Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Aplikačné rozhranie pre vyhodnotenie sieťovej prevádzky v reálnom čase

Bakalárska práca

Príloha F

SYSTÉMOVÁ PRÍRUCKA JXColl v3.9

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)
Školiteľ:	Ing. Adrián Pekár
Konzultant:	Ing. Adrián Pekár

Košice 2013 **Pavol Beňko**

Obsah

1 Funkcia programu **6**

2 Popis programu **7**

2.1 Popis tried, členských premenných a metód 8

2.1.1 Trieda RecordDispatcher 8

2.1.2 Trieda ACPIPFIXWorker 10

3 Preklad programu **15**

3.1 Zoznam zdrojových textov 15

3.2 Požiadavky na technické prostriedky pri preklade 16

3.3 Požiadavky na programové prostriedky pri preklade 16

3.4 Náväznosť na iné programové produkty 17

3.5 Vlastný preklad 17

3.6 Vytvorenie inštalačného DEB súboru 17

3.7 Opis známych chýb 19

4 Zoznam použitej literatúry **20**

Zoznam obrázkov

Zoznam tabuliek

1 Funkcia programu

Program JXColl (Java XML Collector) slúži na zachytávanie a spracovávanie informácii o tokoch v sieťach získané exportérom. Tvorí súčasť meracej architektúry BasicMeter, ktorý na základe nastavených parametrov konfiguračného súboru vie dáta získané z aktuálnej sieťovej prevádzky ukladať do databázy alebo ich sprístupniť pomocou vlastného protokolu pre priame spracovanie (protokol ACP) používateľovi. Údaje uložené v databáze sú určené pre neskoršie vyhodnotenie prídavnými modulmi spomínanej meracej architektúry a sú v súlade s požiadavkami protokolu IPFIX. JXColl tiež generuje účtovacie záznamy, ktoré slúžia na analýzu sieťovej hierarchie konkrétnym používateľom z hľadiska protokolov, portov, IP adries a časových charakteristík. Program bol vytvorený Ľubošom Koščom, neskôr zoptimalizovaný a doplnený novými funkciami Michalom Kaščákom, Adriánom Pekárom a Tomášom Vereščákom.

2 Popis programu

Jednotlivé časti programu sú umiestnené v nasledujúcich balíkoch:

- sk.tuke.cnl.bm.JXColl.export - triedy určené na export údajov do databázy alebo protokolom ACP
- sk.tuke.cnl.bm.JXColl.input - triedy slúžiace na prijímanie dát z exportérov
- sk.tuke.cnl.bm.JXColl.IPFIX - triedy s manuálnou implementáciou IPFIX
- sk.tuke.cnl.bm.JXColl.accounting - triedy účtovacieho modulu
- sk.tuke.cnl.bm.JXColl - hlavné triedy samotného programu JXColl
- sk.tuke.cnl.bm.OWD - balík pre modul merania jednosmerného oneskorenia
- sk.tuke.cnl.bm - pomocné triedy a výnimky

2.1 Popis tried, členských premenných a metód

Kedže sa počas vývoja a odstraňovania chýb pôvodné triedy a ich metódy nezmenili, v nasledujúcich častiach budú uvedené len tie, ktoré boli doplnené počas riešenia jednotlivých problémov uvedených v analýze riešenia. Popis ostatných tried a metód je uvedený v príručkách predošlých verzií programu.

2.1.1 Trieda RecordDispatcher

Trieda predstavuje systém parsovania hodnôt pre databázu, ACP a pre účtovanie.

Konštruktor

public **RecordDispatcher**()
Konštruktor slúži na inicializovanie objektov databázy, ACP a účtovania. Konštruktor je singleton, čo značí že inicializácia objektu prebehne jeden krát.

Metódy

public static **RecordDispatcher**()
Metóda slúži na prístup k singleton objektu danej triedy.
Návratová hodnota:
Singleton objekt triedy RecordDispatcher.

public synchronized void **dispatchIPFIXRecord**(IPFIXTemplateRecord template,
IPFIXDataRecord data,
InetSocketAddress ipmb)

Odovzdá prijaté dáta a im zodpovedajúcu šablónu všetkým aktívnym modulom kolektora, ktoré pracujú s IPFIX správami.

