

树莓派开发

24 树莓派与云服务器

申请苏宁公有云

□ 访问苏宁公有云平台



申请苏宁公有云

□注册-登录

登录



苏宁公有云

本系统暂时只支持Firefox、Chrome浏览器

zhjh@seu.edu.cn

.....

登录

[忘记密码？](#)

[注册](#)

申请苏宁公有云

- 在苏宁公有云界面上，选择：控制台|其他|免费试用
- 按照菜单提示，完成免费试用申请，得到**200元**试用费

免费试用

登录后，您可以申请免费试用。我们会在1~2个工作日内审核您提供的资料，并根据您提供的资料为您的账户赠送不同的金额以供您试用，如有!

注册与登录

 →

实名认证

 →

填写申请信息

 →

人工审核

 →

开始试用

立即申请

查看我的申请

- 在1-2个工作日后，申请或许得到批准，可获得约**200元**试用费，试用费用实时计算，没有使用期限。

- 然后对申请到的云服务器，进行配置。

申请苏宁公有云

□ 免费试用的资源

当前配额使用情况

×

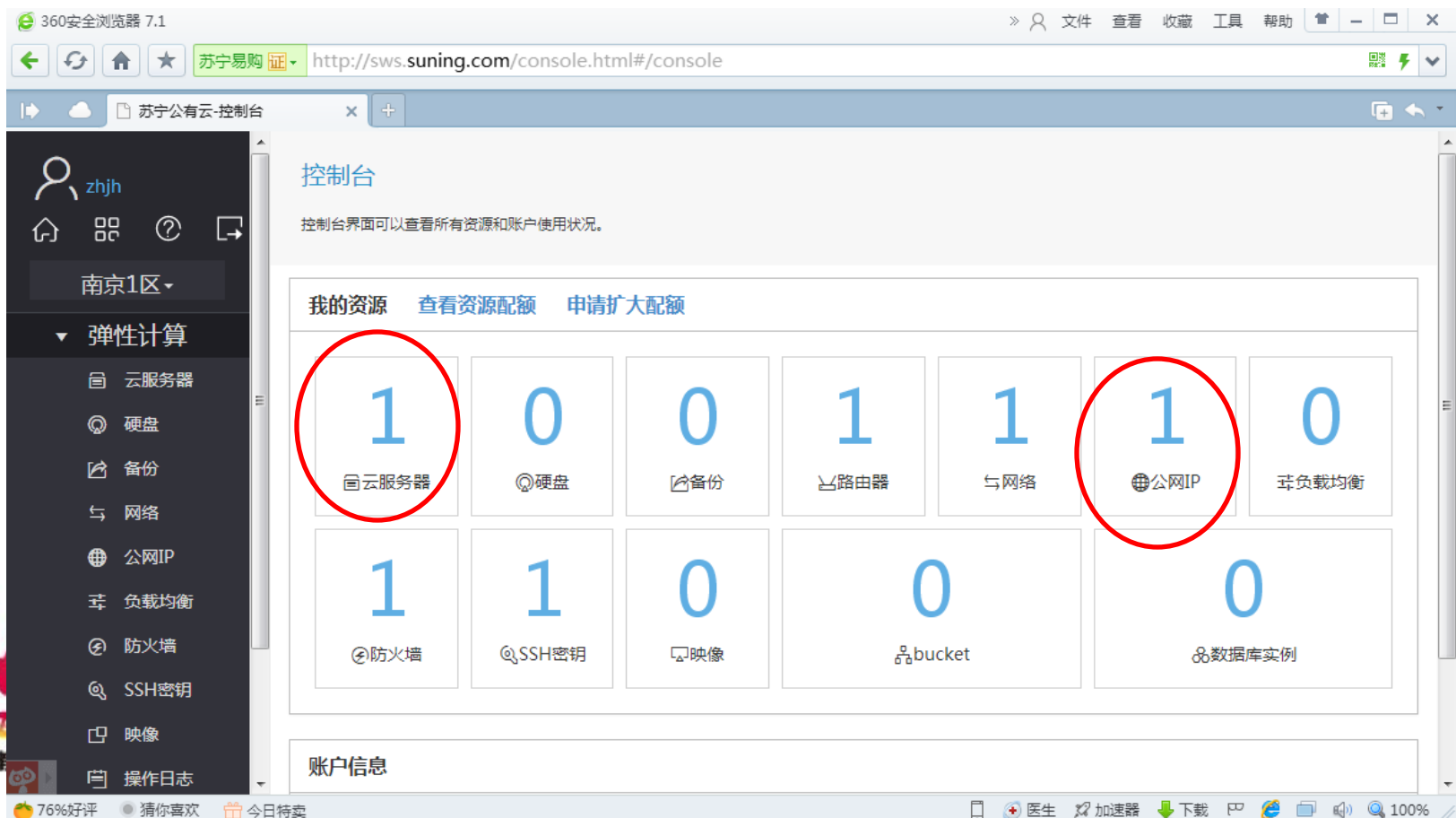
*限定配额是为了防止资源滥用，如果有额外需求您可以申请扩大配额

资源类型	已使用	资源配额
云服务器(个)	0	10
公网IP(个)	0	2
SSH密钥(个)	0	无限制
硬盘(个)	0	10
路由器(个)	0	3
映像(个)	0	无限制
防火墙(个)	0	无限制
网络(个)	0	15

资源类型	已使用	资源配额
负载均衡(个)	0	无限制
Paas应用(个)	0	10
PostgreSQL(个)	0	5
Redis(个)	0	5
MongoDB(个)	0	5
内存(GB)	0	10240
带宽(Mbps)	0	无限制

