SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Fakulta informatiky a informačných technológií
Odbor: Počítačové inžinierstvo

Penetračné testovanie

Tímový projekt

Tím č. 3: Bc. Rami Al Beyrouti
Bc. Martin Blesák
Bc. Peter Daniš
Bc. Martin Močol
Bc. Peter Štuller

OBSAH

0. Úvod

Cieľom tohto dokumentu je priblížiť a vysvetliť riešenie zadania Tímového projektu s názvom Penetračné testovanie. Výsledkom riešenia je web aplikácia, ktorá má za úlohu prezentovať základné a aj mierne pokročilé vedomostné informácie o Penetračnom testovaní.
Druhou úlohou aplikácie je zastrešovať aj integračné prostredie, ktorým bude možné uskutočňovať vybraté penetračné testy.
Dokument je rozdelený na viacero kapitol, ktoré zodpovedajú jednotlivým činnostiam nasledujúcim po sebe pri vytváraní konečného riešenia zadania.
V analýze problematiky sa pozrieme na Penetračné testovanie ako také, na jeho použitie, a na jeho dostupnosť pre ľudí, ktorí by sa ním chceli zaoberať. Tiež sa pozrieme na zdroje našej tvorby, ktoré môžeme použiť pri implementácii.
V kapitole špecifikácia si podrobnejšie opíšeme požiadavky na funkcie systému, na údaje, ktoré budeme využívať a určíme si, ako by sa mala naša aplikácia správať.
Kapitola návrh riešenia ukazuje, akým smerom by sme sa chceli vydať pri implementácii riešenia nášho problému.
Konečný produkt nášho snaženia vychádza zo všetkých týchto fáz tvorenia, preto sa snažíme splniť ich čo najvernejšie a tomuto snaženiu čo najvernejšie prispôsobiť aj tento dokument.

1. Analýza problematiky

Najprv by bolo vhodné vysvetliť, čo vlastne penetračné testovanie ako termín znamená. **Penetračné testovanie** – je termín z oblasti bezpečnosti počítačových a informačných systémov, ktorým označujeme činnosť, pri ktorej sa pokúšame nájsť slabiny vlastného systému, pričom ich hľadáme formou útoku na vlastný systém. Čiže sa tvárime, ako útočník na systém a prenikáme do neho (z latinčiny penetratio – prenikanie). Samotný útok sa väčšinou uskutočňuje sadou nástrojov, ktorým sa hovorí testy.

Bolo by ešte vhodné vysvetliť niekoľko pojmov, ktoré sa bezprostredne viažu na penetračné testovanie.

Pozrieme sa aj, aké existujúce implementačné prostredia by sme mohli využiť.

1.1. Metodiky

Metodiky sú súbory pravidiel, podľa ktorých sa testy vykonávajú. Všeobecne sa dá súbor testov rozdeliť na oblasti testovania, ktoré testujú niektorú časť informačného systému organizácie. Jednotlivé oblasti sa pritom môžu prekrývať a delia sa na moduly. Jeden modul spúšťa jednu testovaciu úlohu. Výsledky testovania sa zapíšu a porovnajú s úplnou bezpečnosťou, ktorú si definuje organizácia ako požiadavku.

1.2. Existujúce nástroje vykonávajúce penetračné testovanie

Jestvuje veľmi veľa nástrojov, ktorými sa dajú uskutočniť testy. Niektoré sú komerčné, niektoré voľne šíriteľné a niektoré sú tzv. open source (ich zdrojový kód je voľne šíriteľný). Keďže máme implementovať aj integračné prostredie na spúšťanie týchto nástrojov, uznali sme za vhodné použiť open source nástroje.

Na tému penetračného testovania existuje veľmi veľa materiálov, ale väčšinou sa jedná o zahmlené informácie firiem, ktoré ponúkajú produkty z tejto oblasti. Preto je výhodnejšie zamerať sa na informácie o voľne šíriteľných nástrojoch vykonávajúcich testy.

Najprístupnejšie sa javia nástroje Nessus, S.A.T.A.N. a niektoré ďalšie.

1.3. Existujúce web rozhrania

Prvou úlohou nášho zadania je vytvoriť webovú prezentáciu informácií o penetračnom testovaní. Tu nájde vzdelaný laik prípadne odborník prehľadne spracované informácie o penetračnom testovaní od základov cez všeobecnú metodológiu penetračného testovania až po popis nástrojov a systémov slúžiacich na penetračné testy.

Nakoľko sa bude jednať hlavne o informácie, je vhodné použiť už existujúci systém správy obsahu (CMS – Content Management System, ďalej v texte len cms systém) . V tomto prípade nám odpadá práca s navrhovaním web stránky a môžeme sa sústrediť hlavne na informácie v nej obsiahnuté.

CMS systémov existuje veľké množstvo, nakoľko ho chceme použiť pri našom školskom projekte, výber obmedzíme na voľne šíriteľné riešenia. Tieto sú väčšinou založené na kombinácii webového servera Apache, jazyka PHP a databázy Mysql. Umožňuje to jednoduchú správu a prenositeľnosť medzi platformami.

1.3.1. Drupal

CMS systém s otvoreným zdrojovým kódom napísaný v jazyku PHP šírený pod licenciou GPL. Umožňuje užívateľom jednoducho publikovať, spravovať a organizovať rozmanitý obsah na web stránke.

Hlavné výhody:

- Jeho funkčnosť možno rozširovať mnohými modulmi.
- Používa užívateľsky prívetivé URL adresy. Využíva na to modul *mod_rewrite* z webového servera Apache. Užívateľ si môže URL adresy prispôsobiť podľa svojich potrieb. Zároveň sú tieto adresy prívetivé aj k vyhľadávacím systémom.
- Oprávnenia sa nemusia priraďovať zvlášť každému používateľovi. Stačí mu priradiť rolu v systéme a následne určovať oprávnenia jednotlivých rolí.
- Na stránke sa dá jednoducho vyhľadávať, všetok obsah je indexovaný.
- Prístup do databázy sa uskutočňuje pomocou abstrahujúcej vrstvy, ktorá zabezpečuje databázovú nezávislosť. Stačí napísať pre túto vrstvu rozhranie na konkrétnu databázu a táto sa môže používať.
- Systém má podporu mnohých jazykov.
- Systém má vlastné logovacie záznamy o všetkých aktivitách na stránke. Udržiava si štatistiky prístupov na jednotlivé časti stránky.
- Pri veľkej záťaži poskytuje možnosť vyrovnávacej pamäte pre požiadavky na databázu.
- Umožňuje vkladať obsah ako text, html ale aj ako php kód.

1.3.2. PHP-Nuke

Automatizovaný systém pre vytváranie interaktívnej webovej stránky. Administrátor má úplnú kontrolu nad celou stránkou, zároveň má k dispozícii mnoho nástrojov na jej správu. Opäť ide o systém založený na PHP. Najviac testovaná platforma pre jeho spúšťanie je OS Linux, webový server Apache, databáza Mysql.

Hlavné výhody:

- Veľká miera flexibility, používateľ si systém ľahko upraví do podoby, ktorá mu najviac vyhovuje.
- Jazyková podpora – viac ako 20 jazykov.
- Rozšíriteľný pomocou mnohých plug-inov, kde veľa z nich je súčasťou základnej inštalácie.

1.3.3. Plone

CMS systém založený na aplikačnom serveri Zope. Riešenie je s otvoreným zdrojovým kódom v jazyku Python.
Hlavné výhody:

- Jazyk Python zabezpečuje dobrú platformovú prenositeľnosť.
- Objektovo orientovaný aplikačný server Zope zabezpečuje robustnosť riešenia, databázovú nezávislosť. Nie je potrebný webový server.
- Jazyk Python umožňuje v prípade vlastného rozširovania tohto systému vysokú programátorskú produktivitu.
- Preložený do asi 50 jazykov.

1.4. Nároky používateľov

Cieľom nášho snaženia je vytvoriť web aplikáciu, ktorá bude zrozumiteľná, bude obsahovať len dôležité údaje, graficky bude príjemná, k informáciám sa používateľ ľahko dostane, integrácia spúšťania testov bude nenásilná. Takéto vlastnosti by uvítali používatelia aplikácie.

2. Špecifikácia

Na každý systém sa kladú určité požiadavky, nech už tým systémom je rozsiahla databáza, operačný systém alebo web aplikácia. Preto je potrebné presne špecifikovať požiadavky aj na našu aplikáciu.

2.1. Špecifikácia funkcií systému

Funkcie nášho systému sú dané požiadavkami človeka, ktorý ho bude používať. Ten by chcel, aby bol systém (v našom prípade web aplikácia) jednoduchý na ovládanie, poskytoval mu čo najviac ľahko prístupných a dôležitých informácií podaných v uhladenej forme, a dovolil mu spustiť niektoré testy.

Zo znenia nášho zadania vyplýva, že aplikácia musí mať dve hlavné funkcie. Sú to tieto dve:

- Prezentovanie informácií o penetračnom testovaní
- Spúšťanie penetračných testov v integrovanom prostredí

Preto rozdeľujeme aplikáciu na dva moduly: prezentačnú časť a integračnú časť.

Prezentačná časť

Vstupy: Informácie o penetračnom testovaní

Výstup: Grafická podoba prezentácie reprezentovaná článkami o jednotlivých témach o penetračnom testovaní spustiteľná vo web prehliadači a uložená na serveri.