Parametre:

template - šablona obsahujúca dáta.

data - dátový záznam.

ipmb - adresa.

public synchronized void **dbExport**(IPFIXTemplateRecord template,
IPFIXDataRecord dataRecord)

Metóda slúži na export všetkých nameraných dát poslaných protokolom IPFIX do databazy a uloženie do hashtable pre ACP. Pri prvom prechode funkciou sa generuje pamäťový záznam o informačných elementoch (ie) z XML súboru. Získajú sa informácie o IE, ktoré sa nachádzajú v šablóne, dekódujú sa ich dátové typy a príslušnosť k skupine pre uloženie hodnôt do databázy.

Parametre:

template - šablóna obsahujúca dáta.

dataRecord - dátový záznam.

public synchronized void **ParseForACP**(IPFIXTemplateRecord template,
IPFIXDataRecord dataRecord)

Metóda slúži na uloženie nameraných hodnôt do hašovacej tabuľky v prípade, že je vypnutý export pre databázu.

Parametre:

template - šablóna obsahujúca dáta.

dataRecord - dátový záznam.

public synchronized void **closeDBConnection**()

Metóda slúži na korektné uzatvorenie spojenia s databázou.

public Hashtable **getData**()
Metóda slúži na prístup k hašovacej tabuľke, ktorá obsahuje názov IE ako kľúč a hodnotu predstavujúcu konkrétnu nameranú hodnotu.
Návratová hodnota:
Vráti hašovaciu tabuľku obsahujúcu dáta.

2.1.2 Trieda ACPIPFIXWorker

Trieda predstavuje vlákno ktoré sprostredkúva komunikáciu a prenos údajov protokolom ACP.

Konštruktor

public **ACPIPFIXWorker**()
Prázdny konštruktor pre vytvorenie vlákna.

public **ACPIPFIXWorker**(ThreadGroup group,
int i)

Konštruktor nastaví meno a id pre tento worker.
Parametre:
group - ThreadGroup ku ktorému worker patrí
i - identifikačné číslo

public void **die**()
Metóda pre korektné zastavenie vlákna.

public static void **processRequest**(Socket request)
Metóda spracuje spojenie z pool.

Parametre:

request - Socket na ktorom prišlo spojenie.

public boolean **ACPauthentication**(Socket connection,
String login,
String passwd)
throws IOException

Metóda, ktorá slúži na overenie autentifikáciu spojenia.

Parametre:

connection - spojenie, cez ktoré sa očakávajú autentifikačné údaje

login - prijatý login

passwd - prijaté heslo

Návratová hodnota:

true - ak došlo ku korektnej autentifikácii.

false - ak analyzér poslal nesprávne autentifikačné údaje.

Výnimky:

java.io.IOException

public void **ACPCommunication**(int messageCode)
throws IOException

Metóda slúži na prijímanie správ od analyzéra a reakcie na nich.

Parametre:

messageCode - kód prijatej správy

Výnimky:

java.io.IOException

private void **makeconnection**(Socket connection)
throws IOException

Metóda, ktorá zabezpečuje fyzickú komunikáciu protokolom ACP.

Parametre:

connection - spojenie medzi kolektorom a príslušným analyzérom

Výnimky:

java.io.IOException

private void **ACPExport**(Inet4Address ipmb,
IPFIXTemplateRecord template,
IPFIXDataRecord data)

Metóda, v ktorej prebieha príprava exportovaných údajov podľa prijatej šablóny

Parametre:

ipmb - IP adresa meracieho bodu

template - IPFIX šablóna

data - IPFIX údaje

private boolean **ACPFilter**(Inet4Address ipmb,
IPFIXTemplateRecord template,
IPFIXDataRecord data)

Metóda, ktorá filtruje exportované údaje podľa zvolených kritérií.

Parametre:

ipmb - IP adresa meracieho bodu

template - IPFIX šablóna

data - IPFIX údaje

Návratová hodnota:

true - ak údaje vyhovujú prijatému filtru, inak false.

public int **stringToInt**(String string)

Metóda na konvertovanie string na int.

Parametre:

string - string, ktorý je určený na konvertovanie

Návratová hodnota:

Výsledná hodnota po konvertovaní.

private String **decodeIpfixType**(String type,
ByteBuffer buffer)
throws UnknownHostException

Dekóduje dátový typ informačného elementu špecifikovaného v IPFIX pre uloženie do databázy.

Parametre:

type - typ informačného elementu podľa IPFIX

buffer - honota informačného elementu uložená v buffry

Návratová hodnota:

Dekódovaná hodnota informačného elementu.

