IPK - Počítačové komunikace a sítě

7. dubna 2021 14:22:30

Projekt 2 » zpět na seznam variant

Zpět na termíny

Varianta termínu - Varianta ZETA: Sniffer paketů - přihlášen

Termín č.:	20		
Zahájení:	2021-02-16	**Ukončení:**	2021-04-25
Přihlašování od:	2021-02-21 09:27:32	**Přihlašování do:**	2021-04-23 00:00:00
Přihlášeno:	348	**Kapacita:**	500
Max. bodů:	20		
Získáno bodů:	**0**		

» **Odevzdané soubory**

Společná část popisu:

CÍL

Vytvořte komunikující aplikaci podle konkrétní vybrané specifikace obvykle za použití libpcap a/nebo síťové knihovny BSD sockets (pokud není ve variantě zadání uvedeno jinak). Mezi další povolené knihovny patří libnet, System.Net.Sockets pro C#, případně standardně přítomné hlavičkové soubory na referenčním stroji. Implicitně předpokládajte k implementaci podporu jak IPv4, tak IPv6 (pokud není explicitně řečeno jinak).

Projekt bude vypracován v jazyce C/C++/C#. V případě C# můžete například použít wrapper script, který zavolá zkompilovanu binárku, a nebo přeloží projekt jako self-contained aplikaci, a tu si potom zkopírovat do kořenové složky. Pokud se rozhodnete projekt implementovat v C#, nechť je Vaše implementace kompatibilní s .NET Core 3.1+ a out-of-the-box provozuschopná na referenční virtuálce.

Co se výběru zadání týče, tak platí, že ti z Vás, kteří minulý rok měli varianty ZETA či OMEGA letos povinně:

- **si buď musí nechat uznat body (protože nelze vypracovávat znovu ty samé projekty);**
- **a nebo si zapsat zadání varianty EPSILON či DELTA.**

DOKUMENTACE

Dokumentaci lze realizovat v libovolném textovém procesoru a jazycích čeština, angličtina, slovenština. Při tvorbě dokumentace citujte v souladu s https://www.fit.vut.cz/study/theses/citations/.cs. Za přečtení stojí i http://www.fit.vutbr.cz/~martinek/latex/citace.html.

VIRTUALIZACE

Individuální zadání specifikuje vlastní referenční systém (spustitelný např. pomocí aplikace VMWare Player nebo VirtualBox), pod kterým musí být projekt přeložitelný a spustitelný. Pokud jste ještě nikdy nevirtualizovali, třeba vám pomůže následující článek http://www.brianlinkletter.com/how-to-use-virtualbox-to-emulate-a-network/. Vyvíjet si můžete na libovolném systému, opravovat a hodnotit odevzdaný kód se však bude na referenčním. Vzhledem k nátuře projektu můžete předpokládat, že projekt bude spouštěn s rootovskými privilegii.

Referenční image (přihlašovací údaje student/student s možnou eskalací privilegií jako root) je ke stažení z těchto odkazů:

- univerzitní OneDrive
- NES fakultní server

ODEVZDÁNÍ

Vypracovaný projekt uložený v archívu .tar a se jménem xlogin00.tar odevzdejte elektronicky přes IS.

Termín odevzdání je 25.4.2020 (hard deadline). Odevzdání e-mailem po uplynutí termínu, dodatečné opravy či doplnění kódu není možné.

Odevzdaný projekt musí obsahovat:

1. soubor se zdrojovým kódem (dodržujte jména souborů uvedená v konkrétním zadání),
2. funkční *Makefile* či jiné pomocné provozy pro úspěšný překlad či interpretaci zdrojového souboru,
3. dokumentaci (soubor *manual.pdf*), která bude obsahovat uvedení do problematiky, návrhu aplikace, popis implementace, tesování.
4. plain-textový soubor *README* obsahující krátký popis programu s případnými rozšířeními/omezeními, příklad spuštění a seznam odevzdaných souborů,
5. další požadované soubory podle konkrétního typu zadání.

V rámci projektu si můžete vytvářet adresářovou strukturu; jen se pokuste dodržet, že výsledná aplikace, dokumentace a README a pomocné provozy ke kompilaci (jako Makefile) se budou vyskytovat v kořeni.

