zranitelností. Dále popisuji, jak s těmito standardy pracovat a možnosti jejich vzájemného využívání. Tyto standardy jsou zaměřeny na popis bezpečnostních konfigurací systémů, produktových názvů, softwarových chyb a zranitelností a na základě těchto dat umožňují využívání metod, které ohodnotí bezpečnostní rizika dané chyby.

3.2.1 CVE (Common Vulnerabilities and Exposure)

Jedná se o standard pro záznam známých zranitelných míst a chyb, který je veřejně přístupný [15, 16]. Podle tohoto seznamu můžeme zjistit zranitelnosti jednotlivých systémů. Vytváření tohoto seznamu začalo v roce 1999. Do této doby měl každý systém vlastní databázi a dané databáze byly nekompatibilní. A tak nešlo určit, zda například data v různých databázích jsou stejná či ne. Díky tomuto standardu mohl vzniknout nástroj, který může porovnávat zranitelnosti mezi různými systémy (viz obrázek 3.1).

Každá zranitelnost má přiřazeno unikátní identifikační číslo. Pokud někdo objeví novou zranitelnost a chce ji přidat do seznamu, musí se nejprve daná zranitelnost schválit, a to následujícím způsobem. Nejprve se nové zranitelnosti přidělí status „Candidate“, což zajišťuje CNA (CVE Candidate Numbering Authority). Poté se řeší, zda danou zranitelnost začlenit do seznamu či ne. Pokud se rozhodne kladně, je daná zranitelnost přijata a status se změní na „Entry“. V opačném případě jsou sděleny důvody nepřijetí.

CVE-ID

CVE-2000-0005
(under review)

Learn more at National Vulnerability Database (NVD)
• Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings

Description

HP-UX aserver program allows local users to gain privileges via a symlink attack.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- BUGTRAQ:19991230 aserver.sh
- BUGTRAQ:20000102 HPUX Aserver revisited.
- HP:HPSBUX0001-108
- OVAL:oval:org.mitre.oval:def:5635
- URL:http://oval.mitre.org/repository/data/getDef?id=oval:org.mitre.oval:def:5635
- XF:hp-aserver

Status

Candidate	This CVE Identifier has "Candidate" status and must be reviewed and accepted by the CVE Editorial Board before it can be updated to official "Entry" status on the CVE List. It may be modified or even rejected in the future.

Phase

Modified (20090302)

Votes

ACCEPT(3) Armstrong, Baker, Stracener
MODIFY(1) Frech
RECAST(1) Christey
REVIEWING(1) Levy

Comments

```
Christey> BUGTRAQ:20000102 "HPUX Aserver revisited." indicates that two
  different versions of aserver have symlink problems, but with
  different files.  So CD:SF-LOC says we should split this.
Frech> XF:hp-aserver
Christey> BID:1928 and BID:1930?  Which one is being described in
  this candidate?
Christey> BID:1930
```

Obrázek 3.2: Příklad položky CVE ve stavu „Candidate“ (převzato z [17])

3.2.2 CCE (Common Configuration Enumeration)

Poskytuje jedinečné identifikátory související s bezpečnostní konfigurací systému[18]. Bezpečnostní konfigurací se například myslí, kolikrát se může zadat špatně heslo, než se daný účet zablokuje a uživatel musí požádat o odblokování. Na základě těchto bezpečnostních identifikátorů můžeme porovnávat nastavení různých systémů (tohoto standardu využívá například CVE pro informování o zranitelnosti systému).

CCE tedy přiděluje unikátní identifikátor k bezpečnostní konfiguraci systému (podobně jako CVE přiděluje identifikátory jednotlivým zranitelnostem). Identifikátor CCE poskytuje most mezi dokumentací, přirozeným jazykem a možností zpracovávat daný záznam automaticky (například pro audity, konfigurace systému atd.). Ukázka záznamu v CCE je na obrázku 3.3.

Záznam se skládá z následujících částí:

- **CCE Identifing Number (Identifikační číslo),** jež přiřadí identifikátor každé uznané bezpečnostní konfiguraci. Slouží jako jedinečný identifikátor, který není popisný. Bezpečnostní identifikátor je ve tvaru (CCE-####-#).
- **Description (Popis)** obsahuje text, který lidským způsobem (přirozeným jazykem) popisuje danou bezpečnostní konfiguraci. Popis není zamýšlen jako konkrétní popis, jak by měla nebo neměla být daná bezpečnostní situace nastavena. Možný záznam je například „Minimální délka hesla by měla být nastavena vhodně“. Z toho plyne, že neurčuje konkrétní minimální délku hesla. Snaží se popisovat problém globálně.
- **Conceptual Parameters (Parametry**). Záznam obsahuje seznam parametrů, které slouží ke splnění CCE na daném systému. Například „Smard Card služba by měla být povolena nebo zakázána vhodně“. Parametry tohoto záznamu jsou Manual, Disabled, Automaticaly.
- **Associated Technical Mechanisms (Technické parametry)** je seznam způsobů, které popisují, jakým způsobem je možné provést změny v systému, jež popisuje záznam CCE. Následuje ukázka dvou způsobů změny bezpečnostní konfigurace CCE-4224-2:
  - *Computer Configuration\Administrative Templates\System\Internet Communication Settings;*
  - *HKLM\Software\Policies\Microsoft\Messenger\Client!CEIP.*
- **Citations (Citace).** Každý zápis má řadu citací, které odkazují na konkrétní dokumenty (jejich části), nástroje, kde jsou detailní popisy dané bezpečnostní konfigurace. Citace slouží ke třem účelům:
  - obsahuje výše zmíněné vazby na více podrobné informace;
  - potvrzuje, že je třeba CCE-ID pro danou bezpečnostní konfiguraci;
  - ověřuje, že citace CCE-ID je popsána na požadovaném stupni abstrakce, která je použita a akceptována.

3.2.3 CPE (Common Platform Enumeratio)

Tak, jak slouží CVE k popisu zranitelnosti a CCE k popisu bezpečnostní konfigurace, CPE slouží k popisu systému, platforem a balíčků. Všechny tyto standardy slouží k automatizaci zpracovávání dat, umožňují vzájemné porovnávání mezi různými systémy a v neposlední řadě značně urychlují zpracovávání těchto informací a jejich sběr [19].

CPE definuje schéma strukturovaného pojmenování pro systémy informačních technologií, platforem a programovacích balíků. Princip je podobný URI (Uniform Resource Identifiers).

CCE ID	CCE Description	CCE Parameters	CCE Technical Mechanisms	Citation	
				DISA Gold Disk for WXP	NSA Security Guide for WXP (NSA-XP-C44-026-02.pdf)
CCE-3110-4	The "Let Everyone permissions apply to anonymous users" policy should be set correctly.	enabled disabled	(1) HKEY_LOCAL_MACHINE \System\CurrentControlSet \Control\Lsa \EveryoneIncludes Anonymous (2) defined by Local or Group Policy	Let Everyone permissions apply to anonymous users Value (CID:783)	Network access: Let Everyone permissions apply to anonymous users: Disabled

Obrázek 3.3: Ukázka záznamu CCE

Struktura jména je ukázána na obrázku 3.4, dále je na obrázku ukázán klasický zápis a zápis v XML.

CPE je využíváno u záznamů CVE a CCE, kde uvádí na jaké platformě, systému či balíčku se daná zranitelnost nachází nebo ke kterému systému se vztahuje bezpečnostní nastavení.

3.2.4 CVSS (Common Vulnerability Scoring System)

Obsahuje metody na výpočet stupně rizika zranitelnosti [13, 20]. Na základě výsledku stupně zranitelnosti můžeme jednotlivé zranitelnosti mezi sebou porovnávat, a tak určit, která zranitelnost je závažnější. Podle stupně rizika můžeme řídit další kroky na jeho omezení.

Základní princip CVSS

Pokud je zadán vstupní vektor základním metrikám, metriky spočítají numerickou hodnotu v rozsahu od 0 do 10, která udává stupeň rizika dané zranitelnosti (10 je největší riziko, nula nejmenší riziko). Vektor je textový řetězec, který obsahuje zadané hodnoty do metrik, nese v sobě informaci, jak byla daná zranitelnost vypočítána. Proto je vždy důležité uvádět vytvořený vektor s vypočítaným ohodnocením zranitelnosti. Princip ohodnocení rizika viz obrázek 3.5.

Vypočítané ohodnocení pomocí základních metrik (Base Metrics) se může dále zpracovávat dalšími dvěma skupinami metrik, a to pomocí časových metrik (Temporal Metrics) a metrik prostředí (Enviromental Metrics). Obě tyto skupiny metrik jsou volitelné a nemusejí se pro výpočet zranitelnosti použít. Výhodou těchto rozšiřujících skupin metrik je přidání informace o kontextu zranitelnosti a tím zpřesnění hodnocení míry rizika na základě uživatelského prostředí. Zda použít jen základní skupinu metrik, nebo i další skupiny metrik, záleží na povaze a účelu záměru, protože ohodnocení skupinou základních metrik může být pro některé účely dostačující.

Struktura CPE jména

```
cpe:/ {part} : {vendor} : {product} : {version} : {update} : {edition} : {language}
```

Příklad zápisu CPE jména

```
cpe:/o:microsoft:windows_2000::sp4:pro
```

Příklad zápisu CPE jména v XML

```
<cpe:platform id="456">
    <cpe:title>Sun Solaris 5.8 or 5.9 or 5.10</cpe:title>
    <cpe:logical-test operator="OR" negate="FALSE">
        <cpe:fact-ref name="cpe:/o:sun:solaris:5.8" />
        <cpe:fact-ref name="cpe:/o:sun:solaris:5.9" />
        <cpe:fact-ref name="cpe:/o:sun:solaris:5.10" />
    </cpe:logical-test>
</cpe:platform>
```

Obrázek 3.4: Zápis CPE jména (převzato z literatury [19])

Metriky CVSS:

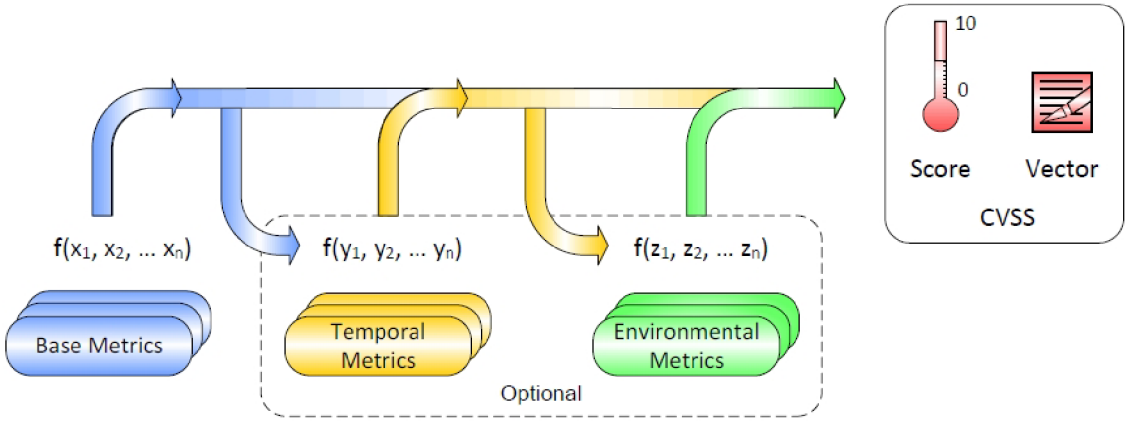
- Základní metriky: Zakládají se na vnitřních vlastnostech zranitelnosti, nejsou ovlivněny časem a uživatelským prostředím.
- Časové metriky: Popisují vlastnosti zranitelnosti, které se mění v čase, ale nejsou závislé na uživatelském prostředí.
- Metriky prostředí: Popisují vlastnosti zranitelnosti, které jsou užitečné a unikátní vzhledem k uživatelskému prostředí.

Vektory CVSS

Jak již bylo zmíněno výše, vektor je textový řetězec, který slouží k popisu vypočítání ohodnocení zranitelnosti. Zde je uveden jen základní popis vektorů, protože problematika je natolik rozsáhlá, že by nebylo vhodné ji zde popisovat celou. Každá metrika obsahuje svoje specifické vektory. Vektor je zapsán stylem: název vektoru, dvojtečka a za dvojtečkou se nachází hodnota, kterou může daný vektor nabývat. Jednotlivé vektory se oddělují lomítkem. Ukázka vektoru je v následujícím příkladu. Detailní informace k vektorům je možné nalézt v literatuře [20].

Příklad vektoru: V tomto příkladu uvedu, jak vypadá vektor pro základní metriky, a popíši jeho význam.