配置云服务器

- 根据云服务商的菜单提示，完成以下配置：
 - 完成申请，配置一个最低配置的云服务器（最少费用）
 - 确定一个用户名和密码（一般密码或SSH密码，注意两者的区别）
 - 申请一个公网IP，并与自己的服务器绑定，记住IP地址



配置云服务器

- ❑ 选择云服务器下自己的服务器（可以有多个，玩大数据用）
- ❑ 点击“启动”：自己的服务器启动后，可以看到：
 - ❑ 服务器状态为：“运行中”
 - ❑ 公网IP，地址是：218.2.204.233
 - ❑ CPU1个、内存1G，没有硬盘（用作数据盘，没钱买。有30G的系统也可以了。）
 - ❑ 数据库可以直接装在系统盘上，不需要申请另外的数据库。

新建	启动	重启	重置	关机	更多操作▼	↺	请输入搜索关键字	Q
<input checked="" type="checkbox"/>	云服务器ID	云服务器名称	状态 ▼	公网IP	CPU(核)	内存(G)	硬盘(G)	创建时间
<input checked="" type="checkbox"/>	v-9m4Dzq7YBNL 	MyServer	运行中	218.2.204.233	1	1	0	2015-06-29 11:17:21

配置云服务器

❑ 在关机状态下，选择云服务器下自己的服务器，选择重置：我们 · 始于1993年

新建	启动	重启	重置	关机	更多操作▼	↺	请输入搜索关键字	Q
<input checked="" type="checkbox"/>	云服务器ID	云服务器名称	状态 ▼	公网IP	CPU(核)			
<input checked="" type="checkbox"/>	v-9m4Dzq7YBNL	MyServer	已关机	218.2.204.233	1	1		