POZNÁMKY

- Pokud v projektu nestihnete implementovat všechny požadované vlastnosti, je nutné veškerá omezení jasně uvést v dokumentaci a v souboru README.
- Co není v zadání jednoznačně uvedeno, můžete implementovat podle svého vlastního výběru. Závažnější designová rozhodnutí popište v dokumentaci a README.
- Při řešení projektu respektujte zvyklosti zavedené v OS unixového typu (jako je například formát textového souboru). Program by však měl být přenositelný.
- Vytvořené programy by měly být použitelné a smysluplné, řádně komentované a formátované a členěné do funkcí a modulů. Program by měl obsahovat nápovědu informující uživatele o činnosti programu a jeho parametrech. Případné chyby budou intuitivně popisovány uživateli.
- Aplikace nesmí v žádném případě skončit s chybou SEGMENTATION FAULT ani jiným násilným systémovým ukončením (např. dělení nulou).
- Pokud přejímáte krátké pasáže zdrojových kódů z různých tutoriálů či příkladů z Internetu (ne mezi sebou), tak je nutné vyznačit tyto sekce a jejich autory dle licenčních podmínek, kterými se distribuce daných zdrojových kódů řídí. V případě nedodržení bude na projekt nahlíženo jako na plagiát.
- Před odevzdáním zkontrolujte, zda jste dodrželi všechna jména souborů požadovaná ve společné části zadání i v zadání pro konkrétní projekt. Zkontrolujte, zda je projekt přeložitelný.

HODNOCENÍ

- **Maximální počet bodů za projekt je 20 bodů.**
  - Maximálně 13 bodů za plně funkční aplikaci.
  - Maximálně 7 bodů za dokumentaci. Dokumentace se hodnotí pouze v případě alespoň nějak funkčního kódu. Pokud kód není odevzdán nebo nefunguje podle zadání, dokumentace se nehodnotí.
  - Za zajímavé přídavky lze získat i bonusové body, tyto v součtu se zbytkem nepřesáhnou maximum za projekt, lze jimi však lepit bodové ztráty z povinných částí.
- Příklad kriterií pro hodnocení projektů:
  - nepřehledný, nekomentovaný zdrojový text: až -7 bodů
  - nefunkční či chybějící Makefile: až -4 body
  - nekvalitní či chybějící dokumentace: až -7 bodů
  - špatná bibliografie: až -2 body
  - bez otestování: až -3 body
  - jiný než TARový formát archivu, případně "nepořádek" (především dočasné soubory nesouvisející s projektem) v odevzdávaném archivu: až -2 body
  - odevzdaný soubor nelze přeložit, spustit a odzkoušet: 0 bodů
  - odevzdáno po termínu: 0 bodů
  - nedodržení zadání: 0 bodů
  - nefunkční kód: 0 bodů
  - opsáno: 0 bodů (pro všechny, kdo mají stejný kód), návrh na zahájení disciplinárního řízení.

Popis varianty:

ZADÁNÍ:
1) Navrhněte a implementujte síťový analyzátor v C/C++/C#, který bude schopný na určitém síťovém rozhraním zachytávat a filtrovat pakety (13 b)
2) Vytvořte relevantní manuál/dokumentaci k projektu (7b)

UPŘESNĚNÍ ZADÁNÍ:
Ad 1)

Volání programu:

./ipk-sniffer [-i *rozhraní* | --interface *rozhraní*] {-p *port*} {[--tcp|-t] [--udp|-u] [--arp] [--icmp] } {-n *num*}

kde

- -i *eth0* (právě jedno rozhraní, na kterém se bude poslouchat. Nebude-li tento parametr uveden, či bude-li uvedené jen -i bez hodnoty, vypíše se seznam aktivních rozhraní)
- -p *23* (bude filtrování paketů na daném rozhraní podle portu; nebude-li tento parametr uveden, uvažují se všechny porty; pokud je parametr uveden, může se daný port vyskytnout jak v source, tak v destination části)
- -t nebo --tcp (bude zobrazovat pouze TCP pakety)
- -u nebo --udp (bude zobrazovat pouze UDP pakety)
- --icmp (bude zobrazovat pouze ICMPv4 a ICMPv6 pakety)
- --arp (bude zobrazovat pouze ARP rámce)
- Pokud nebudou konkrétní protokoly specifikovány, uvažují se k tisknutí všechny (tj. veškerý obsah, nehledě na protokol)
- -n *10* (určuje počet paketů, které se mají zobrazit; pokud není uvedeno, uvažujte zobrazení pouze jednoho paketu)
- argumenty mohou být v libovolném pořadí