Vektor: “AV: L/AC:H/Au:N/C:N/I:C/A:C”



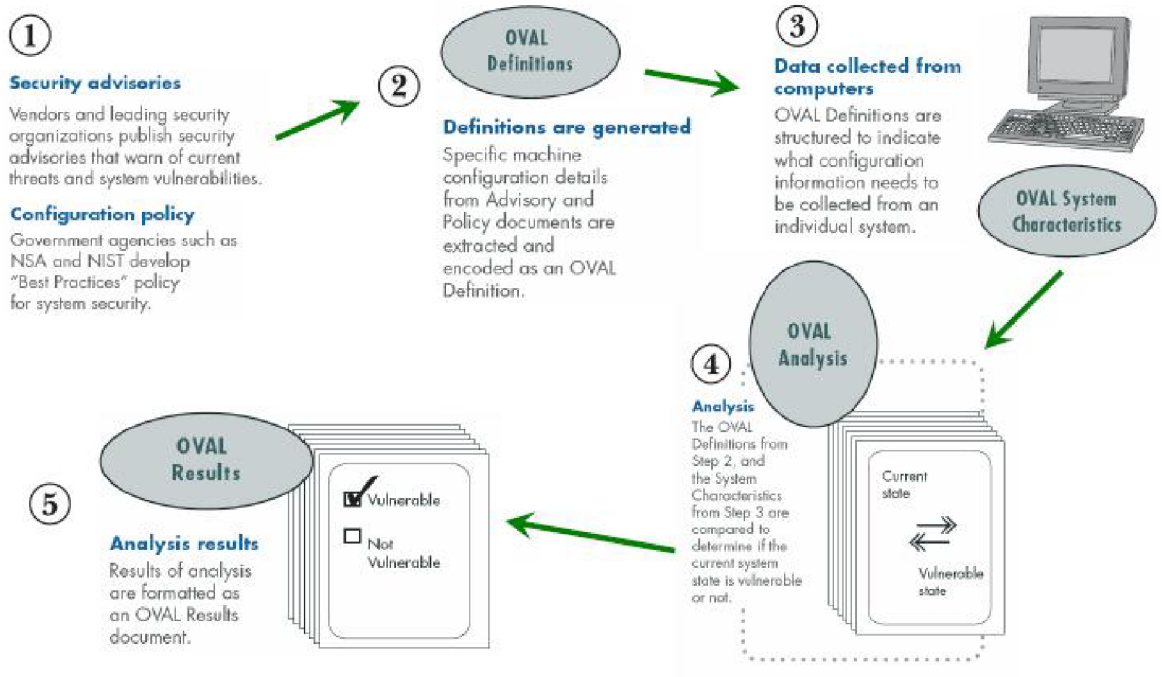
Obrázek 3.5: CVSS metriky princip (převzaté z literatury[20])

Popis vektoru:

- **AV (Access Vector)** popisuje, jakým způsobem může být daná metrika využita. **L (Local):** zranitelnost může být využita pouze v případě, že útočník má lokální přístup, a to fyzický nebo přístup k lokálnímu účtu.
- **AC (Access Complexity)** popisuje náročnost útoku jedním útočníkem. **H (Hight):** náročnost je vysoká, ohodnocení zranitelností je menší.
- **Au (Authentication)** popisuje, kolikrát se útočník musí přihlašovat, než dosáhne požadovaného cíle. **N (None):** přihlašování není požadováno.
- **C (Confidentiality Impact)** popisuje, k jakým důvěrným informacím se může útočník po úspěšném útoku dostat. **N (None):** nedostane se k žádným důvěrnostem systému.
- **I (Integrity Impact)** popisuje, jestli útočník po úspěšném útoku může narušit integritu systému. **C (Complete):** útočník může modifikovat všechny soubory na cílovém systému.
- **A (Availability Impact)** popisuje, jakou měrou může útočník po úspěšném útoku změnit dostupnost systému. **C (Complete):** útočník může kompletně zamezit přístup k systému.

3.2.5 OVAL (Open Vulnerability and Assessment Language)

OVAL je standard, který podporuje popis otevřených a veřejně dostupných bezpečnostních informací a standardizuje přenos těchto informací z oblasti bezpečnosti napříč celým spektrem nástrojů, které se zabývají bezpečností a bezpečnostními službami [21]. OVAL obsahuje jazyk využívaný k zakódování systémových informací, které je možno uložit, a tyto uložené informace mohou být využívány v rámci celého společenství. Jazyk standardizuje tři hlavní kroky procesu hodnocení: reprezentuje informace o konfiguraci systému pro testování, analyzuje současný



Obrázek 3.6: Funkce OVAL (převzato z literatury [21])

stav systémů (zranitelnosti, nastavení, záplaty atd.) a zobrazuje výsledky ohodnocení systému. Funkce standardu OVAL je znázorněná na obrázku 3.6.

Přínosy jazyka OVAL:

- Jednoduchost při určování zranitelnosti, nebezpečného nastavení nebo kontrola, zda jsou nainstalované dostupné záplaty na daném systému.
- Standardizované schéma, které popisuje nezbytné bezpečnostní nastavení.
- XML dokument pro detailní zápis konkrétního bezpečnostního problému.
- Podpora bezpečnostních specialistů, správců systémů a softwarových vývojářů.

Jazyk OVAL je založen na třech základních krocích:

1. Standardizované testy: Využívají OVAL *Definition schema*, které slouží pro zápis OVAL definicí v jazyce XML. OVAL definice zapisují specifikace stavového automatu, který určuje, kdy je systém zranitelný, kdy je systém v pořádku, umožňuje automatické testování systému atd. OVAL definice jsou vytvořeny podle doporučených bezpečnostních pravidel. Na vytváření těchto pravidel se podílí velká část profesionálů na počítačovou bezpečnost, kteří diskutují o detailech bezpečnosti a určují, jestli je zranitelnost přítomna v systému, zdali konfigurace odpovídá bezpečnostní politice nebo jestli systém obsahuje bezpečnostní

záplaty. Schéma pro popis OVAL definicí se skládá ze dvou částí. První část obsahuje základní (obecný) popis schématu a druhá část obsahuje individuální schémata pro testování různých systémů, platforem nebo aplikací. V druhé částí jsou například schémata obsahující testy pro operační systém UNIX, WINDOWS atd.

2. Získání informací ze systému: Využívá *OVAL System Characteristics schema* pro sběr dat ze systému pro testování. Definuje standard XML souboru pro reprezentaci konfiguračních informací systému, které obsahují parametry systému – například nastavení instalovaného softwaru a další bezpečnostně důležité informace o nastavení. Tedy schéma obsahuje charakteristiku systému. Porovnání je realizované s OVAL definicemi, viz krok 1. Na základě tohoto porovnání zjistíme aktuální bezpečnostní stav systému. Toto schéma může sloužit k výměně informací o systému mezi různými nástroji.

3. Zobrazení výsledků: Využívá *OVAL Results schema defines*. Jedná se o standard, který slouží pro zobrazování výsledků stavu systému. Výsledek obsahuje současný stav systému s danou systémovou konfigurací, který je porovnáván s množinou OVAL definicí. Schéma umožňuje aplikacím interpretovat výsledky a přijmout nezbytná opatření na zmírnění zranitelnosti a konfiguračních konfliktů, například instalace záplat nebo změnění bezpečnostní konfigurace systému.

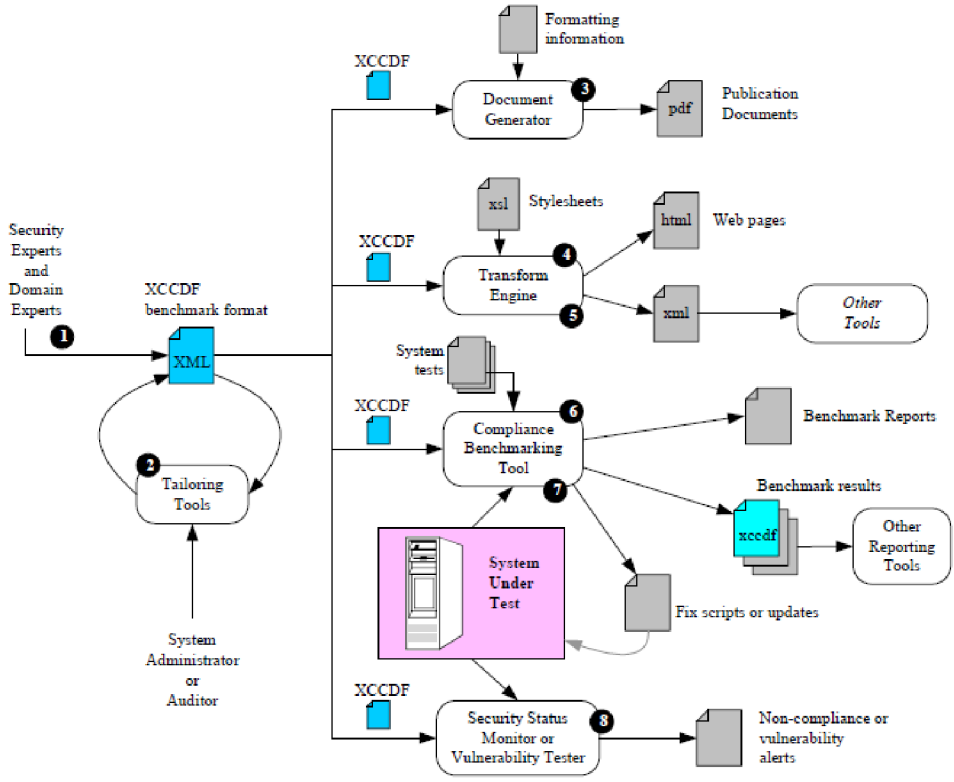
3.2.6 XCCDF (The Extensible Configuration Checklist Description Format)

Jedná se o standard, který definuje jazyk na vytváření dokumentů [13, 22, 23]. Cílem tohoto standardu je podpořit výměnu informací mezi různými nástroji aplikacemi, generování dokumentů, automatizované porovnávání obsahu vytvořených dokumentů. Standard definuje formát na uchování výsledku testování. XCCDF poskytuje jednotný základ pro vyjádření bezpečnostních kontrolních seznamů a dalších konfigurací souvisejících s bezpečností. XCCDF dokumenty jsou specifikované ve formátu XML. Kompletní popis XCCDF formátu najdete v literatuře [22].

Hlavním cílem XCCDF je umožnit bezpečnostní analýzu a vytvořit efektivní automatizované kontrolní seznamy IT specialistům. Na obrázku 3.7 jsou uvedeny některé případy užití XCCDF. Popis jednotlivých případů užití:

1. Odborníci na bezpečnost vytvoří bezpečnostní pokyny, kde jsou uspořádána pravidla ke konkrétním systémům nebo platformám. Aby mohl být tento požadavek reálný, musí být XCCDF otevřený, standardizovaný formát a musí umožňovat editování tohoto dokumentu různými nástroji. Musí být dostatečně přesně reprezentovány podmínky a vztahy systému, kterého se dané pravidlo týká. Také musí obsahovat potřebné informace pro provedení kontroly daného pravidla (navázání na OVAL 3.2.5).

2. Administrátoři systému mohou použít nástroje pro přizpůsobení bezpečnostních pokynů a seznamu podle jejich bezpečnostní politiky. Například pravidlo bude určovat požadavky na heslo (délka hesla, doba platnosti atd.) a administrátor tyto pravidla bude moci upravit.



Obrázek 3.7: Use case XCCDF (převzato z literatury [22])

3. Ačkoliv XCCDF je vytvořeno pro zpracovávání počítačem (XML), lze z něj vytvořit dokument čitelný pro uživatele.

4. Struktura dokumentu XCCDF by měla podporovat transformaci do HTML pro uvedení na web.

5. XCCDF dokument by měl jít také transformovat do jiných formátů XML, aby mohl být používán i jinými programy (podpora přenositelnosti).

6. Hlavním účelem XCCDF dokumentu je umožnit měnit jeho nastavení pomocí nástrojů pro zabezpečení. Takový nástroj by měl umět zpracovávat XCCDF dokument spolu s podporou pro definici testování systému, aby mohl určit, zda pravidla uvedená v XCCDF dokumentu cílový systém splňuje nebo ne. Po vyhodnocení pravidel by mělo být možno vytvořit zprávu o výsledku testování s ohodnocením (skóre 3.2.4).

7. Navíc některé nástroje mohou ke zprávě o výsledku využít XCCDF dokument a s ním související data, aby navrhly nápravu nebo upravily zjištěné zranitelnosti nebo chyby.

8. XCCDF dokument může být použit také jako skener pro zranitelnosti v systému. Otestuje, jestli je cílový systém odolný proti konkrétnímu druhu zranitelnosti. V tomto případě XCCDF dokument upozorňuje na zranitelnosti, umožňuje popsat zranitelnost a automatické ověření zranitelnosti v systému.

Kapitola 4

Analýza

V této kapitole si definujeme základní požadavky vytvářeného nástroje a základní návrh systému. Protože tento návrh odpovídá první iteraci v životním cyklu vývoje softwaru, není potřebné definovat všechny požadavky, ale jen ty základní. Stanovíme aktéry tohoto nástroje. Vytvoříme use-case diagram případu užití a na základě definovaných požadavků se pokusíme stanovit kritické požadavky, které by mohly zapříčinit neúspěch projektu.

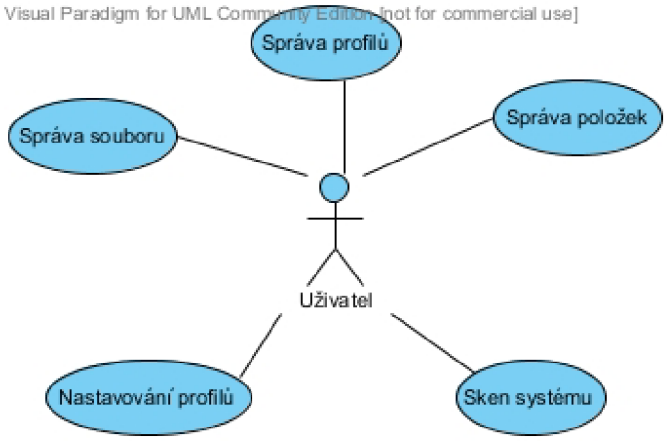
4.1 Požadavky

První podkapitola specifikuje požadavky na systém a zároveň je to i první krok analýzy. Tyto požadavky zpravidla určuje zákazník. V případě této diplomové práce to bylo vedení týmu Security standards firmy Red Hat. Vytvářený nástroj bude nadstavbou nad knihovnu OpenSCAP (viz kapitola 3), která využívá standardů SCAP. Tuto knihovnu tým Security standards vyvíjí a stále rozšiřuje. Jelikož to bude první nástroj nad touto knihovnou, bude vyžadována zpětná vazba z důvodu nalezení implementačních chyb a chybějících funkcí, které bude potřeba opravit nebo případně implementovat. V neposlední řadě bude také třeba konzultovat implementace funkcí, které jsou příliš pomalé, aby proběhla jejich optimalizace.

