一种多架构应用软件开发及运行方法研究与实现

鲁 振,胡坚升,李名扬
(中软信息系统工程有限公司,北京 102209)

摘 要: 对国产基础软件运行环境及运行依赖库进行了研究,提出了一种多架构应用软件开发及运行方法,并在 FT1500A、X86(CPU)与麒麟(OS)相组合的两类架构环境上实现了该方法。运行结果表明,该方法能够提供良好的多架构应用软件开发及运行服务,为国产基础软硬件生态的软硬件适配、整体解决方案提供技术支撑。

关键词: 多 CPU 架构;应用软件开发;生态体系;核心运行框架;沙箱
中图分类号: TP311 **文献标识码:** A **DOI:** 10.19358/j.issn.2096-5133.2020.09.006
引用格式: 鲁振,胡坚升,李名扬. 一种多架构应用软件开发及运行方法研究与实现[J].信息技术与网络安全,2020,39(9):28-33.

Research and implementation of a multi-architecture application software development and operation method

Lu Zhen, Hu Jiansheng, Li Mingyang
(CS&S Information System Engineering Co., Ltd., Beijing 102209, China)

Abstract: This paper studied the running environment and running dependency library based on the domestic basic software ecosystem, and proposed a multi architecture application software development and a running method. The running method has been implemented on FT1500A and X86(CPU)+Kylin(OS) architecture environments. The running results show that the method can provide good services for multi architecture software development and operation, and can provide technical support for the software and hardware adaptation and solid foundation for domestic basic software and hardware ecology.

Key words: multi-CPU architecture; application software development; ecosystem; core operating framework; sandbox

1 背景介绍

1.1 现状分析

近年来,以自主 CPU+OS 为核心的国产基础软硬件生态体系不断发展和完善,但仍存在一些短板和弱项,比如基础软件方面,操作系统多是基于开源的 Linux 系统,经常会导致软件开发与运行存在运行依赖库不规范、应用软件版本混乱、冲突等问题。基于系统开发和运行的实践,当前国产基础软件生态体系主要存在以下四个方面的问题。

(1)开发严重碎片化。Linux 上存在太多的开发库,国产操作系统缺少一套类似微软.NET 框架的统一开发解决方案,开发者难以选择最佳的开发语言、开发库和开发环境,比如对 C/C++而言,主流的用户界面(UI)程序开发有 Gtk、Qt、WxWidget 三种,同时 Gtk 本身又有 Gtk2 和 Gtk3 系列[1],Qt 有 Qt4 和 Qt5 区分。Linux 应用软件开发者需要一套开发和运行的行业标准,解决开发框架选择难、开发文档少或无、开发库版本多、开发接口不统一等问题。

(2)权限控制机制存在安全隐患。主流的国产操作系统使用基于用户角色的权限控制机制,应用一般具备诸如访问用户文件[2]、访问其他应用数据[3]、使用网络和外部设备[4]等权限。虽然大多数的 Linux 发行版(典型的如 Debian、Ubuntu、Redhat、Centos)都提供自身的软件包维护机制,用户往往也使用值得信赖的源下载应用,但病毒往往也会利用应用程序这条路径植入传播。用户对一些程序的非法操作往往会导致严重的安全问题,给自身带来极大的困扰。例如用户使用的应用程序需要访问网络时,有

可能会从不安全的站点下载恶意程序，执行一些非法操作，如盗取用户的敏感信息，干扰用户的日常工作、数据安全和个人隐私等，用户需要一种有效的保护计算机安全的方法，比如利用沙箱技术给应用程序提供隔离的运行空间。

(3)跨架构应用移植不统一。应用软件可移植性越来越受到关注，诸如 Java、Python 等语言都提供一套虚拟机用于屏蔽底层处理器和操作系统差异[5]，但是对于 C/C++等平台相关编程语言，目前缺少一种跨架构的可移植开发运行解决方案[6]。