Výnimky:

java.net.UnknownHostException

public void **dispatchIPFIXRecord**(Inet4Address ipmb,
IPFIXTemplateRecord template,
IPFIXDataRecord data)

Metóda sa stará o sprostrekovanie IPFIX úajov na spracovanie.

Parametre:

ipmb - IP adresa meracieho bodu

template - IPFIX šablóna

data - IPFIX údaje

public void **run**()

Hlavný loop pre ACP, kde sa kontroluje spojenie, prebieha príjem riadiacich správ a samotné posielanie dát vyhovujúcich šablóne a zvoleným filtračným kritériám.

Špecifikované:

run in interface java.lang.Runnable

Overrides:

run in class java.lang.Thread

public boolean **isActive**()

Metóda slúži na získanie príznaku spojenia.

Návratová hodnota:

true - ak je spojenie v danom okamihu aktívne.

false - ak nastalo prerušenie spojenia.

public void **setActive**(boolean active)

Metóda slúži na nastavenie príznaku spojenia.

Parametre:

active - nadobúda hodnotu true, ak je spojenie aktívne, v opačnom prípade false.

3 Preklad programu

3.1 Zoznam zdrojových textov

Zdrojové texty sú k dispozícii v prílohe A CD, časť JXColl bakalárskej práce.

Tieto zdrojové texty sú rozdelené do nasledujúcich balíkov:

- balík sk.tuke.cnl.bm:
  - ACPIPFIXTemplate.java
  - DataException.java
  - DataFormatException.java
  - Filter.java
  - InetAddr.java
  - InvalidFilterRuleException.java
  - JXCollException.java
  - Sampling.java
  - SimpleFilter.java
  - TemplateException.java
  - Templates.java
- balík sk.tuke.cnl.bm.JXColl:
  - Config.java
  - IJXConstants.java
  - IpfixDecoder.java
  - IpfixElements.java
  - JXColl.java
  - NetConnect.java
  - PacketCache.java
  - PacketObject.java
  - RecordDispatcher.java
  - Support.java
- balík sk.tuke.cnl.bm.JXColl.export:
  - ACPServer.java
  - ACPIPFIXWorker.java
  - DBExport.java
  - PGClient.java
- balík sk.tuke.cnl.bm.JXColl.accounting:
  - AccountingManager.java
  - AccountingRecord.java
  - AccountingRecordsCache.java
  - AccountingRecordsExporter.java

```
- balík sk.tuke.cnl.bm.JXColl.accounting:
   OWDCache.java
   OWDFieldSpecifier.java
   OWDFlushCacheABThread.java
   OWDListener.java
   OWDObject.java
   OWDTemplateCache.java
   OWDTemplateRecord.java
   Synchronization.java
- balík sk.tuke.cnl.bm.JXColl.IPFIX:
   ExporterKey.java
   FieldSpecifier.java
   IPFIXDataRecord.java
   IPFIXMessage.java
   IPFIXOptionsTemplateRecord.java
   IPFIXSet.java
   IpfixSingleSessionTemplateCache.java
   IpfixUdpTemplateCache.java
   TemplateHolder.java
```

3.2 Požiadavky na technické prostriedky pri preklade

Preklad programu si vyžaduje minimálne uvedenú hardvérovú konfiguráciu:

- CPU Intel Pentium III 1Ghz alebo ekvivalent
- grafická karta novej generácie s minimálne 64MB pamäťou
- sieťová karta 100Mb/s
- pevný disk s 1GB voľného miesta
- operačná pamäť 512MB

3.3 Požiadavky na programové prostriedky pri preklade

- operačný systém GNU/Linux s verziou jadra 2.6 a vyššou (odporúča sa kvôli podpore SCTP)

- Java Runtime Environment (JRE) verzie 1.7.0_03 a vyššej
- balík lksctp-tools
- knižnice dodávané na inštalačnom médiu

3.4 Náväznosť na iné programové produkty

Program umožňuje ukladanie prijatých dát do databázy alebo ich sprístupnenie priamym pripojením, ktoré budú následne vyhodnotené príslušnými prídavnými modulmi. Je implementáciou zhromažďovacieho procesu nástroja BasicMeter. Z toho vyplýva jeho závislosť na merací a exportovací proces - BEEM, alebo iné implementácie.