Jedním ze základních požadavků na tento nástroj je umožnit pracovat s dokumentem XCCDF (viz kapitola 3.2.6). Bude umět načíst a zobrazit relevantní informace z dokumentu XCCDF. Umožní načtené informace editovat a po editaci dokument uložit. Také by měl umět zkontrolovat, zda dokument odpovídá standardu XCCDF, a pokud ne, vhodnou formou o tom uživatele upozornit.

Editace dokumentu bude zahrnovat:

- Správa profilů (Edit profiles), vytváření, rušení a úpravu profilů, podle kterých se bude provádět skenování systému, na kterém nástroj poběží.
- Nastavování profilů (Tailoring) bude umožňovat zvolit, která pravidla, případně skupiny pravidel, se budou v systému kontrolovat (zda systém tato pravidla, popř. skupiny pravidel, splňuje či nikoliv). Umožní základní nastavení jednotlivých pravidel pro kontrolu



Obrázek 4.1: Obchodní model použití

(závažnost pravidla, hodnoty pravidla atd.). Jelikož v souboru XCCDF je velké množství pravidel a skupin, bude zapotřebí umožnit jejich vyhledávání, a tak zjednodušit práci s pravidly.

- Správa položek (Edit items), vytváření, rušení a úprava:
  - Pravidel (rules) pro zjišťování přítomnosti zranitelností na počítači.
  - Skupin (groups), ve kterých jsou sdružována související pravidla.
  - Hodnot (values), které využívají pravidla jako vstupní parametry při kontrole daného pravidla.

  V pravidlech a skupinách je spousta informací, které se budou moci editovat, proto by měl nástroj všechny tyto informace vhodně zobrazovat a umožnit k nim jednoduchý a přehledný přístup.
- Správa souboru (Edit Benchmark), změna základních informací o souboru XCCDF, také vytváření nového XCCDF souboru a zjišťování, zda je soubor validní.

Dále tento nástroj bude umožňovat zjištění přítomnosti zranitelností na tomto počítači (Scan). Skenování počítače bude probíhat na základě zvoleného profilu, který je uložen v XCCDF souboru. Výsledek skenování bude možno uložit pro případné další zpracování. Návrh systému bude zapotřebí provést tak, aby bylo možné tento nástroj do budoucna rozšiřovat o další funkce.

4.1.1 Aktéři

V tomto nástroji bude jen jeden aktér, a tím je uživatel. Není zapotřebí vytvářet jiné role, protože nástroj bude zjišťovat přítomnost zranitelných míst systému.

4.1.2 Funkční požadavky

Funkční požadavky definují očekávané služby nástroje, které zákazník po nástroji požaduje. V případě našeho nástroje je to tento základní seznam.

1. Nástroj bude umožňovat načítat soubor XCCDF, kde jsou uloženy profily, seznam zranitelností a související informace pro skenování počítače.
2. Nástroj bude umět skenovat počítač na základě vybraného profilu a zobrazit výsledek skenování.
3. Nástroj bude umět pracovat s profily, na základě nichž bude probíhat skenování počítače (vytváření, mazání, editace).
4. Nástroj bude umět pracovat s jednotlivými položky XCCDF souboru načítání, prezentace, filtrování, mazání, editace.
5. Nástroj bude moci nové nastavení souboru XCCDF vyexportovat.

4.1.3 Nefunkční požadavky

Nefunkční požadavky jsou požadavky na systém, které se netýkají chování systému. Zaměřují se spíše na kvalitu systému.

1. Nástroj bude implementován v jazyce Python.
2. Grafika nástroje bude tvořena pomocí knihovny GTK.
3. Nástroj bude využívat knihovnu OpenSCAP.
4. Nástroj bude odpovídat standardu SCAP.
5. Systém musí fungovat pod distribucí Fedora od firmy Red Hat.

4.1.4 Kritické požadavky

Kritickými požadavky máme na mysli oblast funkcí, které jsou nejzákladnějšími požadavky od zákazníka na systém. Tyto požadavky by měly tvořit okolo 20 % celkových požadavků. Proto jsem do těchto požadavků zařadil kontrolu počítače, který zjistí požadované nastavení a zranitelnost počítače a zobrazí tuto informaci uživateli. Navazujícím požadavkem je načtení XCCDF souboru, který obsahuje seznam pravidel pro testování, bez tohoto souboru by ani testování nemohlo být provedeno. Dalším navazujícím důležitým požadavkem je volba profilu, podle kterého se bude testovat počítač. Tento požadavek byl zvolen také proto, že ve velkém počtu případů nebude uživatel vytvářet pravidla pro testování, ale využije právě nadefinovaného profilu zkušeným uživatelem.

4.1.5 Model případů užití

Při vytváření modelu případu použití jsem vycházel z požadavků od týmu Security standards firmy Red Hat, který je popsán v předešlé podkapitole. Tento model by měl pokrývat všechny základní požadavky, které jsou na tento nástroj v současnosti kladeny.

4.1.6 Příklad případu užití

K diagramu s případy užití uvádím jeden textový případ užití nástroje. Záměrně jich zde neuvádím více, protože rozsah tištěné práce by tím neúměrně narostl a informační hodnota pro čtenáře by tomu neodpovídala, přestože textový popis je podstatnou součástí diagramu případu užití.

Případ použití:	Načtení XCCDF
ID:	1
Vytvořeno:	Vladimír Oberreiter
Popis:	Načtení souboru XML, který je v souladu se standardem XCCDF.
Primární aktéři:	Uživatel
Sekundární aktéři:	Žádný
Předpoklad:	Žádný
Následné podmínky:	Načtený dokument
Hlavní tok:	1. Případ použití se spustí, když uživatel zvolí načíst soubor. 2. Objeví se formulář pro vybrání souboru. 3. Uživatel vybere soubor a potvrdí. 4. Formulář pro vybrání souboru se zavře a nástroj načte dokument.
Následné podmínky:	Načten požadovaný dokument
Alternativní toky:	1. Storno

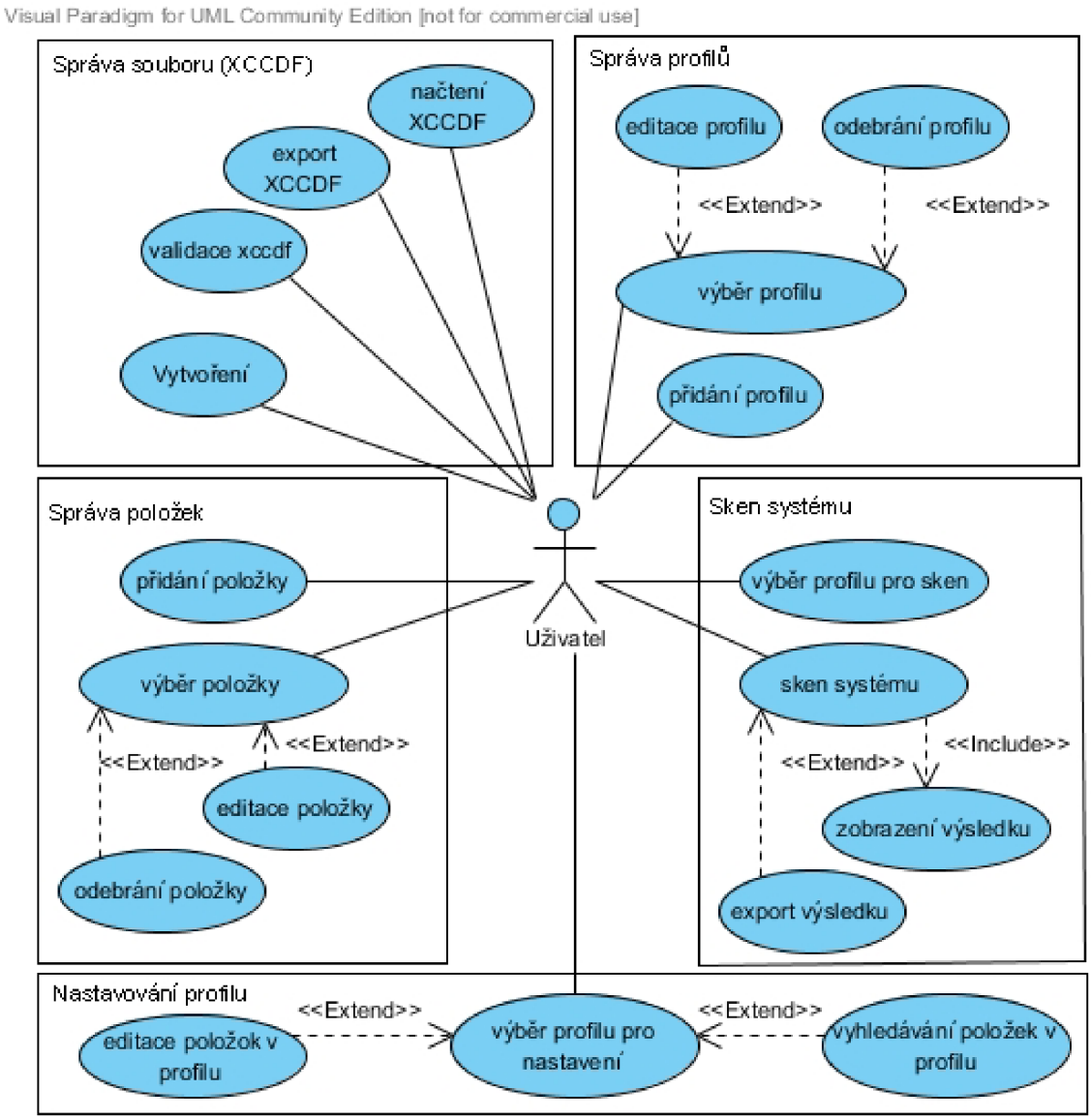
Tabulka 4.1: Příklad specifikace případu užití

Alternativní tok:	Storno
ID:	1.1
Popis:	Uživatel se rozhodl načtení ukončit.
Primární aktéři:	Uživatel
Sekundární aktéři:	
Předpoklady:	Uživatel stiskl tlačítko storno.
Alternativní tok:	
Následné podmínky:	Zavřelo se okno pro výběr dokumentu.

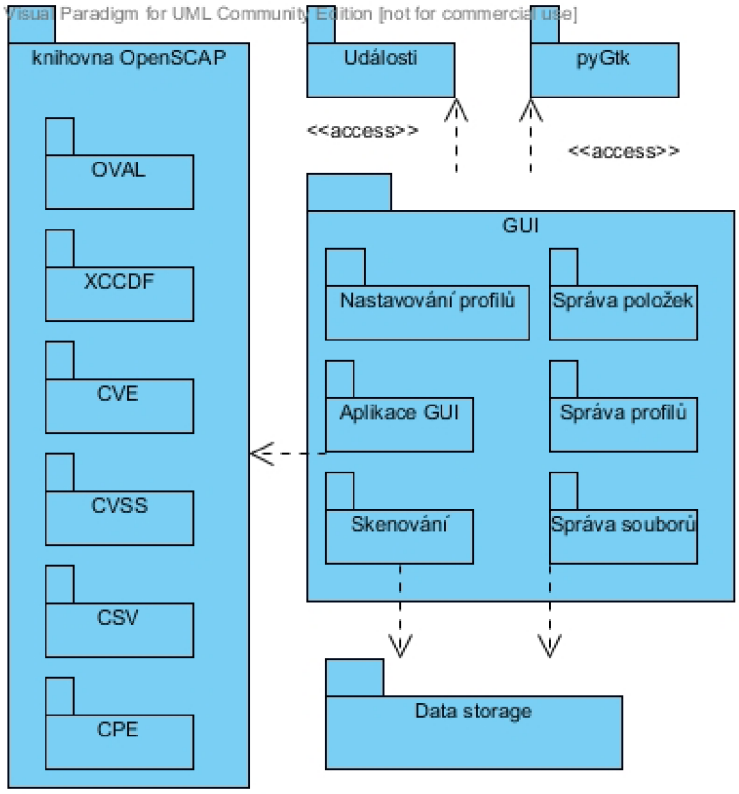
Tabulka 4.2: Alternativní tok případu užití 1.1

4.2 Analýza balíčků

Analýza balíčků seskupuje logicky prvky modelu, které jsou sémanticky podobné. Balíčky umožňují lepší pohled na architekturu systému. Tento návrh balíčků vychází z modelu případu užití. Z modelu balíčků můžeme také vidět závislosti mezi jednotlivými třídami.



Obrázek 4.2: Model případu užití



Obrázek 4.3: Analýza balíčků

V diagramu můžeme vidět, že práce a komunikace s knihovnou OpenSCAP je řízena pouze jedním balíčkem, který tvoří rozhraní mezi vytvářeným nástrojem a knihovnou OpenSCAP. Ostatní balíčky, které chtějí komunikovat s knihovnou OpenSCAP, musí využít tento balíček. Pro komunikace mezi jednotlivými balíčky se využívá událostí, které do budoucnosti budou zjednodušovat přidání nových balíčků využívajících knihovnu OpenSCAP. K vykreslování je využit balíček pyGtk.

4.3 Analýza struktury XCCDF

Tato podkapitola se zabývá analýzou struktury dokumentu XCCDF, jakým způsobem jsou v něm data uložena a jednotlivými závislostmi mezi daty. Bez této analýzy není možné provést návrh nástroje na zpracování souboru XCCDF. Seznámení se strukturou dokumentu XCCDF také usnadní pozdější implementaci nástroje [22].