Prezentačná časť bude teda obsahovať funkcie zobrazovania informácií, ich zoradenia do kapitol, vyhľadávania, otestovania znalostí v danej problematike a bude obsahovať odkazy na obsiahlejšie informácie o niektorých podoblastiach, na ktoré používateľ narazí. Celá prezentačná časť musí mať príjemné používateľské rozhranie a aj nevtieravú a harmonickú grafickú podobu.

Potrebujeme jednoduchým a prehľadným spôsobom uverejniť na stránke spísané informácie o penetračnom testovaní. Vytvoriť menu, ktoré bude umožňovať štruktúrovaný prístup k týmto informáciám. Uverejňovať na stránke text alebo html prezentáciu spolu s obrázkami a animáciami.

Integračná časť

Vstupy: Vybrané nástroje na penetračné testovanie systémov

Výstup: Integrované prostredie na spúšťanie nástrojov z web aplikácie

Integračná časť umožní spúšťanie vybratých nástrojov priamo z web aplikácie, pričom by nemala byť narušená logická, sémantická ani grafická nadväznosť na prezentačné prostredie.

2.2. Špecifikácia údajov v systéme

Údaje v systéme sú v našom prípade informácie o penetračnom testovaní. Získavame ich štúdiom materiálov, ktoré by mali byť uvedené aj v aplikácii, ak by používateľ chcel zistiť o problematike niečo viac. Údaje by mali byť do systému ľahko vkladateľné, keďže predpokladáme v tejto oblasti informačných technológií ďalší rozvoj. Oprava údajov by mala byť tiež zabezpečená, ako aj ich prípadné rušenie.

2.3. Správanie sa systému

Systém by sa mal správať ako všeobecná webová aplikácia, ktorej zameranie je poskytovať používateľovi informácie, ktoré si môže pomocou integrovaného rozhrania na spúšťanie testov aj v praxi overiť. Mal by byť teda stabilný s rýchlou odozvou, obsahujúci množstvo údajov, ktoré sú rýchlo dostupné. Podľa toho je dôležité správne si zvoliť implementačné prostredie.

3. Návrh riešenia

V tejto kapitole je rozpísaný náš návrh, ako budeme implementovať jednotlivé časti aplikácie, a ako chceme, aby výsledok nášho snaženia vyzeral. Keďže máme na starosti dva typy úloh, a to prezentačné a integračné prostredie, ich návrhy sa budú líšiť, aj keď navonok by mali pôsobiť ako jednoliaty celok.

3.1. Návrh prezentačného prostredia

Ako prvé je dobré navrhnúť prezentačné prostredie, keďže obsahuje informácie aj o integračnom prostredí. Pri jeho návrhu musíme vychádzať z už spomínaných požiadaviek. Systém je navrhovaný ako web aplikácia, keďže to udáva zadanie.

Ako prvé si musíme zvoliť implementačné prostredie, navrhnúť scenár alebo štruktúru web aplikácie, a nakoniec aj jej výzor a rozhranie s používateľom.

3.1.1. Návrh prezentačného prostredia

V časti analýzy sme uviedli 3 CMS systémy, z ktorých musíme vybrať jeden najviac spínajúci naše požiadavky. Tieto boli stanovené v časti špecifikácie.

Každý zo systémov do istej miery spĺňa naše požiadavky, a keby sme sa nemohli slobodne rozhodnúť, hociktorý zo systémov by bol vyhovujúci. Rozhodli sme sa ale uprednostniť systém *Drupal*. Zvyšné 2 systémy sú totiž na naše použitie príliš robustné a zbytočne komplexné. Naproti nim Drupal poskytuje potrebnú funkčnosť pri zachovaní jednoduchého a intuitívneho ovládania. Zároveň je možné vyjsť z jeho hodnotenia podľa [4], kde jednoznačne dominuje spomedzi ostatných CMS systémov s otvoreným zdrojovým kódom.

3.1.2. Návrh kapitol

Rozvrhnutie stránok vo web aplikácii by sa dalo realizovať prostredníctvom kapitol. Jedna kapitola reprezentuje jednu stránku, ktorá sa objaví v prehliadači. Všetky údaje by sa dali čítať sekvenčne, alebo by sa v nich dalo pohybovať pomocou nadpisov kapitol.

Návrh kapitol vyzerá takto:

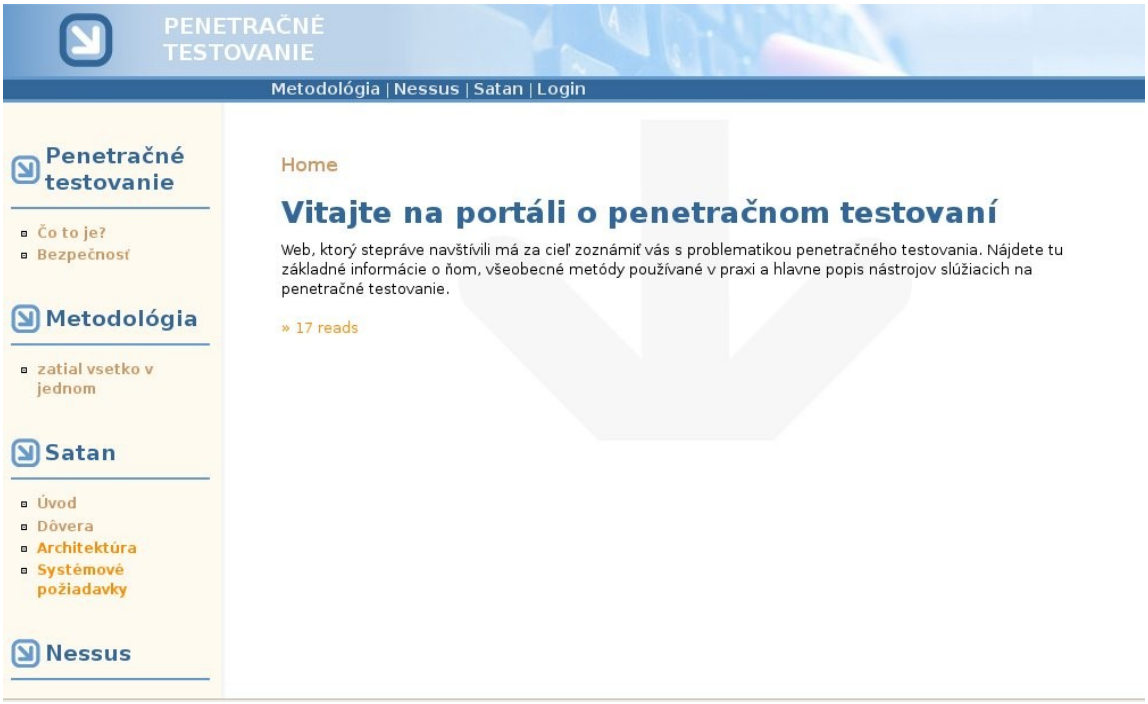
- 1.kapitola : Úvod
- 2.kapitola: Metodológie testovania
- 3. kapitola: Niektoré používané nástroje
  - 3.1.: Nessus
  - 3.2.: S.A.T.A.N.
  - 3.3.: Ďalšie nástroje
- 4. kapitola: Integrované prostredie
- 5. kapitola: Test znalostí
- 6. kapitola: Zaujímavé odkazy
- 7. kapitola: O autoroch

Každá kapitola bude naplnená údajmi v podobe článkov, integračného prostredia, testu, a odkazov. Na konci kapitoly bude možnosť plynulého prechodu na ďalšiu kapitolu.

3.1.3. Návrh rozhrania

Rozhranie web prezentácie je do istej miery dané vybratým CMS systémom. Tento je možné samozrejme prispôsobovať podľa potreby. Používateľ bude mať k dispozícii prehľadne štruktúrované menu, kde každá jeho časť sa bude týkať jednej kapitoly. Návrh menu možno vidieť na obrázku č. 1.

V hornej časti vidíme linky pre rýchly prístup k jednotlivým témam, linka smeruje vždy na prvú stránku s danou témou (úvod). Na ľavej strane sa nachádza menu pre každú tému. Jednotlivé body sú zoradené podľa odporúčaného poradia čítania, každý bod môže byť štruktúrovaný do ďalších podbodov.



Obr. č. 1: Návrh rozhrania web prezentácie

3.2. Návrh integračného prostredia

Integračné prostredie má za úlohu spúšťať vybrané testy a nemalo by sa vymykať designu prezentačného prostredia.

Bližší návrh a implementácia integračného prostredia budú nasledovať v kapitolách venovaným integračnej časti projektu.

4. Prototyp

4.1. Cieľ prototypovania

Podľa návrhu z predchádzajúcej kapitoly sme vytvorili prototyp, na ktorom sme pozorovali, ktoré časti môžeme vylepšiť. Keďže v tomto semestri sme sa sústredili na prezentačnú časť aplikácie, pozorovali sme hlavne grafickú a štylistickú podobu web aplikácie. Sústredili sme sa samozrejme aj na plnenie a korektnosť funkcií, ktoré sme si zadefinovali v špecifikácii.

