Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Flexibilní logování pro embedded Linuxové systémy

David Vavřička

Vedoucí práce: Ing. Matěj Laitl

3. března 2016

Poděkování

Doplňte, máte-li komu a za co děkovat. V opačném případě úplně odstraňte tento příkaz.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 3. března 2016 .....

Odkaz na tuto práci

Vavřička, David. *Flexibilní logování pro embedded Linuxové systémy.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Doplňte

Klíčová slova logování, vestavěné systémy, logovací démoni, Linux, Rsyslog

Abstract

Sem doplňte ekvivalent abstraktu Vaší práce v angličtině.

Keywords logging, embedded systems, logging daemons, Linux, Rsyslog

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Kapitola 1

Technické požadavky

Cílem je upravit logovací řešení pro set-top box EKT DID7006mTF [1] tak, aby splňovalo technické požadavky popsané v této kapitole. Řešení musí fungovat a být otestováno na zmíněném modelu set-top boxu a pokud možno by mělo být přenositelné i na jiné typy set-top boxů. Požadavky jsou rozděleny na základní a rozšířené. Rozšířené požadavky není nutno implementovat.

1.1 Základní technické požadavky

1.1.1 Snížení objemu logů

Je žádoucí umožnit snížit objem zasílaných logů z důvodu přílišného zatížení sítě a serverových disků. Původní řešení všechny logy odesílalo na vzdálené servery. Nové řešení by mělo odesílat pouze důležité zprávy, tedy zprávy s nízkou severitou.

1.1.2 Vzdálená konfigurace

Technické řešení musí být schopno za běhu pomocí SHELL-ového API měnit minimální povolenou severitu zpráv pro jednotlivé komponenty a dále toto API musí mít výchozí severitu, která se použije pro komponenty ji nemají explicitně nastavenou. Takto změněné nastavení musí být perzistentní i po restartu STB. Výchozí nastavení se obnoví až po factory resetu. API navrhne dle své libovůle sám řešitel.

1.1.3 Rate-limiting odesílaných zpráv

Nové logovací řešení musí být schopné provádět rate-limiting odesílanych zpráv tak, aby nepřekročilo maximální vyhrazenou šířku pásma. Bude umožněno nastavit jak dlouhodobé tak krátkodobé limity. Naivní rate-limiting je i v existujícím řešení, řešitel navrhne výchozí nastavení nového řešení tak, aby přibližně odpovídalo současnému chování.

1.1.4 Formát logů

Je nutno zachovat formát logů jako ho má původní řešení, aby se jednalo o drop-in replacement bez nutnosti jakkoli měnit konfiguraci serveru, který sbírá logy od set-top boxů.

1.1.5 Razítkování zpráv

Každé zprávě se musí přidat textový prefix if=N, kde N monotonicky roste s každou zprávou. To slouží pro detekci ztracených zpráv. Id přeteče po 32 nebo 64 bitech, to záleží na rozhodnutí řešitele. Po rebootu STB id znovu začíná od 1.

1.1.6 Post-processing zpráv

Zadavatel má pouze částečnou kontrolu nad zprávami generovanými aplikacemi na set-top boxu, například nedokáže ve všech případech eliminovat dlouhé prefixy u zpráv. Je proto nutno takové prefixy rozpoznat a vhodně odfiltrovat před odesláním. Ze stejného důvodu mají některé zprávy nevhodně vyplněnou severitu a položku app-name. Jejich správné hodnoty jsou uloženy v textu zprávy, jejíž formát je konstantní. Řešení bude schopné tyto údaje z těla zprávy extrahovat a nahradit jimi původní metadata. Tato pravidlo musí být možné definovat a měnit bez nutnosti nového sestavení softwaru. Řešitel vytvoří pro ukázku 2 pravidla, která budou sloužit zadavateli jako šablony pro možná budoucí filtrovací pravidla.

1.1.6.1 Pravidlo pro filtrování zpráv pro dané komponenty

Zprávy s nastavenou severitou INFO a komponentou sld_br je třeba změnit podle následujícího vzoru.

Originální zpráva:

```
2016-02-18T14:05:24+01:00 cc-b8-f1-00-6f-07 sld_br: id=559
:[stbhal.cpp:debug:520]: INFO: [94mDEBUG: InformationService:
 Reading 'nangu.video.forcedScart': false[0m
```

Upravená zpráva:

```
2016-02-18T14:05:24+01:00 cc-b8-f1-00-6f-07 nangu-portal: [94m InformationService:
 Reading 'nangu.video.forcedScart': false[0m
```

1.1.6.2 TODO další pravidla

doplnit v průběhu implementace pravidel

1.1.6.3 Převod severit

Aplikace na STB používají pro logování TODO-DOPLNIT formát logů. Problémem je, že tento formát není kompatibilní s dnes za standard považovaným syslog formátem. Zadavatel proto požaduje změnit severity zpráv podle následující tabulky.

Tabulka 1.1: Převodní tabulka

Portal	Syslog
ERROR: 1	ERR
WARN: 1	WARN
INFO: 1	NOTICE
DEBUG: 1	INFO
TRACE: 1	DEBUG

1.2 Rozšířené technické požadavky

1.2.1 Komprese zpráv

Bylo by vhodné zvážit pro a proti komprese zpráv. Vyplatí se ušetřená přenesená data oproti režiji spojené s kompresí a dekompresí zpráv?

1.2.2 Rozšíření C++ komponenty

Zadavatel na STB provozuje malého démona dmd napsaného v C++, který mimo jiné obsahuje minimalistický HTTP webserver. Dále v browseru běží Javascript aplikace (nangu.TV portál), která pomocí messagingu komunikuje s centrálním serverem. Tato Javascript aplikace ovšem nemůže přímo používat Shell API. Požadavkem je rozšířit C++ komponentu dmd tak, aby umožnila Javacript aplikaci řídit konfiguraci logování (viz bod Vzdálená konfigurace).