Základní datový model obsahuje čtyři základní datové objekty (tyto objekty obsahují spoustu dalších objektů):

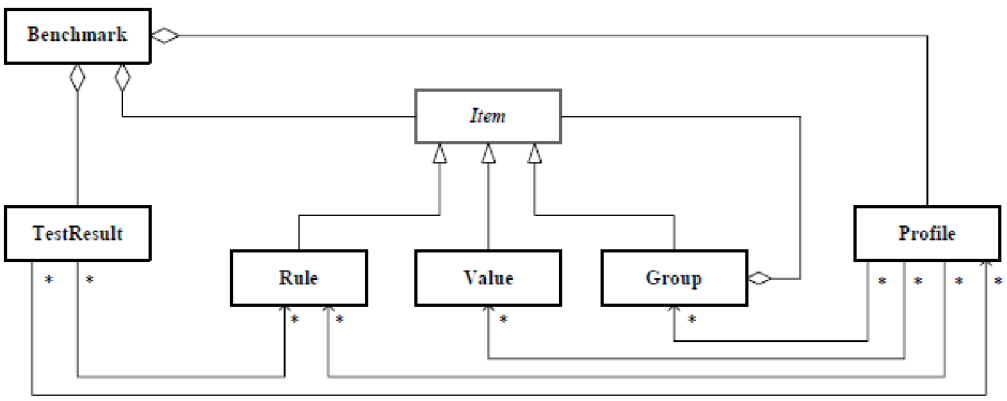
1. Hlavní objekt zvaný Benchmark je v dokumentu XCCDF pouze jednou. V tomto objektu jsou uloženy základní informace o dokumentu XCCDF a dále slouží jako kontejner pro ostatní objekty, které může obsahovat.

2. Položka (Item) je základní objekt pro Benchmark. Na tuto položku lze odkazovat pomocí unikátního identifikačního čísla a mimo jiné obsahuje vlastnosti pro její popis. Z položky je odvozeno několik dalších tříd:

   - Skupina (Group) může obsahovat libovolný počet ostatních položek. U skupiny lze zvolit, zda je vybrána či není vybrána. Pokud skupina není vybrána, tak také položky, které obsahuje, jsou implicitně nevybrány. Pokud je skupina vybrána, tak pravidla, která obsahuje, jsou použita pro skenování systému, pokud jsou vybrána.
   - Pravidlo (Rule), tato položka v sobě má uložen odkaz na kontrolu daného pravidla, dále obsahuje informace, například hodnotu závažnosti chyby. U pravidla je také možné zvolit, jestli je či není nastaveno, stejně jako u skupiny. K pravidlu mohou být přiřazeny hodnoty (Value).
   - Hodnota (Value), tato položka je pojmenovaná datová hodnota, která může obsahovat seznam hodnot pro výběr. Má uveden datový typ a může mít také uvedena metadata o tom, jak má být použita, jakých hodnot může nabývat, například číslo od 1 do 100. Tato hodnota může být substituovaná do ostatních položek.

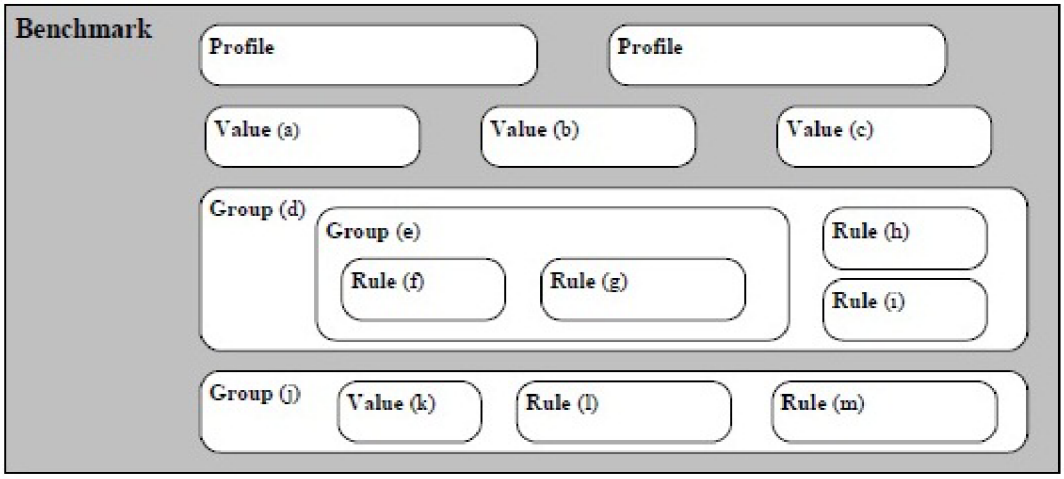
3. Profil (Profile) je kolekce atributů odkazujících na objekty skupin, pravidel a hodnot. Díky profilům je umožněno vytvářet různé nastavení dokumentu XCCDF. V profilech můžeme měnit nastavení specifikovaných položek a tím vytvářet profily šité na míru našemu systému. Podle nastavení profilu se provede skenování počítače.

4. Výsledky testů (TestResult) slouží k ukládání testů, které byly provedeny vzhledem k jednotlivým systémům.



Obrázek 4.4: Datový model XCCDF (převzato s literatury [22])

Závislosti mezi jednotlivými objekty jsou definovány následovně: Benchmark může obsahovat spoustu položek, ale každá položka může patřit pouze do jednoho Benchmarku. Také skupiny



Obrázek 4.5: Struktura XCCDF (převzato z literatury [22])

mohou obsahovat spoustu položek, ale žádná položka nemůže patřit do více skupin. Vzniká tedy stromová struktura položek, kde kořenový uzel stromu je tvořen hlavním objektem Benchmark, uzly tvoří skupiny a listy jsou tvořeny pravidly a hodnotami. Objekt profilu odkazuje na objekty skupin, pravidel a hodnot. Objekt výsledků testů odkazuje také na objekty skupin, pravidel, hodnot a také může odkazovat na objekt profilu. Výše popsaná struktura je znázorněna na obrázku 4.4. Definice skupin, pravidel a hodnot mohou být rozšířeny dalšími skupinami, hodnotami a pravidly (takto rozšířená položka dědí hodnoty vlastnosti).

Typická struktura Benchmarku je znázorněna na obrázku 4.5. Benchmark zpravidla obsahuje jednu nebo více skupin, každá skupina obsahuje nějaká pravidla, hodnoty a další skupiny. Skupiny umožňují autorovi seskupovat související skupiny a pravidla do struktury, popsat je a také uložit odkazy související s těmito položkami. Díky skupinám může uživatel vybírat pravidla a skupiny, které spolu souvisejí.

4.4 Závěr analýzy

V průběhu analýzy byly s firmou Red Hat prodiskutovány všechny nejdůležitější případy použití. Na základě informací od firmy Red Hat a zjištěných informací během analýzy se nejeví žádný problém, který by mohl zabránit úspěšné implementaci projektu.

Nevýhodou je, že knihovna OpenSCAP je stále ve vývoji a některé části nejsou ještě stoprocentně implementované. Úspěch implementace projektu závisí tedy i na programátorech firmy Red Hat. Jelikož jsem se o projekt okolo knihovny OpenSCAP zajímal už více než před rokem, vidím, jak velký pokrok byl proveden, a doufám, že tato nevýhoda nebude komplikací, i když za tu dobu se práce s knihovnou hodně změnila a některé zkušební implementace, které byly vytvořeny, už nejsou nyní na aktuální knihovně OpenSCAP funkční.

Kapitola 5

Návrh

Tato kapitola se zabývá návrhem nástroje a vychází z předcházející analýzy požadavků. Fáze návrhu zahrnuje upřesnění posledních požadavků a navrhnutí architektury nástroje. Firma Red Hat klade velký důraz na přehledné grafické prostředí, které by mělo být co možná nejintuitivnější. Samozřejmě nelze očekávat, že tento nástroj, který umožňuje celou řadu nastavení souvisejících se skenováním počítače (nastavování závažnosti chyb, souvisejících pravidel, metrik pro CVSS, nastavování seznamu hodnot pro výběr vhodných parametrů při skenování a mnoho dalších), bude uživatel umět používat v plném rozsahu, aniž by o této problematice něco věděl. Nástroj bude vhodný spíše pro zkušené uživatele. Ale obyčejný uživatel, který si jej stáhne (i se specifikovanými dokumenty pro kontrolu XCCDF a OVAL), nainstaluje si jej a bude si chtít zkontrolovat, zda systém odpovídá pravidlům v dokumentu XCCDF, by měl umět po krátké nápovědě tento nástroj bez problému použít. Tento požadavek je podpořen tím, aby nástroj, který bude vytvořen, byl komerčně úspěšný.

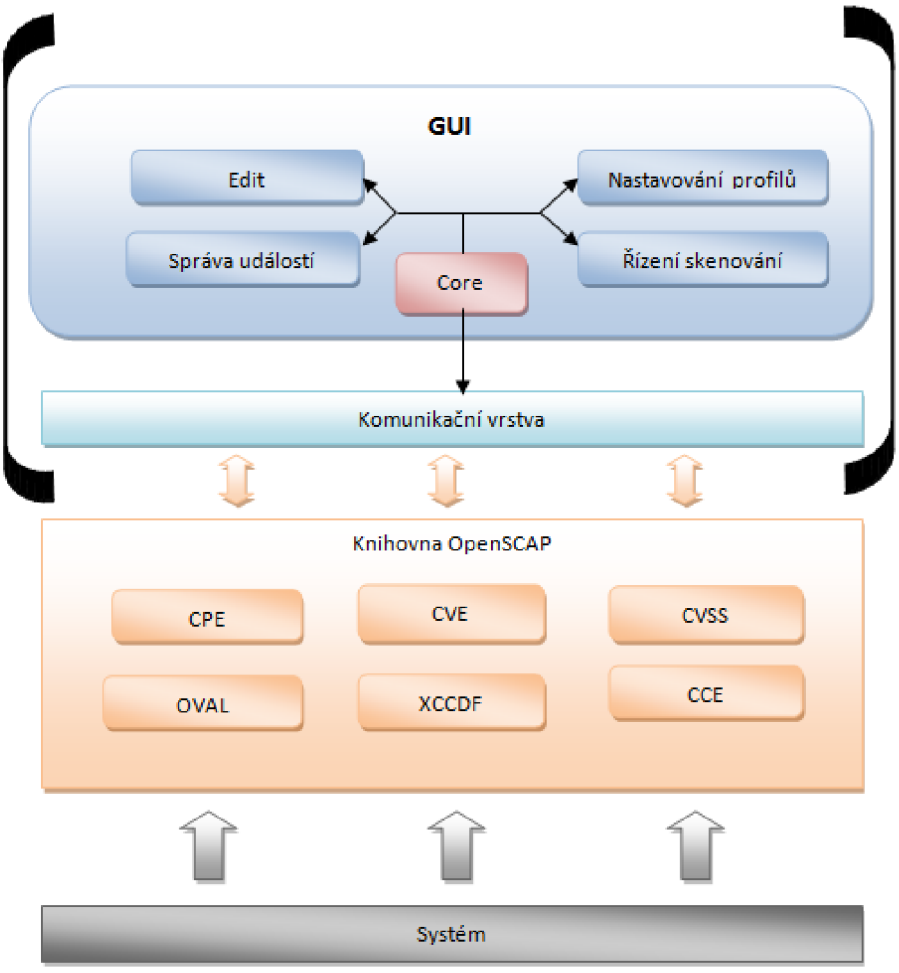
Z výše uvedeného plyne, že nástroj je z hlediska uživatele rozdělen do dvou částí: skenování s výběrem profilu, podle kterého se samotné skenovaní bude provádět, a zobrazení výsledku, případně export výsledku. Druhá část bude zaměřena na zkušené uživatele, kteří si budou moci uzpůsobit dokument XCCDF dle svých požadavků.

5.1 Architektura nástroje

Architektura nástroje se zabývá fyzickým rozdělením systému, vzájemným propojením jednotlivých částí a komunikací. Důležitou částí je knihovna OpenSCAP, která zjednodušuje práci se soubory pro konfiguraci systému a je také využívána pro kontrolu systému pro zjišťování zranitelností přítomných na systému. Tato knihovna musí být instalovaná na systému uživatele, který chce nástroj na kontrolu zranitelností využívat. Další částí je komunikační vrstva mezi touto knihovnou OpenSCAP a grafickým prostředím, které obsahuje kompletní management pro správu testování. Na obrázku 5.1 je znázorněná výše popsaná architektura.

Nyní popíšu funkci jednotlivých modulů v architektuře:

Core Jak vidíme na obrázku 5.1, tento modul je základním jádrem programu, který řídí spo-



Obrázek 5.1: Architektůra nástroje ve spolupráci s knihovnou OpenSCAP

lupráci všech ostatních částí nástroje a obsahuje důležité informace související s během programu, které musí byt dostupné pro všechny ostatní součásti.

Správa událostí Tento modul se také stará o spolupráci mezi ostatními moduly a umožňuje jejich synchronizaci. Do tohoto modulu se každý objekt může zaregistrovat, ať již jako příjemce nebo jako vysílající. Pokud se objekt zaregistruje jako vysílající, může vyslat událost kdykoliv potřebuje, a objekty, které jsou zaregistrovány pro příjem této události, můžou provést požadovanou operaci.

Nastavování profilů Stará se o profily, které umožňují nastavit, jestli dané pravidlo půjde kontrolovat nebo ne. Umožňuje nastavovat hodnoty, které se využijí jako vstupní parametry při kontrole.

Edit Tento modul umožňuje editaci dokumentu XCCDF (vytváření, mazání a úprava profilů, skupin, pravidel a hodnot).

Řízení skenování Tento poslední modul se stará o skenování systému a zobrazení výsledku po ukončení skenování.

