应用移植

版权所有© **2019** 华为技术有限公司

前言

本章将介绍了华为鲲鹏平台应用移植的相关知识。包括程序运行原理和软件迁移至鲲鹏计算平台的整个实施过程。并从服务器和容器两种应用载体出发，介绍了**Kunpeng Porting Advisor**迁移工具的使用和容器迁移操作步骤。

版权所有© **2019** 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

本节概述和学习目标

- 本节将从软件迁移原理开始介绍，以了解软件迁移的背景与必要性。并通过软件迁移过程介绍了解迁移的方法以及流程。
- 学完本节后，您将能够：
  - 了解程序运行原理
  - 了解软件迁移至鲲鹏计算平台的过程

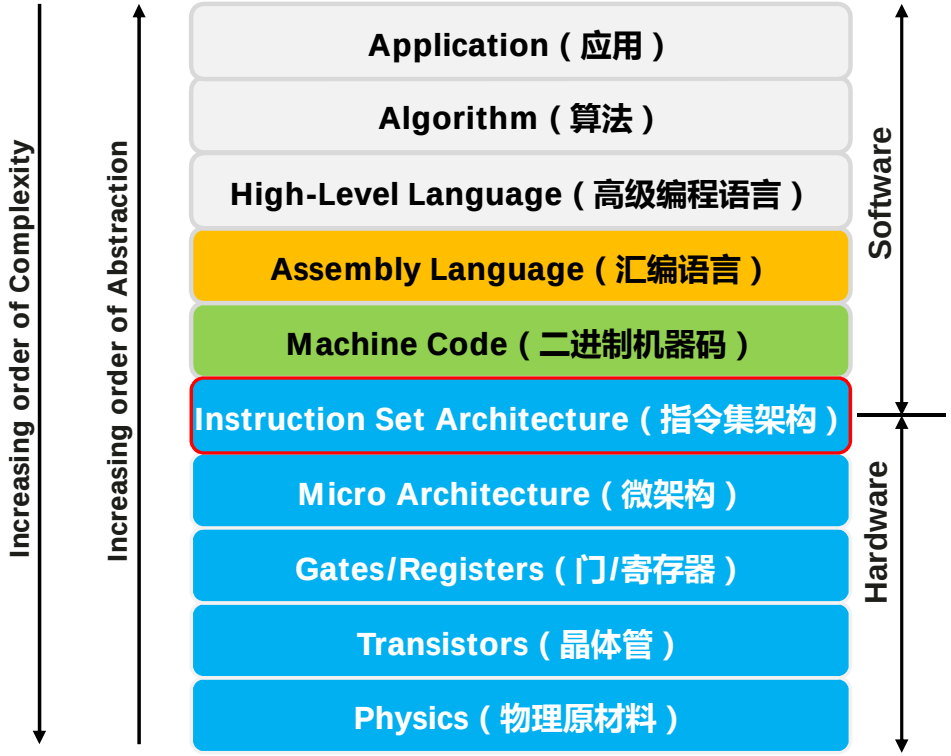
版权所有© 2019 华为技术有限公司

目录

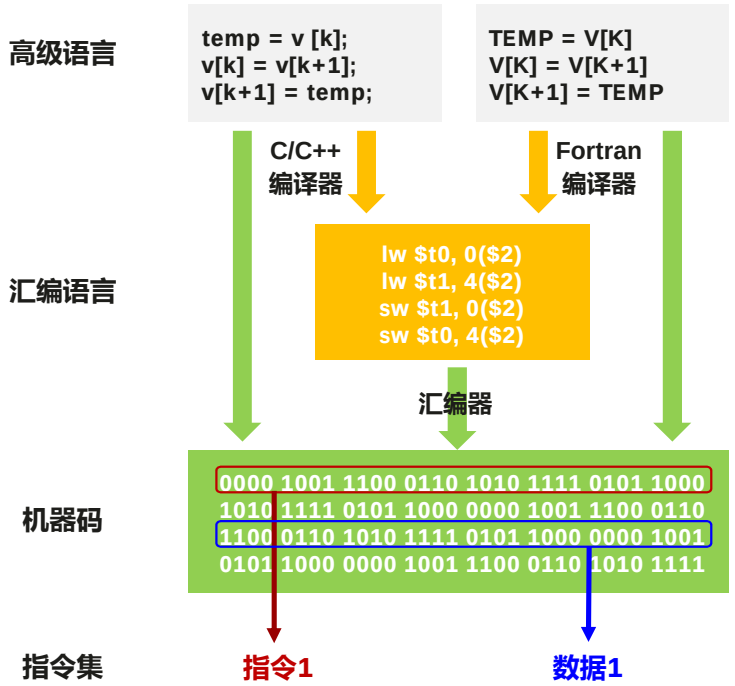
版权所有© 2019 华为技术有限公司

计算技术栈与程序执行过程

计算技术栈



程序执行过程



版权所有© 2019 华为技术有限公司

鲲鹏处理器与x86处理器的指令差异

程序代码（C/C++）：

```
int main()
{
    int a = 1;
    int b = 2;
    int c = 0;
    c = a + b;
    return c;
}
```

编译

鲲鹏处理器指令

指令	汇编代码	说明
b9400fe1	ldr x1, [sp,#12]	从内存将变量a的值放入寄存器x1
b9400be0	ldr x0, [sp,#8]	从内存将变量b的值放入寄存器x0
0b000020	add x0, x1, x0	将x1(a)中的值加上x0(b)的值放入x0寄存器
b90007e0	str x0, [sp,#4]	将x0寄存器的值存入内存（变量c）

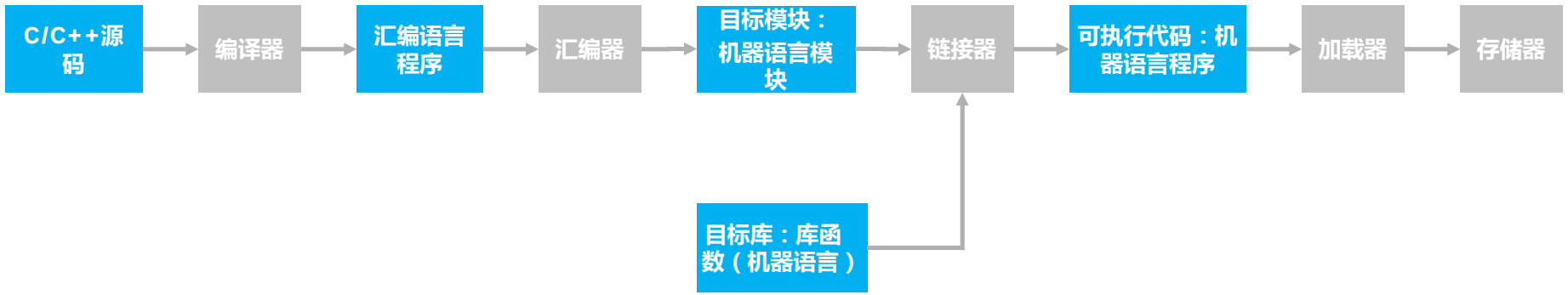
x86处理器指令

指令	汇编代码	说明
8b 55 fc	mov -0x4(%rbp),%edx	从内存将变量a的值放入寄存器edx
8b 45 f8	mov -0x8(%rbp),%eax	从内存将变量b的值放入寄存器eax
01 d0	add %edx,%eax	将edx(a)中的值加上eax(b)的值放入eax寄存器
89 45 f4	mov %eax,-0xc(%rbp)	将eax寄存器的值存入内存（变量c）

版权所有© 2019 华为技术有限公司

从源码到可执行程序 - 编译型语言

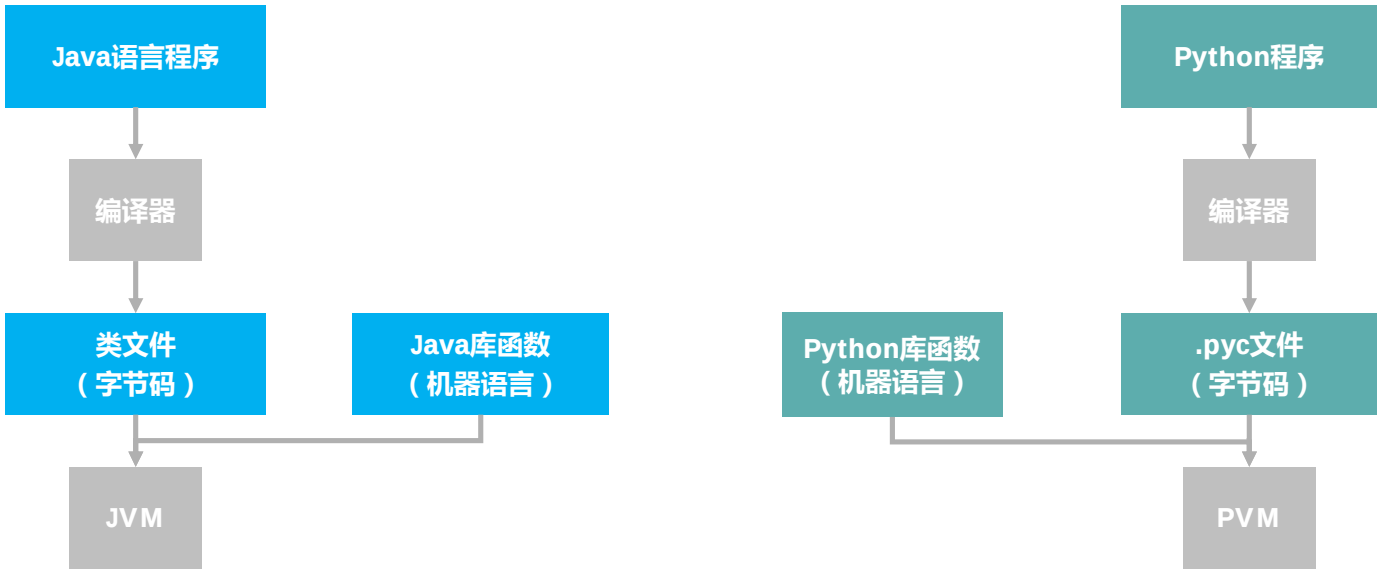
- **编译型语言**：典型的如**C/C++/Go**语言，都属于编译型语言。编译型语言开发的程序在从**x86**处理器迁移到鲲鹏处理器时，必须经过重新编译才能运行。
- 从源码到程序的过程：源码需要由编译器、汇编器翻译成机器指令，再通过链接器链接库函数生成机器语言程序。机器语言必须与**CPU**的指令集匹配，在运行时通过加载器加载到内存，由**CPU**执行指令。



版权所有© 2019 华为技术有限公司

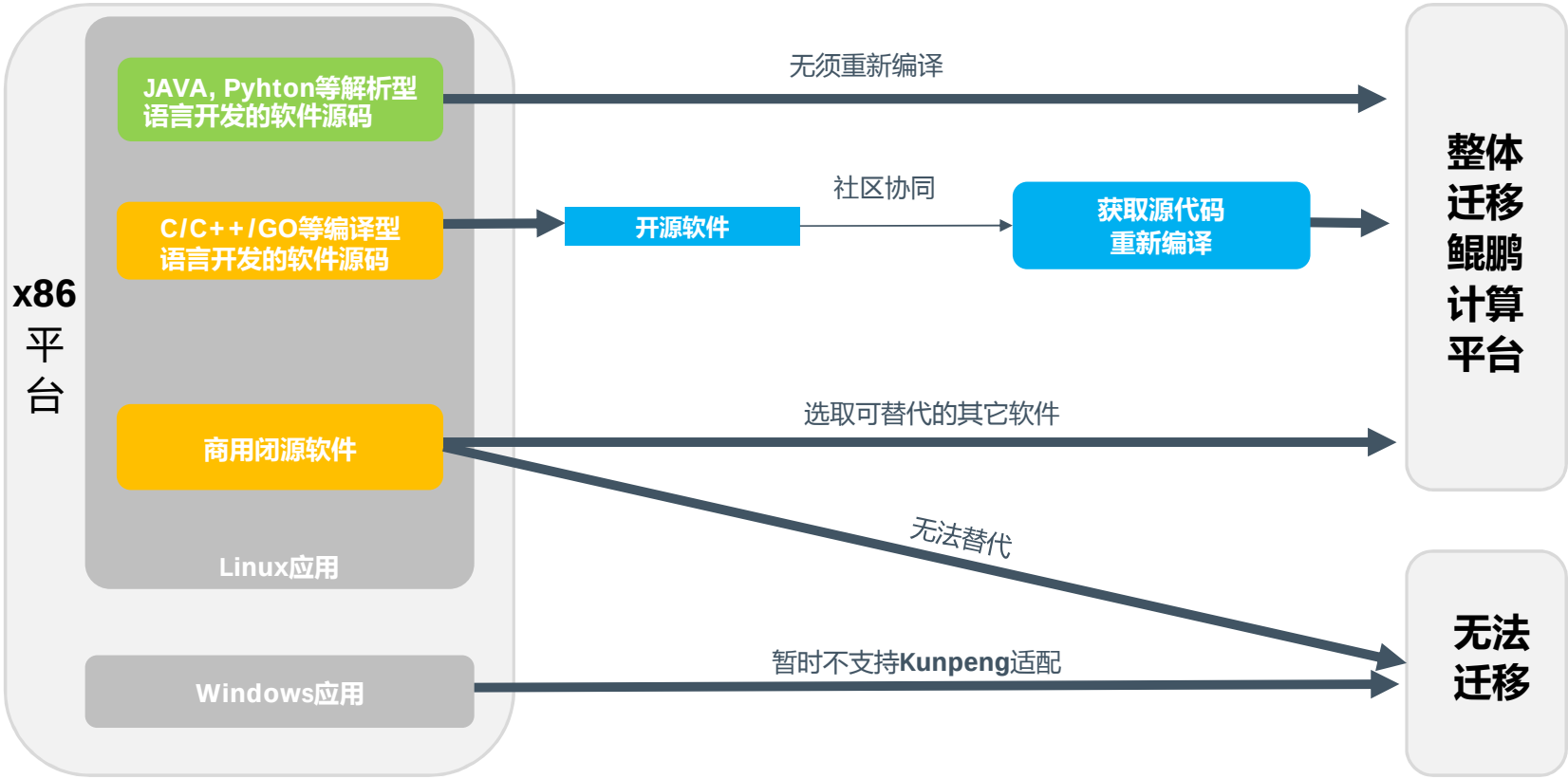
从源码到可执行程序 - 解释型语言

- **解释型语言**：典型的如**Java/Python**语言，都属于解释型语言，解释型语言开发的程序在迁移到鲲鹏处理器时，一般不需要重新编译。
- 解释型语言的源代码由编译器生成字节码，然后再由虚拟机解释执行。虚拟机将不同**CPU**指令集的差异屏蔽，因此解释型语言的可移植性很好。但是如果程序中调用了编译型语言所开发的**so**库，那么这些**so**库需要重新移植编译。



