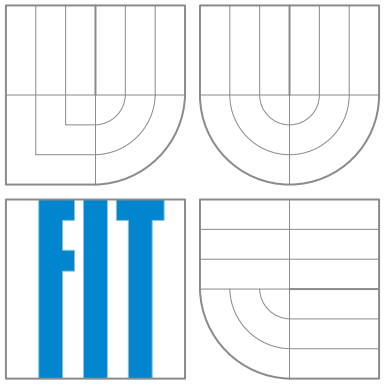

Fakulta informačních technologií
Vysokého učení technického v Brně

Dokumentácia k predmetu ISA

Zoznam služieb bežiacích na zadaných počítačoch

4. novembra 2012

Autor: Richard Chomjak, xchomj00@stud.fit.vutbr.cz

Obsah

Kapitola 1

Úvod

Tento dokument obsahuje problematiku, návrh a popis implemetácie programu.
Dokument taktiež obsahuje ovládateľnosť a popis programu `tcpsearch`.

Kapitola 2

Úvod do problematiky

Zisťovanie bežiacich služieb na počítačoch je jedna zo základných bezpečnostných metodík. Z pohľadu správcu systému ma pomôcť zaistiť bezpečnosnosť spravovaného systému. Z druhej strany bojového frontu ma uľahčiť použitie správnych programových nástrojov[1] pre vniknutie do systému. Existuju viaceré metodiky skenovania napr. TCP/UDP skenovanie, ICMP skenovanie, TCP/IP odtlačky OS[2].
Každá metodika skenovania sa zameriava na určitý typ služby a chovania.[3]

Jedna z najjednoduchších metodík skenovania služieb počítačov je TCP skenovanie.
Táto metodika bola použita v tomto projekte.

[1] V anglickom jazyku exploit.
[2] V anglickom jazyku TCP/IP OS fingerprint
[3] Pre najpresnejšie zistenie bežiacích služieb sa dané metodiky kombinujú.

2.1 TCP skenovanie

TCP skenovanie ako bolo už spomenuté je jedna z najjednoduchších metodik skenovania. Protokol TCP využíva synchronnú komunikáciu, ktorá je inicializovaná trojkrokovým potvrdzovaním[4].
Týmto spravanim protokolu TCP sa predcháchadza vytvoreniu asynchronnej chyby[5] na danom sokete[6] pri odosielaní a príjmani dát ako je to u protokolu UDP.

Overenie spojenia

Pri pripájani alebo pri posielaní uživateľských dát na cieľový systém može vzniknuť asynchrónna chyba, ktorá je sposobená chybou spojenia alebo filtráciou[7] dat cieľovým systémom. Táto skutocnost nezaručuje, že na danom cieľovom sokete nebeží žiadna služba.
Po úspešnom naviazaní spojenia a komunikáciou s cieľovým systémom je možne potvrdiť, že daný soket je otvorený a umožňuje prijímať alebo posielať dáta.[8]

Zistenie používaného protokolu

Zistenie konkrétneho protokolu na danom sokete je možné zistiť jeho správanim[9]. Tato metóda je exaktná.
Okrem tejto metódy existujú aj mnoho ďalších. Program `tcpsearch` využíva metódu zisťovania bannera[10], ktorú nie všetky protokoly umožňujú. Dana metóda je pasívna, to znamená neposiela žiadne dáta, ale len prijíma.[11]

[4] V anglickom jazyku three-way handshake.
[5] Asynchronná chyba može vznikať aj pri inicializovaní spojenia.
[6] V anglickom jazyku socket.
[7] Na cieľovom systéme može byť spustená služba, ale pre konfiguráciu zdrojového systému je komunikácia zamietnuta.
[8] Funkcia `shutdown` umožňuje nastaviť duplexitu soketu.
[9] Na zaklade odosielania a prijímania konkrétnych dát.
[10] Banner môže slúžiť na zaistenie kompatibility verzie protokolov. Tento postup využíva protokol SSH.
[11] Z pohľadu aplikácie.