- ❑ 可以选择登录密码方式
- ❑ 二者不可兼得

重置服务器

该功能将重置您的云服务器的登录方式。加载SSH密钥以后，将禁止密码登录。

登录方式 ☒ SSH密钥 ☐ 密码

用户名 root

SSH密钥

确定

取消

通过IE登录云服务器

□ 点击云服务器ID，可以看到这台服务器的更多情况：

□ 点击远程登录

基本属性

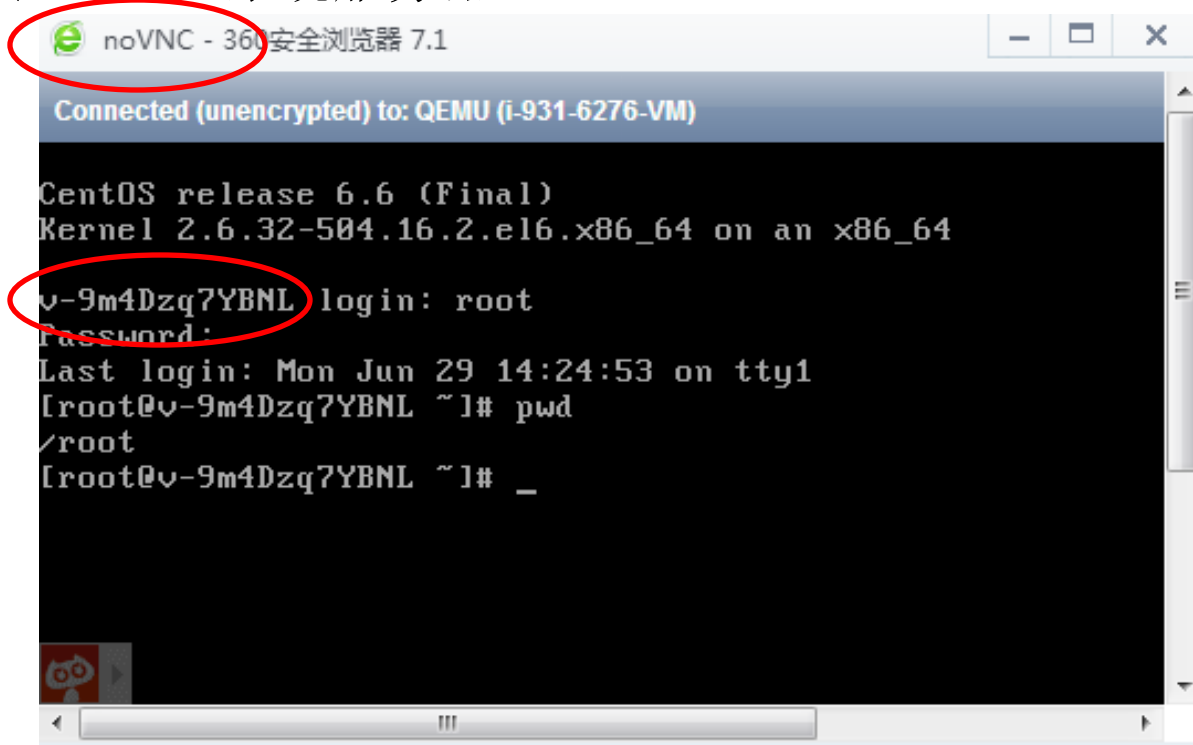
主机ID	v-9m4Dzq7YBNL	主机名称	MyServer	运行状态	运行中
远程桌面	登录远程桌面	CPU核数	1核	初始密码	zhjh6453
主机内存	1024MB	系统硬盘	30GB	主机单价	0.1元/小时 查看消费详情
创建时间	2015-06-29 11:17:21				
描述	无				

关联资源

公网IP	218.2.204.233	网络	n-JKskzq7YW4H(192.168.0.235)	映像	CentOS 6.5 64bit
				SSH密钥	k-edFXzq7YiXO
硬盘0GB	未加载硬盘 加载硬盘				