5.1.1 Diagram tříd

Na obrázku diagramu tříd 5.2 je znázorněn logický pohled na statickou strukturu nástroje. Tento diagram znázorňuje třídy a vztahy mezi těmito třídami.

Stručný popis jednotlivých tříd:

Core Tato třída slouží k řízení aplikace a zajišťuje komunikaci mezi třídami. Ke komunikaci a synchronizaci mezi třídami využívá také třídu HandlerEvents.

HandlerEvents Tato třída slouží k registraci událostí a jejich rozesílání třídám, které se registrovaly pro jejich příjem. Po přijetí události je provedena odpovídající funkce.

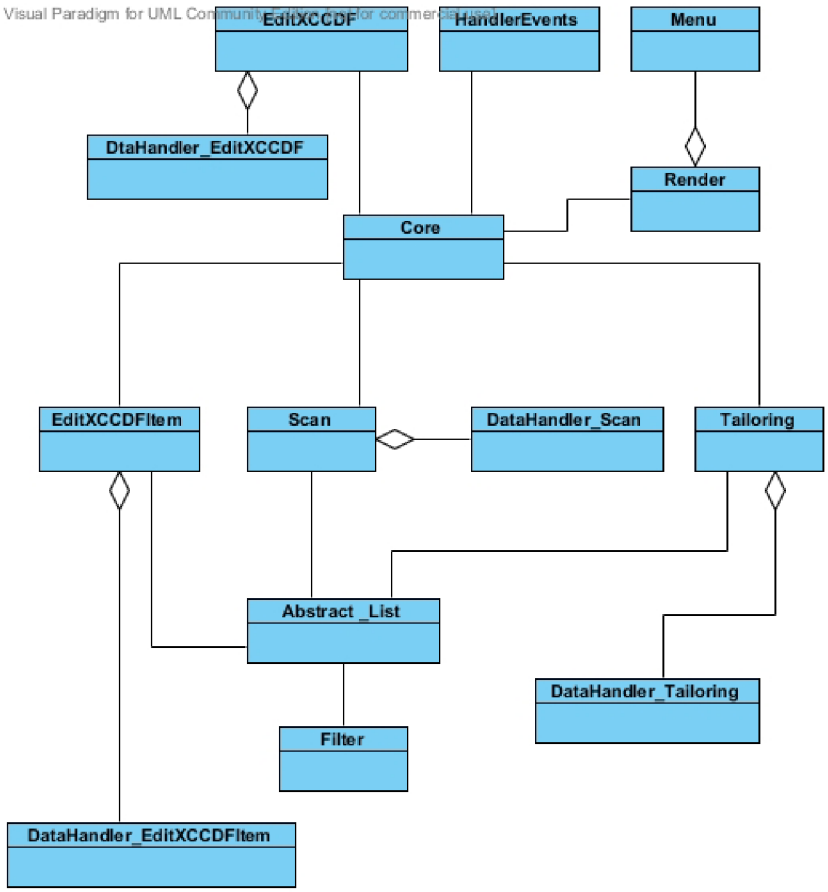
Render Tato třída se stará o vykreslení hlavního okna aplikace a hlavního dvouúrovňového menu aplikace a dále se stará o zobrazování grafického rozhraní souvisejícího s tlačítky menu.

Menu Tato třída se stará o hlavní menu aplikace, které je dvouúrovňové. Je navržené tak, aby do něj šly v budoucnu pomocí balíčků přidávat nové položky menu, a tak aplikaci rozšiřovat o novou funkčnost.

Abstract_List Tato třída je rozšíření widgetu TreeList, které mimo jiné umožňuje vyhledávání a filtrování v TreeListu.

Filter Tato třída se stará o grafické rozhraní k funkcím filtru a vyhledávání. Zadané informace o filtrování zprostředkovává třídě Abstract_List, která filtrování, vyhledávání provede.

EditXCCDF Grafické rozhraní k třídě DataHandler_EditXCCDF.



Obrázek 5.2: Diagram tříd

DataHandler_EditXCCDF Tato třída se stará o načítání dokumentu XCCDF, jeho export a také kontrolu validace dokumentu XCCDF.

Tailoring Grafické rozhraní k třídě DataHandler_Tailoring.

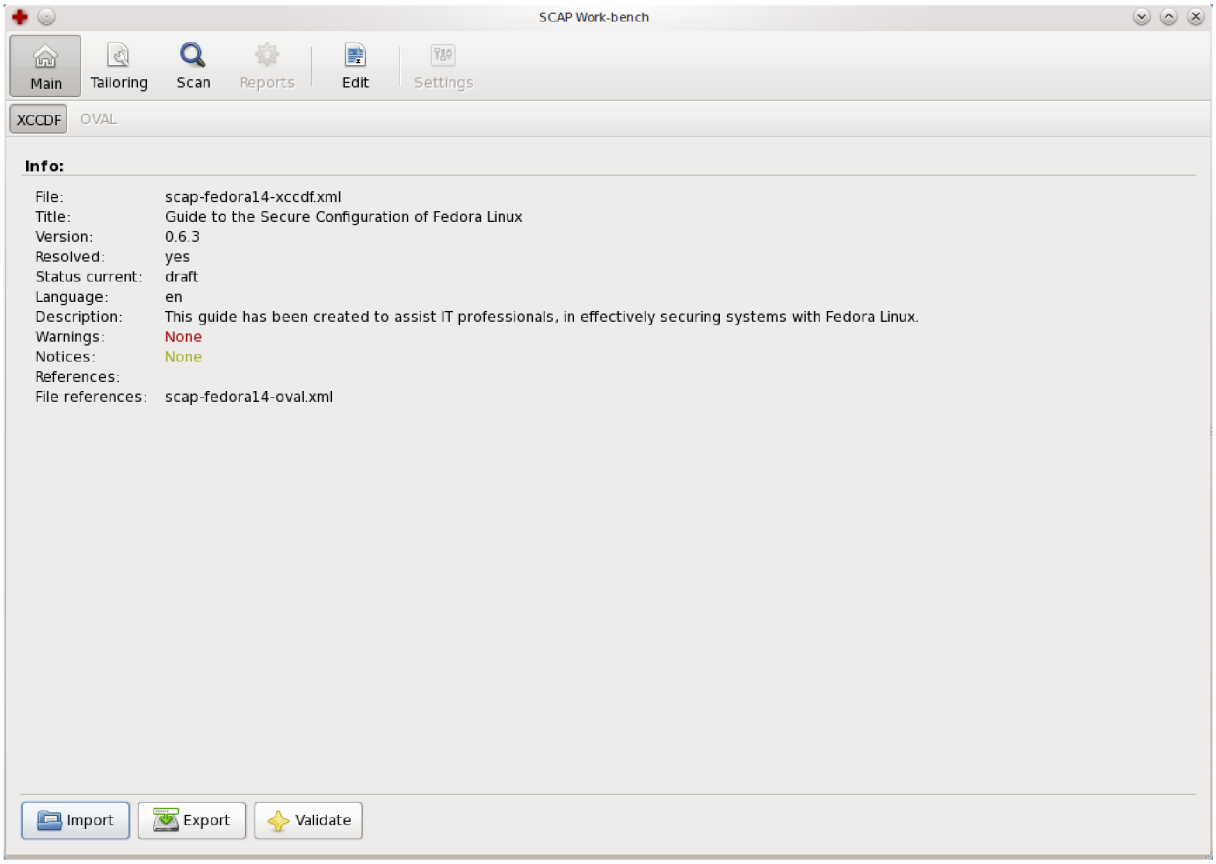
DataHandler_Tailoring Tato třída se stará o úpravu profilů, podle kterých se provádí skenování systému (např. výběr pravidel, které se budou skenovat).

Scan Grafické rozhraní k třídě DataHandler_Scan.

DataHandler_Scan Tato třída se stará o skenování systému podle zvoleného profilu, zobrazení výsledku a možnost exportu výsledku do XML dokumentu pro případné pozdější zpracování.

EditXCCDFItem Grafické rozhraní k třídě DataHandler_EditXCCDFItem

DataHandler_EditXCCDFItem Tato třída se stará o zobrazení všech skupin s pravidly pro jejich editaci.



Obrázek 5.3: Hlavní okno

5.2 Grafický návrh jednotlivých částí

V návrhu jednotlivých částí podrobněji popíšu, co by jednotlivé části měly obsahovat a návrh grafického rozhraní. Nebudu zde uvádět všechny návrhy, ale jen výběr těch nejdůležitějších, na kterých systém stojí.

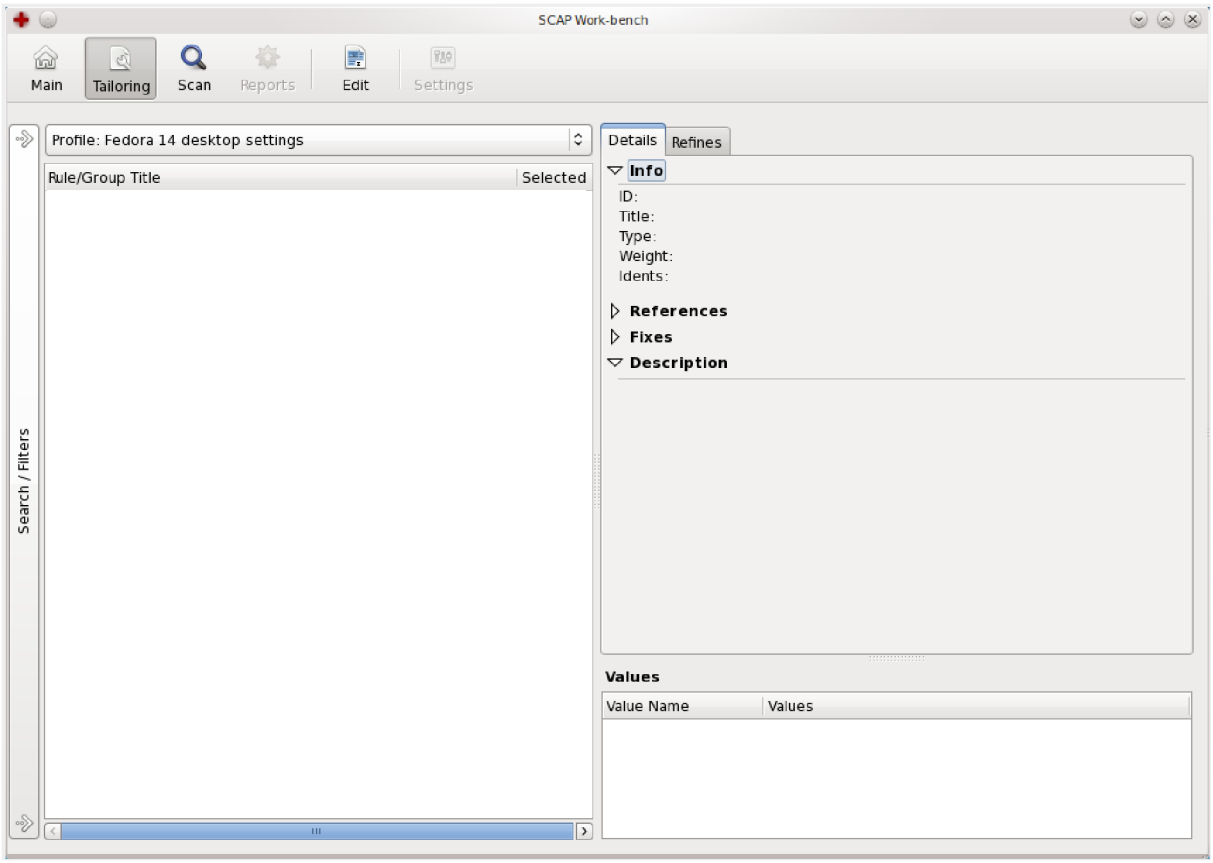
Hlavní okno

V hlavním okně nástroje se bude provádět načítání dokumentu, kontrola jeho validace, export editovaného dokumentu a samozřejmě vytvoření nového dokumentu XCCDF. Po načtení dokumentu se zobrazí základní informace o tomto dokumentu. Tyto informace nepůjdou měnit, měnit je lze v Editu. Pokud ve stejném adresáři bude i dokument OVAL, na který dokument XCCDF odkazuje, načte se i on a zobrazí se o něm základní informace. Grafický návrh hlavního okna je na obrázku 5.3.