(4)应用软件版本混乱。主流的国产操作系统发行版都使用类似 RPM、DPKG 等打包系统构建，最大的特点是上游开发者和下游软件包维护者(打包者)明显地区分开。上游应用开发者编写代码，下游发行者获取并将其转化(编译、编写规则并重新打包)为 RPM 或 DEB 包；最后安装到本地系统中。这种场景在一定程度上解决了包的风险问题，因为软件包维护者往往会选择值得信赖和功能可靠的应用，但是也难以避免地暴露一些问题，上游应用开发者往往希望更高的发布速度，而事实上完全依赖下游发行者打包开发的应用，下游发行者决定具体的调度、申明、打包、提供支持等规则。应用本身的实际测试变得十分困难，因为最终用户往往可能使用不同的包版本，应用在某个发行版的某个版本下的测试，无法确定应用在其他发行版和其他版本的任意组合下都能正常运行；要测试应用在某个发行版的某个版本下的运行，开发者往往需要安装该发行版的版本环境，并编译运行该应用，这将是一项繁杂的工作。

1.2 研究目标

针对国产基础软硬件生态环境下应用软件开发与运行存在的问题，设计多架构应用软件开发及运行库服务系统，目标主要有以下三点：

(1)提供跨平台多架构的统一运行库，屏蔽底层软硬件差异，使应用更易于分发到不同的平台架构，该运行库覆盖最基本的 C/C++标准库、Python 虚拟机、Java 虚拟机等应用基本运行环境。

(2)提供统一和容易使用的开发接口，解决 Linux 上开发库的碎片化问题。

(3)提供应用运行沙箱，沙箱提供了文件系统隔离、系统资源隔离、物理资源隔离、权能限制和强制访问控制等策略，尽可能地防止上层应用直接访问底层主机，减少应用程序对主机造成的影响，为用户提供一个独立和安全的运行环境。