版权所有© 2019 华为技术有限公司

迁移策略选择



版权所有© 2019 华为技术有限公司

目录

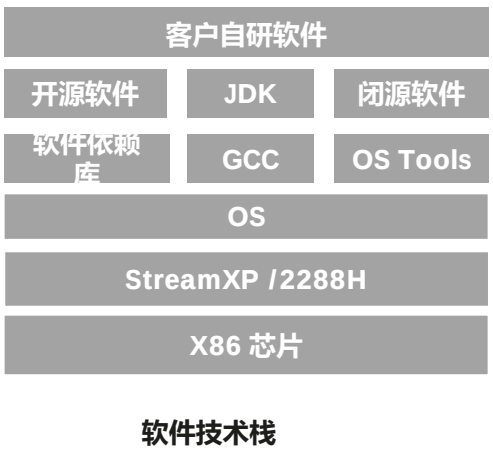
版权所有© 2019 华为技术有限公司

迁移过程概述 - 五个阶段完成软件迁移

	阶段一 技术分析	阶段二 编译迁移	阶段三 功能验证	阶段四 性能调优	阶段五 规模商用
软件移植	• 软件栈分析（应用软件、OS、数据库、中间件组件等） • 编程语言/代码、依赖库分析	• 重写汇编代码 • 修改编译选项 • 代码编译（含依赖库替换）	• 全量功能验证 • 交付工具适配	• 产品关键性能指标测试和调优 • 全面性能测试和调优	• 可靠性、可服务性验证和配置工具开发 • 上市资料刷新
迁移环境	• 准备调试编译环境（准备测试样机服务器/OpenLab线上服务器）	• 搭建编译调试环境（OS/编译器/CI工程等）	• 搭建功能测试环境	• 部署测试工具	• 部署生产系统 • 灰度/割接上线
管理工作	• 成立项目组 • 制定迁移计划 • 协调相关人力/物料资源	• 例行监控与沟通汇报	• 例行监控与沟通汇报	• 例行监控与沟通汇报	• 项目总结关闭

版权所有© 2019 华为技术有限公司

阶段一：软件移植 - 软硬件技术栈整体迁移策略



软件技术栈

业务软件

开源软件

- **开源软件**：获取开源软件**ARM64**软件包或下载源码重新编译
- **依赖组件**：获取**ARM64**包或获取源码重新编译，闭源组件需升级替换**ARM**版本或替换类似组件库

自研软件

- **编译语言**：**C/C++/Go**类编译语言重新编译
- **解释语言**：**Java/Python**等解释型语言替换**ARM**版本**JDK**或**PVM**虚拟机

商用软件

- 获取支持鲲鹏芯片或**TaiShan**服务器的软件版本
- 如无法获取到兼容版本，需更换其他类似软件

运行环境

JDK/PVM

- 替换支持鲲鹏、**TaiShan**版本，如**OpenJDK**、**Huawei JDK**等

编译器

- 替换支持鲲鹏、**TaiShan**版本编译器，如**GCC 9.1**以上

操作系统

- 替换支持鲲鹏、**TaiShan**版本操作系统，可参考软件兼容性清单

驱动

- 替换支持鲲鹏、**TaiShan**版本的驱动

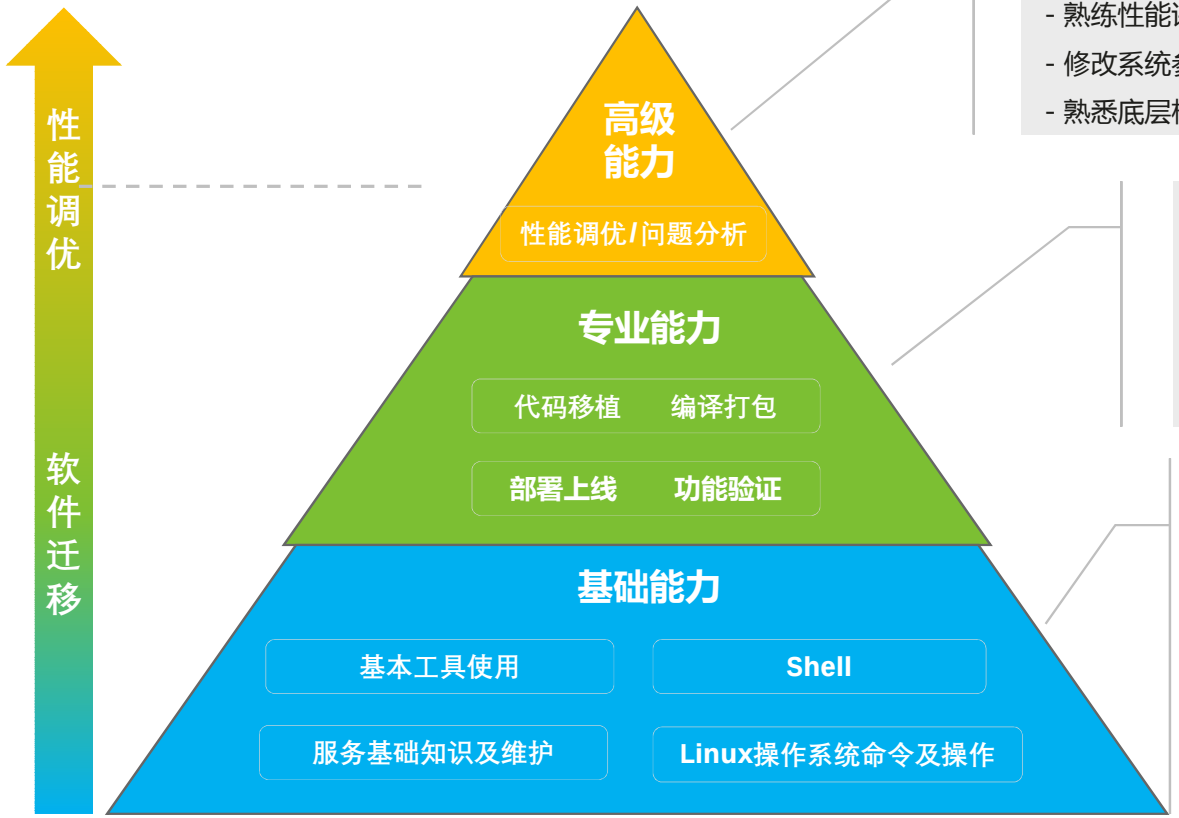
迁移环境

- 获取**Taishan**服务器测试样机/或申请**OpenLab**线上服务器资源

说明：在华为云社区的鲲鹏论坛，已经建立了软件仓库，部分软件包可以在软件仓库直接下载

版权所有© 2019 华为技术有限公司

阶段一：管理工作 - 软件迁移团队能力模型与要求



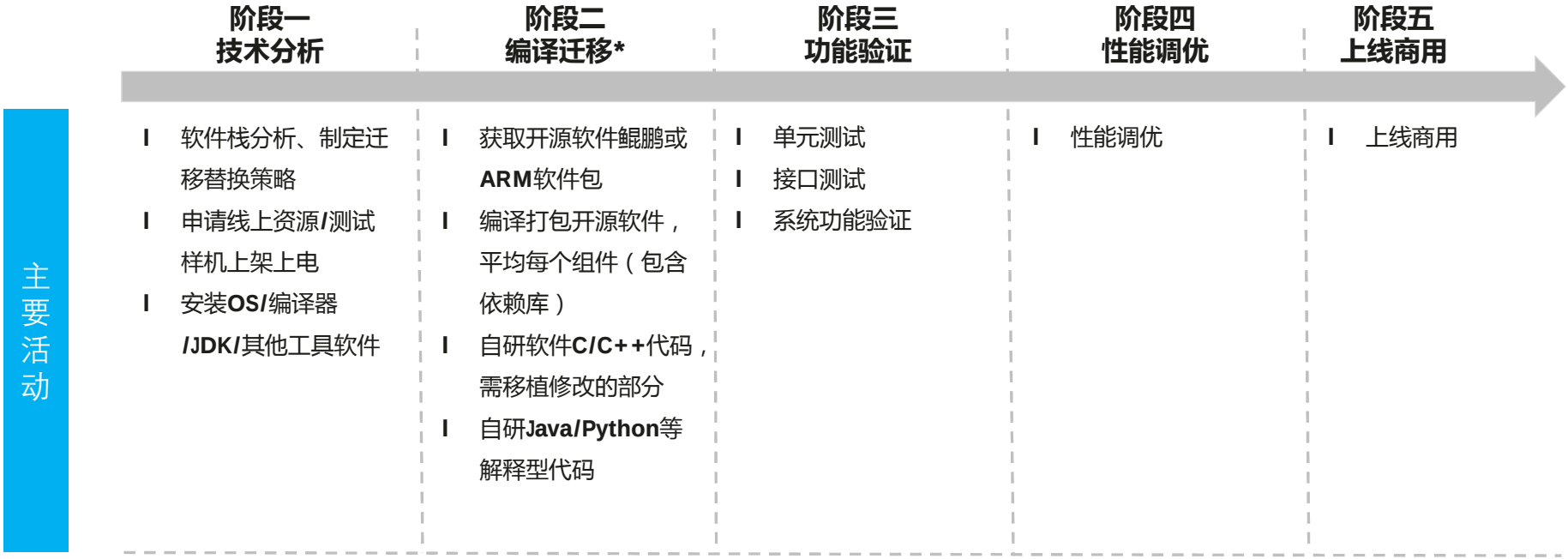
- 评估性能瓶颈（CPU/IO/MEM/NET）
- 熟练性能评估工具(perf/strace/numactl)
- 修改系统参数/业务代码完成性能调优
- 熟悉底层机制(NUMA/中断/流水线)

- 熟悉gcc/maven等编译工具/编译代码
- 阅读修改MakeFile/Pom.xml等编译脚本
- 制作jar包/RPM包
- 安装包启动服务完成部署
- 软件功能验证测试

- TaiShan服务器基本结构/装卸组件/启动上下电/iBMC/Bios/RAID
- Linux系统: cd/rm/ln/chmod/cp/find/grep...
- 读懂Shell脚本，编写简单Shell程序
- 熟悉工具：vim/tar/jar/wget/git/yum/rpm/BeyondCompare
...

版权所有© 2019 华为技术有限公司

阶段一：管理工作 – 制定迁移计划

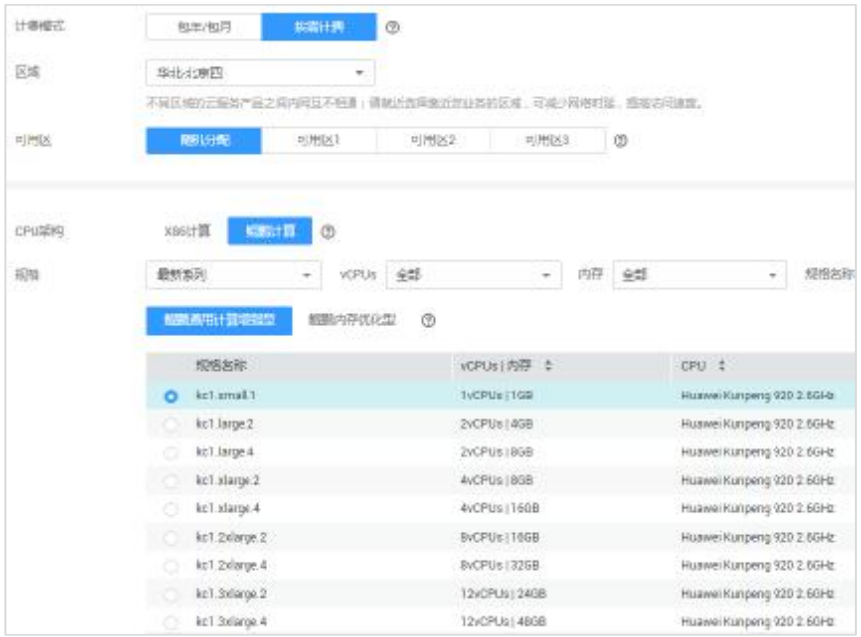


版权所有© 2019 华为技术有限公司

阶段一：迁移环境 - 准备迁移环境

通过华为云鲲鹏云服务器搭建开发环境

1. 登录华为云官网，查看弹性云服务器https://www.huaweicloud.com/product/ecs.html
2. 通过华为云账号登录，点击立即购买
3. CPU架构选择鲲鹏计算，然后按照指导完成鲲鹏弹性云服务器的购买

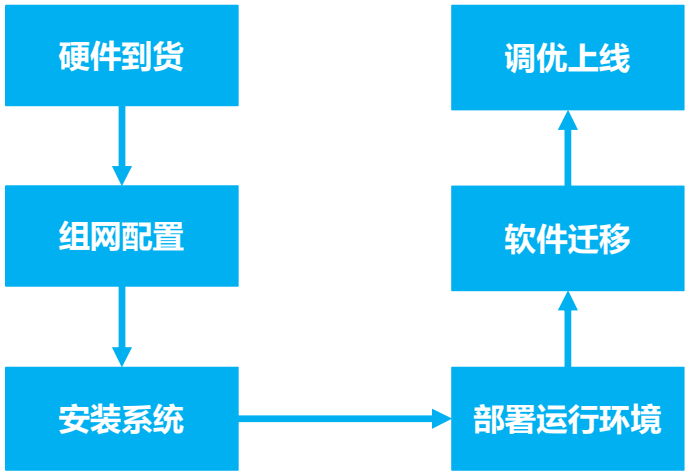


版权所有© 2019 华为技术有限公司

阶段一：迁移环境 - 准备迁移环境

本地TaiShan服务器环境

- 本地迁移流程



- 服务器要连接公网环境
- 配置物理服务器，组网，需要一定周期
- 仅本地可用，无认证

版权所有© 2019 华为技术有限公司

阶段一：迁移环境 - 准备迁移环境

远程Openlab

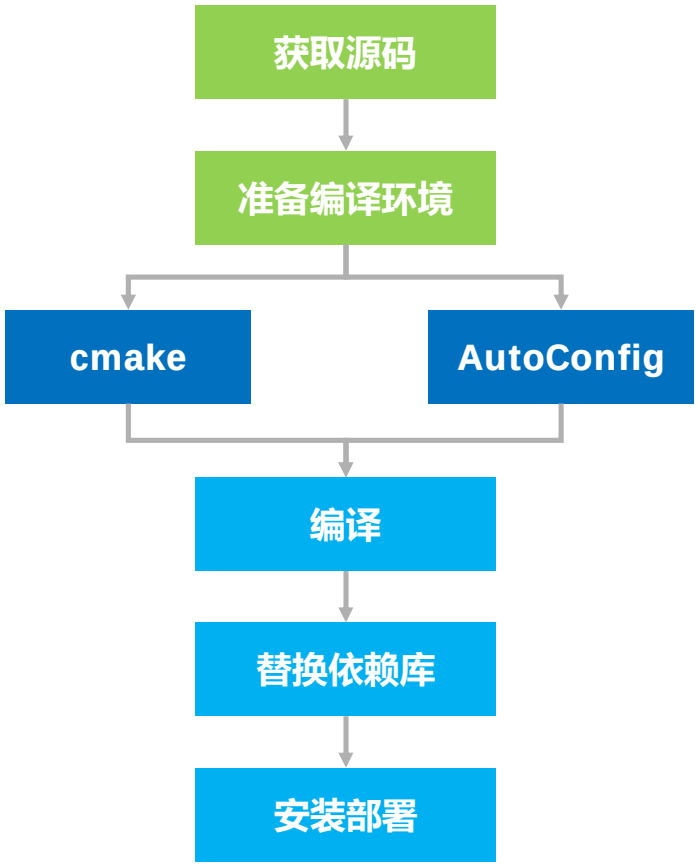
通过开放工具化、流程化能力，支持面向全软件栈的鲲鹏生态建设

业务流	源码移植检查	编译构建	性能调优	自动化检测	认证发布
测试套	兼容性测试	稳定性测试	性能测试	安全测试	功耗测试
工具链	移植知识库	扫描工具	移植工具	性能分析工具	自动化检测工具