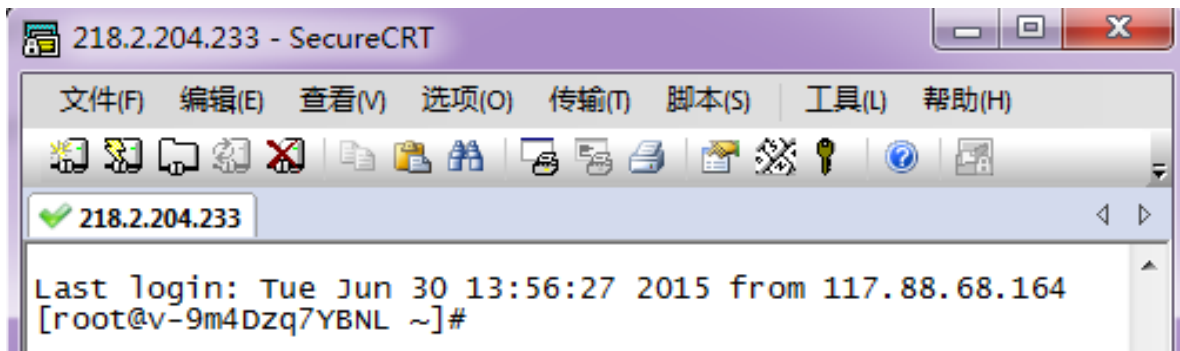
在WIN7下使用VNC登录云服务器

- 自动使用VNC远程登录到服务器上：
 - 缺省用户名：root，使用自己定义的密码。
 - 登录后的当前目录是：/root
 - 操作系统的版本是CentOS6.6，服务器主机用户是申请时分配的名字
 - 此时，就像使用本地机器一样，使用这台云服务器了。
 - 目前苏宁云不提供windows系统服务器