根据以上目标，应用软件开发及运行库服务整体框架如图 1 所示。

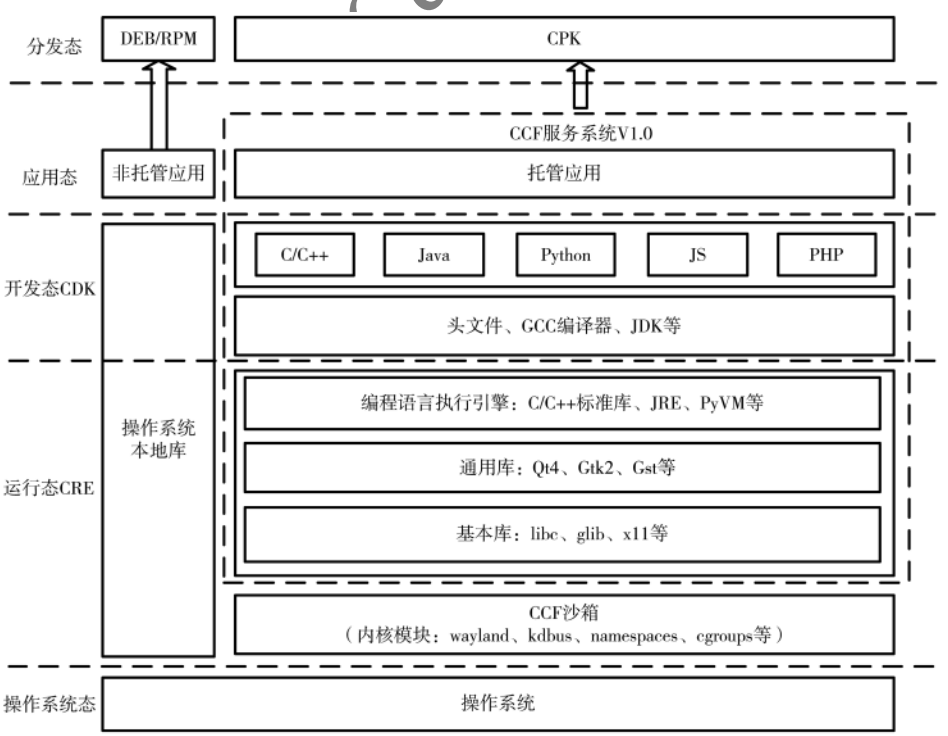

图 1 多架构应用软件开发及运行库服务系统整体设计

2 系统设计

2.1 架构设计

多架构应用软件开发及运行库服务系统(CCF)自底向上分为操作系统态、运行态(CRE)、开发态(CDK)、应用态和分发态。其中,操作系统层包括当前所有主流的 Linux 发行版和底层处理器架构,多架构应用软件开发及运行库服务系统可支持所有主流 Linux 发行版和绝大部分处理器架构。CCF 可屏蔽操作系统和处理器架构差异,基于 CCF 开发和运行的应用软件仅依赖于相应类型的 CCF,与操作系统差异无关。使用操作系统类库,并直接运行于主机系统的应用软件属于非托管应用,这些应用一般与平台类型相关。运行层包括沙箱、核心基础库(如 libc、glib、xlib 等库)、编程语言通用库(分为用户界面、多媒体、进程通信、数据库操作等 20 多种类型)、编程语言执行引擎(如 C++标准库 libstdc++、Java 虚拟机、Python 虚拟机、PHP 执行引擎、JavaScript 解析引擎等)。开发层实际上是运行态加上开发应用必须的头文件、编译器、jar 包、python 模块等文件组成的。使用 CCF 开发并依赖 CCF 运行的应用是“托管”应用,并以“CPK”的方式分发。

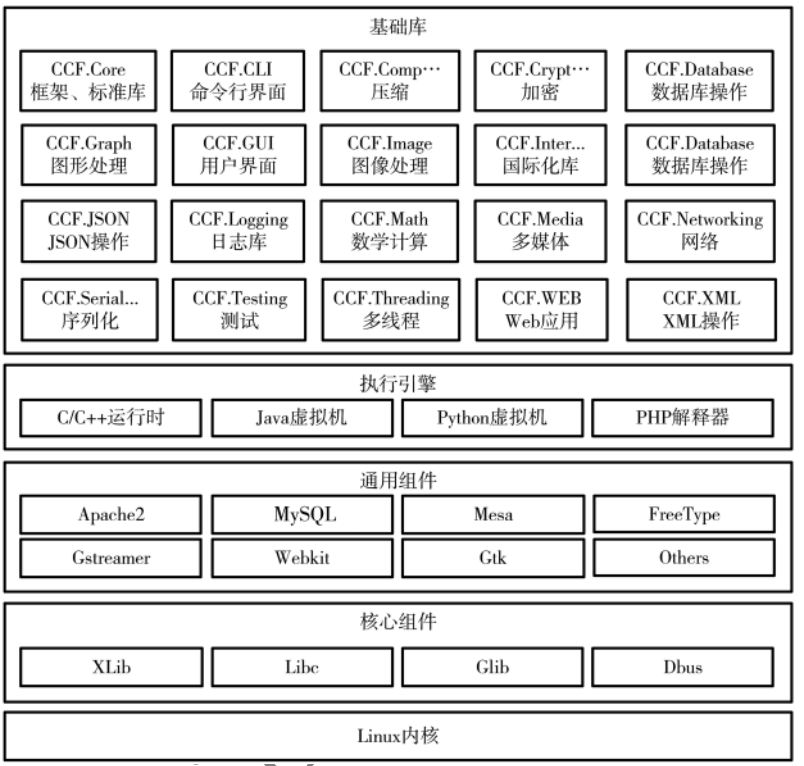

图 2 多架构应用软件运行时框架

2.1.1 多架构应用软件开发及运行核心框架运行时(CRE)

多架构应用软件运行时提供了基于良好定义且高度优化的环境,以支撑应用软件的运行。这里指的运行时实际上是一系列不同架构、不同版本的运行时。运行时包括 C/C++、Python、Java 等高级语言基本运行环境,具体表现为可执行文件、动态链接库、资源文件、配置文件、脚本等。运行时本身又分为基本运行时和开发运行时,前者是支撑一个应用运行的最小环境,后者则在包含前者的基础上,添加编译和调试应用所需的基本环境。运行时的整体设计如图 2 所示。

2.1.2 多架构应用软件开发及运行核心框架开发工具(CDK)

多架构应用软件开发工具由开发运行时、工具链、基础类库、公共接口、应用框架组成,支持 C/C++、Java、PHP、Python 等多种编程语言,提供统一的公共编程接口和通用基础类库。利用工具链编译、虚拟机运行等方式实现应用程序在沙箱中的最终运行。图 3 展示了开发框架的各个层次。

2.1.3 多架构应用软件运行沙箱(CCF Sandbox)

