Linux

基本介绍

Linux是一种自由和开放源码的类UNIX操作系统（操作系统是什么不细讲，都是大白话，可以以windows类比）

> 严格说是模仿unix开发的系统（冷知识：安卓是基于Linux开发的）

Linux发行版

linux发行版，就是为一般用户预先集成好的Linux操作系统及各种应用软件。一般用户不需要重新编译，在直接安装之后，只需要小幅度更改设置就可以使用。

> 说人话：内核加工之后就是发行版，里面有现成的应用软件

Linux的优势

- 安全，作为一个开源的操作系统，内部的代码都是对外可见的
- 开发支持友好，Linux的包管理比任何其他操作系统都强大得多。与Windows相比，在Linux中安装软件非常容易。它可以显著增加程序员的工作流程。
- 可移植性好，Linux 可以在各种设备上运行
- 稳定，免费等等

版本选择

版本名称	网 址	特 点	软件包
Debian Linux	www.debian.org	开放的开发模式，且易于进行软件包升级	apt
Fedora Core	www.redhat.com	Fedora Linux（第七版以前为Fedora Core）是较具知名度的Linux发行包之一，由Fedora项目社群开发、红帽公司赞助，目标是建立一套新颖、多功能并且自由（开放源代码）的操作系统。	up2da
CentOS	www.centos.org	CentOS 8系统2021年12月31日已停止维护服务，CentOS 7系统将于2024年06月30日停止维护服务。 CentOS官方不再提供CentOS 9及后续版本，不再支持新的软件和补丁更新。 CentOS用户现有业务随时面临宕机和安全风险，并无法确保及时恢复	rpm
SUSE Linux	www.suse.com	专业的操作系统，易用的 YaST 软件包管理系统	YaST
Mandriva	www.mandriva.com	操作界面友好，使用图形配置工具，有庞大的社区进行技术支持，支持 NTFS 分区的大小变更	rpm
KNOPPIX	www.knoppix.com	可以直接在 CD 上运行，具有优秀的硬件检测和适配能力，可作为系统的急救盘使用	apt
Gentoo Linux	www.gentoo.org	高度的可定制性，使用手册完整	portag
Ubuntu	www.ubuntu.com	优秀已用的桌面环境，基于 Debian 构建	apt

上面的内容后续有需要再来看看就好，现阶段较为热门的 ubuntu，debian 足以满足大部分需求

后续课程我们使用ubuntu讲解

Linux命令

包管理

包管理器又称**软件包管理系统**，它是在电脑中**自动安装、配置、卸载和升级**软件包的工具组合

- 降低安装维护软件的成本
- 避免安装大量软件造成的路径污染
- 不必查找和安装软件的其他依赖项
- 避免捆绑和垃圾软件
- 彻底地卸载

Linux发行版都有一个或多个软件仓库，它基本上是软件包的集合。软件库包含不同种类的软件包（类比应用市场）

一个包可能有**依赖关系**，可能需要安装其他软件包。软件包管理器通常会处理这些依赖关系，并将其与你正在安装的软件包一起自动安装。当你删除一个包时，他的依赖也会被自动删除，或者询问你是否删除

apt常用命令

```
apt [options] [command] [package ...]
```

- **options：**可选，选项包括 -h（帮助），-y（当安装过程提示选择全部为"yes"），-q（不显示安装的过程）等等。
- **command：**要进行的操作。
- **package**：安装的包名

常用命令	说明
sudo apt update	更新本地软件包索引
sudo apt upgrade	升级软件包
sudo apt install <package_name>	安装指定的软件
sudo apt update <package_name>	更新指定的软件
sudo apt show <package_name>	显示软件包具体信息
sudo apt remove <package_name>	删除软件包
sudo apt autoremove	清理不再使用的依赖和库文件
sudo apt purge <package_name>	移除软件包及配置文件
apt list --installed	列出所有已安装的包