- 办公环境可以连接公网环境
- **无需配置物理服务器，向Openlab申请远程服务器资源**
- **提供认证，生态推广**

版权所有© 2019 华为技术有限公司

阶段二：开源软件源码编译移植（C/C++代码）



1、获取源码：通过**github**或第三方开源社区获取源码；

2、准备编译环境：安装编译器**gcc**等；

3、使用开源软件源码中的**cmake**或**AutoConfig**脚本生成**makefile**；

4、执行**makefile**编译可执行程序；

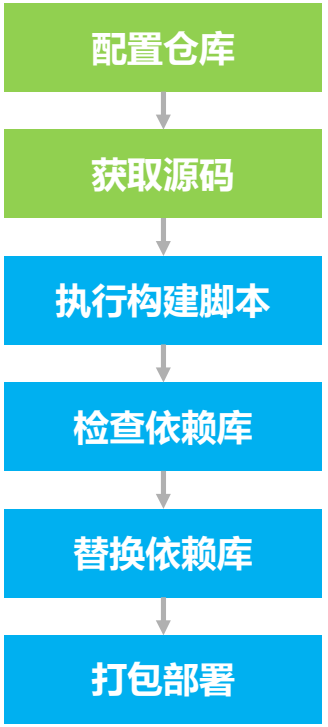
5、替换依赖库：通过开源软件**readme**文件、编译时的**so**库缺失或链接错误，重新编译或替换依赖库；

6、将可执行程序安装部署到生产或测试系统。

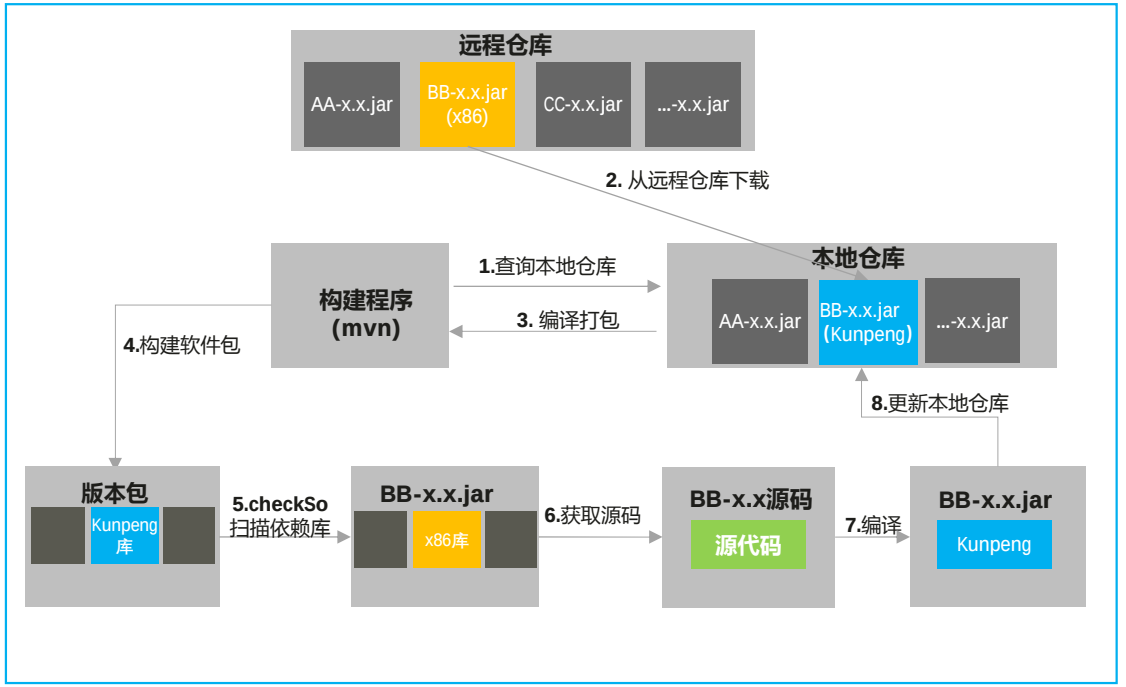
版权所有© 2019 华为技术有限公司

阶段二：开源软件Maven仓库软件迁移（Java）

开源软件迁移过程（Maven软件仓库）



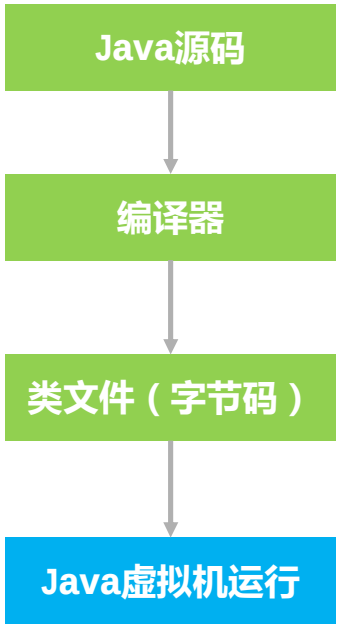
Maven仓库软件构建流程（Maven软件仓库）



版权所有© 2019 华为技术有限公司

阶段二：Java自研代码移植

Java程序编译运行的过程



Java程序迁移至鲲鹏处理器

1、安装鲲鹏或ARM版本JDK

2、配置JDK环境变量

3、编译Java源码生成字节码

4、【可选】移植jar包中的依赖库

（替换鲲鹏/ARM版本或获取C/C++源码重新编译）

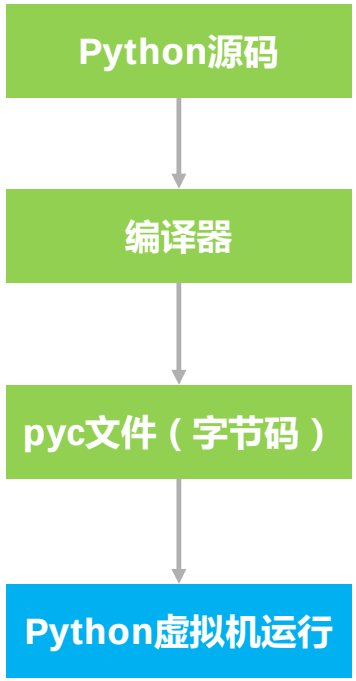
5、【可选】使用新的依赖库重新打jar包

6、启动Java程序，调试功能

版权所有© 2019 华为技术有限公司

阶段二：Python自研代码移植

Python程序编译运行的过程



Python程序迁移至鲲鹏处理器运行

1、使用操作系统自带的**Python**，或安装兼容鲲鹏/**ARM**版本的**Python**软件

2、设置**Python**执行环境变量

3、【可选】编译生成**pyc**文件（字节码）

4、【可选】移植外部依赖库

（替换鲲鹏/**ARM**版本或获取**C/C++**源码重新编译）

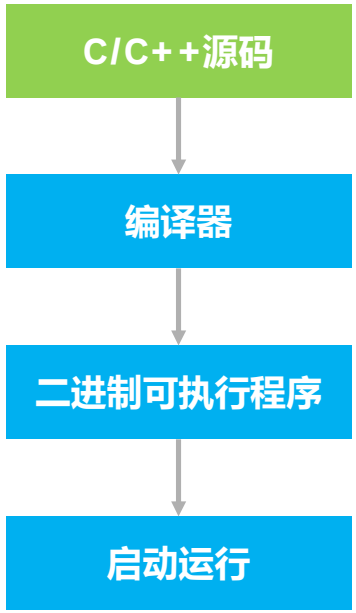
5、【可选】更新代码中新的外部依赖库调用代码

6、执行**Python**源码程序

版权所有© 2019 华为技术有限公司
HUAWEI

阶段二：C/C++自研代码移植

C/C++程序编译运行的过程



C/C++程序迁移至鲲鹏处理器

1、安装编译器（推荐**GCC 7.3.0**以上版本）

2、配置**GCC**环境变量

3、修改**C/C++**源码

4、编译**C/C++**源码，生成可执行程序

5、启动**C/C++**程序，调试功能

版权所有© 2019 华为技术有限公司

阶段二：C/C++代码builtin函数、数据类型移植

	功能说明	X86代码	鲲鹏代码
数据类型	显示定义char类型变量为有符号型	**char a = 'a'**	**signed char a = 'a'**
Builtin函数	使用编译器自带的builtin函数	**__builtin_ia32_crc32qi (__a, __b);**	**__builtin_aarch64_crc32b (__a, __b);**
	把内存数据预取到cache中	asm volatile("**prefetcht0** %0" :: "m" (*(unsigned long *)x));	#define prefetch(_x) **__builtin_prefetch(_x)**

版权所有© 2019 华为技术有限公司

阶段二：C/C++代码编译选项、编译宏移植

	功能	X86编译选项	鲲鹏编译选项
64位编译	定义编译生成的应用程序为64 位，编译选项不同	-m64	-mabi=lp64
ARM指令集	Makefile中定义指令集类型，由X86修改为ARM	-march=broadwell	-march=armv8-a
编译宏	原有X86版编译宏替换成ARM版	__X86_64__	__ARM_64__

版权所有© 2019 华为技术有限公司

阶段二：其他小语种代码移植

Go　lua　Ruby　PHP

Scala　Perl　TypeScript　JavaScript

编译型：源码需要重新编译

解释型：源码不用编译，安装解释器即可

编译型　解释型

搭建程序运行环境方法:

1、操作系统自带解释器软件，直接安装即可使用；

2、从官网上下载支持**ARM**的解释器软件包，直接解压使用；

3、从官网下载解释器源码，编译后使用。

版权所有© **2019** 华为技术有限公司

阶段二：商用闭源软件迁移（商用数据库软件）

版权所有© 2019 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

某行业伙伴快速实现软件迁移的案例

案例：行业伙伴核心应用系统迁移

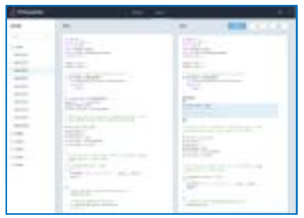
业务系统 **C++/**汇编**/Java** **520+**源码文件	**业务系统** **55**个依赖库移植 **39**个编译问题，**195**行代码修改
Debian OS	**Debian OS** 适配补丁（**30**个）
原有平台 **2.4GHz / 2P / 48**核	**鲲鹏/TaiShan** **2.6GHz / 2P / 96**核

5人月完成520+源码文件移植，性能提升56%

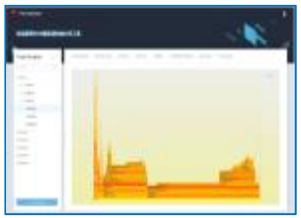
环境准备：**1**人月 → 代码编译和修改：**2**人月 → 性能优化：**2**人月

技术分析
硬件环境
代码评估
环境部署

代码移植
C++/汇编语言移植
安装**JDK**
编译开源代码
更换依赖库



性能优化
热点函数分析
多核绑定
内存就近访问
网卡队列优化



版权所有© **2019** 华为技术有限公司

思考题

1. 为什么x86架构处理器上的软件在鲲鹏处理器使用时需要移植？（　）

A. 两种处理器的指令集不同

B. 源代码需要按照目标处理的指令集编译成指令才能运行

C. 编译型语言由编译器静态编译成指令和数据

D. 解释型语言由语言的虚拟机在运行时将源码/字节码编译成指令和数据

版权所有© 2019 华为技术有限公司

本节小结

- 本节概述了迁移的原理，为什么需要进行软件迁移，软件迁移所具备的环境是什么，以及软件迁移的实施方法与流程。

版权所有© 2019 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

HUAWEI

本节概述和学习目标

- 上一节中我们讲述了软件迁移的原理和实施过程，本节将通过学习华为鲲鹏代码迁移工具，掌握迁移工具的应用场景和操作过程。
- 学完本节后，您将能够：
  - 了解华为鲲鹏代码迁移工具的定位和应用场景
  - 掌握**C/C++**类应用程序移植的操作过程
  - 掌握**Java**类应用程序移植的操作过程

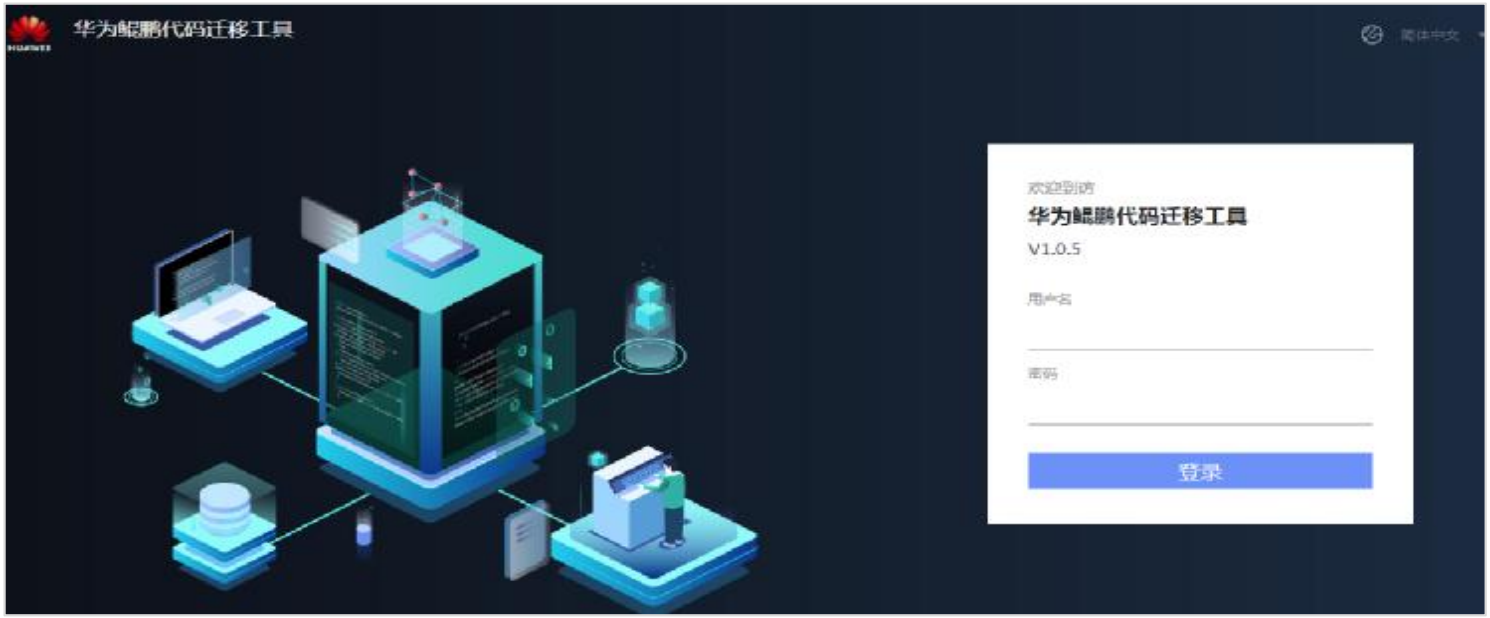
版权所有© **2019** 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