Formát výstupu:

čas IP : *port* > *IP* : *port*, length *délka*

offset_vypsaných_bajtů: *výpis_bajtů_hexa výpis_bajtů_ASCII*

přičemž:

- *čas* je ve formátu dle RFC3339
- *délka* je v bytech

(takto vypíšete úplně celý paket)

Příklady volání:

./ipk-sniffer -i eth0 -p 23 --tcp -n 2
./ipk-sniffer -i eth0 --udp
./ipk-sniffer -i eth0 -n 10
./ipk-sniffer -i eth0 -p 22 --tcp --udp --icmp --arp .... stejné jako:
./ipk-sniffer -i eth0 -p 22
./ipk-sniffer -i eth0

Příklady výstupu:

./ipk-sniffer -i

lo0
eth0

./ipk-sniffer -i eth0

2021-03-19T18:42:52.362+01:00 147.229.13.223 : 4093 > 10.10.10.56 : 80, length 112 bytes

0x0000: 00 19 d1 f7 be e5 00 04 96 1d 34 20 08 00 45 00 ....4 ..

0x0010: 05 a0 52 5b 40 00 36 06 5b db d9 43 16 8c 93 e5 ..R[@.6. [..C....
0x0020: 0d 6d 00 50 0d fb 3d cd 0a ed 41 d1 a4 ff 50 18 .m.P..=. ..A...P.

0x0030: 19 20 c7 cd 00 00 99 17 f1 60 7a bc 1f 97 2e b7 . ...`z.....
0x0040: a1 18 f4 0b 5a ff 5f ac 07 71 a8 ac 54 67 3b 39 ....Z._. .q..Tg;9
0x0050: 4e 31 c5 5c 5f b5 37 ed bd 66 ee ea b1 2b 0c 26 N1._.7. .f...+.&
0x0060: 98 9d b8 c8 00 80 0c 57 61 87 b0 cd 08 80 00 a1 .....W a.....

Netisknutelné znaky budou nahrazeny tečkou, vypisovat můžete i případný padding.

Ad 2)

V dobré dokumentaci se OČEKÁVÁ následující: titulní strana, obsah, logické strukturování textu, výcuc relevantních informací z nastudované literatury, popis zajímavějších pasáží implementace, sekce o testování (ve které kromě vlastního programu otestujete nějaký obecně známý open-source nástroj), bibliografie, popisy k řešení bonusových zadání.

DOPORUČENÍ:

- Při implementaci použijte knihoven pcap / libnet
  Pcap: http://www.tcpdump.org/pcap3_man.html
  Libnet: http://www.packetfactory.net/projects/libnet/
- U syntaxe vstupních voleb jednotlivým programům složené závorky {} znamenají, že volba je nepovinná (pokud není přítomna, tak se použíje implicitní hodnota), oproti tomu [] znamená povinnou volbu. Přičemž pořadí jednotlivých voleb a jejich parametrů může být libovolné. Pro jejich snadné parsování se doporučuje použít funkci getopt().
- Výsledky vaší implementace by měly být co možná nejvíce multiplatformní mezi OS založenými na unixu, ovšem samotné přeložení projektu a funkčnost vaší aplikace budou testovány na referenčním Linux image (viz obecné pokyny k zadání) pro síťové předměty (přihlašovací údaje student / student).
- Očekává se použití promiskuitního módu síťové karty.
- Program by se měl dát v kterémkoli okamžiku korektně ukončit pomocí Ctrl+C.

ODEVZDÁNÍ:
Součástí projektu budou zdrojové soubory přeložitelné na referenčním operačním systému, funkční Makefile (či pomocné provozy C#), soubor manual.pdf a README (viz obecné pokyny). Projekt odevzdejte jako jeden soubor xlogin00.tar, který vytvoříte programem tar.

LITERATURA:

- Wikipedia, the free encyclopedia: http://en.wikipedia.org/wiki/Pcap
- TCPDUMP/LIBPCAP public repository: http://www.tcpdump.org/
- Odkazy na knihovnu http://packetfactory.openwall.net/projects/libnet/
- RFC 792 - Internet Control Message Protocol a RFC 4443 - ICMPv6
- RFC 826 - ARP

PATCH-NOTES:

- bude doplněno na základě případné diskuze na fóru