在这我们会发现一个问题，在Linux上基本上所有操作都是由命令来完成的，但光是包管理器就有这么多命令，实际上没必要全背下来，只需要记住有哪些操作就差不多了

许多指令提供了 usage，通常可以通过 -h 或 --help 参数触发。

```
apt -h
```

同时在 Linux 中，有一个历史悠久的帮助命令：man。

想要查阅一个命令的帮助手册，只需要 man 一下就可以了。

```
man ls
```

dpkg

`dpkg` 是 Debian 系列的底层包管理工具，专门用于处理 `.deb` 包文件。它不会自动解决软件包的依赖关系，所以通常需要和 APT 配合使用。

常用 dpkg 命令：

- **安装 `.deb` 文件**：
- `sudo dpkg -i <package-name>.deb`
- **列出已安装的软件包**：
- `dpkg -l`
- **删除软件包**：
- `sudo dpkg -r <package-name>`

文件管理

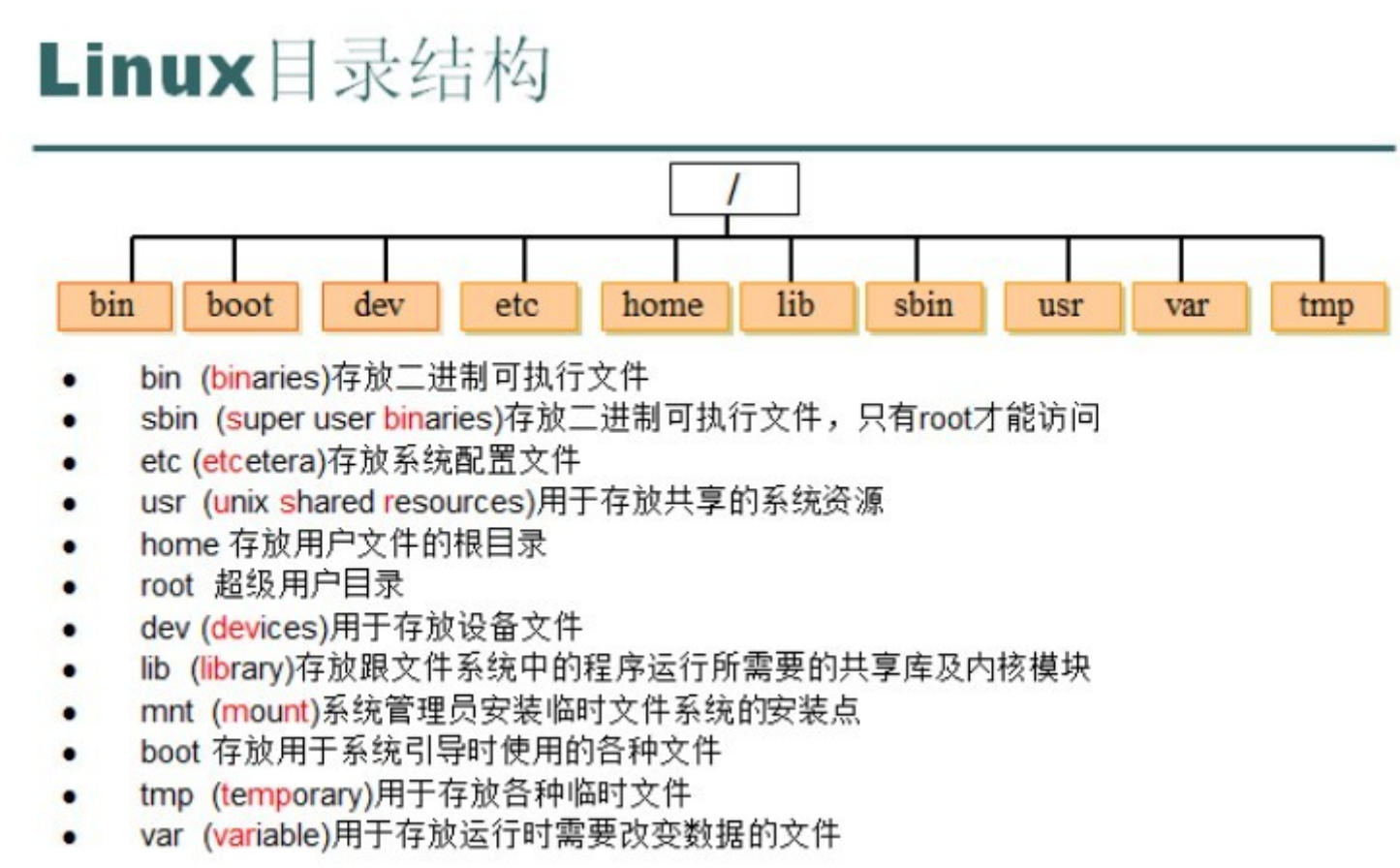
Linux 的发行版虽然有很多，但他们的文件系统**基本上**都遵循一个标准，即FHS标准，这个标准定义了有哪些目录，并且在这些目录下的文件应该起什么样的作用。不过当你使用包管理器下载文件时，包管理器会自动帮你选择路径。