Nastavování profilů (Tailoring)

Formulář s profily bude umožňovat výběr profilu. V této části se profily nebudou moci vytvářet ani mazat, to bude ponecháno do části Edit, kterou budou využívat zkušení uživatelé. Ve vybra-

ném profilu se bude volit, zda se dané pravidlo použije při skenování nebo ne. Aby se pravidlo mohlo použít, musí být povoleny všechny skupiny, které jsou nad pravidlem. Pro výběr pravidel a skupin bude k dispozici filtr a vyhledávání, které zjednoduší orientaci v seznamu položek (skupiny, pravidla). Po vybrání se zobrazí detailní informace o položce, budou se moci pravidlu nastavit hodnoty, které se využijí při skenování, a další hodnoty pro určení závažnosti zranitelnosti, kterou pravidlo kontroluje, bližší popis těchto hodnot v literatuře [22]. Grafický návrh je na obrázku 5.4



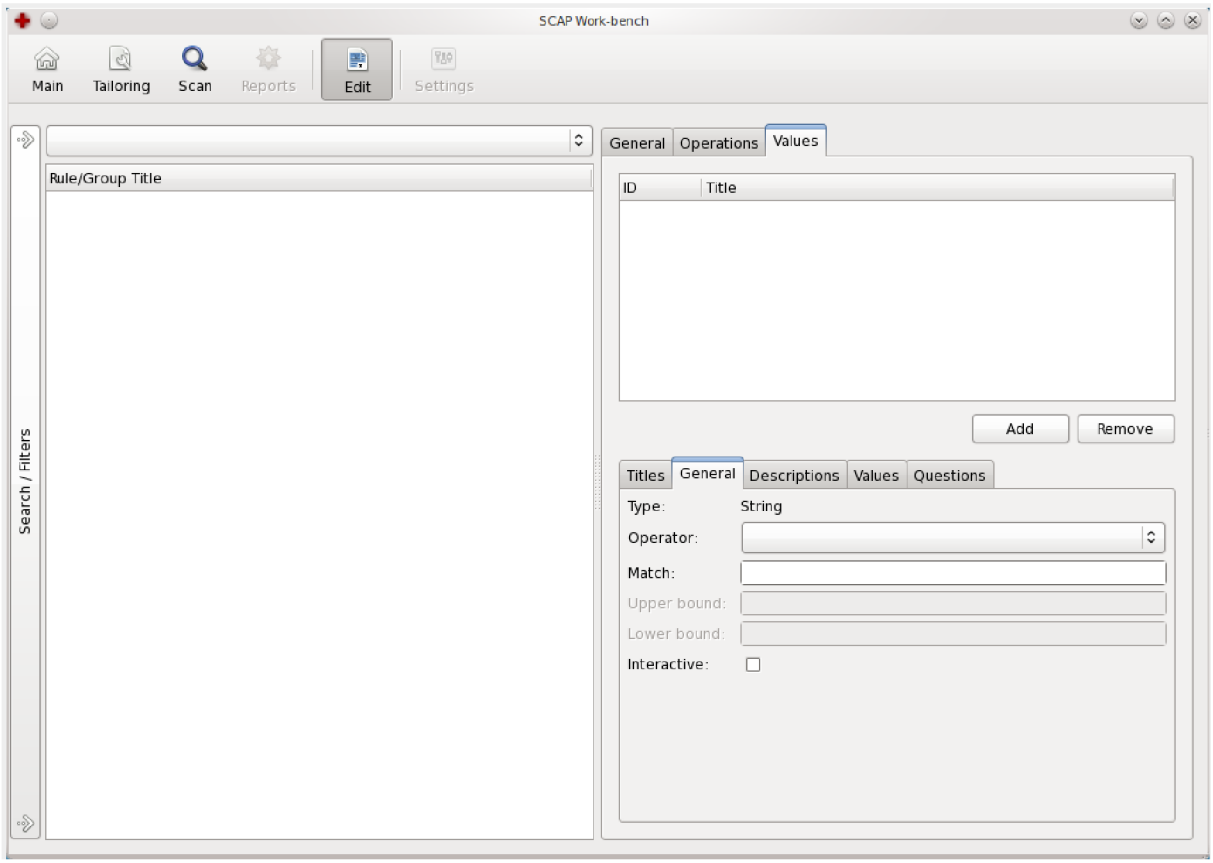
Obrázek 5.4: Grafický návrh pro profily

Edit

Ve formuláři Edit se bude moci přepínat mezi správou položek dokumentu XCCDF, správou profilů a správou Benchmarku. Grafický návrh editace je na obrázku 5.5.

Jednotlivé správy budou obsahovat:

- Správa položek umožní editovat skupiny, pravidla a hodnoty, bude zobrazovat jejich detailní informace a bude je umožňovat měnit. Každá položka obsahuje velké množství informací, které se u ní mohou nastavit - název, popis, seznam souvisejících položek, seznam konfliktních položek, hodnoty nabízející se pro výběr v profilu přes platformu, pro kterou



Obrázek 5.5: GUI návrh pro editaci položek XCCDF

tato položka platí, atd. Bližší popis všech hodnot, které se vážou k dané položce, naleznete v literatuře [22]. Z tohoto důvodu bylo při navrhování této karty nejnáročnější, jak informace rozdělit do skupin, aby uživatel potřebné informace intuitivně našel, a vhodně je zobrazit uživateli, aby je mohl editovat. Návrh tohoto formuláře trval dlouho, protože se vytvářel v několika etapách a mnohokrát se dle požadavku firmy Red Hat přepracovával.

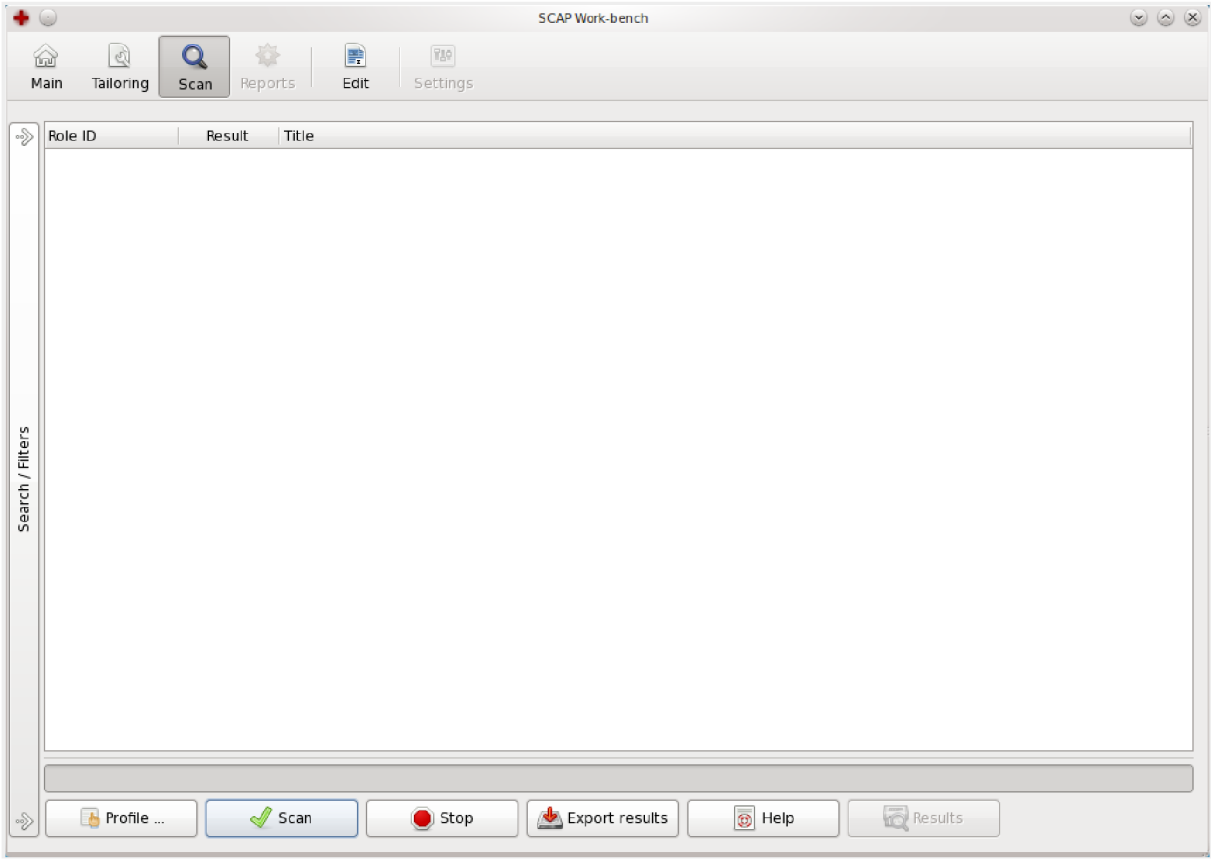
- Správa profilů umožní vytvářet, mazat a upravovat základní informace o profilech (například popis profilu), nebude zde možno nastavování profilů pro skenování, které půjde provádět pouze v Nastavování profilu (Tailoring).

- Správa Benchmarku umožní nastavovat základní informace o dokumentu XCCDF (název, popis, verze atd.).

Skenování systému

Formulář pro skenování bude obsahovat výběr profilu pro skenování a tlačítka pro ovládání skenování (start, stop, export výsledku skenování). Také by měl obsahovat filtr a vyhledávání, které ulehčí orientaci ve výsledku skenování. Výsledek skenovaných pravidel v systému bude barevně rozlišen podle závažnosti, a to od červené barvy, která indikuje FAIL (pravidlo nesplněno), až

po zelenou, která indikuje PASS (pravidlo splněno). Grafický návrh editace je na obrázku 5.6.



Obrázek 5.6: GUI návrh pro skenování systému

5.3 Akceptační testy

Jedním z požadavků je vytvořit protokol o testování nástroje, proto při návrhu vytvořím několik základních akceptačních testů, na kterých jsem se dohodl s firmou Red Hat. Tyto testy budou převážně pokrývat množinu kritických požadavků, které jsou popsány v kapitole analýzy. Konkrétně jsou to tyto oblasti:

- Editace pravidel, podle nichž se provádí skenování a editace skupin, do kterých jsou pravidla umístěna.
- Úprava profilů pro skenování.
- Skenování systému a zobrazení výsledku uživateli.

Tyto testy budou provedeny po dokončení implementace. Podrobný popis nejdůležitějších akceptačních testů najdete v příloze.

5.4 Závěr návrhu

V předchozí kapitole byl popsán návrh nástroje pro práci s knihovnou OpenSCAP. Byl proveden návrh architektury nástroje a jeho spolupráce s knihovnou OpenSCAP. Byl navržen diagram tříd a popsány jednotlivé třídy. Nakonec bylo navrhnuto grafické prostředí pro načítání, exportování dokumentu XCCDF a jeho validaci, nastavování profilů pro skenování, správu skupin, pravidel, hodnot, profilů a Benchmarku a skenování systému na základě vybraného profilu a zobrazování výsledku.

Návrh grafického prostředí probíhal v úzké spolupráci s firmou Red Hat. Grafický návrh byl vytvářen v iteracích, kde vždy po vytvoření nějaké části návrhu byl tento návrh představen firmě Red Hat a byl zkonzultován, na základě konzultace byl návrh dále upravován až do výsledné podoby. Někdy bylo vytvořeno i více návrhů stejného druhu, aby při konzultaci mohl být vybrán ten nejlepší.

Kapitola 6

Implementace

Implementace se již od začátku prolínala s procesem návrhu nástroje. V části návrhu nástroje byly vytvářeny prototypy nástroje, na kterých byl prezentován jeho grafický vzhled. Tyto prototypy se upravovaly dle požadavků zákazníka (firma Red Hat) až do výsledné podoby nástroje, která byla schválena. Díky tomuto postupu byl návrh úspěšný a ušetřil mnoho času při implementaci, aby se jednotlivé části nemusely předělávat. I přesto se na základě zpětné vazby návrh obměnil v pokročilé fázi implementace.

Implementace byla rozdělena do několika fází. První byla načítání dokumentu XCCDF, nastavování profilu, skenování, správa pravidel a skupin v souboru XCCDF a správa profilů. Tento postup byl zvolen proto, aby nástroj byl funkční už během vývoje a mohl se uvést jeho první release, díky čemuž jsem mohl získávat zpětnou vazbu na funkci nástroje a grafický vzhled nástroje. Paralelně s implementací probíhala i úprava knihovny OpenSCAP na základě zjištěných nedostatků během implementace (úprava chybných funkcí a implementace některých nových funkcí, na které se během její tvorby pozapomnělo). Bohužel díky těmto požadavkům na úpravu knihovny OpenSCAP se implementace nástroje značně protahovala, protože během úprav knihovny OpenSCAP práce na nástroji stagnovala.

6.1 Projektový tým

Na vývoji nástroje pro knihovnu OpenSCAP se podílel tým dvou lidí. Kromě mě byl vývoji přiřazen ještě zkušený programátor z týmu Security standards firmy Red Hat, jehož úlohou bylo pomáhat řešit problémy při práci s knihovnou OpenSCAP, podílet se na implementaci nástroje a dohlížet na kvalitu mé práce. Moje úloha byla kompletní analýza a návrh nástroje, implementace probíhala ve spolupráci s výše zmíněným programátorem firmy Red Hat.

6.2 Použité technologie

Nástroj na měření zranitelnosti a editaci XCCDF je vytvořen v jazyce Python, grafika je implementována pomoci grafického toolkitu GTK+ pro Python (knihovna pygtk). Pro samotné vy-

tváření formulářů byl použit nástroj Glade, jehož velkou výhodou je oddělení grafického návrhu od jádra aplikace. Výstup nástroje Glade je XML soubor, který obsahuje kompletní informace o vzhledu, a v programu si stačí vyžádat konkrétní grafický objekt (widget), se kterým chceme pracovat. Tento přístup podporuje třívrstvou architekturu a při požadavku na změnu grafického prostředí se v řadě případů nemusí měnit kód aplikace. Díky jazyku Python je tento nástroj přenositelný i na jiné platformy než Linux, na kterém byl nástroj vyvíjen, což podstatně zvětšuje skupinu uživatelů, kteří by mohli mít o daný nástroj potencionální zájem. Pro vytváření diagramu byly zvoleny bezplatné nástroje Umbrello a Visual Paradigm for UML Community Edition 8.1. Vývoj nástroje probíhal na systému Fedora od firmy Red Hat, do jehož distribuce bude i po implementaci zahrnut.