Kapitola 3

Aplikácia tcpsearch

3.1 Návrh aplikácie

Program `tcpsearch` využíva k uloženiu informácii zanorený lineárny jednosmerný zoznam. Kde prvá úroveň lineárneho zoznamu sa vytvára pri načítaní informácii o cieľovom systéme zo súboru. Druha úroveň uchováva informácie o výsledkoch komunikácie so cieľovým systémom. Zanorený lineárny jednosmerný zoznam je uložený v štruktúre `nodemain`, ktorá obsahuje informácie o parameteroch programu.

Implementácia programu je členená na viacero časti, ktoré spracovávajú určitú množinu úloh.

3.2 Implementácia programu

Program je implementovaný v programovacom jazyku `C`. Návrh programu zohľadnuje možnosť pridávania ďalších funkcií. Komentáre v zdrojovych súborov sú kompatibilne s programom `doxygen`.

Spracovanie parametrov

Program `tcpsearch` na spracovanie parametrov využíva funkciu `mgetopt`. Funkcia `mgetopt` na zaistenie požiadaviek využíva aj iné funkcie. Funkcie na spracovanie parametrov sa nachádzajú v súboroch `mgetopt.c` a `mgetopt.h`.

Výsledky zo spracovania parametrov sa ukladajú do štruktúry `mgetopt`. Popis štruktúry sa taktiež nachádza v súbore `mgetopt.h`.

Spracovanie portov

Po overení vstupných parametrov a uložení údajov do štruktúry program začne so spracovaním cieľových portov. Pre overenie validnosti cieľových portov sa využíva deterministický automat. Automat využíva funkcie:
`parser_automata_state_0`, `parser_digit_state_1`, `parser_dash_state_2`, `parser_comma_state_3`. Jednotlive porty su ukladané do pola `ports` v štruktúre `nodemain`. Na ukladanie portov slúži funkcia `ports_push`, ktorá je popísaná v súboroch `ports_opretions.c` a `ports_operations.h`. Každý vkladaný port sa overí pomocou funkcie `ports_convert`, aby jeho hodnota nenabúdala väčšiu hodnotu ako je definovaná v makre `MAXIMUM_PORT_VALUE` v súbore `ports_operations.h`.

Pri použití rozsahu portov sa pomocou funkcie `ports_compare` overí, ze prvý parameter rozsahu je menší ako druhy parameter. Následne na to sa zavolá funkcia `ports_push_range`, tato funkcia ma za úlohu vkladať rozsah portov cez funkciu `ports_push`.

Automat pri spracovaní portov využíva makro `MAX_DIGIT`[1], ktoré mu indikuje maximálny počet bajtov pre načítavanie udajú.

Načítavanie zo súboru

Načítanie zo súboru využíva deterministický automat, ktorý je implementovaný funkciou `file_read` v súbore `file_read_parser.c`. Stav automatu popisujú štyri stavy `Fstart`, `Fget`, `Fingore` a `Fnewline`. Stav `Fstart` je počiatočný stav. Stav `Fget` indikuje stav, kedy je možné načítavať dáta do pomocného zásobnika [2]. Automat sa dostane do stavu `Fignore` keď narazí na prázdny znak. Ak automat načíta vstupný znak, ktorý je reprezentovaný ako `\n` stav automatu sa zmení na `Fnewline`. Zmena stavu z `Fnewline` na stav `Fget` nastáva pri prečítaní validného vstupu [3].

Pri ukončení načítavania dát v stave `Fget`.[4] Tak sa vytvorí položka v lineárnom zozname, ktorý sa nachádza v štruktúre `nodemain->host`, štruktúra `nodemain` sa nachádza v súbore `tcpsearch.h`. Do položky lineárneho zoznamu bude pomocou funkcie `memcpy` kopírovaný obsah pomocného zasobnika. Po načítaní vstupného súboru, program začne vykonávať sieťové operácie.