> 一切皆文件

目录结构

FSH (Filesystem Hierarchy Standard):标准的目录结构

```
[root@linux-server ~]# ls /
bin   dev  home  lib64  mnt  proc  run   srv  tmp  var
boot  etc  lib   media  opt  root  sbin  sys  usr
```



文件分类

在Linux系统中**一切皆文件**

- 普通文件（文本文件，二进制文件，压缩文件，电影，图片。。。）

`d 目录文件（蓝色）`

`b 设备文件 block device 设备文件，如硬盘U盘;`

`c 设备文件 字符设备文件，比如我们的终端tty1，打印机。`

`l symbolic link 即符号链接文件，又称软链接文件 （浅蓝色）`

`s socket 即套接字文件，用于实现两个进程进行通信`

`p 管道文件`

文件权限

- 读取（r）： 允许查看文件内容，显示目录列表
- 写入（w）： 允许修改文件内容，允许在目录中新建、删除、移动文件或者子目录
- 可执行（x）： 允许运行程序，切换目录
- 无权限（-）： 没有权限

而不同的人对文件的权限各不相同，在Linux中分为三种人：**文件拥有者，拥有者所在用户组中的其他用户，其他用户**

设置一个文件权限的时候需要设置3*3也就是9个权限,常用命令有chown,chmod,chattr 等等.....

我们这里用chmod举例，chmod有两种格式，下面以八进制形式举例

https://www.runoob.com/linux/linux-comm-chmod.html（具体教程）

```
chmod nnn(三位八进制数) 文件/目录
```

文件管理命令

touch 创建文件

`pwd 显示当前工作目录`

`cd 切换文件路径`

`ls 会列举出当前工作目录的内容（文件或文件夹）`

常用参数：

`-a ：列出当前目录全部的文件，包含开头为 . 的隐藏文件`

`-l ：列出文件的属性与权限等`

`mkdir 创建文件夹`

常用参数：

`-v 详细`

`-p 递归 目录`

`cp 复制文件`

常用参数：

`-f：强制覆盖目标文件`

`-i：若目标文件已经存在，覆盖时会先询问`

`-l：创建硬链接，而非复制文件本身`

`-p：连同文件的属性一起复制`

`-r：递归复制`

`-u ：若目标文件已经存在，且原文件比较新，才会覆盖`

`mv 移动文件`

常用参数：

`-f` ：如果目标文件已经存在，不会询问而直接覆盖

`-i` ：若目标文件已经存在，会询问是否覆盖

`-u` ：若目标文件已经存在，且原文件比较新，才会覆盖

`rm` 删除文件

常用参数：

`-r` 递归

`-f` force强制

`-v` 详细过程

`cat` 查看文件

常用参数：

`-n` 显示行号

`-A` 包括控制字符（换行符/制表符）

`vim` 编辑文件

Vim的使用

VIM 常用的有四个模式：

- 正常模式 (Normal-mode)
- 插入模式 (Insert-mode)
- 命令模式 (Command-mode)
- 可视模式 (Visual-mode)