在WIN7下使用VNC登录云服务器

- 在windows下，可以使用支持SSH的VNC远程仿真软件登录到服务器上：
- 运行SecureCRT，输入服务器IP、用户名和密码。



在树莓派上登录云服务器

- ❑ (1) 在WIN7下，先使用VNC远程仿真软件登录到树莓派
- ❑ (2) 在树莓派仿真情况下，登入云服务器：

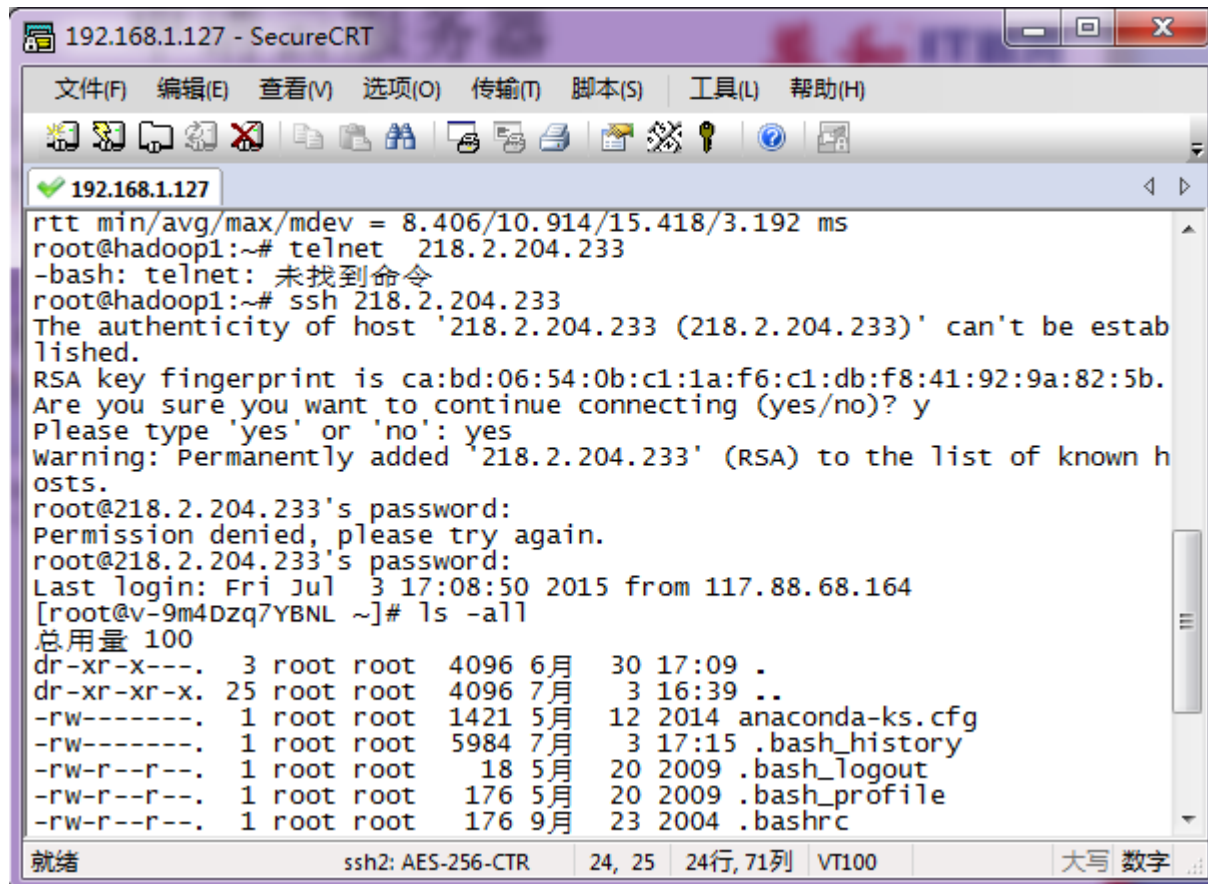
❑ 在树莓派下，使用：

ssh 218.2.204.233

❑ 注意：

❑ 在树莓派上我的用户名是root，所以，登录云也是root，对方问的密码也是root

❑ 链接成功！



```
192.168.1.127 - SecureCRT
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)
192.168.1.127
rtt min/avg/max/mdev = 8.406/10.914/15.418/3.192 ms
root@hadoop1:~# telnet 218.2.204.233
-bash: telnet: 未找到命令
root@hadoop1:~# ssh 218.2.204.233
The authenticity of host '218.2.204.233 (218.2.204.233)' can't be established.
RSA key fingerprint is ca:bd:06:54:0b:c1:1a:f6:c1:db:f8:41:92:9a:82:5b.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '218.2.204.233' (RSA) to the list of known hosts.
root@218.2.204.233's password:
Permission denied, please try again.
root@218.2.204.233's password:
Last login: Fri Jul 3 17:08:50 2015 from 117.88.68.164
[root@v-9m4Dzq7YBNL ~]# ls -all
总用量 100
dr-xr-x---. 3 root root 4096 6月 30 17:09 .
dr-xr-xr-x. 25 root root 4096 7月 3 16:39 ..
-rw-----. 1 root root 1421 5月 12 2014 anaconda-ks.cfg
-rw-----. 1 root root 5984 7月 3 17:15 .bash_history
-rw-r--r--. 1 root root 18 5月 20 2009 .bash_logout
-rw-r--r--. 1 root root 176 5月 20 2009 .bash_profile
-rw-r--r--. 1 root root 176 9月 23 2004 .bashrc
```

在树莓派上登录云服务器

- ❑ 直接在树莓派的LX终端上,执行ssh 218.2.204.233
- ❑ 结果也是一样的。



```
root@v-9m4Dzq7YBNL:~  
文件(E) 编辑(E) 标签(T) 帮助(H)  
root@hadoop1: ~# ssh 218.2.204.233  
root@218.2.204.233's password:  
Permission denied, please try again.  
root@218.2.204.233's password:  
Last login: Fri Jul 3 17:25:15 2015 from 117.88.68.164  
[ root@v-9m4Dzq7YBNL ~]#
```