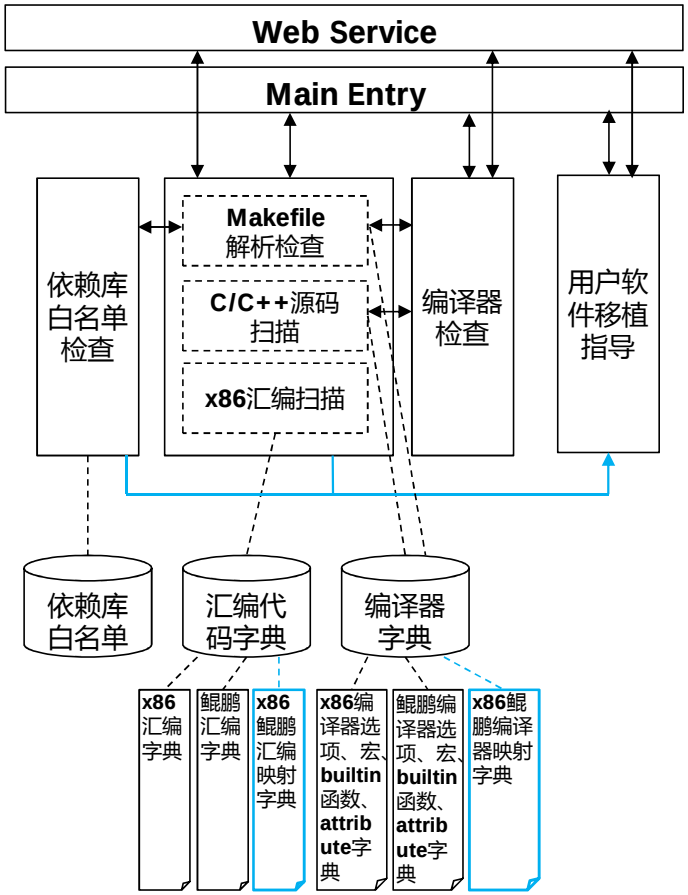
华为鲲鹏代码迁移工具是什么

- **华为鲲鹏代码迁移工具**主要面向鲲鹏平台的开发者、用户和第三方待移植软件提供方开发工程师，用来分析待移植软件源码文件，并给出代码移植指导报告，同时能够自动分析出需修改的代码内容，并指导如何修改，帮助用户顺利完成应用从**x86**平台向鲲鹏平台的移植。



版权所有© 2019 华为技术有限公司

华为鲲鹏代码迁移工具逻辑架构



模块名称	功能说明
Web Service	**Web**服务器。
Main Entry	代码迁移工具命令行入口。
依赖库白名单检查	检查软件构建工程文件中的**SO**文件名列表，对比**SO**依赖库白名单，得到需要移植的**SO**依赖库的详细信息。
Makefile解析检查	检查软件构建工程文件中使用的链接库、编译选项、宏定义，并提供移植修改建议。
C/C++源码扫描	检查源码中使用编译器宏、**builtin**函数、**attributes**，并提供移植修改建议。
x86汇编扫描	检查源码中使用**x86**汇编，提供移植修改建议。
编译器检查	根据编译器版本确定**x86**平台与鲲鹏平台相异的编译器宏、编译选项、**builtin**函数、**attribute**函数等列表。
用户软件移植指导	**1.** 根据包扫描和**C/C++**源码扫描结果合成用户软件移植分析报告。 **2.** 根据源码检查结果，结合软件替换规则字典，为每个修改点增加修改建议。

版权所有© **2019** 华为技术有限公司

华为鲲鹏代码迁移工具业务流程

输入

源码文件：

- C/C++源代码文件
- 汇编源代码文件
- Makefile文件

分析处理

源码文件扫描分析：

- 用户C/C++需要移植部分识别
- 汇编源码同功能指令集、兼容指令集移植部分
- 从Makefile中识别需要移植/替代的编译依赖库
- 根据知识库给出移植指导建议

输出

报告文件：

- 分析文件、分析时间戳等信息
- 关键修改内容综述，简单说明
- 详细csv或html报告，需修改的代码行号，更改点及指导建议
- 编译依赖库移植或替换建议

版权所有© 2019 华为技术有限公司

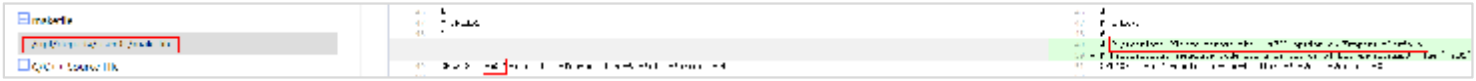
功能特性 (1)

华为鲲鹏代码迁移工具可以作为独立软件提供给用户安装使用，支持的功能特性如下：

- 检查用户软件**C/C++**软件构建工程文件，并指导用户如何移植该文件。

举例： **Makefile**文件检查结果中，原始文件中存在编译选项**-m32**，工具建议移植到鲲鹏平台时请删除该编译选项。

-m32是**x86 32**位应用编译选项，**m32**选项设置**int**为**32bits**及**long**、指针为**32 bits**，为**AMD**的**x86**架构生成代码。在鲲鹏平台无法支持。



- 检查用户软件**C/C++**软件构建工程文件使用的链接库，并提供可移植性信息。

举例： 移植报告检查结果中，原始文件中存在软件构建工程文件使用的**SO**链接库，工具建议下载已有二进制安装包，然后上传至目标服务器进行安装使用即可。

版权所有© 2019 华为技术有限公司

功能特性 (2)

- 检查用户软件**C/C++**源码，并指导用户如何移植源文件。

  举例：对于用户软件**C/C++**源码，工具主要检查编译器宏、**builtin**函数、**attributes**等，并提供移植建议。

  1）对于编译器宏中的**__x86_64__**字段，工具建议移植到鲲鹏平台时请删除该字段。

  2）对于使用编译器自带的builtin函数**__builtin_ia32_crc32qi (__a, __b);**，工具建议修改为**__builtin_aarch64_crc32b (__a, __b);**

- 检查用户软件中**x86**汇编代码，并指导用户如何移植。

  举例：对于用户软件**x86**内存操作的汇编指令，需要从**mov**指令替换为**ldr x0, [x1] & str x0, [x2]**指令。**CRC32C**校验值计算的汇编指令，需要从**crc32q**指令替换为**crc32cb**、**crc32ch**、**crc32cw**、**crc32cx**指令。无条件跳转的汇编指令，需要从**JMP**指令替换为**b**指令。

- 支持命令行和**Web**两种访问方式。

- 支持**SO**依赖库白名单升级。

版权所有© **2019** 华为技术有限公司

应用场景

含有源代码的软件从**x86**平台移植到鲲鹏平台

- 从**x86**平台移植到**TaiShan**服务器
- 从**x86**平台移植到华为鲲鹏云主机

版权所有© **2019** 华为技术有限公司

部署方式

- 华为鲲鹏代码迁移工具采用单机部署方式，即将工具部署在用户的开发、生产环境的**x86**服务器或云服务器上。部署环境要求如下表所示。

类别	子类	要求
硬件	服务器	• **x86**物理服务器 • **TainShan 200**服务器
虚拟机	服务器	• 弹性云服务器**ECS**
操作系统	**CentOS**	• **CentOS 7.6**（**x86**、**TaiShan**服务器、云服务器） • **openEuler 1.0**（**TaiShan**服务器）

版权所有© 2019 华为技术有限公司

如何访问和使用

华为鲲鹏代码迁移工具提供**CLI**和**Web**两种访问方式，安装时由用户选择，只能安装一种。

- **CLI**方式：

  通过命令行方式使用代码迁移工具各功能，最终移植分析结果输出到**.csv**文件，用户可以根据移植建议进行处理。

- **Web**方式：

  通过浏览器远程使用代码迁移工具各功能，最终移植分析结果输出到**.csv**或者**.html**文件中，用户可以根据移植建议进行处理。工具只允许一个用户工作，不支持多用户在线和并发访问。工具具有用户管理功能，由管理员创建和管理。每个用户需要创建自己的工作空间，每个用户的代码扫描任务都在各自的工作空间内完成。

版权所有© 2019 华为技术有限公司

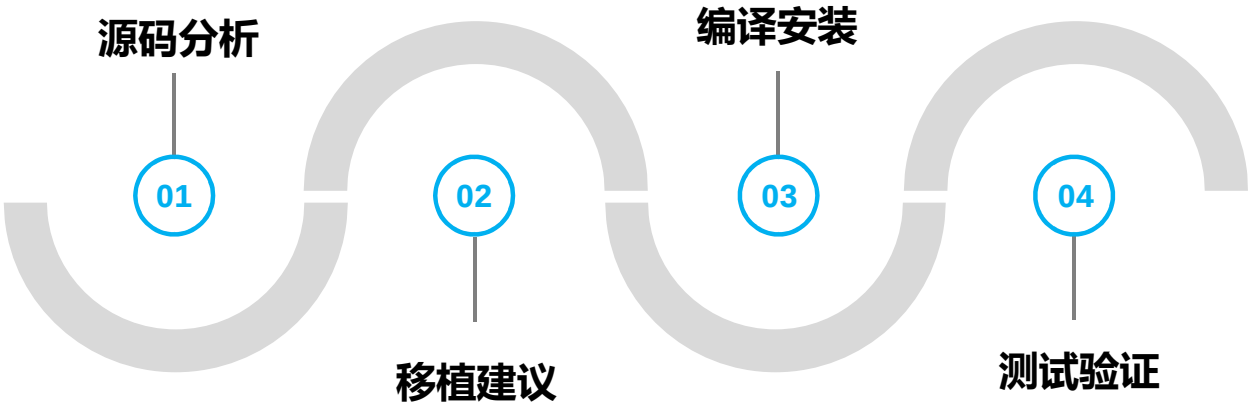
目录

版权所有© 2019 华为技术有限公司

C/C++类应用移植

基于编译型语言开发的应用程序，例如**C/C++**语言应用程序，其编译后得到可执行程序，可执行程序执行时依赖的指令是**CPU**架构相关的。因此，基于**x86**架构编译的**C/C++**语言应用程序，无法直接在**TaiShan**服务器或华为鲲鹏云服务器上运行，需要进行移植编译。

应用移植的流程如下：



版权所有© 2019 华为技术有限公司

源码分析 (1)

- **Step 1** 登录代码迁移工具**Web**界面。
  - 在浏览器地址栏输入**https://*部署服务器的IP*:*端口号***

    例如：https://10.116.239.242:8084，默认用户名为**portadmin**，默认密码为**Admin@9000**
- **Step 2** （可选）如果需要使用其他用户登录**Web**界面进行移植分析时，请单击界面右上角用户名下拉列表选择“用户管理”，然后创建用户。创建用户后，工具自动创建该用户的工作空间。
- **Step 3** 使用**WinSCP**工具将源码文件上传至代码迁移工具所在服务器的“**/opt/portadv/*用户名***”路径下。
  - 例如管理员用户上传至“**/opt/portadv/portadmin**”下。

版权所有© 2019 华为技术有限公司

源码分析 (2)

- **Step 4** 使用默认管理员或新创建的普通用户登录代码迁移工具**Web**界面。
- **Step 5** 新建源码移植分析任务。参数说明如下表所示。
- **Step 6** 单击“分析”，立即启动源码可移植性分析。

参数名称	参数说明
源代码存放路径	源代码存放路径子目录。即**Step3**中上传的源码文件的路径。
编译器版本	选择编译器版本。当前版本只支持**GCC 4.8**。
构建工具	选择构建工具。当前版本只支持**make**。
编译命令	选择源码编译命令。当前版本只支持**make**。
目标操作系统	选择源码将要移植到的目标服务器操作系统型号和版本。
目标系统内核版本	选择目标操作系统内核版本。

版权所有© 2019 华为技术有限公司

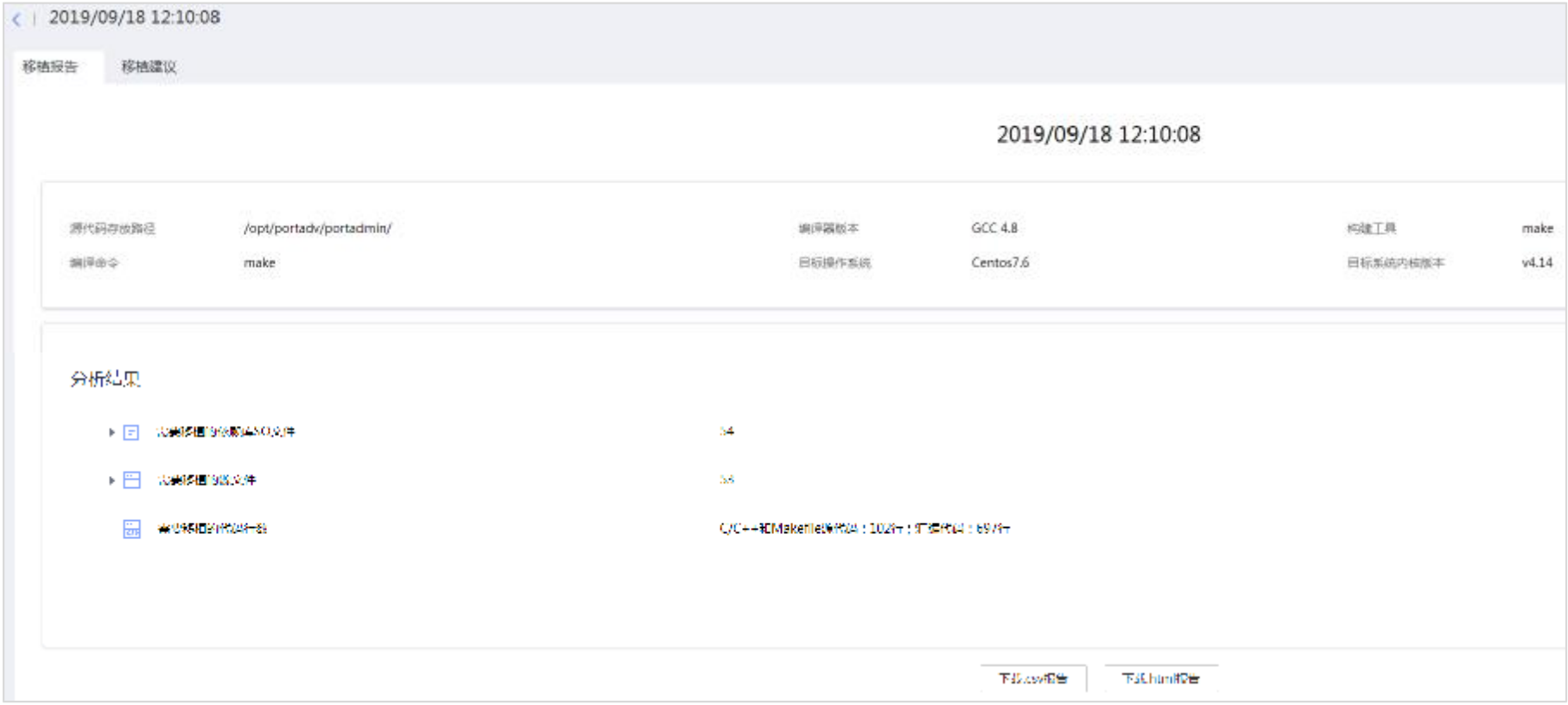
源码分析 (3)