4.2. Výsledky prototypovania

Výsledkom prototypovania ja terajší stav vývoja aplikácie. Dá sa povedať, že prezentačná časť aplikácie má už finálnu podobu, kým na integračnej časti budú pokračovať práce v letnom semestri. Počas vývoja aplikácie boli do web prostredia pridané vždy nové články o penetračnom testovaní, o jednotlivých nástrojoch pre vykonávanie penetračného testovania, boli pridané nové obrázky, diagramy a animácie. Vylepšoval sa celkový dizajn stránky, farebná kombinácia a rozloženie jednotlivých elementov na stránke.

4.3. Upravený návrh

Návrh implementácie sa po prototypovaní veľmi nezmenil. Hlavnou zmenou je základná farebná schéma prezentačnej stránky. V upravenom návrhu sa nachádza viac animácií, obrázkov, diagramov a článkov. Nový návrh rozhrania je na obr. 2.

Obr. č. 2: Upravený návrh rozhrania web prezentácie

5. Implementácia prezentačnej časti

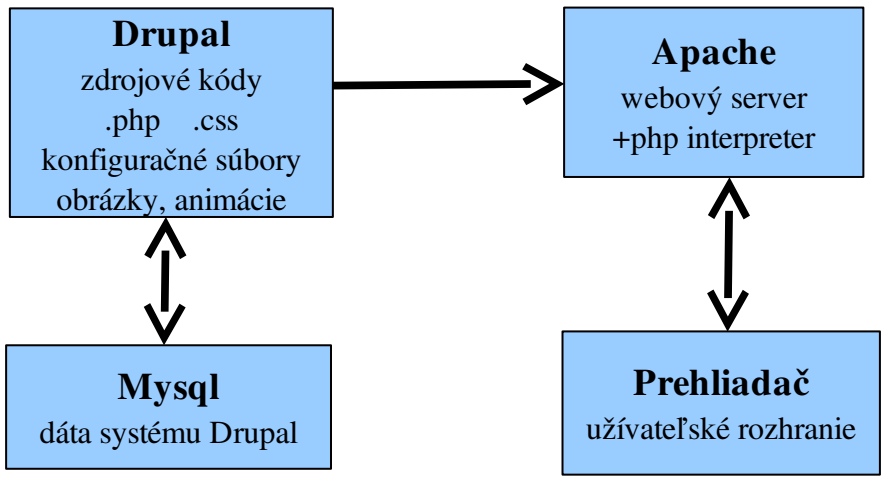
Na prezentovanie informácií o penetračnom testovaní sme využili CMS systém Drupal ako už bolo uvedené pri návrhu. Celá implementácia pozostávala z nasledovných krokov:

1. Inštalácia a sprevádzkovanie systému Drupal na školskom serveri.
2. Vyhľadávanie, triedenie, prekladanie informácií o penetračnom testovaní.
3. Úprava informácií do podoby prezentácie, logické rozčlenenie, príprava animácií a obrázkov k prezentovaným informáciám.
4. Vkladanie prezentácie ako celku do systému Drupal. Vytvorenie systému menu pre prístup k informáciám.
5. Úprava systému Drupal spolu s vloženými informáciami na požadovanú formu. Zahŕňalo hlavne úpravu systému Drupal na úrovni kaskádových štýlov (CSS súbory) a šablón (tie sú v systéme vo forme php kódu).

Nakoľko riešime projekt ako tím, činnosť súvisiaca s väčšinou krokov prebiehala paralelne.

5.1. Architektúra

Architektúra prezentačnej časti je jednoznačná už zo zadania. Ide o klasickú webovskú prezentáciu, t.j. architektúru klient/server. Konkrétnu architektúru nášho systému znázorňuje obrázok č. 3.



Obr. č. 3: Architektúra prezentačnej časti

Architektúra je znázornená všeobecne. Naše implementačné prostredie má nasledovné parametre:

- OS Linux Debian 3.0 (jadro 2.4.18-1-686-smp)
- Webový server Apache 1.3.31-2

- PHP 4.3.8-2
- Databázový server Mysql 4.0.18-2
- Testované prehliadače:
  - Firefox
  - IE
  - Mozilla
  - Epiphany

6. Overenie funkčnosti prezentačnej časti

Overovali sme funkčnosť prezentačnej časti, keďže integračná časť bude hotová až v letnom semestri. Overovali sme ju na dvoch druhoch operačných systémoch (Linuxového aj Windows typu), ako aj na viacerých druhoch internetových prehliadačov.

Prezentačná časť je v tejto chvíli, na konci svojho vývoja, plnohodnotne pripravená plniť požiadavky používateľov.

Grafické rozhranie je príjemne navrhnuté, jednotlivé kapitoly spĺňajú informačné nároky používateľa.

Stabilita systému je dobrá, odkazy a linky sú funkčné. Zaručuje to dobrá voľba CMS systému.

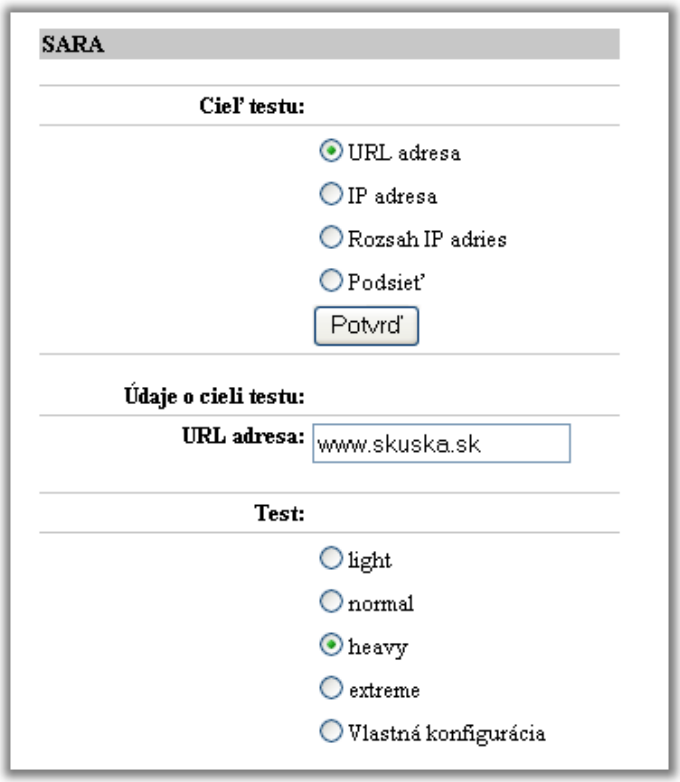
V grafickom znázornení sa nevyskytujú žiadne rušivé prvky. Štylisticky je prezentačná časť na celkom dobrej úrovni, aj keď by sa určite štylistika dala ešte

7. Návrh integračnej časti

Pri návrhu integračnej časti musíme myslieť na návrh prostredia, architektúry a algoritmov. Taktiež treba vziať do úvahy dostupné podmienky na implementáciu.

7.1. Návrh prostredia

Prostredie bude nadväzovať na prostredie prezentačnej časti, čiže bude na klientskej strane realizované cez web rozhranie. Na strane servera bude podporované technológiou PHP s prostredím Pear, v ktorom budú implementované formuláre.



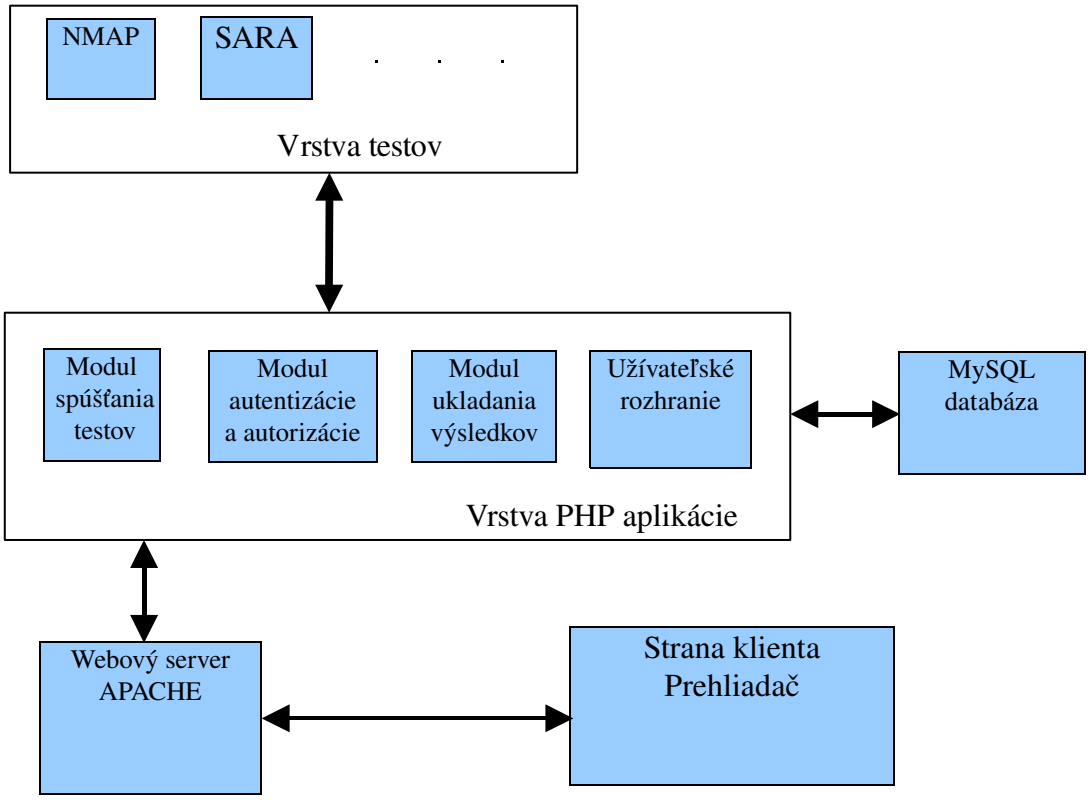
Obr.č.3. Jedna z navrhovaných obrazoviek

Po načítaní integračnej časti sa zobrazí prihlasovací formulár. Po prihlásení sa zobrazí menu, v ktorom si používateľ zvolí, ktorý nástroj použije. Na výber budú nástroje SARA, NSAT, NIKTO a NMAP.