正常模式

启动 VIM 后默认位于正常模式，一般用于**浏览**文件，也包括一些**复制、粘贴、删除**等操作。这时击键时，一些组合键就是 vim 的功能键，而不会在文本中键入对应的字符

在这个模式下，我们可以通过键盘在文本中快速移动光标。不论是什么模式，按一下 `<Esc>` 键 (有时可能需要按两下，插入模式按一下 `Esc` ，就会切换到正常模式，命令模式或者可视模式下执行完操作以后，就会自动进入正常模式，如果进入命令模式或者可视模式没有执行任何操作，按两下 `Esc` 即可 ）都会进入正常模式。

插入模式

按 `i,a` （较常用，i是在当前光标前插入，a是光标后） 键会进行插入模式。该模式启动以后，就会进入编辑状态，通过键盘输入内容

命令模式

在正常模式中，按下 `:` 键或者 `/` ，会进入命令模式。在命令模式中可以执行一些输入并执行一些 VIM 或插件提供的指令，就像在shell里一样。这些指令包括设置环境、文件操作、调用某个功能等等

命令	说明
:set nu	显示行号
:set nonu	取消行号
:n	定位到 n 行
/{目标字符串}	关键字查找
:起始行号,结束行号d	删除多行文本

可视模式

在正常模式按下 `v` ，可以进入可视模式。可视模式中的操作有点像拿鼠标进行操作，选择文本的时候有一种鼠标选择的即视感，有时候会很方便

以上是关于 VIM 四种模式的解读，我们在使用 VIM 操作文本的时候，编辑区底部一般都会显示当前处于什么模式下（插入模式会有 INSERT 提示，可视模式会有 VISUAL 或者 VISUAL LINE 的提示）

https://zhuanlan.zhihu.com/p/68111471

用户管理

su

su(switch user) 可以帮助我们完成用户的切换。

```
1 su - username
```

输入目标用户的密码即可切换至该用户

切换至 root 用户可以省去后面的参数

sudo

可以理解为windows中的以管理员身份运行

在命令前加上sudo即可，需要输入密码

执行指令时， 系统会先在 `/etc/sudoers` 中查找用户是否有运行 `sudo /` 目标程序的权限以及是否能免密执行

磁盘管理

Linux 磁盘管理好坏直接关系到整个系统的性能问题

Linux 磁盘管理常用三个命令为 **df**、**du** 和 **fdisk**

- **df**（英文全称：disk free）：列出文件系统的整体磁盘使用量
- **du**（英文全称：disk used）：检查磁盘空间使用量
- **fdisk**：用于磁盘分区

不做详细的介绍，感兴趣可以课下自己看

网络管理

- ping命令 – 测试主机间网络连通性
- netstat命令 – 显示网络状态
- ifconfig命令 – 显示或设置网络设备

网络配置文件

配置文件	说明
/etc/sysconfig/network	系统网络配置文件，包含了主机最基本的网络信息用于系统启动
/etc/sysconfig/network-scripts/ifcfg-ethX	以太网接口配置文件
/etc/sysconfig/network-scripts/route-ethX	以太网接口的静态路由配置文件
/etc/hosts	完成主机名映射为IP地址的静态解析功能
/etc/resolv.conf	配置域名服务客户端的配置文件，用于指定域名服务器的位置
/etc/host.conf	配置域名服务客户端的控制文件

进程管理

每当你在 Linux 中执行一个命令，它都会创建，或启动一个新的进程。比如，当你尝试运行命令 `ls -l` 来列出目录的内容时，你就启动了一个进程。如果有两个终端窗口显示在屏幕上，那么你可能运行了两次同样的终端程序，这时会有两个终端进程。