- **Step 7** 分析完成后，在右侧“历史报告”区域单击具体执行时间点的报告，即可查看详细的移植分析结果。

版权所有© 2019 华为技术有限公司

移植建议 (1)

- **Step 1** 打开移植分析报告。



版权所有© 2019 华为技术有限公司

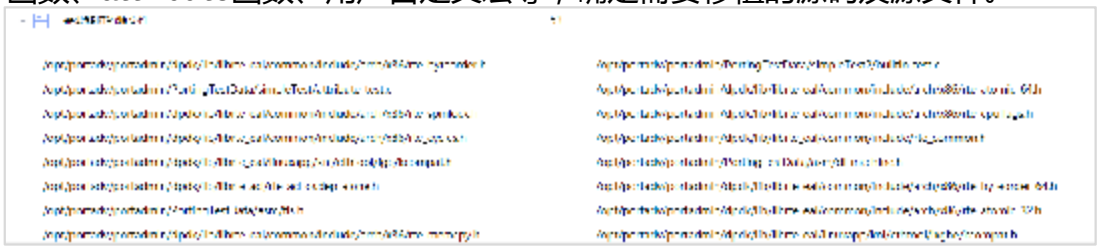
移植建议 (2)

- **Step 2** 查看分析结果。
  - 查看需要移植的依赖库**SO**文件。

根据源码包中的**{SO**文件名**}**列表，对比**SO**依赖库白名单，得到需要移植的**SO**库的详细信息。

  - 查看需要移植的**C/C++**源文件。

扫描分析用户软件目标二进制文件依赖的源文件集合，根据编译器版本信息，检查源码中使用的架构相关的编译器宏、**buildin**函数、**attribute**函数、用户自定义宏等，确定需要移植的源码及源文件。

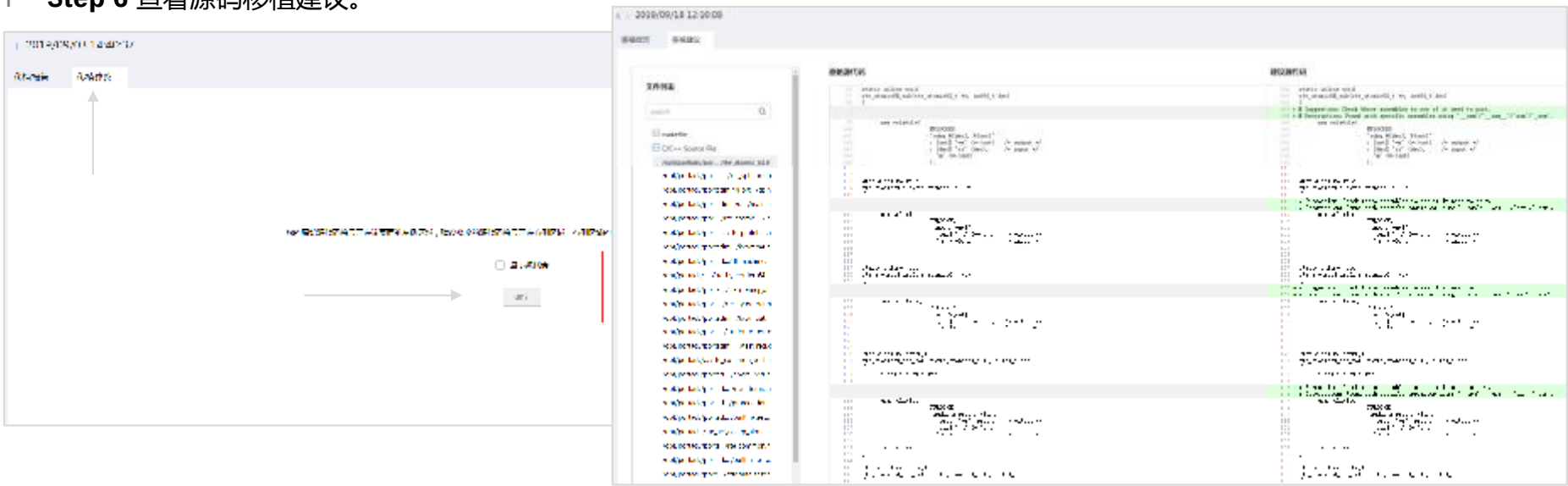


  - 查看需要移植的**C/C++**代码、汇编代码的总行数统计信息

版权所有© **2019** 华为技术有限公司

移植建议 (3)

- **Step 3** （可选）您可以将分析报告下载到本地进行查阅。
  - 支持下载**.csv**格式和**.html**格式的分析报告。
- **Step 4** 单击“移植建议”页签。
- **Step 5** 阅读源码查看说明，勾选“显示源代码”，单击“确定”。
- **Step 6** 查看源码移植建议。



版权所有© 2019 华为技术有限公司

编译安装与测试验证

编译安装：

- **Step 1** 配置编译环境。
  - **1.** 根据移植分析报告中对依赖库**SO**文件的操作建议，将依赖库**SO**文件下载并上传到**TaiShan**服务器或华为鲲鹏云服务器。
  - **2.** 安装或编译依赖库**SO**文件。
- **Step 2** 修改源代码。
  - **1.** 根据移植建议中的源码修改建议，修改源代码。
  - **2.** 将修改后源代码文件上传至**TaiShan**服务器或华为鲲鹏云服务器。
- **Step 3** 编译安装。
  - **1.** 执行编译。
  - **2.** 执行安装。

应用程序编译安装后，需要通过一些测试工具、测试方法等验证应用是否能够正常运行在TaiShan服务器或华为鲲鹏云服务器上：

- **Step 1** 启动并运行应用程序。
- **Step 2** 验证应用程序的功能和性能。

版权所有© **2019** 华为技术有限公司

CLI方式进行代码分析

命令功能

对用户上传的源代码进行扫描分析，并输出详细的移植建议和指导。

命令格式

/opt/portadv/tools/cmd/bin/porting_advisor -S <*source*> -C <*compiler*> --cmd <*cmd*> --tos <*tos*> --tk <*tk*> -D <*debug*> -O <*ouput*>

参数说明

命令	参数	说明
-S/--source	**source**	必配参数，待扫描的**C/C++**源文件所在路径。 路径使用全路径。可以输入多个路径，用“,”隔开。例如：**/path/to/sourcecode**
-C/--compiler	**compiler**	可选参数，指定**GCC**编译器版本，当前只支持**gcc4.8**。
-T/--tools	**tools**	可选参数，指定构建工具及命令行，**make**或者**cmake**，默认是**make**（当前版本只支持**make**）。
--cmd	**cmd**	必配参数，提供完整的软件构建命令，构建命令中必须要有**make**字段。例如： ‘**make all**’
--tos	**tos**	必配参数，软件需要移植到的目标服务器操作系统型号和版本。例如：**CentOS7.6**
--tk	**tk**	可选参数，软件需要移植到的目标操作系统内核版本。例如：**4.14**
-D/--debug	***debug***	可选参数，工具调试日志信息等级，分**INFO**、**WARN**、**ERR**三级。默认等级是**INFO**。 **INFO log**最多，**ERR log**最少。
-O/--ouput	***ouput***	可选参数，指定输出报告格式，**csv**或**html**格式（当前版本只支持**csv**）。例如：**csv**

使用实例

以分析**PortingTestData**应用源代码并输出**csv**格式的分析报告为例，请根据实际情况将“**/opt/code/PortingTestData/**”替换成实际需要扫描的代码路径。

/opt/portadv/tools/cmd/bin/porting_advisor -S /opt/code/PortingTestData/ -C gcc4.8 --cmd 'make all' --tos centos7.6

版权所有© **2019** 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

Java类应用移植

- 基于解释型语言开发的应用程序，是**CPU**架构不相关的，例如**Java**、**Python**，将这类应用程序移植到**TaiShan**服务器或华为鲲鹏云服务器，无需修改和重新编译，按照与**x86**一致的方式部署和运行应用程序即可。
- **Java**应用程序**jar**包内，可能包含基于**C/C++**语言开发的**SO**库文件，这类**SO**库需要移植编译，使用编译得到的**SO**库重新打包**jar**包。移植具体方法与**C/C++**类应用相同。本节不再详细赘述。

版权所有© 2019 华为技术有限公司

思考题

1. 华为鲲鹏代码迁移工具适用于以下哪些类型的应用程序？（　）

   A. C/C++

   B. Java

   C. 汇编

   D. Python

2. 华为鲲鹏代码迁移工具能够提供（　）方面的移植评估结果。

   A. 扫描源码中有多少个安装包

   B. 扫描源码中有多少可以移植的依赖库**SO**文件

   C. 扫描源码中有多少行可以移植的**C/C++**代码、汇编代码

   D. 预估移植所需的工作量

版权所有© 2019 华为技术有限公司

本节小结

- 简要介绍代码迁移工具的定位、应用场景，支持的功能特性，以及用户如何访问和使用工具的方式。
- 介绍使用代码迁移工具如何对**C/C++**类应用进行移植的具体操作指导。
- 介绍需要进行移植的**Java**类应用程序场景说明。

版权所有© **2019** 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

HUAWEI

本节概述和学习目标

- 上一节讲述了使用迁移工具迁移已有的源码到华为鲲鹏平台上，若华为鲲鹏平台的载体不是云服务器或物理服务器，而是基于鲲鹏平台开发的容器的时候如何进行容器的迁移呢？
- 学完本课程后，您将能够：
  - 能够理解容器和镜像的相关概念，容器与虚拟机的区别
  - 能够在鲲鹏服务器产品上通过**Docker pull**和**Dockerfile**方式构建基础镜像
  - 能够通过**docker**安装**redis**应用
  - 能够验证**docker**安装的**redis**应用是否成功

版权所有© **2019** 华为技术有限公司

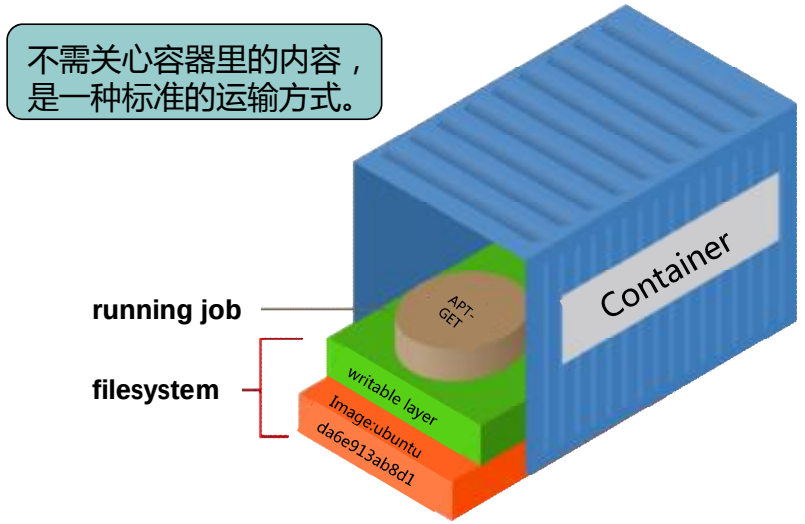
目录

版权所有© 2019 华为技术有限公司

什么是容器

- 容器是一种轻量级、可移植、自包含的软件打包技术，使应用程序可以在几乎任何地方以相同的方式运行。开发人员在自己笔记本上创建并测试好的容器，无需任何修改就能够在生产系统的虚拟机、物理服务器或公有云主机上运行。

运输业



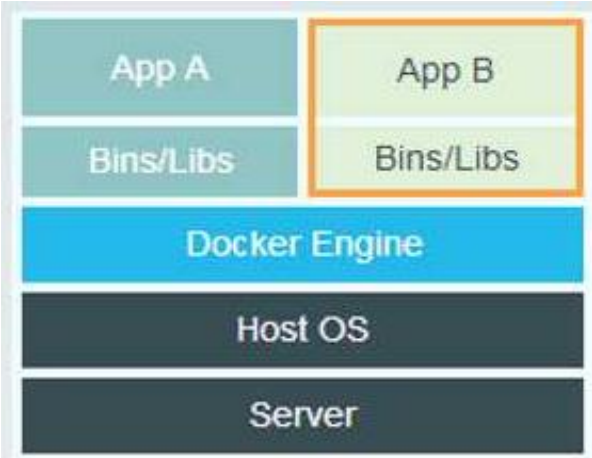
Container

版权所有© 2019 华为技术有限公司

容器与虚拟机

- 容器和虚拟机之间的主要区别在于虚拟化层的位置和操作系统资源的使用方式。

虚拟机



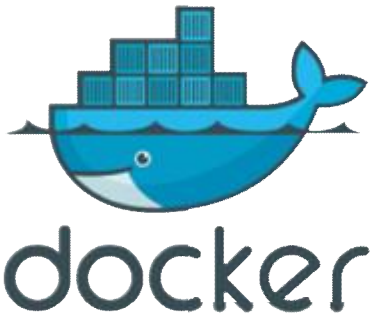
容器

版权所有© 2019 华为技术有限公司

什么是Docker

- **Docker** 是最受大众关注的容器技术，并且现在“几乎”成为事实上的容器标准。

- 简单的应用场景
  - **Web** 应用的自动化打包和发布。
  - 自动化测试和持续集成、发布。



版权所有© 2019 华为技术有限公司

容器与虚拟机参数对比

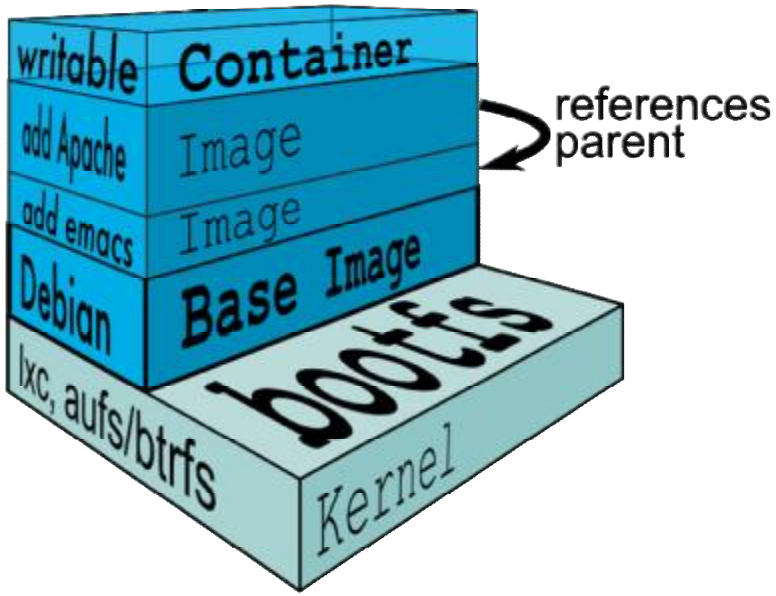
参数	容器（Docker）	对比	虚拟机
快速创建，删除	启动应用	>	启动Guest OS+启动应用
交互，部署	容器镜像	=	虚拟机镜像
密度	单Node 100~1000	>	单Node 10~100
更新管理	迭代式更新，通过修改Dockerfile，对增量内容进行分发，存储，节点启动	>	向虚拟机推送安装，升级应用软件补丁包
启动时间	秒级启动	>	分钟级启动
轻量级	镜像大小通常以M为单位	>	虚拟机以G为单位
性能	docker共享宿主机内核，系统级虚拟化，占用资源少，没有Hypervisor层开销，性能基本接近物理机	>	虚拟机需要Hypervisor层支持，虚拟化一些设备，具有完整的GuestOS，虚拟化开销大，因而降低性能，没有容器性能好。
安全性	Docker具有宿主机root权限，有一定安全隐患	<	硬件隔离，相对安全
高可用性	通过业务本身的高可用性来保证	<	武器库丰富：快照，克隆，HA，动态迁移，异地容灾，异地双活
使用要求	共享宿主机内核，不用考虑CPU是否支持虚拟化技术	>	基于硬件的完全虚拟化，需要硬件CPU虚拟化技术支持