Pri každom nástroji bude zoznam možností, čo robiť, resp. ako obmedziť alebo rozšíriť jeho základnú funkcionalitu. Po stlačení príslušného tlačidla sa vykoná zvolená množina testov a zobrazí sa výsledná správa. Po skončení testov sa bude dať vrátiť na výber nástrojov.

7.2. Návrh architektúry

Integračná časť bude fungovať ako Klient-Server aplikácia vďaka web prehliadaču na klientskej strane a Apache serveru na serverovej strane. Server bude podporovať technológiu PHP s prostredím Pear. Na serveri budú nainštalované potrebné testovacie nástroje a budú sa na ňom dať spúšťať pomocou príkazového riadku volaného klientom pomocou web rozhrania. Výsledky testov sa budú ukladať do premenných na serveri a zobrazia sa klientovi v prehliadači.



Obr. č.4 : Návrh architektúry integračnej časti

Integračné prostredie nebude svojou podstatou poskytovať len prostriedky nevyhnutné pre intregráciu, ale aj také, ktoré umožnia komplexnú funkčnosť prostredia. Navrhovaná funkčnosť je:

1. Prostriedky pre integráciu testov

   Vlastný framework založený na prostriedkoch jazyka PHP a iných frameworkoch (napríklad pear). Základné vlastnosti, ktoré chceme týmto dosiahnuť:

   a) Prístup k databáze má byť na databáze nezávislý, aby bolo možné prevádzkovať aplikáciu na rôznych databázach.

   b) Prostriedky spúšťania testov. Niektoré testy totiž vyžadujú pre svoju činnosť spustenie s administrátorskými oprávneniami pod príslušným OS.

   c) Prostriedky pre ľahké vytváranie užívateľského rozhrania k testu. Pri webovej aplikácii ide hlavne o vytváranie formulárov a ich spracovávanie.

d) Prostriedky pre ukladanie výsledkov testov, ich neskoršie prezeranie.

e) Prostriedky pre ľahkú jazykovú lokalizáciu aplikácie.

2. Prostriedky pre riadenie prístupu k aplikácii

   Aplikácia má umožňovať prístup k vykonávaniu penetračných testov, ktoré môžu byť svojou podstatou nebezpečné. Preto je veľmi dôležité vedieť obmedziť prístup k týmto testom.

   a) Prostriedky autentizácie užívateľa.

   b) Prostriedky autorizácie užívateľa.

   c) Prostriedky správy aplikácie – prideľovanie oprávnení, vytváranie užívateľov a pod.

3. Prostriedky pre bezpečné prevádzkovanie aplikácie

   Tieto nesúvisia len so samotnou aplikáciou, ale aj s jej prevádzovaním pri reálnom nasadení. Je ale dôležité navrhnúť ich a používať už pri vývoji aplikácie. Následne prezentovať hotové riešenie v tomto navrhnutom bezpečnom prostredí. Ďalej je už na administrátorovi servera, na ktorom sa bude prevádzkovať táto aplikácia, ako bude tieto bezpečnostné opatrenia implementovať. Dôležitými oblasťami sú:

   a) Prevádzkovať aplikáciu na bezpečnom a spoľahlivom operačnom systéme, pre ktorý sú dostupné bezpečnostné updaty.

   b) Prevádzkovať verzie programov, ktoré tvoria aplikáciu (apache, php, mysql) vo verziách s dostupnými bezpečnostnými záplatami. Tento bod je spravidla splnený, ak je splnený bod a).

   c) Interpret jazyka PHP prevádzkovať v bezpečnom režime (safe mode). Kontrolovať užívateľské vstupy na správny formát (prevencia pred sql injection, cross site scripting).

   d) Aplikáciu navrhnúť s ohľadom na zvýšenie bezpečnosti. Typickým príkladom môže byť princíp nesprístupňovať v koreni webového servera php skripty, ktoré sa nemusia spúšťať priamo pomocou url. Typicky stačí do koreňového adresára umiestňovať hlavný skript index.php. Zvyšná časť aplikácie je v inom fyzickom adresári. Tieto skripty sú síce čitateľné pre webový server (skripty sa „includujú“), ale nie je možné ich spúšťať individuálne pomocou url linky.

Zamýšlaná funkčnosť integračného prostredia môže byť aj intregrácia rozmanitých bezpečnostných testov a ich sprístupnenie pomocou jednoduchšieho rozhrania ako ony samotné poskytujú. Užívateľ potom nemusí mať prehľad o všetkých možnostiach nastavenia viacerých druhov testovacích nástrojov, ale tieto sa mu budú prezentovať v užívateľskom rozhraní. Týmto sa uľahčí a zefektívni pravidelný bezpečnostný test, napríklad na podnikových serveroch a sieťach.

Užívateľ však nemôže byť laik v problematike bezpečnostných testov. Aplikácia má pomôcť spúšťať rôzne testy a nie zjednodušovať ich pre menej skúsených používateľov.

V prípade použitia aplikácie na testy podnikovej siete resp. serverov, môže byť aplikácia umiestnená vnútri siete alebo v Internete (zrejme častejší prípad). Pri umietnení v Internete je veľmi dôležitá

vyššie popísaná bezpečnosť, aby nemohli byť penetračné testy zneužité na útoky. Veľmi vhodné by bolo prevádzkovať aplikáciu sa samostatnom počítači (serveri, fyzické oddelenie).

7.3. Návrh algoritmov

Pri návrhu integračnej časti a jej následnom implementovaní treba brať ohľad na správne konštruovanie príkazového riadku pre daný nástroj. Preto postup na jeho vytvorenie treba poriadne premyslieť. Najprv treba ponúknuť používateľovi všetky dostupné možnosti daného nástroja pomocou elementov formulára prostredia Pear. Po naplnení premenných elementov sa musí vykonštruovať príkazový riadok, ktorý sa následne vykoná, spustí funkcie daného nástroja a získa sa výstup testov, ten sa spracuje a zobrazí používateľovi v zrozumiteľnej forme na jeho prehliadači.

8. Ohraničenia integračnej časti

Niektoré ohraničenia vyplývajú z dostupných podmienok na implementáciu, iné z požiadaviek na integračnú časť.

Prvým ohraničením je nemožnosť implementovania všetkých existujúcich nástrojov, keďže by dlho trvalo už len zisťovanie, aké všetky existujú.

Implementovať je možné len funkcionality nástrojov, ktoré sú dostupné na zvolenom serveri, kde bude integračná časť nainštalovaná.

Na klientskej strane môže vzniknúť ohraničenie týkajúce sa nepodporovaného web prehliadača.

V časti návrhu bolo spomenuté obmedzenie na užívateľa, ktorý môže pracovať s aplikáciou. Nemôže ísť o laika v oblasti penetračných testov.

9. Implementácia integračnej časti

Kapitola hovorí o implementácii jednotlivých oblastí integračnej časti. Opisuje použitý server, implementáciu prihlásenia sa a jednotlivých funkcionalít penetračných nástrojov.

9.1. Implementačné prostredie

Aby sme mohli dosiahnuť požadovanú funkcionalitu produktu, je potrebné zvoliť si vhodné implementačné prostredie.

9.1.1. Server

Implementácia bude prebiehať v prostredí OS Linux (Debian Etch). Hlavné nástroje a programové prostriedky sú:

- PHP5 – aplikácia pobeží na verziách rady 5.0 aj 5.1. Vyvíjané a testované na verzii 5.1.2.
- APACHE2 – vyvíjané a testované na verzii 2.0.55.
- MYSQL 5.0 – nakoľko majú naše tabuľky pomerne jednoduchú štruktúru a nevyužívajú pokročilé databázové vlastnosti zavedené od verzie 5.0, nie je problém prevádzkovať aplikáciu aj na staršej rade 4.0 resp 4.1. Vyvíjané a testované na verzii 5.0.20.

Hardvérové požiadavky aplikácie sú minimálne, preto sme pre vývoj a testovanie vybrali dostupné staršie PC:

- procesor Pentium 200MHz
- 128MB RAM
- 8GB hard disk

9.1.2. Pear

Aby sme urýchlili vývoj v prostredí jazyka PHP, zvolili sme si používať prostredie Pear. Ide o sadu rozmanitých tried voľne dostupných a s otvoreným zdrojovým kódom, ktoré poskytujú rôznu funkcionalitu:

- databázovo nezávislý prístup do databázy
- abstraktný prístup do databázy pomocou objektov (DataObjects)
- šablóny
- vytváranie a validovanie formulárov (QuickForms)
- a mnoho iného, výhodou je objektový prístup

Keďže je integračná časť implementovaná v prostredí Pear, je vhodné spomenúť niekoľko základných prvkov tohto prostredia.
Ak chceme použiť prostredie Pear pre vytváranie užívateľského rozhrania vo forme formulárov, musíme mať v zdrojovom súbore riadky :

```
set_include_path("/cesta/pear:.");
require_once "HTML/QuickForm.php";
```

kde `cesta` je cesta, kde je Pear nainštalovaný na serveri.