多架构应用软件运行沙箱是一套整合了内核 cgroups、namespaces、selinux、kdbus、systemd 和 wayland 显示服务器的应用运行沙箱机制。基本的原则是,应用以普通用户身份执行,只具备最低的访问权限。更多的权限需要通过权限定义和提权。与一般的沙箱不同,多架构应用软件运行沙箱采用的是“资源是否可见”原则,而非“资源访问控制”。与限制应用访问操作系统资源权限不同,这里的沙箱默认应用软件对操作系统资源不可见。当沙箱初始化完成后,首先建立文件系统,同时,某些白名单内的文件和路径(一般是最基本的资源)随之被挂载到命名空间(namespace),只有那些经过筛选、审查且被认为运行足够稳定的文件或目录才会被沙箱中的应用软件访问。图 4 展示了沙箱和应用之间的关联。

2.2 流程设计

2.2.1 多架构应用软件开发流程

多架构应用软件开发及运行库服务系统主要

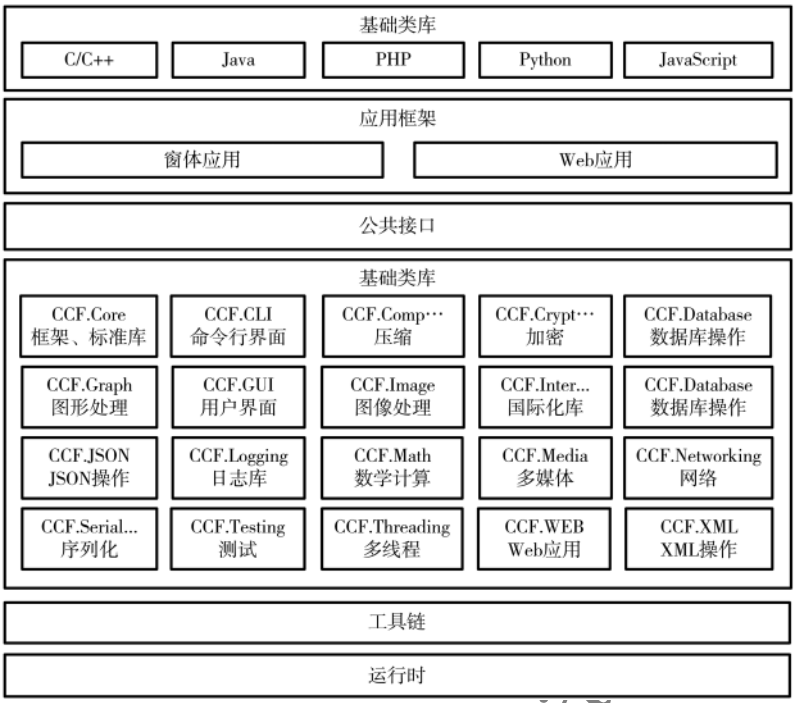

图 3 多架构应用软件核心开发框架

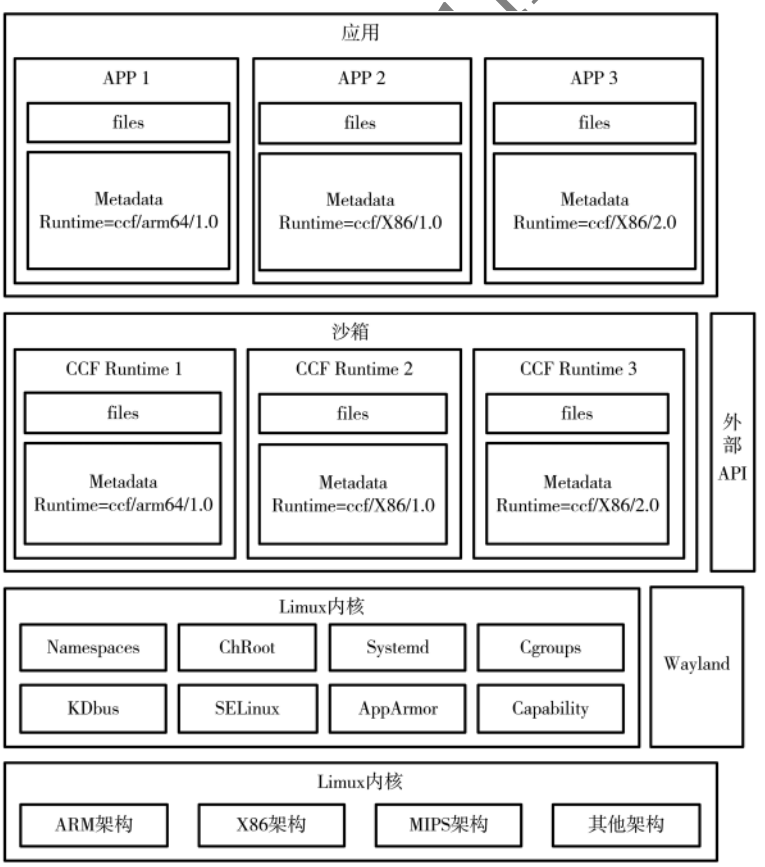

图 4 安全沙箱设计框架

由托管式应用的开发和运行两个过程组成。

多架构应用软件开发流程如图 5 所示。

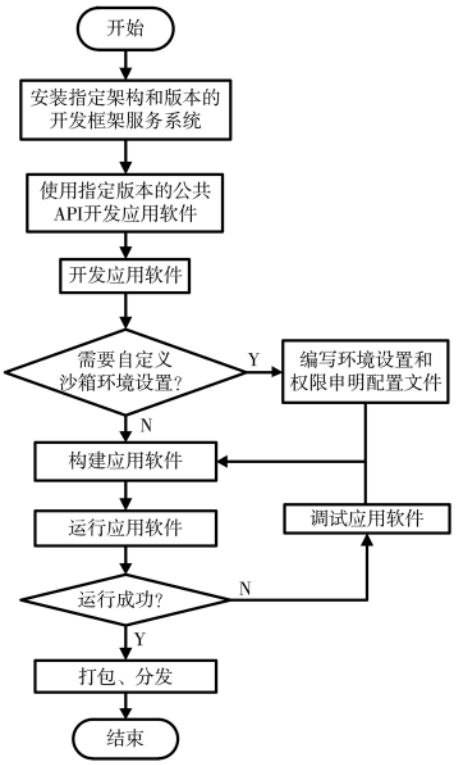