版权所有© 2019 华为技术有限公司

Docker容器与镜像

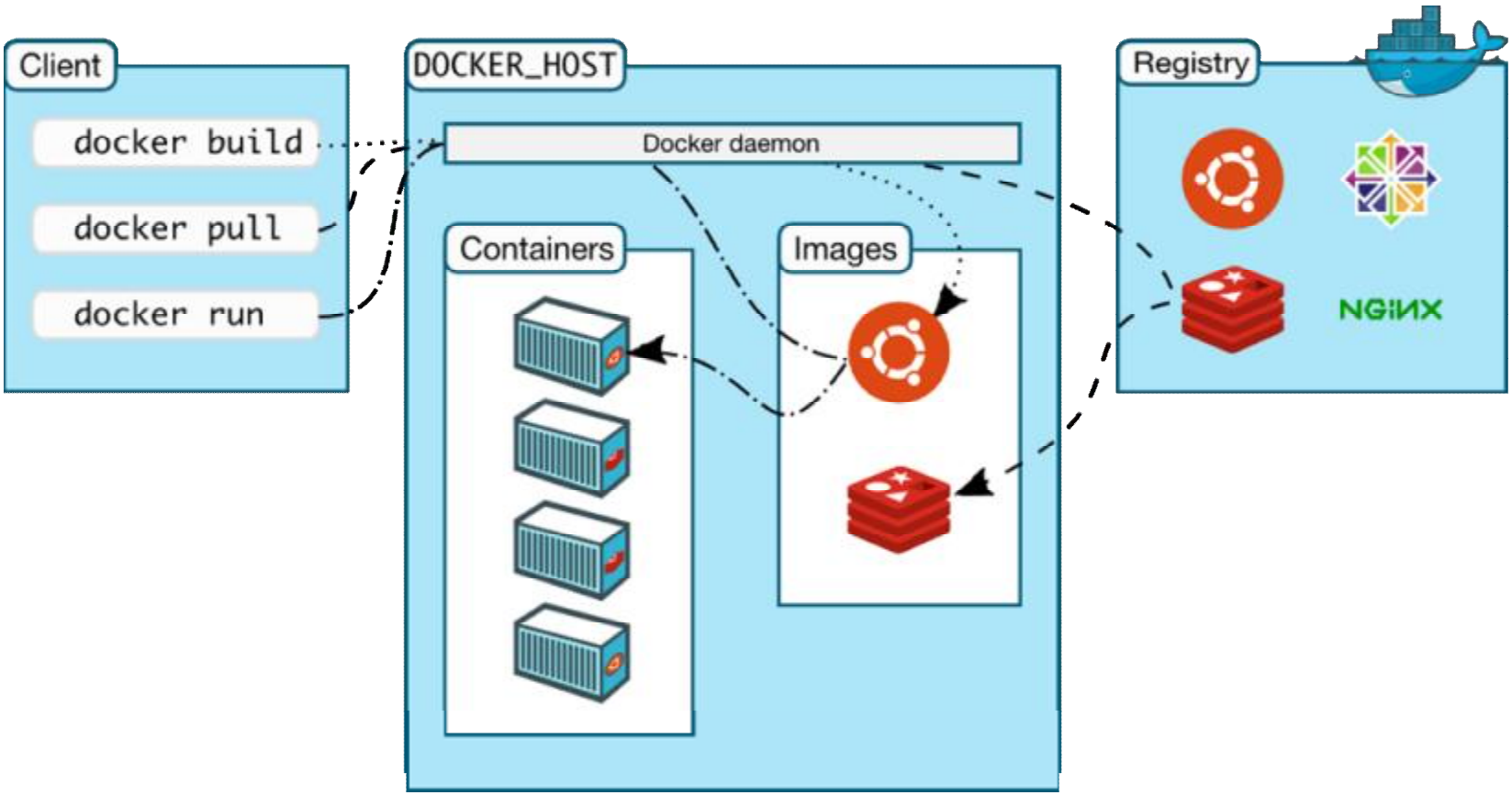
- **Docker容器通过 Docker 镜像来创建**
- 容器与镜像的关系类似于面向对象编程中的对象与类

Docker	面向对象
容器	对象
镜像	类



版权所有© 2019 华为技术有限公司

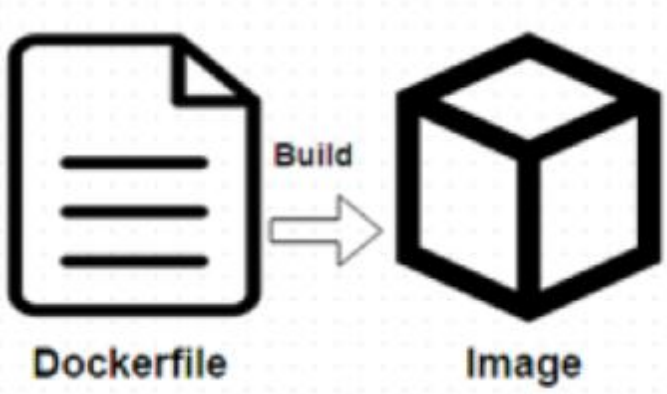
Docker架构



版权所有© 2019 华为技术有限公司

Dockerfile

- **Dockerfile**是一个文本格式的配置文件，包含创建镜像所需要的全部指令。基于在**Dockerfile**中的指令，用户可以快速创建自定义的镜像。
  - **Docker image**的表述
  - 简单语法用来构建镜像
  - 自动化的脚本创建镜像



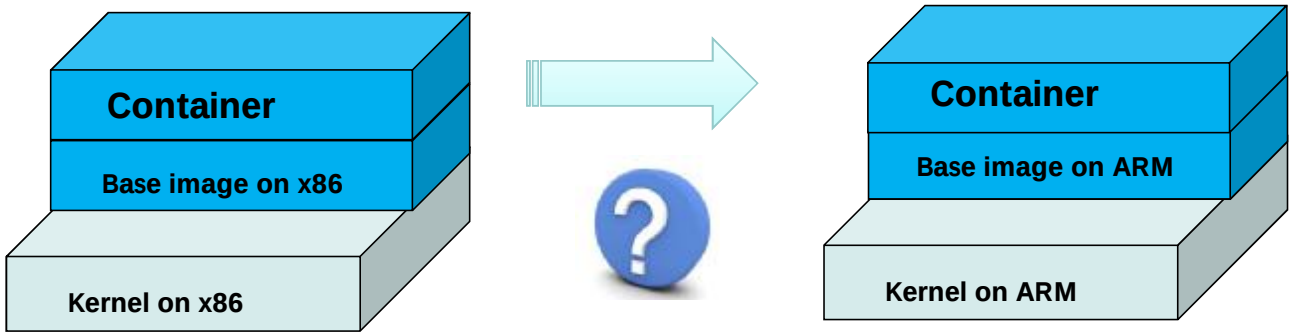
版权所有© 2019 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

容器迁移的背景

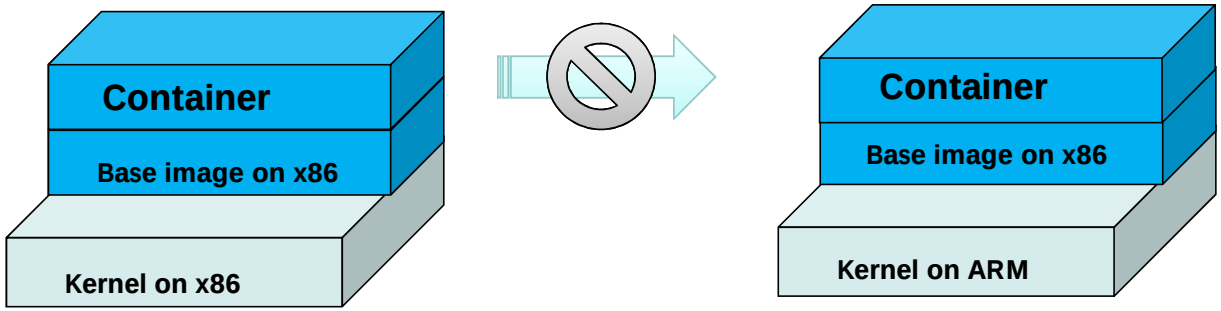
- 随着**Docker**技术应用领域的不断拓展，**Docker**容器使用所遇到的问题也显露无遗，主要表现在：系统负载不均衡和主机硬件维护困难。
- 为了能够有效解决上述问题，彰显**Docker**容器的优势，研究人员提出了**Docker**容器迁移策略。



版权所有© 2019 华为技术有限公司

容器运行的前提

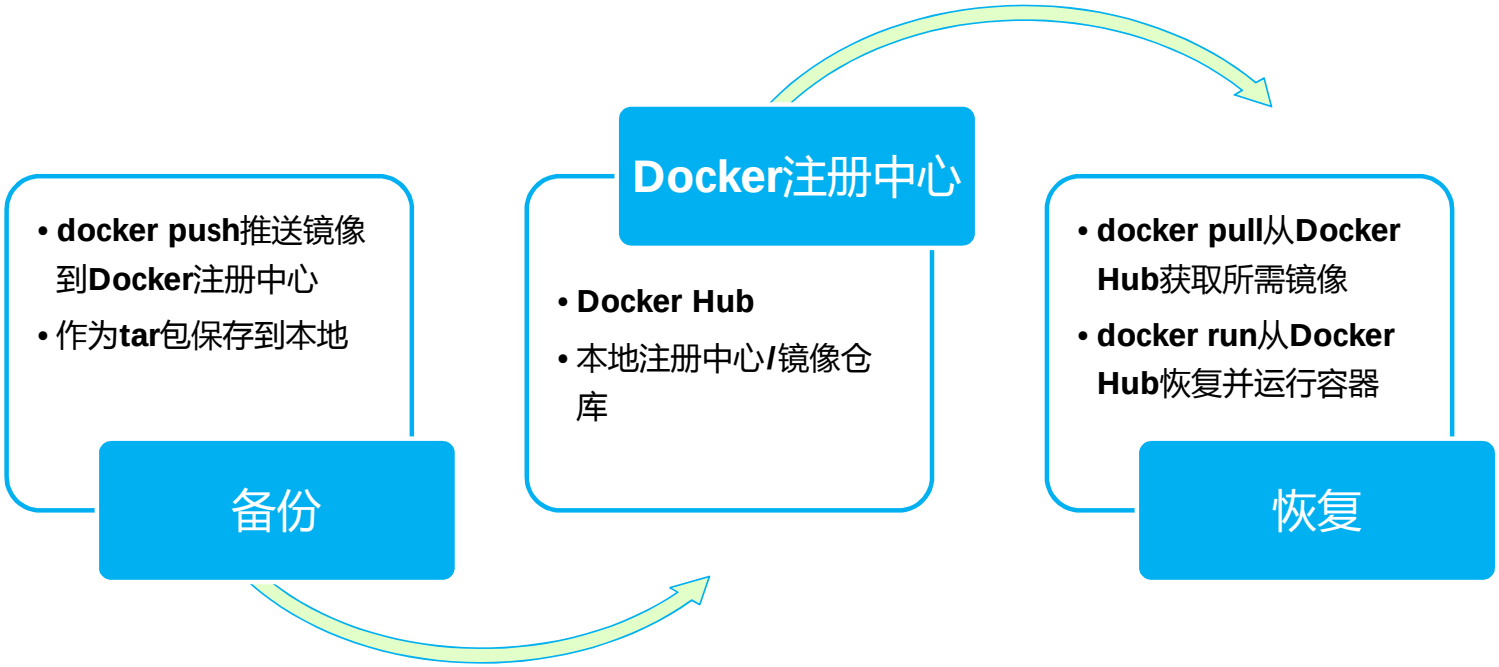
- 跨平台的容器无法运行，会出现格式错误
  - **x86**平台获取的镜像是适用于**x86**平台，当迁移到鲲鹏平台，容器无法执行。
  - 在基于**ARM**的平台中，**docker pull**方式或者**Dockerfile**方式获取或者构建的镜像均为基于**ARM**平台的，同样也无法在**x86**上运行。



版权所有© 2019 华为技术有限公司

容器迁移的原理

迁移容器同时涉及到了上面两个操作，备份和恢复。我们可以将任何一个**Docker**容器从一台机器迁移到另一台机器。在迁移过程中，首先我们将把容器备份为**Docker**镜像快照。然后，该**Docker**镜像或者是被推送到了**Docker**注册中心，或者被作为**tar**包文件保存到了本地。如果我们将镜像推送到了**Docker**注册中心，我们简单地从任何我们想要的机器上使用 **docker run** 命令来恢复并运行该容器，或者通过**docker pull**命令拉取我们想要的镜像。



版权所有© 2019 华为技术有限公司
HUAWEI

Docker容器迁移策略

- **Docker容器迁移有两种策略：使用Docker pull获取镜像或使用Dockerfile构建镜像。**

Docker pull获取镜像	Dockerfile构建镜像
• **Docker pull**命令直接获取基于**ARM**平台的**docker**镜像 • 使用**Docker pull**命令的对象必须是适用于**ARM**的镜像，如**arm64v8/centos:7**	• 使用**dockerfile**构建基于**ARM**平台的**docker**镜像 • 基础镜像和构建后的镜像均只能使用于**ARM**平台