Nový formulár sa vytvára príkazom:

```
$form = new HTML_QuickForm('meno_formulára', 'get');
```

Ak chceme použiť nadpis (hlavičku):

```
$form->addElement('header', 'meno_hlavicky', 'Zobrazený nadpis');
```

Textové pole sa pridáva príkazom:

```
$form->addElement('text', 'meno_premennej', 'Text pred');
```

Statický text:

```
$form->addElement('static','meno_premennej','Text');
```

Checkbox:

```
$premenna=&HTML_QuickForm::createElement('checkbox',
                                         'meno_premennej',
                                         null,
                                         'text_za');
$form->addElement($premenna,'pos_id');
```

Radio tlačidlo:

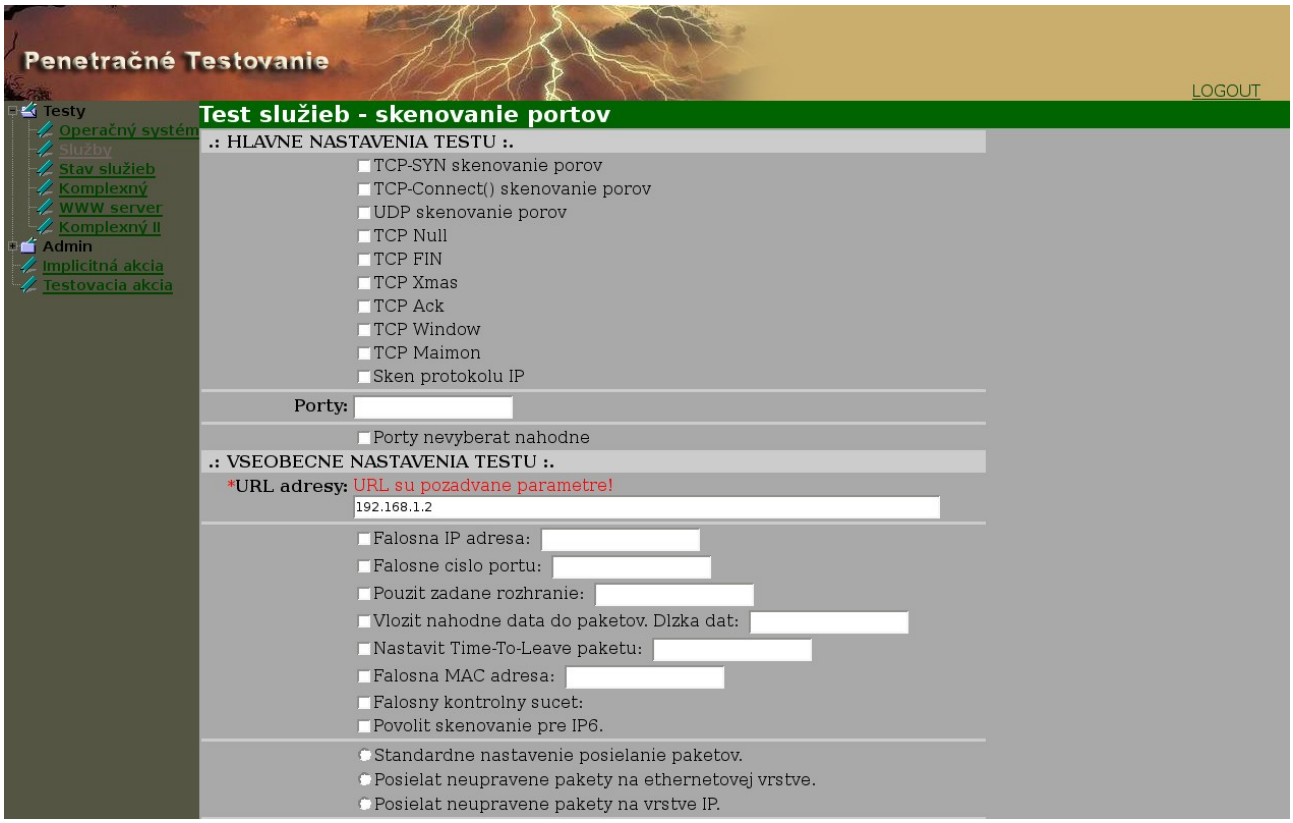
```
$premenna = &HTML_QuickForm::createElement('radio',
                                        'meno_premennej',
                                        'text_pred',
                                        'text_za',
                                    hodnota_premennej_po_zaskrtnuti);
$form->addElement($premenna);
```

Nakoniec, formulár sa zobrazí príkazom:

```
$form->display();
```

9.1.3. Používateľské rozhranie

Používateľské rozhranie je tvorene HTML stránkami, do ktorých je obsah vkladaný pomocou šablón. Šablóny umožňujú potrebnú flexibilitu pri zobrazovaní výsledkov realizovaných penetračných testov.



Obr. č. 5: Používateľské rozhranie

Na ľavej strane obrazovky je menu, kde si používateľ zvolí test, ktorý chce realizovať. Pod hlavičkou je priestor pre informácie, ktoré napomáhajú používateľovi k lepšej orientácií v systéme. V tomto mieste sa tak tiež dynamicky zobrazujú chybové hlásenia.

Rozhranie pôsobí jednoducho a prehľadne, čo bolo naším cieľom. Používateľ sa nestratí v množstve neprehľadných informácií.

9.2. Autorizácia a autentizácia

Jednou z pridaných hodnôt integračného prostredia oproti klasickému použitiu testov na príkazovom riadku je aj obmedzenie prístupu k nim. Používateľ sa musí do systému prihlásiť a môže používať len testy, na ktoré má oprávnenie. Administrátor aplikácie určuje oprávnenia na jednotlivé testy. Užívatelia sú zadelení do skupín, takže oprávnenia sa môžu prideľovať skupinovo aj jednotlivo.

Autentizácia je implementovaná pomocou PHP sessions. Z prihlasovacieho formulára sa zisťuje meno a heslo, ktoré sa následne overí podľa informácií v databáze. Ak je heslo platné, vytvorí sa pre používateľa session, ktorú následne používa pomocou prideleného session id. Toto číslo sa automaticky pripája za každú linku (napríklad: linka.php?SID=78ct4no72349cvn379ct).

Aby sa nemohlo zneužiť session id užívateľa, ktorý sa neodhlásil, session sa automaticky zruší po 25 minútach užívateľovej nečinnosti.

Aplikácia je dostupná len cez protokol https, takže nie je možné odchytiť meno a heslo posielané v procese autentizácie od klienta na server.

9.3. SARA

Pri implementácii funkcionalít systému SARA sme vychádzali zo štruktúry tohto systému. Keďže sa dá ovládať aj z príkazového riadku, použili sme pri implementácii syntax uvedenú v jeho manuálových stránkach [5] . Pre náš projekt je dôležitá syntax príkazu :

sara *[prepínač] [cieľ]*

a prepínače:

- **-a** určuje úroveň útoku od 0 po 3
  -a 0 je najslabší útok, -a 3 je najsilnejší
- **-f** nastaví niektoré testy, aby šli aj za firewall
- **-O** vynechá adresy zadané za prepínačom, adresy sa oddeľujú medzerou
- **-s** expanzia útoku na celú podsieť.

Preto sme sa zamerali na spomenutú syntax. Ako všetky časti integračnej časti, aj implementácia funkcionalít systému SARA je realizovaná pomocou nástroja PEAR.

V prvom kroku sa vyberie cieľ útoku. Výber je realizovaný elementmi formulára tzv. radio button-mi. Na výber je URL adresa, IP adresa, rozsah IP adries a podsieť s maskou.

Po potvrdení výberu sa zobrazia ostatné prvky formulára, kde sa pomocou elementov PEAR text, radio a checkbox naplnia premenné, pomocou ktorých vytvoríme celkový príkaz, pričom podmienkami, nastavenými elementmi radio a checkbox , určujeme hodnoty premennej `$options`. Výsledný príkaz potom vyzerá takto:

```
$prikaz = $cesta.'sara'.$options.' '.$target;
```

Kde `$cesta` je cesta na serveri, kde je nainštalovaný systém SARA, `$options` je premenná udržiavajúca prepínače s ich možnosťami a `$target` je cieľ útoku.

Po zostavení príkazu sa príkaz vykoná volaním

```
exec($prikaz,$pole,$reval);
```

Kde `$pole` je premenná na uloženie výpisu z konzoly po vykonaní príkazu a `$reval` je návratová hodnota. Po načítaní textu do `$pole` , sa tento text už len vypíše.

9.4. NSAT

Nástroj na analýzy sieťovej bezpečnosti (NSAT) má syntax zadávania ako príkazu v príkazovom riadku [8] :

nsat *[voľby] [cieľ]*

Nástroj na analýzu sieťovej bezpečnosti je navrhnutý, aby kontroloval všetky vzdialené služby, ktoré môžu obsahovať závažné bezpečnostné zraniteľnosti a slabiny, a vytvára sumár výsledkov v tvare log-súborov existujúcich na serveroch (logy z automatického ťaženia, párovanie OS odtlačkov, bežiace služby, ICMP odpovede, štandardné akcie, zistené zadné dvere, rpc služby, krehké CGI skripty, a ďalšie).