6.3 Změny v návrhu

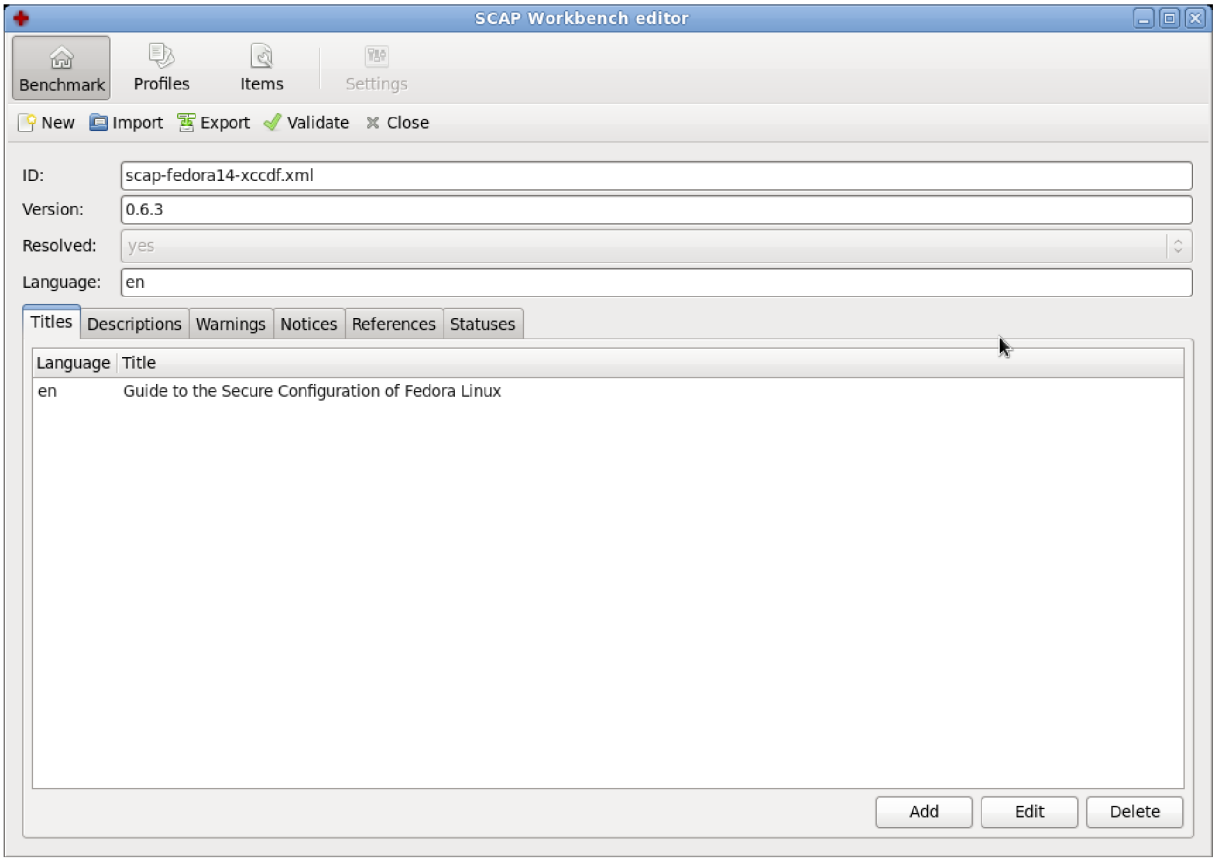
Oproti návrhu byl nástroj rozdělen na dva samostatné spustitelné programy (z důvodu implementačních závislostí mezi jednotlivými moduly), aby nedocházelo ke konfliktům mezi editací dokumentu a nastavováním profilů pro skenování. První spustitelný program umožňuje nastavování profilu, na základě těchto profilů dále umožňuje skenovat systém a zobrazit výsledek. Druhý program umožňuje správu skupin, pravidel, hodnot, profilů a hlavních informací o dokumentu XCCDF (Benchmark). Tím došlo i ke striktnímu oddělení části programu, která je jen pro zkušené uživatele. Z hlediska návrhu to znamená, že návrh Editace byl vyčleněn do samostatného spustitelného programu a do tohoto nově vzniklého samostatněspustitelného programu byly doplněny funkce z hlavního okna (načtení dokumentu XCCDF, export, validace atd.). Grafický návrh této změny je na obrázku 6.1.

Dále došlo k drobným změnám v návrhu pro tailoring, správu skupin, pravidel a hodnot, které vedly k dalšímu zjednodušení a zpřehlednění správy těchto položek. Tyto změny byly prováděny na základě zpětné vazby od uživatelů, kteří tento systém zkoušeli.

6.4 Řešené problémy

Jeden z problémů při implementaci byl s načítáním struktury XCCDF, kdy se načítá velké množství dat, proto se muselo zvolit načítání postupné, podle informace, které uživatel vyžadoval. V praxi to znamená načítat veškeré informace, až je uživatel skutečně potřebuje, ale základní informace o všech pravidlech a skupinách musely být načteny vždy na začátku editace (pokud to bylo potřeba). I přesto, že načítání ostatních informací bylo až na základě vybraného pravidla, hodnoty či skupiny, načítání trvalo příliš dlouho. Tento problém s dlouho trvajícím načítáním byl vyřešen pomocí vícevláknového zpracování, aby při načítání program „nezamrzl“ a uživatel mohl případně pracovat v jiné části nástroje.

Problém s dlouho trvající odezvou byl také řešen při provádění skenování, kde by bylo pro uživatele nepříjemné, kdyby program „zamrznul“ až na několik minut (podle počtu testovaných pravidel, specifikovaných v profilu zvoleném pro skenování systému). Tento problém byl také



Obrázek 6.1: Změněná grafická podoba pro část Edit

vyřešen pomocí vícevláknového zpracování, podobně jako problém popsaný v odstavci výše.

Vyskytl se také problém závislosti mezi správou dokumentu XCCDF v sekci Edit a sekcí skenování a nastavováním profilu (Tailoring), který nakonec vedl k rozdělení nástroje na dva spustitelné programy. Jeden umožňuje skenování a nastavování profilů pro skenování a druhý umožňuje editaci dat v dokumentu XCCDF. Tato změna je popsaná v 6.3.

Další drobnou nepříjemností byla práce s GTK+. Jelikož tento toolkit pro práci s grafikou není tak propracovaný a některé možnosti jsou v něm omezeny, musel jsem některé grafické objekty upravovat a některé záměry musely být upraveny vzhledem k možnostem GTK+.

6.5 Testování dle akceptačních testů

Všechny dohodnuté testy s firmou Red Hat viz 5.3, které jsou sepsány v příloze, byly úspěšně provedeny. S jejich výsledkem byla firma Red Hat seznámena. Samozřejmě testy, které jsou sepsány v příloze, nejsou všechny, které byly provedeny na implementovaném nástroji. Je to jen zlomek seznamů nejzákladnějších testů nástroje, které byly dohodnuty, že musí být úspěšné, aby byl projekt uznán a mohl být začleněn do distribuce. Nástroj byl také testován během implementace, podle toho, jak byly jednotlivé části implementovány.

6.6 Závěr implementace

Implementace byla provedena v požadovaném rozsahu a u žádných částí, které byly navrženy v sekci návrh, se nevyskytl tak závažný problém, aby bránil implementaci nástroje. Změny oproti původnímu návrhu jsou uvedeny v 6.3.

V příloze se nachází postup instalace tohoto nástroje, stručný popis, jak s tímto nástrojem pracovat, a na přiloženém CD se nachází soubor s XCCDF, který lze použít pro práci s nástrojem. [1] Samotný nástroj na zjišťování zranitelností je také na přiloženém nosiči CD.

[1] XCCDF soubor je také možno stáhnout z http://www.open-scap.org, nachází se ve zdrojových souborech knihovny OpenSCAP.

Kapitola 7

Závěr

Na projektu jsem začal pracovat více než před rokem a během této doby se některé přístupy k práci s knihovnou OpenSCAP značně změnily. Tyto změny probíhaly v době analýzy, kdy jsem ještě s knihovnou nepracoval, ale některé změny proběhly i během návrhu. Důsledkem toho jsou některé vytvořené prototypy, které sloužily v době návrhu k prezentaci na pravidelných schůzkách, kde se jednalo o návrhu, v dnešní době nefunkční. Naštěstí v době implementace už žádné zásadní změny neprobíhaly. Změny, které probíhaly v době implementace, se týkaly především opravy funkcí, které nepracovaly správně nebo byly příliš pomalé. Několikrát se stalo, že potřebná funkce vůbec neexistovala a musela se doimplementovat. To, že knihovna OpenSCAP je stále ještě ve vývoji, podstatně prodloužilo dobu implementace, protože v některých případech se čekalo, až se daná funkčnost opraví nebo doimplementuje. Naštěstí práce na nástroji začala v dostatečném předstihu a vše se stihlo v požadovaném termínu. Zjišťovat nedostatky v implementaci knihovny OpenSCAP byl i jeden ze zadaných úkolů, protože nástroj na měření zranitelností, který jsem vytvářel, je první program tohoto typu, který knihovnu OpenSCAP využívá.

V projektu jsem splnil všechny body zadání, seznámil jsem se s knihovnou OpenSCAP a s protokolem automatického managementu zranitelností (SCAP). Provedl jsem analýzu a návrh nástroje na měření zranitelností operačního systému a editaci XCCDF databáze zranitelností pro systém Fedora od firmy Red Hat. Pomocí knihovny OpenSCAP jsem implementoval uvedený nástroj v jazyce Python, který byl zvolen pro svou přenositelnost. Výsledný nástroj jsem otestoval ve spolupráci s firmou Red Hat, protokol o testování nástroje je v příloze. Analýza možnosti dalšího rozšíření nástroje je popsána v sekci 7.3. Projekt byl úspěšně začleněn do distribuce Fedora.

7.1 Zhodnocení projektu

Projekt měl pro mě velký přínos, a to především v seznámení se standardy pro bezpečnost a s přístupy zjišťování zranitelností. Rozšířil jsem si znalosti o platformě Linux, naučil se programovat v jazyce Python, ve kterém jsem nikdy dříve neprogramoval, a musím říci, že mě práce v jazyce

Python bavila. Dále jsem si také prohloubil znalosti o grafickém toolkitu GTK+ pro Python (tedy pyGtk). V neposlední řadě to byla zkouška, zda dokážu pracovat v týmu programátorů pro větší firmu a zda bych mohl být jejich platným členem, což mi pomůže při rozhodování, kam směřovat svoje úsilí při hledání zaměstnání po ukončení studia.

7.2 Vlastní přínos

Můj přínos do projektu spočíval v kompletní analýze, návrhu a případných změnách v návrhu nástroje. Můj přínos dále spočívá v implementaci nástroje a v neposlední řadě ve zpětné vazbě programátorům týmu Security standards firmy Red Hat, kteří na základě zjištěných informací při implementaci upravovali knihovnu OpenSCAP, ať již se jednalo o úpravu stávajících funkcí (špatná funkčnost či jejich optimalizace), nebo implementaci nových funkcí, které v dosavadní implementaci chyběly.

7.3 Možnosti rozšíření

Tento projekt rozhodně není u konce, je hotová základní část, která umožňuje skenování počítače na základě vybraného profilu a jeho nastavení, dále umožňuje upravovat dokument XCCDF podle standardu SCAP.

Nástroj by měl být do budoucna rozšířen o práci s dokumentem OVAL (XML soubor, který je v souladu se standardem OVAL), který obsahuje potřebné informace o jednotlivých zranitelnostech pro skenování systému. Bez těchto informací nemůže být pravidlo, které je obsaženo v XCCDF skenováno. Rozšíření o úpravu dokumentu OVAL je dalším naplánovaným krokem vývoje tohoto nástroje a od začátku návrhu se s tímto rozšířením počítá, protože mezi dokumenty XCCDF a OVAL je blízká spolupráce.

Další možnost rozšíření je práce s výsledky skenování na přítomnost různých zranitelností, kde by se mohly jednotlivé uložené výsledky zranitelností mezi sebou porovnávat, a tak například poznat, jestli nastavení, která byla provedena v poslední době, přispěla k bezpečnosti systému či se systém naopak stal zranitelnější. Při zobrazování porovnání výsledků by si uživatel mohl volit různé způsoby vizualizace, například různé typy grafů, histogramy atd. Výsledky by se nemusely porovnávat jen dva mezi sebou, ale i ve skupině výsledků, aby byl vidět časový vývoj zabezpečení a přítomnost potencionálních zranitelností systému.

7.4 Budoucnost projektu

Budoucnost tohoto projektu je podle mého názoru optimistická, nasvědčuje tomu zájem uživatelů o tento nástroj. Dále pro světlou budoucnost tohoto nástroje hovoří zájem firmy Red Hat, která neukončila práci na tomto nástroji a stále jej rozvíjí. V současné době se nástroj rozšiřuje o práci s dokumentem OVAL. Aktuální verzi nástroje si můžete stáhnout z adresy git://git.fedorahosted.org/scap-workbench.git.

Literatura

[1] HANÁČEK, Petr; STAUDEK, Jan. Bezpečnost informačních systémů: Metodická příručka zabezpečování produktů a systémů budovaných na bázi informačních technologií. Praha: ÚSIS, 2000. 127 s. ISBN 80-238-5400-3.

[2] HANÁČEK, Petr. Bezpečnost informačních systémů 1 [online prezentace]. [cit. 2010-01-15]. Dostupné z WWW: <https://www.fit.vutbr.cz/study/courses/BIS/private/bis01.pdf>.

[3] CERT [online]. © 1995-2011 [cit. 2011-04-03]. Advisories. Dostupné z WWW: <http://www.cert.org/advisories/>.

[4] MANADHATA, Pratyusa. Measuring a System's Attack Surface. [s.l.], 2004. 22 s. Odborná studie.

[5] KEMMERER, R.; ECKMANN, S.; VIGNA, G. STATL: An Attack Language for State-Based Intrusion Detection. Journal of Computer Security. 2002, 10, s. 71-104.

[6] ILGUN, K.; KEMMERER, R.A.; PORRAS, P.A. State Transition Analysis: A Rule-Based Intrusion Detection Approach. IEEE Transactions on Sortware Engineering. 1995, 5, s. 181-199.

[7] SCHNEIDER, Fred B. Enforceable security policies. ACM Transactions on Information and System Security (TISSEC). 2000, 3, s. 30-50.