版权所有© 2019 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

容器迁移整体步骤

- **Docker安装**
- **Docker构建基础镜像**
- **Dockerfile创建应用镜像**
- 验证应用镜像

版权所有© 2019 华为技术有限公司

检查环境是否支持Docker

- 检查内核版本
  - **uname –r**

```
[root@worker01 ~]# uname -r
4.14.0-115.5.1.el7a.aarch64
```

- **Docker支持的CentOS版本要求**
  - **CentOS 7**，系统为**64**位、系统内核版本为 **3.10** 以上
  - **CentOS 6.5**或更高，系统为**64**位、系统内核版本为 **2.6.32-431** 或者更高版本

版权所有© 2019 华为技术有限公司
HUAWEI

安装和启动Docker

- 安装依赖软件包
- 安装 **Docker**
  - **yum -y install docker**
- 启动 **Docker**服务
  - **systemctl start docker**
- 详细参考**https://docs.docker.com/install/linux/docker-ce/centos/**

版权所有© 2019 华为技术有限公司

运行hello-world

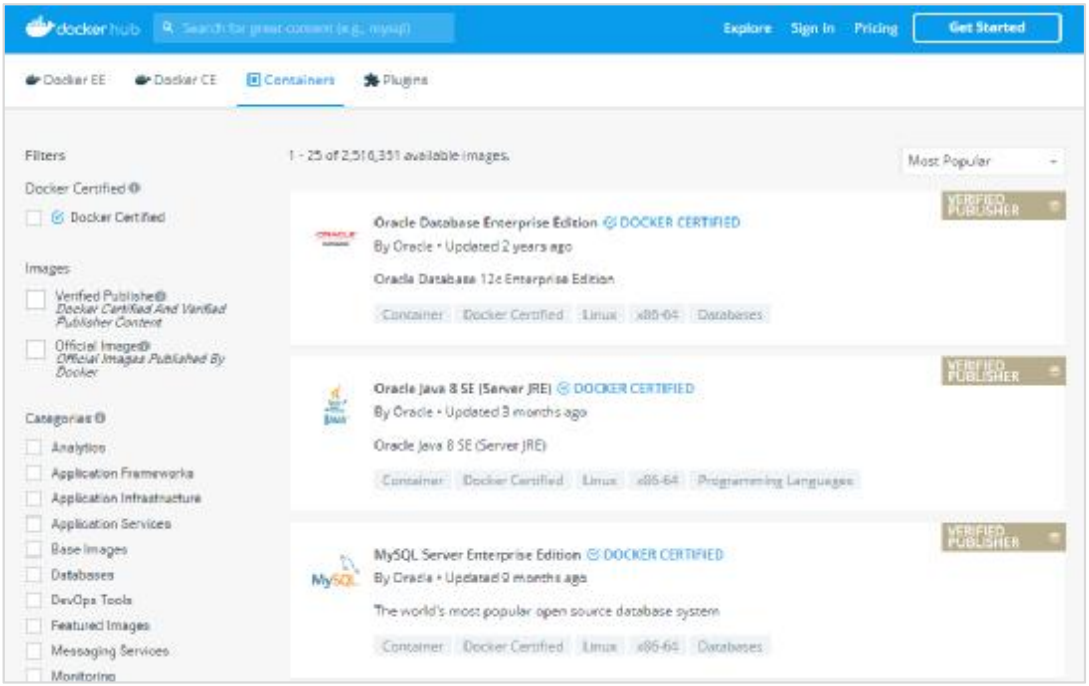
- 运行 **hello-world**
  - 由于本地没有**hello-world**这个镜像，所以会下载一个**hello-world**的镜像，并在容器内运行
  - 对于本地没有的镜像，相当于先**pull**再**run**一个镜像
- **docker run hello-world**

```
[root@worker01 ~]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
3b4173355427: Pull complete
Digest: sha256:b8ba256769a0ac28dd126d584e0a2011cd2877f3f76e093a7ae560f2a5301c00
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (arm64v8)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```


版权所有© 2019 华为技术有限公司

Docker Hub

- **Docker Hub**是目前世界上最大的容器镜像仓库，由**Docker**公司维护。上面有**Docker**公司提供的镜像及大量用户上传的镜像。



版权所有© 2019 华为技术有限公司

查找镜像

- 查找Docker Hub上的CentOS镜像
  - **docker search centos**

```
[root@worker01 ~]# docker search centos
NAME                              DESCRIPTION                                     STARS               OFFICIAL
       AUTOMATED
centos                            The official build of CentOS.                   5566                [OK]

ansible/centos7-ansible           Ansible on Centos7                              123
       [OK]
jdeathe/centos-ssh                CentOS-6 6.10 x86 64 / CentOS-7 7.6.1810 x86…   112
       [OK]
consol/centos-xfce-vnc            Centos container with "headless" VNC session…   99
       [OK]
centos/mysql-57-centos7           MySQL 5.7 SQL database server                   62

imagine10255/centos6-lnmp-php56   centos6-lnmp-php56                              57
       [OK]
tutum/centos                      Simple CentOS docker image with SSH access      45

centos/postgresql-96-centos7      PostgreSQL is an advanced Object-Relational …   39
```


版权所有© 2019 华为技术有限公司

获取镜像

- 从**docker hub**拉取官方的镜像**arm64v8/centos**，标签为**7**
  - **docker pull arm64v8/centos:7**

```
[root@worker01 ~]# docker pull arm64v8/centos:7
7: Pulling from arm64v8/centos
4856e02b0d50: Pull complete
Digest: sha256:df89b0a0b42916b5b31b334fd52d3e396c226ad97dfe772848bdd6b00fb42bf0
Status: Downloaded newer image for arm64v8/centos:7
docker.io/arm64v8/centos:7
```

- 查看拉取官方的镜像**arm64v8/centos**，本地镜像列表里查到**REPOSITORY**为**arm64v8/centos**，标签为**7**的镜像时，说明拉取成功
  - **docker images arm64v8/centos:7**

```
[root@worker01 ~]# docker images arm64v8/centos:7
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
arm64v8/centos      7                   0cb4fb73950e        4 weeks ago         239MB
```


版权所有© 2019 华为技术有限公司

启动和查看容器

- 使用镜像**arm64v8/centos:7**以交互模式启动一个容器，在容器内执行**/bin/bash**命令
  - **docker run -it arm64v8/centos:7 /bin/bash**

```
[root@worker01 ~]# docker run -it arm64v8/centos:7 /bin/bash
[root@f4603f1f918a /]#
```

- 如果要退出就按**Ctrl-D**，或者输入**exit**
- 查看容器运行中和未运行的容器
  - **docker ps -a**

```
[root@worker01 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                      PORTS
  NAMES
f4603f1f918a        arm64v8/centos:7    "/bin/bash"         14 hours ago        Exited (0) 2 minutes ago
  busy_khorana
7c69973c7050        hello-world         "/hello"            14 hours ago        Exited (0) 14 hours ago
  mystifying_brown
```


版权所有© 2019 华为技术有限公司

提交和查看新镜像

- 根据容器ID创建一个新的镜像作为Redis的基础镜像
  - **docker commit -a "huawei.com" -m "redis images" f4603f1f918a t_arm64v8/centos:7**

```
[root@worker01 ~]# docker commit -a "huawei.com" -m "redis images" f4603f1f918a t_arm64v8/centos:7
sha256:ac4b911562b0b8ec231cb4a4e216f17b524f90e088742f557c110de7456f0c25
[root@worker01 ~]#
```

- 查看新构建的基础镜像
  - **docker images**

```
[root@worker01 ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
t_arm64v8/centos    7                   ac4b911562b0        48 seconds ago      385MB
arm64v8/centos      7                   0cb4fb73950e        4 weeks ago         239MB
hello-world         latest              de6f0c40d4e5        8 months ago        4.75kB
```


版权所有© 2019 华为技术有限公司

创建redis目录

- 创建**redis**目录
  - **mkdir -p ~/redis ~/redis/data**
  - **data**目录将映射为**redis**容器配置的**/data**目录，作为**redis**数据持久化的存储目录
- 创建**redis**容器时候会自动产生一些数据，为了不让数据随着**container**的消失而消失，保证数据的安全性。例如：数据库容器，数据表的表会产生一些数据，如果把**container**给删除，数据就丢失。为了保证数据不丢失，这就有了**Volume**的存在。

版权所有© 2019 华为技术有限公司

Dockerfile编写

- 进入创建的**redis**目录，创建**Dockerfile**

  - **Dockfile**编写内容如下：

```
FROM t_arm64v8/centos:7
WORKDIR /home
RUN wget
http://download.redis.io/releases/redis-
5.0.5.tar.gz && \
tar -xvzf redis-5.0.5.tar.gz && \
mv redis-5.0.5/ redis && \
rm -f redis-5.0.5.tar.gz
WORKDIR /home/redis
RUN make && make install
VOLUME /data

EXPOSE 6379
CMD ["redis-server"]
```

  - **Dockfile**参数内容如下：

FROM <image>或者FROM <image>:<tag>：基于哪个镜像，**<image>**首选本地是否存在，如果不存在则会从公共仓库下载

WORKDIR：切换目录用，可以多次切换（相当于**cd**命令），对**RUN**，**CMD**，**ENTRYPOINT**生效

RUN：执行指令

EXPOSE：EXPOSE指令告诉容器在运行时要监听的端口，但是这个端口是用于多个容器之间通信用的（**links**），外面的**host**是访问不到的。要把端口暴露给外面的主机，在启动容器时使用**-p**选项

VOLUME：指定挂载点**/data**

CMD：用于容器启动时指定的服务

版权所有© 2019 华为技术有限公司

通过Dockerfile创建镜像

- 通过Dockerfile创建redis镜像
  - **docker build -t t_arm64v8/centos_redis:5.05 .**

```
[root@worker01 redis]# docker build -t t_arm64v8/centos_redis:5.05 .
Sending build context to Docker daemon  124.6MB
Step 1/7 : FROM t_arm64v8/centos:7
 ---> ac4b911562b0
Step 2/7 : WORKDIR /home
 ---> Running in 27e0d1db3f7f
Removing intermediate container 27e0d1db3f7f
 ---> 6d947ee18245
Step 3/7 : RUN wget http://download.redis.io/releases/redis-5.0.5.tar.gz &&         tar -xvzf redis-5.0.5.tar.gz &&         m
v redis-5.0.5/ redis &&         rm -f redis-5.0.5.tar.gz
 ---> Running in c4380adfd537
--2019-09-21 02:58:18--  http://download.redis.io/releases/redis-5.0.5.tar.gz
Resolving download.redis.io (download.redis.io)... 109.74.203.151
Connecting to download.redis.io (download.redis.io)|109.74.203.151|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1975750 (1.9M) [application/x-gzip]
Saving to: 'redis-5.0.5.tar.gz'
```


版权所有© 2019 华为技术有限公司

执行redis-server

- 运行容器，执行**redis-server**
  - **docker run -p 6379:6379 -v $PWD/data:/data -d t_arm64v8/centos_redis:5.05 redis-server --appendonly yes**

```
[root@worker01 redis]# docker run -p 6379:6379 -v $PWD/data:/data -d t_arm64v8/centos_redis:5.05 redis-server --appendonly ye
s
566f60cc5f6fedc3ac7c92c09ea36d79c1a9880459141dec492eaac3c3637ddf
```

- 命令说明：
  - **-p 6379:6379**：将容器的**6379**端口映射到主机的**6379**端口
  - **-v $PWD/data:/data**：将主机中当前目录下的**data**挂载到容器的**/data**
  - **redis-server --appendonly yes**：在容器执行**redis-server**启动命令，并打开**redis**持久化配置

版权所有© 2019 华为技术有限公司
HUAWEI

连接redis容器

- 执行**redis-cli**命令连接到刚启动的容器，即连接容器中的**redis**服务端
  - **docker exec -it 566f60cc5f6f redis-cli**
  - **127.0.0.1:6379> info**

 版权所有© 2019 华为技术有限公司

HUAWEI

使用redis容器

- 连接容器中的**redis**服务端后，在交互命令中输入命令，即可使用**redis**了

```
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> set runkey "hello redis"
OK
127.0.0.1:6379> get runkey
"hello redis"
127.0.0.1:6379>
```

- 说明
  - **ping**返回**PONG**说明检测到 **redis** 服务已经启动
  - **set runkey "hello redis"**：设置**runkey**值为**"hello redis"**，返回**OK**，说明设置成功
  - **get runkey**：获取**runkey**的值，返回**"hello redis"**说明与设置的相匹配

版权所有© 2019 华为技术有限公司

思考题

1. 关于执行命令“**docker ps -a**”后，显示的标题含义描述，正确的是？（ ）

   A. CONTAINER ID:容器的唯一表示**ID**

   B. IMAGE:创建容器时使用的镜像

   C. COMMAND:容器最后运行的命令

   D. CREATED:创建容器的时间

2. 关于**Docker**的镜像仓库，说法正确的是？**( )**

   A.实现**Docker**镜像的全局存储

   B. 提供**API**接口

   C.提供**Docker**镜像的下载**/**推送**/**查询

   D.可用于租户管理

版权所有© 2019 华为技术有限公司

本节小结

- 本节主要讲述了什么是容器，以及在鲲鹏平台上如何进行容器的迁移，包括如下：
  - 容器的相关概念介绍
  - **Docker**容器迁移策略
  - **Docker**安装
  - **Docker**构建基础镜像
  - 根据基础镜像安装**Redis**
  - 验证**Redis**镜像

版权所有© 2019 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

本节概述和学习目标

- 从前面章节我们了解了迁移原理和过程，以及在不同载体上的迁移流程和操作步骤，最后我们来了解下迁移过程中常见的问题
- 学完本节后，您将能够：
  - 熟悉软件迁移过程中常见的问题
  - 了解常见问题的解决思路

版权所有© 2019 华为技术有限公司

目录

版权所有© 2019 华为技术有限公司

C/C++语言char数据类型默认符号不一致问题 (1)

问题现象

C/C++代码在编译时遇到如下提示：

告警信息：**warning: comparison is always false due to limitedrange of data type**

原因分析

char变量在不同**CPU**架构下默认符号不一致，在**x86**架构下为**signed char**，在**ARM64**平台为**unsigned char**，移植时需要指定**char**变量为**signed char**。

解决方案

1. 在编译选项中加入“**-fsigned-char**”选项，指定**ARM64**平台下的**char**为有符号数；
2. 将**char**类型直接声明为有符号**char**类型：**signed char**。

版权所有© 2019 华为技术有限公司

C/C++语言char数据类型默认符号不一致问题 (2)

鲲鹏弹性云服务器

```
[root@ecs-8225-kunpeng ~]#
[root@ecs-8225-kunpeng ~]# uname -a
Linux ecs-8225-kunpeng 4.14.0-115.5.1.el7a.aarch64 #1 SMP Mon Feb 4 16:38:08 UTC 2019 aarch64
[root@ecs-8225-kunpeng ~]# cat charTest.c
#include <stdio.h>
int main()
{
  char ch=-1;
  printf("ch=%d\n",ch);
  return 0;
}