Podľa nastaveného *cieľa* ja možné bezpečnostné analýzy vykonávať pre jednu klientskú stanicu (host) (-**h** meno), beztriedne zadelenie IP adries počítača
(-**s** štart_ip -**e** koniec_ip), alebo zo súboru obsahujúceho zoznam mien a/alebo IP adries počítačov (-**f** súbor).

Nástroj na analýzu sieťovej bezpečnosti podporuje niekoľko *volieb*, vrátane časových limitov a voľby pre množstvo zobrazených informácií, ktoré pomáhajú pri nastavovaní skenovania a pri nastavení optimálnych hodnôt na rozdielne typov skenovania.

-**L** Používaj utáraný spôsob záznamu

-**E súbor výnimiek** určuje súbor obsahujúci rozsahy sieťových adries v CIDR (RFC 1519) notácii, ktoré budú vyňaté zo skenovania. Tento zoznam môže obsahovať adresy ako 127.0.0.1/8, 192.3.4.5, 200.10.10.0/24, a tak ďalej. Keď skenovací proces začne, adresy vo vnútri súboru budú preskočené.

-**C konfiguračný súbor** určí voliteľný konfiguračný súbor. NSAT ho použije miesto štandardného konfiguračného súboru nsat.konf. Konfiguračný súbor určí správanie sa, služby, porty, vstup a databázový súbor mien, ktorý NSAT používa. Je odporúčané používať upravený konfiguračný súbor podľa priania zákazníka na dosiahnutie presných výsledkov.

-**V virtual host** presne určí virtuálne IP adresy vo vašej sieti, ktoré sa budú skenovat. Ak je virtuálny host zapnutý a smerovaný ako virtuálna adresa na lokálnom stroji, všetko skenovanie (okrem vzdialeného OS odtlačku, ktorý je zriedka zistiteľný) sa javí ako prichádzajúce z IP adresy.

-**n** štandardne sa po spustení odsunie NSAT na pozadie a ukončí sa automaticky po skončení skenovania. Špecifikovaním –n voľby, zostane v popredí a bude zobrazovať informácie o stave skenovania. Toto môže byť užitočné pri ladiacich účeloch, a ak sa rozhodnete používať NSAT v individuálnom okne.

-**t sekundy** presne určí rozdielny časový limit pre I/O operácie, ako v connect(), read() a tak ďalej, štandardný časovač je nastavený na 5 sekúnd. Nastaviť túto hodnotu na vyššie znamená, ukončiť proces skenujúci porty pred tým, ako môže dokončiť všetky testy. I/O časovanie je optimalizované na nie celkovo veľké a mnohonásobné skenovanie.

-**m počet** nastaví maximálne množstvo procesov. Tento limit zabráni prideleniu viacerých procesov a deskriptorov ako zvládne skenovací stroj.

-**l sekundy** čas, ktorý proces smie skenovat daného hostiteľa. Skenujúci proces kontroluje často opakované aktivity, a ak toto maximum času pre jedno vlákno vyprší, vlákno bude ukončené. Štandartne je to 200 sekúnd.

-**i sekúnd** špecifikuje systémový nevyužitý čas v sekundách. Systémový nevyužitý čas je množstvo sekúnd, keď systém nepracuje. Táto hodnota je významná, ak sa aktivuje prepínač -**c** -režim prehľadávania. Štandartne je to 90 sekúnd.

-**p 0/1** zapnúť alebo vypnúť skenovací mód postavený na ping. Pred skenovaním portov, ping zistí, či je každý z cieľov vykonateľný. Ak je táto možnosť zapnutá, počítače odpovedajúce mimo ping časovača, nebudú skenované. Táto voľba je štandardne zablokovaná.

-**c 0/1** zapnutie alebo vypnutie krycieho módu. Ak je povolený, nástroj sa bude maskovať a prerušovať, ak boli skenovacie aktivity spozorovane cieľom. Táto voľba je štandardne zablokovaná.

-**V 0-3** výber intenzity skenovania. Nástroj môže čítať a vyhodnotiť vstup zo stanovených služieb, len hlásiť ich prítomnosť, alebo vypúšťať niektoré skenovania úplne. To môže obvykle urýchliť skenovací proces a zminimalizuje správy o službách, ktoré nie sú skutočne miestom potencionálnych zraniteľností. Kým použitie stručných výpisov môže viesť k dlhšiemu času na hodnotenie log súborov.

Tak ako pri predchádzajúcom nástroji, aj pri tomto sme implementovali funkcionality pomocou elementov formulárov a zreťazenia výsledného príkazu podľa stanovených podmienok.

9.5. NIKTO

Pri implementácii funkcionalít systému NIKTO (nástroj na hľadanie štandardných webových súborov a skúmanie webového servera a CGI bezpečnosti) sme postupovali podobne, pričom sme prihliadali na túto syntax v príkazovom riadku [6]:

nikto *[-h cieľ] [voľby]*

Kde :

-**host** (-**h**) je cieľový počítač(e) na testovanie. Toto môže byť IP adresa počítača alebo hostiteľské meno, alebo súbor s adresami počítačov alebo hostiteľských mien. Je to povinný parameter.

Pričom voľby môžu byť (všetky môžu byť skrátené na prvé písmeno (t.j. -m pre -mutate), s výnimkou -verbose a -debug):

-**Cgidirs** Voliteľný test, CGI adresáre sa budú skenovat. Platné hodnoty sú 'none' nič nekontrolovať, 'all' kontrolovať všetky CGI adresáre (tak ako neschválený -allcgi), alebo cesta špecifikovaná pre CGI adresára, t.j. ' /cgi/'.

-**cookies** Vytlačí názvy cookie a hodnoty, ktoré boli prijaté v priebehu skenovania.

-**evasion** IDS maskovacie techniky. Viacnásobné voľby môžu byť tvorené reťazou čísel spolu, t.j. aktivizovať metódy 1 a 5, používať "-e 15".
Platné voľby sú:

1. náhodné URI zakódovanie (nie UTF8)
2. pridá adresár s vlastným odkazom /./
3. predčasné URL ukončenie
4. úmyselné dlhá náhodná reťaz žiadosti
5. falšuje parametre súboru
6. TAB ako požiadavka nahrádza medzeru
7. náhodná citlivosť
8. používa riadiaci oddeľovač systému Windows namiesto /
9. relačné spájanie

-**findonly** Používať port scan pre nájdenie platných HTTP a HTTPS portov, ale neuskutočňuje kontroly proti ním.

-**Format** Výstupný formát pre špecifikovaný súbor -output voľbou.
Platné formáty sú:

- HTM - HTML formát výstupných dát.
- TXT - text formát výstupných dát. To je predvoľba ak -F nie je špecifikovaný.
- CSV - Comma Seperated Value format.

-**generic** vnútenie plného skenovania namiesto skenovania podľa identifikačného reťazca servera

-**id** Používanie HTTP autentifikácie, formát je identifikácia používateľ:heslo pre autorizovanie Nikto web servera. Pre NTLM oblasti, formát je id:heslo

-**mutate** Mutované testy. Toto spôsobuje, že Nikto odloží všetky súbory so všetkými adresármi z .db súborov a hostiteľov. Možno, že by ste mohli nájsť niektoré zvláštnosti. Všimnite si, že to generuje veľa testov.

-**nolookup** Neuskutočňuje vyhľadávanie na základe mena hostiteľa.

-**output** Píše výstup do zadaného súboru po skončení testov. Formát je text v špecifikovanom formáte.

-**port** Číslo portu na skenovanie, pred voľba je port 80. Môže to byť tiež rozsah alebo zoznam portov, ktorý Nikto bude skenovat na webovom serveri. Ak webový server je nájdený, uskutoční sa plné skenovanie iba ak -f voľba je používaná.

-**root** zmena "/password.txt" na "/directory/password.txt" .

-**ssl** Test v SSL móde na uvedenom porte. Všimnite si, že Nikto sa pokúša určiť automaticky čí ide o port HTTP alebo HTTPS, ale toto môže byť pomalé ak server neodpovedá alebo oneskorí sa odpovedať kvôli chybe. Toto nastaví SSL používanie pre *all* host a porty.

-**timeout** Časový limit pre každú požiadavku, predvoľba je 10 sekúnd.

-**useproxy** Používa definovaný proxy v config.txt pre všetky požiadavky.

-**vhost** Virtuálny host na používanie pre "Host:" hlavičku, v tom prípade to je odlišne od cieľa.

-**Version** Vypíše verziu Nikto nástroja, všetky plugin-y a všetky databázy.

Tieto voľby nemôžu byť skrátene na prvé písmo:

-**dbcheck** Táto voľba skontrolujte syntax v súboroch scan_database.db a user_scan_database.db .To je užitočné iba ak pridávate testy alebo máte s testmi problémy.

-**debug** Vypíše veľké množstvo detailných informácií. Vo väčšine prípadov je to viac informácií než potrebujete, tak skúste najprv -verbose.

-**update** Týmto sa pripojíte k cirt.net a načítate aktualizovanú scan_database.db a plugin-y. Používajte to s vedomím, že sťahujete súbory, možno aj kódy, z "nedôveryhodného" zdroja. Táto voľba nemôže byť v kombinácii s inými, ale požadované premenné (ako PROXY nastavenia) budú načítané z config.txt súboru.