进程和文件一样都有一个唯一的标识： `PID` ，我们可以通过 `pid` 来追踪进程

如果要查看当前的进程可以使用ps -au命令（列出详细资源），一般-A即可（列出所有进程）

```
1 USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
```

-
  - `USER` : 行程拥有者
-
  - `PID` : pid
  - `%CPU` : 占用的 CPU 使用率
  - `STAT` : 该行程的状态
  - `START` : 行程开始时间
  - `TIME` : 执行的时间
  - `COMMAND` :所执行的指令

常见STAT

字符	说明
D	无法中断的休眠状态 (通常 IO 的进程)
R	正在执行中
S	静止状态
T	暂停执行
Z	不存在但暂时无法消除
W	没有足够的记忆体分页可分配

`kill [pid]` ： 杀死进程(默认参数-15)

也可以加参数，常用的是 `kill -9` 即强制终止

`top/htop`： 动态显示进程信息

云服务器

云服务器的简单定义为：在云计算环境中运行的**虚拟服务器**，可由无限用户按需访问

安全性：云计算服务器不会因用户过多而过载，任何软件问题都**与本地环境隔离**

灵活性：通过云服务器工作可让用户从不同位置访问同一服务器，从而实现工作人员灵活性

处理能力：用于云计算的服务器链接在一起，共享各种工作负载的计算能力，使得计算能力大幅提升

最重要的来了，云服务器有自己的公网IP地址！！！可以支持任何联网用户访问其中提供的服务

> 由于目前还没讲计算机网络，可以简单理解为云服务器**拥有世界上的身份证（所有国家通用）**，也就是不论在哪都知道它（可以访问），而没有公网IP只有私有IP的计算机就是拥有某个国家的身份证，在这个国家里其他人可以访问它，但是在世界上无人知晓（不能访问到）

部署go项目

重点来了！！！

我们大多数人使用的都是windosws系统，使用go直接编译出来的是exe文件。exe文件是没法直接在Linux系统下运行的，就好像你电脑的软件不能直接拿到手机上运行一样

这里我们就有**两种办法**

直接在开发机上编译，再传输过去，或者，把源码传输到服务器上，然后在linux服务器上编译

第一种方法通常称为交叉编译，go 语言原生就对交叉编译支持非常好，只需要我们改一下go的参数即可

还记得 `go env` 命令吗，查看go的环境变量

看 GOOS 和 GOARCH 这两个参数，GOARCH 一般是 amd64，这个就是目标平台的 cpu 架构，现在一般都是这个架构

然后看 GOOS，这个就是目标平台的操作系统

```
1 go env -w GOOS=linux
```

再查看一下，然后再编译即可

接下来需要我们把它传到服务器上，我这里使用scp命令

```
1 scp main root@165.154.145.135:/home/ubuntu/hello
```

root为我的用户名，@后面是我的服务器地址，冒号后面为目录的绝对路径

ps：或者用云服务商自带的文件上传功能，也可以用一些软件来上传（我使用的是WinScp）

如果你是一个web程序，需要一直挂着的话，这种情况当我们关闭终端时程序也会关闭，我们可以使用

```
nohup ./main &
```

让它在后台一直保持运行

其中

- &的作用是让它保持后台运行，但如果关闭终端依然会被杀掉。
- nohup的作用是保证它在后台不会被杀掉。
- 使用fg可以让一个后台程序转到前台
- 使用disown可以让已经在后台的任务不挂掉

前后台切换等操作：https://blog.csdn.net/ARPOSPF/article/details/99310770

第二种部署的方法，直接传main.go文件到服务器，然后在服务器上使用go build。不过这种方法需要服务器也装有go的环境，但不用担心本地依赖库和服务器不同或者版本不同造成的各种问题

docker

为什么要在这里介绍docker呢，是因为上面说了第二种部署需要服务器有go环境，安装环境是一个很麻烦的事情，而且我们在服务器上其实对go环境的需求不大，这时候docker就很有作用了