[root@ecs-8225-kunpeng ~]# gcc -o charTest charTest.c
[root@ecs-8225-kunpeng ~]# ./charTest
ch=255
[root@ecs-8225-kunpeng ~]#
```

X86弹性云服务器

```
[root@ecs-82e5-x86 ~]# uname -a
Linux ecs-82e5-x86 3.10.0-1062.1.1.el7.x86_64 #1 SMP Fri Sep 13 22:55:44 UTC 2019 x86_64
[root@ecs-82e5-x86 ~]# cat charTest.c
#include <stdio.h>
int main()
{
  char ch = -1;
  printf("ch=%d\n",ch);
  return 0;
}
[root@ecs-82e5-x86 ~]# gcc -o charTest charTest.c
[root@ecs-82e5-x86 ~]# ./charTest
ch=-1
[root@ecs-82e5-x86 ~]# _
```

可以看到：相同的代码，鲲鹏下**char**默认为**unsigned char**类型，所以赋值为**-1**的时候，输出的为**-1**对**256**取模的结果即**255**，**X86**中的**char**默认为**signed char**类型，输出为**-1**

版权所有© **2019** 华为技术有限公司

C/C++语言中调用汇编指令编译错误

问题现象

C/C++代码在编译时遇到如下提示：

错误信息：**error**：**impossible constraint in ‘asm’ __asm__ __volatile__**

原因分析

代码中使用汇编指令，而汇编指令与**cpu**指令集强相关。在**x86**架构**cpu**中的汇编指令需要修改为鲲鹏处理器平台的指令才能编译通过，实现功能替换。

解决方案

1. 本例中的代码调用了**x86**平台的汇编指令，修改为鲲鹏处理器对应的指令即可；
2. 部分功能可以修改为使用编译器自带的**builtin**函数，在基本不降低性能的前提下，提升代码的可移植性。

版权所有© **2019** 华为技术有限公司

编译错误：无法识别-m64编译选项

问题现象

C/C++代码在编译时遇到如下提示：

错误信息：**gcc: error: unrecognized command line option ‘-m64’**

原因分析

-m64是**x86 64**位应用编译选项，**m64**选项设置**int**为**32bits**及**long**、指针为**64 bits**，为**AMD**的**x86 64**架构生成代码。在鲲鹏处理器平台无法支持。

解决方案

将鲲鹏处理器平台对应的编译选项设置为**-mabi=lp64**，重新编译即可。

版权所有© 2019 华为技术有限公司

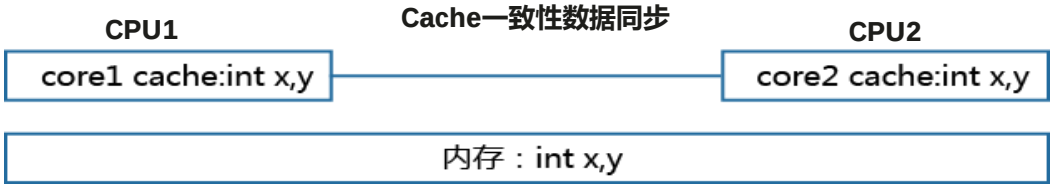
目录

版权所有© 2019 华为技术有限公司

弱内存序问题

弱内存序导致多线程程序执行结果概率性不一致

- **同一份数据，在cache里面存在多份，需要CPU之间进行同步**



- **代码编写顺序和执行顺序可能不一样**

开发人员写如下代码

```
int x = 0;
int y = 0;
...
x = 1;   //x，y为全局变量
y = 1;
```

CPU1上可能的执行顺序与预期不一致：

```
y = 1;
x = 1;

//x在内存中，等待导入，y在cache中，先执行
```

CPU2上的线程在执行如下逻辑，就可能出现问题：

```
if (y ==1){
    assert(x == 1)
}

//通过cache一致性同步y=1，此时x=0
```

- **CPU1：执行x=1时，如果x在内存，y在cache，则core1等待x导入到cache，先执行y=1并同步给CPU2的 core2 cache**
- **CPU2：y=1成立，执行assert(x==1)错误（assert是一种断言操作）**

解决方案

- 使用**内存屏障指令**保证对共享数据的访问和预期一致

smp_wmb(): 保证写操作有序

smp_rmb(): 保证读操作有序

```
int x= 0;
int y = 0;
x = 1;
smp_wmb();    // 等待 x=1 执行完成
y = 1;
```

其它线程在执行如下逻辑，就可能现问题

```
if ( y==1 ) {
    smp_rmb();  // 保证读的数据是最新的
    assert(x==1);
}
```


版权所有© 2019 华为技术有限公司

超出整型取值范围时浮点型转整型与x86不一致

问题现象

C/C++双精度浮点型数转整型数据时，如果超出了整型的取值范围，鲲鹏处理器的表现与**x86**平台的表现不同。测试代码如下：

```
long aa = (long)0x7FFFFFFFFFFFFFFF;
long bb;
bb = (long)(aa*(double)10); // long->double->long
// x86      :  aa=9223372036854775807, bb=-9223372036854775808
// Kunpeng :aa=9223372036854775807, bb=9223372036854775807
```

原因分析

x86（指令集）中的浮点到整型的转换指令，定义了一个**indefinite integer value**——“不确定数值”（**64bit**：**0x8000000000000000**），大多数情况下**x86**平台确实都在遵循这个原则，但是在从**double**向无符号整型转换时，又出现了不同的结果。鲲鹏的处理则非常清晰和简单，在上溢出或下溢出时，保留整型能表示的最大值或最小值。

C/C++语言double类型超出整型取值范围向整型转换参照表

CPU	double值	转为long变量保留值
x86	正值超出long范围	0x8000000000000000
x86	负值超出long范围	0x8000000000000000
鲲鹏	正值超出long范围	0x7FFFFFFFFFFFFFFF
鲲鹏	负值超出long范围	0x8000000000000000

CPU	double值	转为long变量保留值
x86	正值超出long范围	0x8000000000000000
x86	负值超出long范围	0x8000000000000000
鲲鹏	正值超出long范围	0x7FFFFFFFFFFFFFFF
鲲鹏	负值超出long范围	0x8000000000000000

CPU	double值	转为long变量保留值
x86	正值超出long范围	0x8000000000000000
x86	负值超出long范围	0x8000000000000000
鲲鹏	正值超出long范围	0x7FFFFFFFFFFFFFFF
鲲鹏	负值超出long范围	0x8000000000000000

CPU	double值	转为long变量保留值
x86	正值超出long范围	0x8000000000000000
x86	负值超出long范围	0x8000000000000000
鲲鹏	正值超出long范围	0x7FFFFFFFFFFFFFFF
鲲鹏	负值超出long范围	0x8000000000000000

版权所有© 2019 华为技术有限公司
HUAWEI

对结构体中的变量进行原子操作时程序异常

问题现象

程序调用原子操作函数对结构体中的变量进行原子操作，程序**coredump**，堆栈如下：

```
Program received signal SIGBUS, Bus error.
0x000000000040083c in main () at /root/test/src/main.c:19
19          __sync_add_and_fetch(&a.count, step);
(gdb) disassemble
Dump of assembler code for function main:
   0x0000000000400824 <+0>:     sub     sp, sp, #0x10
   0x0000000000400828 <+4>:     mov     x0, #0x1                   // #1
   0x000000000040082c <+8>:     str     x0, [sp, #8]
   0x0000000000400830 <+12>:    adrp    x0, 0x420000 <__libc_start_main@got.plt>
   0x0000000000400834 <+16>: add x0, x0, #0x31 //将变量的地址放入x0寄存器
   0x0000000000400838 <+20>: ldr  x1, [sp, #8] //指定ldxr取数据的长度（此处为8字节）
=> 0x000000000040083c <+24>: ldxr x2, [x0] //ldxr从x0寄存器指向的内存地址中取值
   0x0000000000400840 <+28>:    add     x2, x2, x1
   0x0000000000400844 <+32>:    stlxr   w3, x2, [x0]
   0x0000000000400848 <+36>:    cbnz    w3, 0x40083c <main+24>
   0x000000000040084c <+40>:    dmb     ish
   0x0000000000400850 <+44>:    mov     w0, #0x0                   // #0
   0x0000000000400854 <+48>:    add     sp, sp, #0x10
   0x0000000000400858 <+52>:    ret
End of assembler dump.
(gdb) p /x $x0
$4 = 0x420039 // x0寄存器存放的变量地址不在8字节地址对齐处
```

原因分析

鲲鹏处理器对变量的原子操作、锁操作等用到了**ldaxr**、**stlxr**等指令，这些指令要求变量地址必须按变量长度对齐，否则执行指令会触发异常，导致程序**coredump**。

一般是因为代码中对结构体进行强制字节对齐，导致变量地址不在对齐位置上，对这些变量进行原子操作、锁操作等会触发问题。

解决方案

代码中搜索“**#pragma pack**”关键字（该宏改变了编译器默认的对齐方式），找到使用了字节对齐的结构体，如果结构体中变量会被作为原子操作、自旋锁、互斥锁、信号量、读写锁的输入参数，则需要修改代码保证这些变量按变量长度对齐。

版权所有© 2019 华为技术有限公司

加速器初始化失败

问题现象

使用KAE引擎的测试RSA性能，性能与不启用加速引擎相当。

```
linux-rmw4:/usr/local/bin # ./openssl speed -elapsed -engine kae rsa2048
....
                  sign    verify    sign/s verify/s
rsa 2048 bits 0.001359s 0.000043s    736.0  23458.9
```

原因分析

加速器未加载成功，需要检查加速器驱动、加速器引擎库相关的软链接、环境变量是否正确安装或配置。

解决方案

步骤1 检查如下加速器驱动是否加载成功加载到内核：uacce.ko、qm.ko、sgl.ko、hisi_sec2.ko、hisi_hpre.ko、hisi_zip.ko、hisi_rde.ko

```
[root@localhost ~]# lsmod | grep uacce
uacce 262144 2 hisi_hpre,qm
```

步骤2 检查/usr/lib64和OpenSSL安装目录是否有加速器引擎库，且建立正确的软链接

```
#查询kae是否正确安装并建立软连接，如果有正确安装显示如下内容
[root@localhost home]# ll /usr/local/lib/engines-1.1/ |grep kae
lrwxrwxrwx. 1 root root 22 Nov 12 02:33 kae.so -> kae.so.1.0.1
lrwxrwxrwx. 1 root root 22 Nov 12 02:33 kae.so.0 -> kae.so.1.0.1
-rwxr-xr-x. 1 root root 112632 May 25 2019 kae.so.1.0.1
[root@localhost home]# ll /usr/lib64/ | grep libwd
#查询wd是否正确安装并建立软连接，如果有正确安装显示如下内容
lrwxrwxrwx. 1 root root 14 Nov 12 02:33 libwd.so -> libwd.so.1.0.1
lrwxrwxrwx. 1 root root 14 Nov 12 02:33 libwd.so.0 -> libwd.so.1.0.1
-rwxr-xr-x. 1 root root 137120 May 25 2019 libwd.so.1.0.1
```

步骤3 检查环境变量LD_LIBRARY_PATH是否包含OpenSSL库路径，通过export命令增

```
[root@localhost home]# export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
[root@localhost home]# echo $LD_LIBRARY_PATH
/usr/local/lib
```


版权所有© 2019 华为技术有限公司

目录

 版权所有© 2019 华为技术有限公司

HUAWEI

迁移工具源码路径错误

问题现象

使用**Porting**迁移工具执行分析任务时，填写源码存放路径后，执行分析任务，报"源代码路径错误"：

原因分析

源代码没有放置到**porting**工具的安装目录下

解决方案

在工具**web**界面，重新填写源代码存放路径，单击“分析”。弹窗页面显示任务分析进度，分析完成后，自动跳转至“移植报告”界面。

版权所有© **2019** 华为技术有限公司

Maven软件仓库编译错误：未配置代理

问题现象

在通过代理访问网络的环境，如果没有配置代理编译会报错，如：

repo.maven.apache.org: Name or Service not known

原因分析

maven有自己的网络配置，如果服务器需要代理访问外网，则同时也需要给**maven**设置代理

解决方案

修改**maven**安装路径下的**conf/settings.xml**，在**proxy**标签中添加代理信息：

```
<proxies>
  <proxy>
    <id>optional</id>
    <active>true</active>
    <protocol>http</protocol>
    <username>zhangsan</username>
    <password>*******</password>
    <host>192.168.33.234</host>
    <port>8080</port>
    <nonProxyHosts>local.net|some.host.com</nonProxyHosts>
  </proxy>
</proxies>
```


版权所有© 2019 华为技术有限公司

Maven软件仓库编译错误：无法找到依赖库

问题现象

maven编译报错：

Failed to execute goal on project hadoop-auth: Could not resolve dependencies for project org.apache.hadoop:hadoop-auth:jar:2.7.1.2.3.4.7-4: The following artifacts could not be resolved: org.mortbay.jetty:jetty-util:jar:6.1.26.hwx, org.mortbay.jetty:jetty:jar:6.1.26.hwx, org.apache.zookeeper:zookeeper:jar:3.4.6.2.3.4.7-4: Failure to find org.mortbay.jetty:jetty-util:jar:6.1.26.hwx in https://repository.apache.org/content/repositories/snapshots

原因分析

从远程仓库下载**jetty-util-6.1.26.hwx.jar**包失败，原因是远程仓库中没有对应的**jar**。

解决方案

修改**maven**安装路径下的**conf/settings.xml**，在**mirror**标签中设置**maven**远程仓库路径，比如华为云：

```
<mirror>
  <id>mirror</id>
  <mirrorOf>*</mirrorOf>
  <name>huaweicloud</name>
  <url>https://repo.huaweicloud.com/repository/maven/</url>
</mirror>
```


版权所有© 2019 华为技术有限公司

思考题

1. 在向鲲鹏处理器迁移软件时，以下哪些是可能导致编译错误或告警的原因？（ ）

   A. 编译选项

   B. 数据类型不同

   C. 汇编指令

   D. 弱内存序问题

2. 弱内存序问题主要与如下那些因素相关？（ ）

   A. 多线程

   B. 多进程

   C. 不同**CPU**之间**Cache**同步

   D. 一级、二级、三级**Cache**间数据同步

   E. 不同**core**之间**Cache**同步

版权所有© 2019 华为技术有限公司

本章总结

- 本章主要介绍了从应用从**x86**平台到鲲鹏平台需要进行软件迁移的背景、原理和实施流程，针对服务器和容器两种鲲鹏平台的应用，如何进行迁移，并阐述了迁移过程和主要操作。最后对迁移中所遇到的常见问题给出了解决思路。

版权所有© **2019** 华为技术有限公司

学习推荐

- 《TaiShan代码移植指导》
  - https://bbs.huaweicloud.com/forum/thread-22606-1-1.html
- 华为云鲲鹏社区
  - https://www.huaweicloud.com/kunpeng/
- 《计算机组成与设计（ARM版）》

版权所有© 2019 华为技术有限公司


谢 谢
www.huawei.com