-**verbose** Vypísanie veľkého množstva dát v priebehu testovania. Toto môže byť užitočné v prípade že spadol server, alebo na sledovanie ako server reaguje na každú požiadavku.

Pri implementácii v prostredí PEAR sme postupovali podobne ako pri funkcionalitách nástroja SARA.

9.6. NMAP

Posledným implementovaným nástrojom integračnej časti je nástroj NMAP. Pri implementácii sme vychádzali taktiež z jeho syntaxe uvedenej aj v [7].

`nmap` *[voľby] [cieľ]*

Pričom máme na výber tieto voľby:

-sL	Aktivovanie jednoduchého skenovania
-sP	Ping skenovanie
-P0	Deaktivovanie Ping skenovania
-PS [portlist]	TCP SYN Ping skenovanie
-PA [portlist]	TCP ACK Ping skenovanie
-PU [portlist]	UDP Ping skenovanie
-PE, -PP,-PM	Rôzne typy ICMP skenovania
-PR	ARP Ping skenovanie
-sU	UDP skenovanie
-sN	TCP Null skenovanie
-sF	FIN skenovanie
-sX	Xmas skenovanie
-sA	TCP ACK skenovanie
-sW	TCP Window skenovanie
-sM	TCP Maimon skenovanie
--scanflags	Voliteľné TCP skenovanie
-sI zombie host[:probeport]	Idlescan skenovanie
-sO	IP protokol skenovanie
-sR	RPC skenovanie
-b ftp relay host	FTP skenovanie
-iL inputfilename	Vstup zo zoznamu
-iR num_hosts	Náhodný výber cieľov
--exclude host1[,host2][,host3],...	Vylúčenie cieľov/sietí zo zoznamu
--excludefile exclude_file	Vylúčenie zoznamom - zo súboru
-n	Deaktivovanie DNS rozlíšenia
-R	DNS rozlíšenie pre všetky ciele
--system_dns	Použitie systému DNS rozlišovania
-p	skenovanie iba špecifikovaných portov
-F	rýchle skenovanie portov – limitované
-r	skenovanie portov v sekvenčom poradí => nie náhodne

-sV	Aktivovanie detekcie verzií
--allports	Nevylúč žiadny port z detekcie
--version_intensity intensity	Nastavenie intenzity
--version_light	Nastavenie „light“ módu
--version_all	Detekuje jednotlivo každý port
--version_trace	Zistenie verzie skenovacej aktivity
-O	Aktivuje detekciu OS
--osscan_limit	Udanie limitu pre OS detekciu
--osscan_guess,--fuzzy	Nastavenia pre výsledky OS detekcie
--min_hostgroup milliseconds, **--max_hostgroup milliseconds**	Nastavenie veľkostí skupiny paralelného skenovania
--min_parallelism milliseconds, **--max_parallelism milliseconds**	Nastavenie paralelizmu pre vyšetrovanie Nmap-om
--min_rtt_timeout milliseconds, **--max_rtt_timeout milliseconds,** **--initial_rtt_timeout milliseconds**	Nastavenie časových limitov vyšetrovania Nmap-u
--host_timeout milliseconds	Vzdanie sa vyšetrovania pomalých hostí
--scan_delay milliseconds, **--max_scan_delay milliseconds**	Nastavenie oneskorenia medzi vyšetrovaniami
-T Paranoid|Sneaky|Polite| |Normal|Aggressive|Insane	Nastavenie časovej šablóny

Ako pri ostatných nástrojoch, aj tu je najprv potrebné vybrať cieľ pomocou elementov formuláru, potom nastaviť prepínače, zreťaziť premenné do príkazu a tento následne vykonať a zachytiť výstup už spomenutým spôsobom.

10. Testovanie

Kapitola zahŕňa poznatky získané pri overovaní funkčnosti systému, jeho obidvoch častí.

10.1. Forma testovania

Testovanie systému prebiehalo v dvoch úrovniach. Prvá úroveň je overiť si prehľadnosť a schopnosť poskytnúť poznatky o penetračnom testovaní. Táto úroveň sa vzťahuje predovšetkým na prezentačnú časť systému.

Čo sa týka integračnej časti systému, zvolili sme vyčerpávajúcu formu testu, a to skontrolovať funkčnosť všetkých testov, ktoré integračné prostredie ponúka.

Testovanie sme realizovali na sieti zloženej z 2 počítačov. Na jednom sa prevádzkovala samotná aplikácia (počítač opísaný v časti implementácie), druhý predstavol pracovnú stanicu (desktop) s operačným systémom linux Ubuntu 5.10, a na ktorom sa prevádzkuje mnoho služieb (ssh, mysql, apache, php, samba, ftp).

10.2. Výsledky testovania

Prezentačná časť projektu prehľadným spôsobom podáva informácie o danej problematike. Rozhranie je prehľadné a čitateľné.

Integračné prostredie umožnilo spúšťanie rozmanitých integrovaných testov, poskytovalo výsledky o nich. Boli odskúšané všetky navrhnuté a implementované vlastnosti.

11. Zhodnotenie

Kapitola poukazuje na postrehy autorov a na ich dojem z vykonaného projektu na ďalší profesionálny rast.

11.1. Nesplnené požiadavky

Väčšina požiadaviek zadávateľa projektu a požiadaviek vyplývajúcich z ponuky a špecifikácie bola splnená. Avšak zmenil sa návrh kapitol, a tak z neho vypadla aj kapitola testovanie znalostí. Ostatné požiadavky boli splnené, aplikácia je web aplikáciou, je prehľadná, zrozumiteľná a pre používateľa príjemná po vizuálnej stránke.

11.2. Prínos projektu

V prvom rade sa autori naučili nové a zaujímavé informácie z oblasti bezpečnosti počítačových systémov, konkrétne v penetračnom testovaní. Oboznámili sa s nástrojmi na vykonávanie testov, čo im pomôže v druhej časti projektu, pri vytváraní integračného prostredia.

Naviac sa autori zdokonalili v tvorbe web prezentácií, v navrhovaní a kreslení diagramov, v tvorení animácií a vytváraní obrázkov. Oboznámili sa s CMS systémami a naučili sa ich ovládať a prispôsobovať svojim potrebám.

V integračnej časti sme sa naučili ovládať prostredie Pear, zdokonalili sa v PHP a preštudovali sme možnosti konkrétnych nástrojov na penetračné testovanie.

12. Použité skratky

CSM -	Content Management System
SARA -	Security Auditor's Research Assistant
NIKTO-	Nástroj na hľadanie štandardných webových súborov a skúmanie webového servera a CGI bezpečnosti
NSAT -	Network Security Analysis Tool
NMAP-	Network Mapper

13. Literatúra

[1] - Informácie o systéme Drupal
http://drupal.org/about
http://drupal.org/node/22963
http://drupal.org/features
(posledný prístup 17.11.2005)

[2] - Informácie o systéme PHP-Nuke
http://www.phpnuke.org/modules.php?name=FAQ&myfaq=yes&
id_cat=1&categories=
(posledný prístup 17.11.2005)

[3] - Informácie o systéme Plone
http://plone.org/about/plone/
(posledný prístup 17.11.2005)

[4] - Prehľad a hodnotenie CMS systémov s otvoreným kódom
http://www.opensourcecms.com/index.php?option=content&task=view&
id=388&Itemid=143
(posledný prístup 17.11.2005)

[5] - Manuálová stránka k systému SARA
http://www-arc.com/sara/sara8.html
(posledný prístup 27.4.2006)

[6] - Manuálová stránka k systému NIKTO
http://www.cirt.net/nikto/README_nikto.html
(posledný prístup 27.4.2006)

[7] - Manuálová stránka k systému NMAP
http://www.insecure.org/nmap/man/
(posledný prístup 27.4.2006)

[8] - Manuálová stránka k systému NSAT
http://sourceforge.net/projects/nsat
(posledný prístup 27.4.2006)

[9] - Návod na používanie Pear
http://www.midnightax.com/quickform.php
(posledný prístup 27.4.2006)

Príloha A - Používateľská príručka prezentačnej časti

1. Spustenie web aplikácie

Na spustenie aplikácie je potrebné mať nainštalovaný a spustený prehliadač www stránok, napríklad MS Internet Explorer, Mozilla alebo Mozilla Firefox.

Aplikácia je umiestnená na adrese :
http://www2.dcs.elf.stuba.sk/TeamProject/2005/team17/portal/

2. Úvodná stránka a pohyb v prezentačnej časti

Po zadaní adresy sa načíta úvodná stránka prezentačnej časti aplikácie.
V ľavom pruhu máme na výber, o čom chceme čítať. Na úvodnú stránku sa môžeme kedykoľvek dostať kliknutím na logo v ľavom hornom rohu.

Ľavý pruh ostáva zobrazený po celý čas práce s aplikáciou. Obsahuje kapitoly, ktoré môžu obsahovať ešte podkapitoly. V prípade, že kapitola má nejaké podkapitoly, rozvinie sa v ľavom pruhu možnosť kapitoly po kliknutí na ňu do podkapitol.

Zvolená kapitola alebo podkapitola sa zobrazí vždy napravo od pruhu pre voľbu kapitoly. V kapitole sa dá listovať ako je zaužívané v internetových prehliadačoch.