图 5 多架构应用软件开发流程图

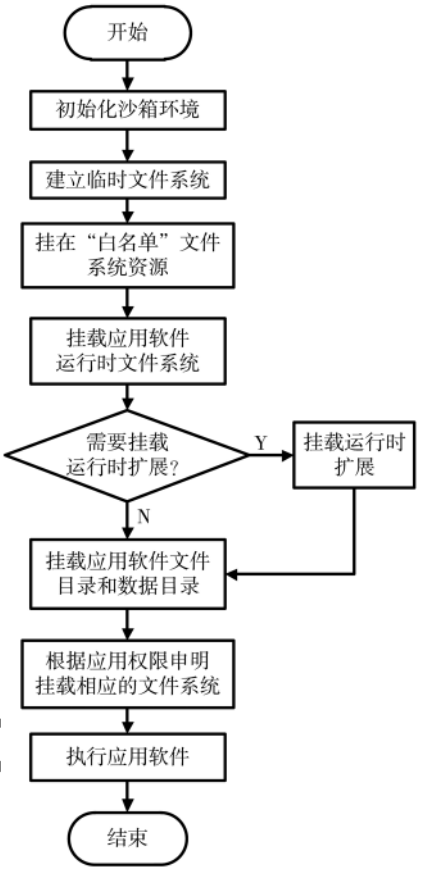

图 6 多架构应用软件运行流程图

2.2.2 多架构应用软件运行流程

多架构应用软件运行流程如图 6 所示。

2.2.3 系统实现及验证

本文以 FT1500A(ARM64)、x86 两类CPU 架构下的 gedit 应用软件为例,验证多架构应用软件开发及运行方法从开发到运行的实现流程。

在构建一个 gedit 之前,先安装应用软件对应的运行时(CRE)、开发包(CDK)和沙箱工具。

首先创建目录并初始化目录,选择对应的运行时和开发包:

```
$ mkdir my-app
$ ccf build-init ./my-app gedit ccf/x86/1.0 ccf/x86/1.0
```

得到一个“metadata”文件(该文件用于定义一些环境配置和权限申明)和空的“my-app/files/”和“my-app/exports/”目录。然后使用“prefix=/app”构建应用软件:

```
$ ccf build ./my-app ./configure --prefix=/app
$ ccf build ./my-app make && make install
```