注意：这是用树莓派的root用户名登陆云服务器，云服务器上正好也有root用户

如果你用的是树莓派的pi用户，就怎么也等不上了，因为没有pi这个用户

不用root登录的方法是：ssh -l xxxxx 218.2.204.233 其中，xxxxx是云服务器上已有的用户

把应用搬到云上去

- ❑ CentOS的安装命令: yum
- ❑ yum 是linux系统的自动安装系统, yum install 仅安装指定的软件
- ❑ yum (全称为 Yellow dog Updater, Modified),能够从指定的服务器自动下载gz包并且安装,可以自动处理依赖性关系,并且一次安装所有依赖的软体包,无须繁琐地一次次下载、安装。
- ❑ yum提供了查找、安装、删除某一个、一组甚至全部软件包的命令,而且命令简洁而又好记。
- ❑ yum的命令形式一般是如下: yum [options] [command] [package ...]
其中的[options]是可选的
- ❑ 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]为所要进行的操作,
- ❑ [package ...]是操作的对象。

为云服务器安装Tomcat6

□ 在centos中安装tomcat6

□ 1) 通过yum自动安装tomcat和dependences

```
root@*****]# yum install tomcat6
```

```
root@*****]# service tomcat6 start
```

```
root@*****]# chkconfig tomcat6 on
```

```
root@*****]# yum install tomcat6-webapps
```

```
root@*****]# yum install tomcat6-admin-webapps
```

□ 如果访问http:// 218.2.204.233:8080/访问不了，那大多是防火墙已经用了8080端口，解决方法如下：

```
iptables -A INPUT -p tcp --dport 8080 -j ACCEPT
```

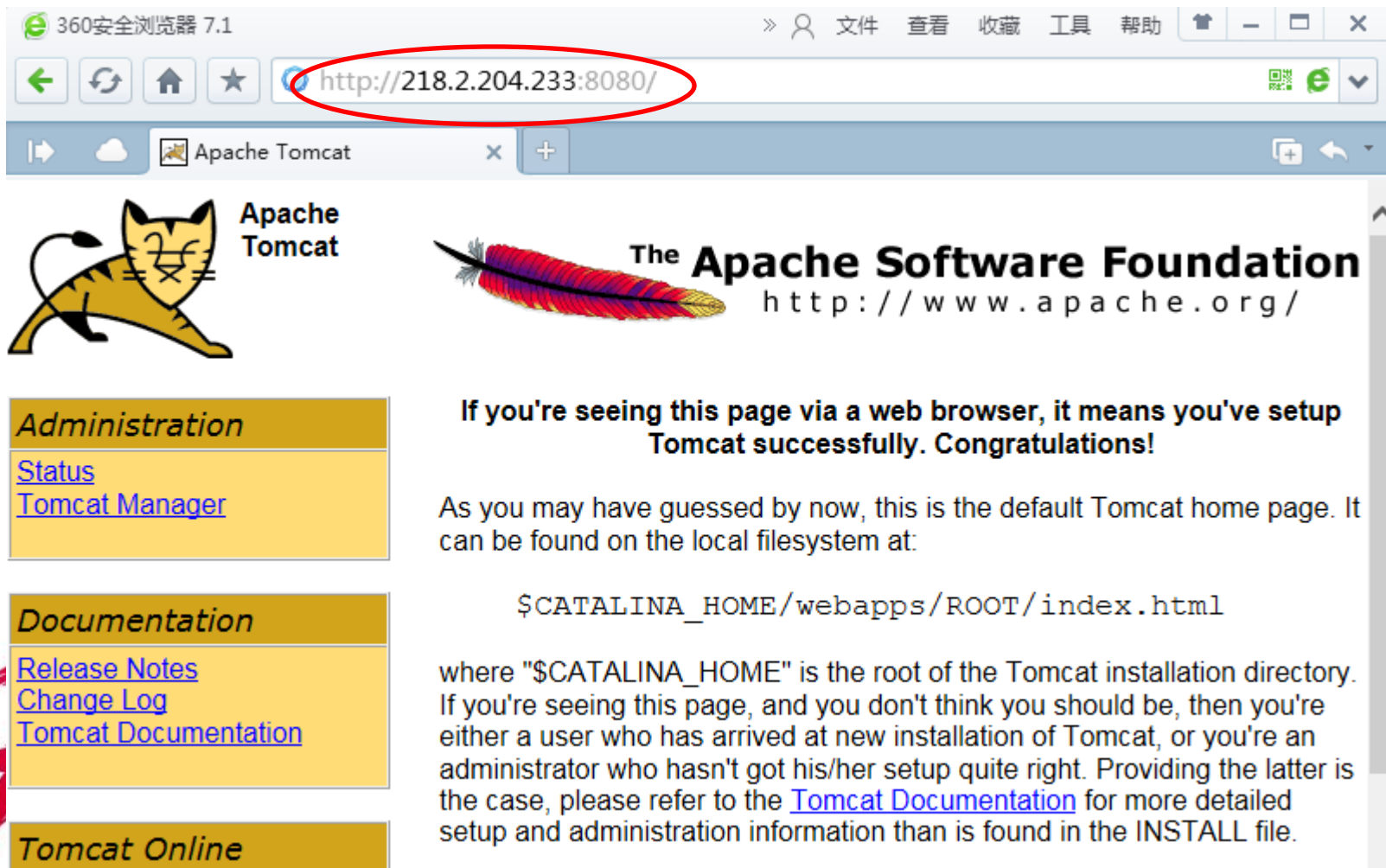
```
iptables -A OUTPUT -p tcp --sport 8080 -j ACCEPT
```