3. Integračná časť

Integračná časť je realizovaná ako kapitola prezentačnej časti, preto je odkaz na ňu zahrnutý v ľavom menu pre výber kapitoly. Po kliknutí na odkaz sa zobrazí integračná časť aplikácie.

4. Ukončenie práce

Pokiaľ sme len v prezentačnej časti aplikácie, nie je potrebné nijak zvláštne uzatvárať bežiacu aplikáciu. Prácu je preto možné ukončiť napísaním novej adresy do internetového prehliadača, alebo jeho zatvorením.

Príloha B - Systémová príručka prezentačnej časti

1. Inštalácia systému

Požiadavky na prevádzku prezentačného systému sú:

- Webový server s interpretom jazyka PHP4 (odporúča sa verzia >= 4.3.8). Nakoľko sa jazyk PHP búrlivo vyvíja a zachovanie spätnej kompatibility nie je vždy na 100%, prevádzku na iných verziách jazyka (4.4, 5.0, 5.1) je potrebné otestovať.
- Databázový server Mysql, verzie 4.0 alebo 4.1.
- Prehliadač webových stránok, bližšie v používateľskej príručke.

Z hľadiska OS je systém platformovo nezávislý. Prostredie na použitom OS musí spĺňať uvedené požiadavky.

Pred inštaláciou je potrebné stiahnuť si najnovšiu verziu prezentačného systému z linky:
http://www2.dcs.elf.stuba.sk/TeamProject/2005/team17/ITdokumenty.htm

Inštalácia zahŕňa tieto kroky:

1. Nakonfigurovať použitý webový server, určiť koreňový adresár dokumentov.
2. Do koreňového adresára webového servera rozbaliť obsah archívu *portal.zip*.
3. Nakonfigurovať Mysql databázu. Vytvoriť prázdnu databázu a používateľa s plnými právami na túto databázu.
4. Naplniť databázu pomocou dodaného sql skriptu *portal.sql*.
5. Nastaviť prístup do databázy pre systém editovaním konfiguračného súboru. Tento sa nachádza relatívne vzhľadom na koreňový adresár webu: *sites/default/settings.php*. Je potrebné zmeniť parameter *$db_url* podľa nasledovnej schémy:
   $db_url='mysql://meno_uzivatela:heslo@nazov_*databazoveho_servera/meno_databazy';*
   Pre hodnoty užívatel=portal_user, heslo=aaa, databáza=portal, kde databázový server je na rovnakom stroji ako webový server bude konfigurácia takáto:
   $db_url='mysql://portal_user:aaa@localhost/portal';
6. Nastaviť východziu url pre systém – parameter *$base_url*. Táto záleží od nastavenia webového servera a OS. V našom prípade:
   *base_url='*http://www2.dcs.elf.stuba.sk/TeamProject/2005/team17/portal*';*

2. Rozširovanie systému

Prezentáciu je možné vďaka použitému CMS systému ľubovolne rozširovať. Je možné pridávať nové informácie a tieto zaraďovať do systému menu, prípadne menu doplniť novými témami.
Pre pridávanie informácií je nutné byť v systéme prihlásený ako užívateľ. Užívateľov môže pridávať administrátor systému, ktorý zároveň vykonáva aj správu systému. V práve nainštalovanom systéme je k dispozícii administrátorské rozhranie pod týmito prihlasovacími údajmi:
meno: admin

heslo: admin
Po prvom prihlásení je z bezpečnostných dôvodov nevyhnutné zmeniť si toto heslo. Prihlasovací dialóg sa vyvolá pridaním parametra *user* do URL adresy. V našom prípade:
http://www2.dcs.elf.stuba.sk/TeamProject/2005/team17/portal/?q=user

V ďalšom texte sa predpokladá, že užívateľ je v systéme prihlásený.

Pridanie nových informácií – obsahu

1. Z menu vybrať položku *Create content.*
2. Zvoliť si typ výstupu *Page* alebo *Story*.
3. Vložiť nadpis (*Title*).
4. Editovať informácie (*Body*). Môže ísť o čistý text s minimom HTML značiek (*Filtred HTML*), o text plne využívajúci formátovanie pomocou HTML (*full HTML*), alebo dynamické informácie využívajúce PHP kód (*PHP code*).
5. Zadefinovať dodatočné parametre (Read only povaha obsahu alebo Read Write...).
6. Potvrdiť vytvorenie nového obsahu (*Submit*).

Štruktúrovanie informácií do menu

Túto činnosť môže vykonávať len administrátor.

1. Z menu vybrať Administer->menus->add menu.
2. Zadať názov a potvrdiť.
3. Z menu vybrať Administer->blocks.
4. Ak chcem novovytvorené menu zviditeľniť, zaškrtnem políčko Enabled.
5. Položka Weight určuje poradie zobrazenia jednotlivých menu.
6. Ďalej vystavíme vytvorený obsah ako linku v rámci menu.
7. Z menu vybrať Administer->menus->add menu item.
8. Zadám nadpis (*title*), ktorý sa zobrazí na začiatku bloku obsahu. Krátky popis (*description*) a cestu (*path*) k obsahu vo forme node/N. N je číslo nášho obsahu vytvoreného vyššie. Čísluje sa od 1, číslo sa pre každý pridaný obsah inkrementuje.
9. Určím, v rámci ktorého menu sa zobrazí táto linka (*Parent item*).
10. Parameter Weight bol popísaný vyššie.
11. Potvrdím vytvorenie linky s obsahom.
12. Pozriem sa do menu a overím si existenciu novej linky s obsahom.

Tu uvedené návody sú len stručné, nakoľko systém Drupal je dobre zdokumentovaný. Preto na tomto mieste uvádzame linku, na ktorej možno nájsť veľa ďalších informácií. Sú určené pre užívateľov, ktorý si chcú prezentáciu rozširovať a upravovať do väčšej hĺbky a podrobností:

http://drupal.org/handbooks

Príloha C – Inštalačná príručka integračnej časti

Aplikácia sa dodáva vo forme 3 súborov:

- tester_app.tgz - Balíček vytvorený pomocou nástroja tar a komprimovaný pomocou gzip. Samotná aplikácia testera v jazyku PHP.
- tester_www.tgz - Balíček vytvorený pomocou nástroja tar a komprimovaný pomocou gzip. Obsahuje len tie časti aplikácie, ktoré musia byť nevyhnutne prístupné pomocou webového servera.
- tester.sql – skript naplní databázu MySQL tabuľkami potrebnými pre aplikáciu.

Z bezpečnostných dôvodov sú umiestnené v koreňovom adresári webového servera len nevyhnutne potrebné súbory (index.php, obrázky a pod.). Ostatné súbory sú mimo neho, ale musia byť čitateľné webovým serverom. Takto sa zabráni spúšťaniu ľubovolných skriptov aplikácie pomocou upravenia url v prehliadači.

- Vytvorte adresár pre koreňový adresár webu. (napríklad /var/tester/www).
- Rozbaľte v ňom balíček tester_www.tgz. (tar -xzf tester_www.tgz).
- Nastavte webový server Apache tak, aby bolo možné pristúpiť k tomuto adresáru cez web (koreňový adresár webu), ale aby bol prístup obmedzený len na https. Príklad konfigurácie:

```
NameVirtualHost *:443
<VirtualHost *:443>
     DocumentRoot /var/tester/www
     SSLEngine on
     SSLCertificateFile /etc/apache2/ssl/apache.pem
</VirtualHost>
```

- Vytvorte adresár pre aplikáciu, ktorý je mimo koreňového adresára webu. (napríklad /var/tester/app).
- Rozbaľte v ňom balíček tester_app.tgz. (tar -xzf tester_app.tgz).
- Podľa umiestnenia samotnej aplikácie (z bodu 4) nakonfigurujte webovú časť testera. Editujte súbor /var/tester/www/index.php. V úvode zmeňte konštantu APP_PATH na adresu aplikácie testera. Príklad:
  define("APP_PATH","/var/tester/app");
- V MySQL databáze vytvorte užívateľa pre prístup k databáze testera (napríklad meno tester, heslo 123 (toto heslo je len príklad, rozhodne doporučujeme vytvoriť bezpečnejšie heslo)).
- Naplňte databázu pre potreby testera. Skript si sám vytvorí aj databázu s názvom tester. Ak to nie je želané, editujte začiatok súboru tester.sql (časť CREATE DATABASE). Príklad:
  mysql -u tester -p < tester.sql
- Nakonfigurujte aplikáciu tester pre prístup do tejto databázy. Editujte súbor /var/tester/app/DataObjects/DataObjects.ini. Na riadku *database* zmeňte údaje podľa údajov z bodu 7. Na riadku *schema_location a class_location* zmeňte umiestnenie podľa adresára aplikácie testera zvoleného v bode 4.
  Príklad:
  [DB_DataObject]

database = mysql://tester:123@localhost/tester
schema_location = /var/tester/app/DataObjects
class_location = /var/tester/app/DataObjects
*class_prefix = DataObject_*

- Na zvýšenie bezpečnosti nastavte minimálne práva na aplikáciu tester. Príklad ak server Apache beží pod užívateľom www-data (Debian):
chown www-data:www-data -R /var/tester
chmod 500 -R /var/tester
chmod 700 /var/tester/app/sess

Príloha D - Riadenie projektu