3.5 Vlastný preklad

Preklad programu spočíva v nakopírovaní zdrojových súborov na disk a spustení kompilátora jazyka Java s potrebnými parametrami a parametrom classpath nastaveným na prídavné knižnice. Odporúča sa použiť java IDE, kde stačí jednoducho nastaviť verziu JDK na 7.0 alebo vyššie a do cesty classpath pridať cesty ku všetkým potrebným knižniciam. Vo vývojovom prostredí Netbeans IDE stačí kliknúť na tlačidlo *Clean and Build*.

3.6 Vytvorenie inštalačného DEB súboru

Vytváranie DEB balíka je možné 2 spôsobmi. Nasledujúci postup predstavuje automatizované vytvorenie. Stačí spustiť skript `buildDeb.sh`, ktorý sa nachádza v priečinku jxcoll/deb.

```
sh buildDeb.sh
```

Výstupom tohto skriptu je súbor s názvom `debian.deb`, ktorý môžme následne premenovať podľa verzie JXColl (napríklad na jxcoll_3.9_i386.deb). Tento skript vykonáva nasledovné operácie:

1. v prípade, ak neexistuje priečinok debian, extrahuje ho z dodávaného archívu debian.tar.gz, inak tento krok preskočí
2. skopíruje binárny súbor aplikácie z projektu do DEB balíčka (predpokladá sa, že bol program kompilovaný v Netbeans IDE pomocou Clean and Build tlačidla)
3. skopíruje konfiguračný súbor aplikácie z projektu do DEB balíčka
4. skopíruje IPFIX definičný súbor aplikácie z projektu do DEB balíčka
5. vymaže prípadné dočasné súbory nachádzajúce sa v DEB balíčku
6. vygeneruje MD5 kontrolné súčty pre všetky súbory DEB balíčka
7. zabezpečí maximálnu kompresiu manuálových stránok a changelog súborov
8. skopíruje binárny súbor z projektu aplikácie do DEB balíčka a nastaví mu práva na vykonávanie
9. vytvorí samotný DEB balíček
10. overí ho pomocou programu lintian - ten vypíše prípadne varovania a/alebo chyby ktoré je následne potrebné manuálne odstrániť
11. archivuje vytvorený DEB balíček do archívu debian.tar.gz

Pred spustením skriptu je nutné skompilovať JXColl pomocou Netbeans IDE tlačidlom *Clean and Build*. Prípadné zmeny control alebo changelog súboru, manuálových stránok je nutné vykonať ručne. Manuálové stránky je vhodné upraviť pomocou programu *GmanEdit*. Po spustení skriptu je automaticky vytvorený DEB balíček s názvom `debian.deb`. Ten je vhodné premenovať podľa aktuálnej verzie pre zachovanie prehľadnosti. Vytvorí sa aj archív debian.tar.gz, ktorý obsahuje najak-

tuálnejšiu adresárovú štruktúru DEB balíčka pre budúce využitie (ak neexistuje priečinok debian, vytvorí sa extrakciou z tohto archívu). Ak je potrebné len aktualizovať kód, stačí spustiť skript a ten sa o všetky potrebné náležitosti postará, pričom vytvorí aj adresár debian. Súbory je v ňom možno upravovať až kým nie je všetko podľa predstáv. Ak je všetko hotové, v Netbeans IDE je potrebné vymazať priečinok debian (vykoná sa SVN DELETE, namiesto obyčajného odstránenia zo súborového systému) a projekt "commitnúť".

3.7 Opis známych chýb

V súčasnosti nie sú známe žiadne vážne chyby.

4 Zoznam použitej literatúry

[1] Koščo, M.: Opis sieťových protokolov prostredníctvom jazyka XML, 2005, Diplomová práca, KPI FEI TU, Košice

[2] Kaščák, M.: Príspevok k problematike aplikačného využitia meraní prevádzkových parametrov počítačových sietí, 2007, Diplomová práce, KPI FEI TU, Košice

[3] Pekár, A.: Meranie prevádzkových parametrov siete v reálnom čase, 2009, Bakalárska práca, KPI FEI TU, Košice

[4] Vereščák, T.: Zhromažďovací proces nástroja BasicMeter, 2010, Bakalárska práca, KPI FEI TU, Košice

[5] Pekár, A.: Optimalizácia zhromažďovacieho procesu nástroja BasicMeter, 2011, Diplomová práca, KPI FEI TU, Košice

[6] Vereščák, T.: Optimalizácia zhromažďovacieho procesu nástroja BasicMeter, 2012, Diplomová práca, KPI FEI TU, Košice