Sieťové operácie

Sieťová časť projektu je implementovaná v suboroch `network.c` a `network.h`. Iterácia nad lineárnym zoznamom `host` vo funkcii `network_operation` ma za úlohu prechod nad každým cieľovým systémom. Na overenie dosiahnutia cieľového systému sa používa funkcia `getaddrinfo`. Výsledná štruktúra z funkcie `getaddrinfo` je uložená vo položke prechadzaneho prvku linearnom zozname.

Ak na cieľový systém je možne incializovať spojenie, tak funkciami `socket` a `connect` sa program snaží zahájiť inicializáciu komunikácie. Asynchrónna chyba, ktorá môže nastať pri používaní funkcie `connect` je ošetrená pomocou signálu `SIGALRM`. Časový interval pre signál `SIGALARM` je definovany v makre `DEFAULT_CONNECT_TIMEOUT`[5]. Po úspešnom pripojení na cieľový systém funkcia `wait_to_read` nastavuje časové prerušenie, ktoré je definované používateľom.[6] Po uplynutí časovača bude pripojený soket odpojený od cieľového systému. Príchodzie dáta program ukladá do pomocného zásobnika. Zásobnik sa plni funkciou `recv`. Po ukončení prijímania sa vytvorí položka lineárneho zoznamu v štruktúre `nodemain->host->result` pre cieľový soket, na ktorý sa bolo možné pripojiť.

Výpis informácií

Implementácia výpisu informácií je uložená v súboroch `print.c` a `print.h`. Procedúra `print_result` prechádza zanorenými lineárnymi zoznamami a vypisuje hodnoty v daných štruktúrach. Vstupný parameter programu `-s` umoznuje prevadzat cisla portov na mena sluzieb funkciou `getservbyport`. Informácie sa vypisujú na súbor `stderr` alebo `stdout`.

---

[1] Makro sa nachádza v súbore ports_range_parser.h.

[2] V zdrojom texte buff.

[3] i $\epsilon\{[A-Z]\} \vee \{[a-z]\} \vee \{[0-9]\} \vee \{:\} \vee \{.\}$, kde i je čítaný znak.

[4] Ak sa zmení stav na `Fnewline` alebo `Fignore` poprípade ak nastane stav `EOF`.

[5] Makro sa nachádza v súbore tcpsearch.h

[6] Časové prerušenie je implementované funkciou `select`.

3.3 Základné informácie programu

Program `tcpsearch` je implementovaný v programovacom jazyku `C`. Implementácia ma 2589 riadkov vrátane komentárov . Aplikácia bola úspešne otestovaná na servere `eva.fit.vutbr.cz`[7] a na `UNIX`-ovom systéme [8].

3.4 Použitie

Použitie programu:

```
tcpsearch -p 0-65535,1,2 vstupny_subor -t 2 -c 2 -v -s

povinne parametre:
 parameter -p slúži k zadaniu portov
 vstupny_subor obsahuje zoznam cieľových systémov, pre načítanie
 zo STDIN sa ako meno použije "-"
nepovinné parametre:
 parameter -t slúži na nastavenie časového limitu prijímania dát
 parameter -c slúži na nastavenie časového limitu pripojenia
 k cieľovému systému
 parameter -v slúži na zobrazenie viacerých chyb pri
 sieťovej komunikácii.
 parameter -s slúži na preklad čísel portov na mena služieb
 parameter -h slúži k zobrazeniu pomocnej textu programu
```

[7] FreeBSD 9.1-PRERELEASE
[8] Mac OS 12.2.0 Darwin Kernel

Literatúra

STEVENS, Richard W. *Unix network programming*. 3rd ed. Boston: Addison-Wesley, 2004, xxiii, 991 s. ISBN 01-314-1155-1.

HALL, Brian. Beej's Guide to Network Programming: Using Internet Sockets. [online]. [cit. 2012-11-04]. Dostupné z: http://beej.us/guide/bgnet/

Dodatok A

Metriky kódu

Počet súborov: 19 súborov

Počet riadkov zdrojového textu: 2589 riadkov