Kapitola 2

Analýza

2.1 Set-top box O2 TV - EKT DID7006mTF

Hardware specifikace

Nainstalovaný software

Gu

PKGBUILD

2.2 Současné řešení

2.3 Volba postupu řešení

Prvně je nutno zvážit, zda problém řešit na straně serveru nebo set-top boxu. Vhodnou konfigurací logovacího démona na straně serveru, který by nepotřebné zprávy zavčas rozpoznal, zahodil a dále nezpracovával bychom splnili požadavek na snížení zátěže serverových disků. Přetížení sítě se takto vyřešit ale nedá a proto toto řešení zavrhuji. Je tedy nutno problém řešit na straně set-top boxu kde původní řešení je postaveno na busy-box syslogd. Nabízí se možnost upravit fungování tím způsobem, aby se logy s nízkou severitou už na set-top boxu zahazovaly a pouze v případě potřeby bylo umožněné na dálku změnit konfiguraci démona tak, aby se povolilo logování pro logy s nastavenou danou komponentou a severitou. To vše přes SHELL-ové API. Součástí zadání je ale i implementovat škrcení zpráv, aby nedocházelo k zahlcení linky. Takovou možnost prostý syslogd neposkytuje a je proto nutno zvážit napsání vlastního démona či nasazení jiného, vyspělejšího logovacího démona.

2.4 Srovnání logovacích démonů

Démon v UNIXovém světě je označení pro takový proces, který oproti běžným procesům neintereaguje přímo s uživatelem, ale běží na pozadí operačního systému a funguje samostatně. Účelem logovacího démona je sběr logů od ostatních procesů, které následně v závislosti na jeho konfiguraci dokáže filtrovat a ukládat na disk či odesílat na požadovaný vzdálený server.

V této kapitole zmíním a popíši vybrané logovací démony a v závěru kapitoly je porovnám.

BusyBox Syslogd

Tato logovací utilita se skládá ze dvou démonu, jmenovitě z Klogd, který má na starost logy linuxového kernelu, druhým démonem je pak syslogd, který spravuje všechny zbylé logy. Oba tyto démoni mají velice omezenou funkcionalitu. Dokáží logy lokálně ukládat, přeposílat je dále po síti, zahazovat duplikáty, rotovat logy v závislosti na velikosti a tím výčet jejich funkcionalit končí.

Syslog-ng

Flexibilní logovací démon zaměřený na centralizované a zabezpečené logování. Má široké možnosti nastavení a poskytuje obrovské množství funkcionalit. Takže jeho vhodným nakonfigurováním se dají snadno splnit všechny vytyčené technické požadavky až na požadavek pro možnost vzdálené změny konfigurace. Je nutno ale zmínit, že pokročilé funkce jako například šifrování zpráv, bufferování nebo message-rate kontrola jsou dostupné pouze v komerční closed-source verzi.

Rsyslog

Výčet funkcionalit Rsyslogu je ještě obsáhlejší než u Syslog-ng. Technické požadavky se s jeho použitím tedy také dají splnit všechny, kromě vzdálené změny konfigurace. Oproti Syslog-ng je Rsyslog kompletně zdarma a open-source. Navíc není jen logovacím démonem, ale i analyzérem logů. Dokáže logy podle obsahu zprávy měnit, třídit a jinak s nimy nakládat. Že je Rsyslog vyspělý a kvalitní program dokazuje fakt, že je defaultním logovacím démonem na spoustě linuxových distribucích, jmenovitě například v Ubuntu. Jeho slabiny shledávám v nedostatečné dokumentaci a ve specifických případech v neefektivním analyzování logů mající za následek (obvzláště na embedded zařízení s pomalým ARM procesorem) rychlostní deficit. Jeho vývoj obstarává z velké většiny pouze jeden člověk, jeho původní tvůrce Rainer Gerhards. A v jednom člověku není snadné dovést tak rozsáhlý projekt k dokonalosti.

Porovnání výše zmíněných logovacích utilit

Pouhým nasazením jakéhokoli známého logovacího démonu není možné splnit všechny vytyčené technické požadavky. V případě ponechání původního BusyBox syslogd démonu by pro splnění technických požadavků bylo nutno doimplementovat tolik funkcionalit, že by to výrazně přesahovalo rozsah bakalářské práce. Výhodněji se jeví nasadit pokročilý logovací démon jako je Syslog-ng či Rsyslog. Oba totiž poskytují námi požadované funkcionality. Syslog-ng však většinu z nich poskytuje pouze v placené closed-source verzi a proto jsem se rozhodl pro Rsyslog.

2.5 Vzdálená konfigurace

Rsyslog při svém zapnutí čte konfigurační soubor rsyslog.conf, který za jeho běhu není možné měnit. Je pro to nutné napsat SHELL-ové API, které umožní na dálku přenastavit tento konfigurační soubor a restartovat rsyslog

Shellové API - 1. způsob

```
set_log_verbosity.sh [component] [severity]
```

TODO rozepsat

Shellové API - 2. způsob

```
/etc/logging.conf

component1  = DEBUG
componentXY = INFO
...
DEFAULT     = INFO
```

TODO rozepsat

Kapitola 3

Realizace

3.1 Nasazení Rsyslogu

build

konfigurace

3.2 Shell API

Kapitola 4

Testování

Závěr

Literatura

[1] *EKT DID7006*. Dostupné z: `http://exploredoc.com/doc/3174828/model-did7006-high-definition-ott-stb`

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

XML Extensible markup language

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
  - `BP_Vavricka_David_2016.pdf` ..... text práce ve formátu PDF