Posudek bakalářské práce

předložené na Matematicko-fyzikální fakultě
Univerzity Karlovy v Praze

Autor práce:	Michal Mižišin
Název práce:	Analyzátor útokov na webový server
Rok odevzdání:	2012
Studijní program a obor:	Informatika, obecná informatika
Autor posudku:	Mgr. Martin Mareš, Ph.D., oponent
Pracoviště:	Katedra aplikované matematiky

K celé práci	lepší	OK	horší	nevyh.
Obtížnost zadání		X	X	
Splnění zadání			X	X
Rozsah práce			X	X

Cílem práce bylo vytvořit aplikaci pro detekci bezpečnostních útoků na webový server na základě analýzy komunikace mezi serverem a klientem.

Autor práce se bohužel tohoto úkolu zhostil poněkud minimalistickým způsobem: navrhl aplikaci, která funguje jako HTTP proxy mezi uživateli a webovým serverem a vyhledává v procházejících datech povětšinou jen podezřelé řetězce vyhovující fixním regulárním výrazům. Proxy navíc pracuje naprosto přímočaře a podstatné části protokolu HTTP vůbec nepodporuje (viz níže). To vše činí celou práci bezmála triviální.

Zadání také požaduje, aby program podle odezvy serveru poznával, zda útok proběhl úspěšně. To práce téměř neřeší, ale nepovažuji to za podstatnou vadu, protože spolehlivá detekce úspěšnosti útoku založená pouze na odpovědích serveru a nezávislá na konkrétní webové aplikaci je takřka nemožná.

Textová část práce	lepší	OK	horší	nevyh.
Formální úprava			X	
Struktura textu			X	X
Analýza				X
Vývojová dokumentace			X	X
Uživatelská dokumentace		X		

Textová část práce obsahuje popis různých druhů webových útoků a stručnou uživatelskou a programátorskou dokumentaci k programu.

Úvod hovoří o principech webových útoků, ale vůbec nezmiňuje, co je hlavním cílem celé práce.

Popis jednotlivých útoků v další kapitole je velice rozvláčný, stále se v něm opakují tytéž informace. U většiny útoků je řečeno, že i tento útok je založen na tom, že webová

aplikace nekontroluje obdržená data, a že doporučená obrana proti takovému útoku je data kontrolovat (což není nijak překvapivé). Navíc jsou u některých útoků uvedena další bezpečnostní opatření, jejichž účinnost je diskutabilní (za všechna jmenujme například pochybné doporučení z kapitoly 2.5.2 odstranit standardním příkazům operačního systému právo spuštění). Mimo to celá kapitola obsahuje už jenom příklady útoků převzaté z citovaných zdrojů (někdy naprosto nekonkrétní, viz kapitola 2.8.3). Útoky by bylo lepší seřadit od jednodušších ke složitějším, například pro pochopení ORM Injection by se čtenáři hodilo vědět, jak funguje SQL Injection.

Dále by měla následovat analýza problému, ze které by vyplynul návrh aplikace. Tato část v práci zcela chybí. Kdyby byla analýza provedena, nejspíš by z ní vyšlo, že použité hledání pomocí regulárních výrazů zabudovaných přímo v kódu aplikace není vhodné – hrozí jak poměrně nízká účinnost detekce, tak mnoho falešných poplachů. Také by bylo jasné, že lepší než zachraňovat pomalost detektoru možností odložit analýzu dat na později by bylo implementovat detekci efektivněji (například v jiném jazyce než Java).

Uživatelská dokumentace je minimální, ale vesměs postačuje. Jen by bylo vhodné v ní zmínit, že mimo hlavního konfiguračního souboru existuje také soubor popisující konfiguraci logování, a prozradit, kde ho program hledá. (Mimochodem, je to jinde, než kde je umístěna jeho ukázková verze na CD.)

Programátorská dokumentace je nedostatečná. Zcela pomíjí nejsložitější část aplikace, totiž HTTP proxy. Navíc dokumentace obsahuje srovnání s jinými nástroji podobného druhu, což logicky patří úplně jinam, nejspíše do závěru práce.

V práci zcela chybí jakékoliv experimentální výsledky. U programu založeného na heuristické analýze nějakých jevů je nutné použitou heuristiku zhodnotit – ukázat, že je účinná proti reálným útokům, a zhodnotit procento falešných poplachů. Domnívám se, že vzhledem k použitým regulárním výrazům by účinnost byla nevelká a falešné poplachy časté. Také by bylo vhodné experimentálně ověřit, zda navržený systém odloženého prohledávání cokoliv řeší – nebylo by nijak překvapivé, kdyby i v Javě bylo rychlejší data přímo prohledat, než je uložit na disk.

Sazba textu je nekvalitní s častými překlepy a pokud mohu posoudit, tak i prohřešky proti slovenskému pravopisu. Sazbě by velice prospělo rozvážněji (a zejména konsistentně) volit typy písma pro různé druhy informací – například pro ukázky výpisů je obvyklé používat neproporcionální písmo, zatímco v práci je použita střídavě kurzíva a obyčejné písmo. Také nebývá zvykem odlamovat neslabičné předložky od následujícího slova.

Implementační část práce	lepší	OK	horší	nevyh.
Kvalita návrhu			X	X
Kvalita zpracování				X
Stabilita implementace			X	

Hlavní věc, kterou bych programu vytkl, je nekvalitní implementace HTTP proxy, která porušuje specifikaci protokolu HTTP téměř každým myslitelným způsobem. Požadavky standardu kladené na chování HTTP proxy nejsou vůbec respektovány. Mimo to například vůbec nepočítá s chunked transfer encoding, s víceřádkovými hlavičkami nebo s tím, že MIME typ v hlavičce Content-Type může mít parametry. Program, který zajišťuje bezpečnost, si nic takového nemůže dovolit – musí protokol implementovat korektně

a velmi striktně a všechny pokusy o obejití pravidel protokolu odmítat nebo rovnou hlásit jako útoky. Také je nutné ošetřovat pokusy o zahlcení proxy a reagovat na ně něčím jiným než pádem aplikace.

Rovněž není zvykem, aby serverová aplikace hledala svůj konfigurační soubor v aktuálním adresáři (nebo relativně k němu) a aby posílala e-maily prostřednictvím externího serveru (který je nutné konfigurovat), místo použití standardních systémových prostředků.

Během testování aplikace několikrát ukončila spojení a ohlásila výjimku, obvykle při zpracování požadavku typu POST s daty, která neodpovídala naivním představám autora o syntaxi.

Celkové hodnocení: neprospěl

Práci navrhuji na zvláštní ocenění: ne

V Praze dne 8. ledna 2013

Martin Mareš