由于本节课并不细讲docker，只是简单结合部署讲个docker的小例子（冰山一角），感兴趣的同学可以自己下去了解学习，后续的课程也会专门细讲docker

虚拟机与容器

虚拟机（VM）是计算机系统的仿真。简而言之，它可以在一台计算机的硬件上运行看似多台单独的计算机。操作系统（OS）及其应用程序共享单个主机服务器或主机服务器池的硬件资源。每个VM都需要自己的基础OS，并且硬件已虚拟化

使用**容器**，可以像虚拟机（VM）一样虚拟化基础计算机，而无需虚拟化OS，而docker是一款优秀的**容器集成软件**

由于Docker的设计是为独立应用封装容器，我们可以很轻松地通过Docker容器组装需要的服务系统体系，整个过程**省去了大量应用程序搭建和调试的时间**。使用Docker时，用很小的修改就能代替我们之前所做的大量工作，节约了大量时间

废话不多说，下面直接进入实操

实操环节

1.安装docker

```
apt install docker.io
```

2.拉取镜像（如果把容器理解为应用程序运行的虚拟环境，那么镜像就可以被看作这个环境的**持久化副本**）

```
docker pull golang
```

3.启动一个容器（基于前面拉取的镜像）

```
docker run --rm -it --name go-http-demo golang bash
```

上面这个命令用镜像 `golang` 创建了一个名为 `go-http-demo` 的容器，在容器中创建了一个 `Bash` 会话。 `--rm` 选项指定容器退出后自动移除容器

4.编写go文件

```go
package main

import (
    "context"
    "fmt"
    "log"
    "net/http"
    "os"
    "os/signal"
    "syscall"
)

func main() {
    mux := http.NewServeMux()
    mux.Handle("/", &helloHandler{})

    server := &http.Server{
        Addr:    ":8080",
        Handler: mux,
    }

    // 创建系统信号接收器
    done := make(chan os.Signal)
    signal.Notify(done, os.Interrupt, syscall.SIGINT, syscall.SIGTERM)
    go func() {
        <-done

        if err := server.Shutdown(context.Background()); err != nil {
            log.Fatal("Shutdown server:", err)
        }
    }()

    log.Println("Starting HTTP server...")
    err := server.ListenAndServe()
    if err != nil {
        if err == http.ErrServerClosed {
            log.Print("Server closed under request")
        } else {
            log.Fatal("Server closed unexpected")
        }
    }
}

type helloHandler struct{}

func (*helloHandler) ServeHTTP(w http.ResponseWriter, r *http.Request) {
    fmt.Fprintf(w, "Hello World! ")
}
```

如果想使用gin框架部署看这篇文章：https://developer.aliyun.com/article/1157900

5.正式部署

```
docker run --rm -it --name go-http-demo -v /home/ubuntu/hello:/go/src -p 8081:8080 golang
```

`-v` 选项允许我们挂载多个本地目录或者数据卷到容器中，更改会在容器内外相互同步，上面的命令将 `/home/ubuntu/hello` 挂载到了容器的 `/go/src` 。

`-p` 指定主机和容器的端口映射，因为代码里 `HTTP` 服务是监听 `8080` 端口的，所以我们将主机的 `80081` 端口和容器的 `8080` 端口做了映射

6.容器内运行

```
1 go run hello.go
```

作业

LV1（重点做）

编写一个打印 Hello World 的项目，并且部署到你的服务器上（两种方法都可以试一下），任何将服务器ip发送给导师

ps：实在不想买服务器就开个Linux系统的虚拟机，本地访问交截图吧（第一次各大云服务厂商都会有试用）

LV2（选做）

尽可能多的完成 Linux 基础入门 中的实验，然后提交截图