“ccf”将会选择对应开发包构建应用程序,构建完成的应用将安装到“my-app/files”目录。

接着需要导出一些资源文件(如 desktop 文件和图标文件)并打包应用软件:

```
$ ccf build-finish ./my-app
```

最后将打包生成的应用转化为 deb、rpm 或者 cpk 格式。并分发到对应架构的 Linux 平台。该平台在安装对应版本(或较高版本)的基础上运行该应用:

```
$ ccf run gedit
```

3 结论

本文对国产 CPU+OS 基础环境应用软件及运行依赖库进行了研究,提出了一种多架构应用软件开发及运行方法,设计了由多架构应用软件开发及运行核心框架运行时(CRE)、多架构应用软件开发及运行核心框架开发工具(CDK)、多架构应用软件运行沙箱(CCF Sandbox)组成的多架构应用软件开发及运行库服务系统(CCF),探索了多架构应用软件开发和运行流程,提供覆盖支持多种高级语言开发、统一开发类库及 API、托管应用运行、多架构可移植方案、沙箱安全机制等多个方面的服务。经过在 FT1500A、X86(CPU)+麒麟(OS)两类架构环境上验证,该方法能够为国产基础软硬件生态体系提供

良好的多架构应用软件开发及运行服务,为国产基础软硬件生态的软硬件适配、整体解决方案提供技术支撑,可以作为国产基础软硬件生态体系应用软件开发及运行的标准。

参考文献

[1] 孙波.跨平台工具软件在控制软件开发中的应用[D].上海交通大学,2009.
[2] 傅晓,王志坚,杨家奇,等.一种基于时间期限和访问次数控制的文件生命周期控制方法[J].计算机应用与软件,2016,33(11):305-309.
[3] 徐协鹏.Web 应用程序数据保护系统[D].复旦大学.2012.
[4] 宋尚.基于计算机网络技术与应用的探讨[J].电子技术与软件工程.2014(3):190.
[5] 肖铭轩.基于元数据的软件框架研究及优化[D].西南交通大学,2017.
[6] 邓广宏,蔡斌,池志强,等.一种实现应用软件跨平台特性的解决方案[J].计算机与数字工程,2008(8):157-161.

(收稿日期:2020-07-14)

作者简介:

鲁振(1983-),通信作者,男,高级工程师,主要研究方向为:计算机应用。E-mail:pzluzhen@163.com。

胡坚升(1990-),男,助理工程师,主要研究方向:计算机应用。

李名扬(1993-),男,助理工程师,主要研究方向:电子信息工程。

(上接第 27 页)

[2] 神州数码云服务平台 SM@RTPAAS 产品白皮书[Z].神州数码信息系统有限公司,2013.
[3] 卢金晨.基于 PaaS 的电子政务服务平台框架研究[D].杭州:浙江工业大学,2015.
[4] 赵小肖.PaaS 模式下私有云政务架构设计与实现[D].曲阜:曲阜师范大学,2013.
[5] 赵恒.面向企业的创新私有云平台的搭建[D].天津:河北工业大学,2014.
[6] 李辉.基于 OpenStack 的私有云计算平台的研究和实现[D].南昌:江西师范大学,2013.
[7] 赵芳,薛元元.一种基于云计算的办公自动化系统解决方案[J].中国科技信息,2018(3):36-37.
[8] 尹霞,周明昱.云计算在省级电子政务系统建设的思路浅析[J].现代电信科技,2012(10):73-78.
[9] 刘瑶.PaaS 云平台技术研究与应用[D].西安:长安大学,2015.
[10] 赵芳.PaaS 平台持续集成自动化测试框架的设计与实现[D].北京:中国科学院大学,2017.
[11] 尹旭刚.基于 Docker 的 PaaS 平台技术研究与实现[D].北京:北京邮电大学,2017.
[12] 袁忠良.容器云计算平台关键技术研究[D].南京:南京大学,2017.

(收稿日期:2020-07-14)

作者简介:

雷阳(1983-),男,硕士,主要研究方向:网络信息系统。

周军(1973-),,硕士,高级工程师,主要研究方向:网络信息系统。

陆平(1980-),男,硕士,主要研究方向:网络信息系统。

版权声明