[8] Openscap [online]. 20011 [cit. 2011-03-30]. Home. Dostupné z WWW: <http://www.open-scap.org/page/Main_Page>.

[9] Guide to Adopting and Using the Security Content Automation Protocol (SCAP) [online]. 2010 [cit. 2011-04-03]. Dostupné z WWW: <http://csrc.nist.gov/publications/nistpubs/800-117/sp800-117.pdf>.

[10] Scap [online]. 2009 [cit. 2011-01-08]. The Security Content Automation Protocol (SCAP) - NIST. Dostupné z WWW: <http://scap.nist.gov/index.html>.

[11] Automating Compliance with Security Content Automation Protocol [online]. c2005-2010 [cit. 2011-01-08]. Dostupné na URL: <http://www.first.org/cvss/grance-tim-slides.pdf>

[12] The MITRE Adoption Programs and the NIST SCAP Validation Programl [online]. 2008 [cit. 2011-01-09]. Dostupné na URL: <http://cve.mitre.org/adoption/Adoption_and_Validation_August2008.pdf>

[13] Barabas, M.: Automated Processes in Computer Security, In: Proceedings of the 16th Conference STUDENT EEICT 2010, Brno, CZ, VUT v Brně, 2010, s. 246-250, ISBN 978-80-214-4079-1

[14] OBERREITER, Vladimír. GUI nástroj na měření zranitelnosti systémů s databází CVE a knihovnou OPEN-SCAP. [s.l.], 2010. 23 s. Semestrální práce. Vysoké učení technické v Brně, Fakulta informačních technologií.

[15] CVE : Common Vulnerabilities and Exposures (CVE) [online]. c1999–2011 [cit. 2011-01-17]. Dostupné z WWW: <http://cve.mitre.org/>.

[16] National Vulnerability Database [online]. 2010 [cit. 2011-04-03]. Dostupné z WWW: <http://nvd.nist.gov/>.

[17] CVE [online]. ©1999–2011 [cit. 2011-04-03]. CVE-2000-0005. Dostupné z WWW: <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0005>.

[18] MANN, David. An Introduction to the Common Configuration Enumeration [online]. July 24, 2008 [cit. 2011-01-17]. Dostupné z WWW: <http://cce.mitre.org/documents/Introduction_to_CCE_White_Paper_July_2008.pdf>.

[19] Common Platform Enumeration (CPE) – Specification [online]. 2009 [cit. 2011-01-20]. Dostupný z WWW: <http://cpe.mitre.org/files/cpe-specification_2.2.pdf>.

[20] MELL, Peter; SCARFONE, Karen; ROMANOSKY, Sasha. A Complete Guide to the Common Vulnerability Scoring System [online]. Version 2.0. June, 2007 [cit. 2011-01-22]. Dostupné z WWW: <http://www.first.org/cvss/cvss-guide.pdf>.

[21] An Introduction to the OVAL™ Language [online]. Version 5.0. c2006 [cit. 2011-01-25]. Dostupné z WWW: <http://oval.mitre.org/oval/documents/docs-06/an_introduction_to_the_oval_language.pdf>.

[22] Quinn, S.D., Ziring, N.: Specification for the Extensible Configuration Checklist Description Format (XCCDF) Verison 1.1.4, NIST a NSA, 2008.

[23] Security Content Automation Protocol [online]. 2011 [cit. 2011-04-03]. Dostupné z WWW: <http://scap.nist.gov/specifications/xccdf/>.

Seznam příloh

Příloha 1
Instalace

Nainstalujte python:

```
$ sudo yum -y install python
```

Nainstalujte potřebné knihovny pro OpenSCAP:

```
$ sudo yum -y install autoconf automake swig pcre-devel libxml2-devel
rpm-devel rpm-build libnl-devel python-devel perl-devel libcurl-devel
gcc libtool libgcrypt-devel libxslt-devel doxygen openldap-devel
c-ares-devel
```

Zkopírujte knihovnu OpenSCAP z přiloženého CD (adresář OpenSCAP), rozbalte, přejděte do rozbaleného adresáře a vytvořte tarball (openscap-x.y.y.tar.bz2):

```
$ cd openscap
$ ./autogen.sh && ./configure && make -j4
$ make dist
```

Tarball zkopírujte do ~/rpmbuild/SOURCES/ a vytvořte rpm balíčky:

```
$ mkdir ~/rpmbuild ~/rpmbuild/SOURCES
$ cp <vytvoreny tarball> ~/rpmbuild/SOURCES/
$ rpmbuild -ba dist/fedora/openscap.spec
```

Nainstalujte vytvořené balíčky:

```
$ sudo rpm -ivh ~/rpmbuild/RPMS/i386/openscap-*
```

Zkopírujte vytvořený nástroj z CD (adresář securityTool), rozbalte, přejděte do rozbaleného adresáře a nainstalujte:

```
$ cd scap-workbench
$ sudo make install
```

Příloha 2
Manuál

Toto je stručný manuál pro práci s vytvořeným nástrojem. Manuál je popsán pro systém Fedora, pro který jsou k dispozici na přiloženém CD dokumenty XCCDF a OVAL, které jsou připraveny pro systém Fedora od firmy Red Hat.

Nastavování profilů (Tailoring) a skenování

- V příkazové řádce zadejte:

  ```
  $ scap-workbench
  ```

- Otevře se nástroj pro nastavování profilů a skenování. Na kartě Main naimportujte soubor XCCDF s pravidly pro skenování pomocí tlačítka import. Soubor je k dispozici v podadresáři knihovny OpenSCAP, kterou jste stáhli z CD a rozbalili při instalaci: /openscap/dist/fedora/scap-fedora14-xccdf.xml (naimportujte soubor XCCDF z tohoto umístění, aby se načetl i odpovídající soubor OVAL, který je umístěn ve stejném adresáři). Na kartě Main můžete provádět ostatní operace s dokumentem XCCDF: provést validaci, vytvořit nový, exportovat změněný a zavřít.

- Přepněte se na kartu Tailoring. V comboBoxu pod hlavním menu vyberte profil, načte se vám požadovaný profil. V načteném profilu můžete vybírat pomocí selectBoxu skupiny a pravidla, které budete chtít zkontrolovat na vašem systému. Na panelu Details (vpravo) jsou zobrazeny základní informace o zvolené položce a dole v tabulce jsou zobrazeny hodnoty, které můžete měnit. Na druhém panelu Refines můžete nastavit parametry pro skenování.

- Po nastavení profilu se přepněte na kartu SCAN. Profil, který se bude skenovat, je ten, který je vybrán na kartě Tailoring (případně můžete změnit pomocí tlačítka Profile). Spusťte sken, po skenování můžete výsledek uložit pomocí tlačítka Export results. Po uložení můžete výsledek zobrazit pomocí tlačítka Results.

Správa dokumentu XCCDF (Edit)

- V příkazové řádce zadejte:

  ```
  $ scap-workbench-editor
  ```

- Otevře se nástroj pro správu (Benchmarku, Profilů a položek). Na kartě Benchmark naimportujte soubor XCCDF s pravidly pro skenování pomocí tlačítka import. Soubor je k dispozici v podadresáři knihovny OpenSCAP, kterou jste stáhli z CD a rozbalili při instalaci: /openscap/dist/fedora/scap-fedora14-xccdf.xml (naimportujte soubor XCCDF z tohoto umístění, aby se načetl i odpovídající soubor OVAL, který je umístěn ve stejném adresáři). Na kartě Benchmark můžete provádět ostatní operace s dokumentem XCCDF: provést validaci, vytvořit nový, exportovat změněný a zavřít.

- Na kartě Profiles můžete přidávat, mazat a upravovat profily. Pro přidání profilu klikněte pravým tlačítkem na pole, kde je seznam profilů, a zvolte Add profile. Podobně tak pro mazání na vybraném profilu stiskněte pravé tlačítko myši a zvolte Remove profile. Pro editaci profilu vyberte požadovaný profil v seznamu a detail profilu se načte do karet na pravé straně formuláře. Ve stromové struktuře pod profilem jsou vidět položky, které jsou odlišné od hlavního dokumentu.

- Na kartě Items můžete přidávat, mazat a upravovat položky. Práce s položkami je obdobná jako práce s profily.

- Po změnách v dokumentu XCCDF na kartě Benchmark klikněte na tlačítko Export pro uložení změn.

Dodatečné informace

- ID položek, profilů, dokumentu atd. musí vždy začínat písmenem, v opačném případě dokument při exportu nebude validní.

- Zkratku jazyka musí tvořit vždy dvě písmena, v opačném případě dokument při exportu nebude validní.

Příloha 3
Akceptační testy

Akceptační testy jsou vytvořeny pro systém Fedora, na kterém byly provedeny. Zde je uvedeno jen několik základních testů. Testů bylo samozřejmě mnohem více a prováděly se v průběhu implementace.

1. Načtení souboru XCCDF

Popis:

Uživatel načte dokument XCCDF pro práci.

Požadavky:

1. Spuštěný *scap-workbench* na kontrolu zranitelnosti systému.
2. K dispozici validní dokument XCCDF.

Postup:

1.
   (a) uživatel v hlavním menu zvolí položku Main, pokud není zobrazena;
   (b) možné reakce:
      i. zobrazí se karta s informací o XCCDF souboru;
      ii. selhání operace;
      iii. selhání systému;
2.
   (a) uživatel zvolí tlačítko Import;
   (b) možné reakce:
      i. zobrazí se formulář pro import XCCDF souboru;
      ii. selhání operace;

iii. selhání systému;

3.

(a) uživatel vybere XCCDF soubor a potvrdí volbu;

(b) možné reakce:

i. načtení souboru XCCDF a zobrazení informace o něm v kartě Main;

ii. selhání operace;

iii. selhání systému.

Úspěch:

Byl načten soubor a uživatel jej může upravovat nebo provést kontrolu počítače.

2. Spuštění kontroly počítače

Popis:

Uživatel vybere profil a spustí skenování systému.

Požadavky:

V nástroji *scap-workbench* na importovaný soubor scap-fedora14-xccdf.xml a k dispozici odpovídající soubor OVAL ve stejném adresáři (oba dokumenty k dispozici na CD v knihovně OpenSCAP: /dist/fedora/).

Postup:

1.

(a) uživatel se přepne v hlavním menu na kartu Scan, pokud není zvolena;

(b) možné reakce:

i. zobrazí se karta pro skenování;

ii. selhání operace;

iii. selhání systému;

2.

   (a) uživatel zvolí tlačítko pro načtení profilu (na kartě Scan), který se bude skenovat;

   (b) možné reakce:

      i. zobrazí se okno pro výběr profilu, který se bude skenovat;

      ii. selhání operace;

      iii. selhání systému;

3.

   (a) uživatel vybere profil „Fedora 14 desktop settings“ a potvrdí výběr;

   (b) možné reakce:

      i. je vybrán profil, který se bude skenovat;

      ii. selhání operace;

      iii. selhání systému;

4.

   (a) uživatel klikne na tlačítko Scan na kartě Scan;

   (b) možné reakce:

      i. spustí se skenování a jednotlivé výsledky skenování jsou průběžně zobrazovány;

      ii. selhání operace;

      iii. selhání systému.

Úspěch:

Byl zkontrolován systém na základě zvoleného profilu a výsledky jednotlivých pravidel, které byly skenovány, jsou zobrazeny.

3. Přidání profilu

Popis:

Uživatel v dokumentu XCCDF přidá nový profil pro skenování.

Požadavky:

V nástroji *scap-workbench-editor* naimportovaný soubor scap-fedora14-xccdf.xml (dokument k dispozici na CD v knihovně OpenSCAP: /dist/fedora/).

Postup:

1.

   (a) uživatel se přepne v hlavním menu na kartu Profile, pokud se na ní nenachází;

   (b) možné reakce:

      i. zobrazí se karta pro správu profilů;

      ii. selhání operace;

      iii. selhání systému;

2.

   (a) uživatel stiskne pravé tlačítko myši, v poli se seznamem profilů;

   (b) možné reakce:

      i. zobrazí se menu, ve kterém bude volba pro přidání profilu;

      ii. selhání operace;

      iii. selhání systému;

3.

   (a) uživatel v zobrazeném menu zvolí přidání profilu;

   (b) možné reakce:

      i. zobrazí se okno pro zadání základních informací o profilu (id, title);

      ii. selhání operace;

      iii. selhání systému;

4.

   (a) uživatel zadá ID, název a potvrdí;

   (b) možné reakce:

      i. vytvoří se nový profil;

      ii. selhání operace;

      iii. selhání systému.

Úspěch:

Byl vytvořen nový profil, který lze upravovat pro skenování.

4. Editace profilu

Popis:

Uživatel přidá k profilu další název profilu v jiném jazyku.

Požadavky:

V nástroji *scap-workbench-editor* naimportovaný soubor scap-fedora14-xccdf.xml (dokument k dispozici na CD v knihovně OpenSCAP: /dist/fedora/). Uživatel se nachází na kartě Profiles.

Postup:

1.

   (a) uživatel zvolí ze seznamu profilů nějaký profil;

   (b) možné reakce:

      i. do General na kartě profilu se načtou informace o zvoleném profilu;

      ii. selhání operace;

      iii. selhání systému;

2.

   (a) v General, na kartě Titles uživatel zvolí tlačítko Add;

   (b) možné reakce:

      i. zobrazí se formulář pro zadání nového názvu;

      ii. selhání operace;

      iii. selhání systému;

3.

   (a) uživatel vyplní jazyk, název a zvolí tlačítko uložit;

   (b) možné reakce:

      i. nový název se uloží k profilu;

      ii. selhání operace;

      iii. selhání systému.

Úspěch:

K zvolenému profilu se uložil další název.