□ 如果安装正确的话，可以在browser中看到tomcat的默认的page。

□

为云服务器安装Tomcat6

- ❑ 2) 访问Tomcat6
- ❑ 在IE浏览器（不是VNC）上输入服务器IP地址和Tomcat6的8080端口



为云服务器安装Tomcat6

❑ 3) 配置tomcat（如果不用manager，可不用此修改）

❑ 配置文件：/etc/tomcat6/tomcat6.conf + /etc/sysconfig/tomcat6

❑ tomcat home目录：/usr/share/tomcat6

❑ 配置tomcat为admin和manager用户：

❑ 修改文件/usr/share/tomcat6/conf/tomcat-users.xml

```
<!-- The host manager webapp is restricted to users with role "admin" -->
```

```
<!--<user name="tomcat" password="password" roles="admin" />-->
```

```
<!-- The manager webapp is restricted to users with role "manager" -->
```

```
<user name="tomcat" password="password" roles="manager,admin" />
```

```
</tomcat-users>
```

❑ 此时可以访问[http:// 218.2.204.233:8080/manager/html](http://218.2.204.233:8080/manager/html)和[http:// 218.2.204.233:8080/host-manager/html](http://218.2.204.233:8080/host-manager/html)来管理tomcat server和host。

直接部署Tomcat的应用

□ 方法1：【使用控制台部署】

□ 访问Http: //218.2.204.233:8080，并通过Tomcat Manager登录，进入部署界面即可。

□ 此方法需要前述修改tomcat manager的用户名及密码

□ 方法2：【利用Tomcat自动部署】

□ 将应用程序复制到Tomcat的webapps路径下，Tomcat启动时将自动加载。

直接部署Tomcat的应用

□ Tomcat的webapps路径在哪里？



If you're seeing this page via a web browser, it means you've setup Tomcat successfully.
Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

`$CATALINA_HOME/webapps/ROOT/index.html`

- 根据Tomcat第一页的提示，应该在\$CATALINA_HOME指定的目录下
- 用echo \$CATALINE_HOME看一下
- 直接查看tomcat6的安装目录/etc/tomcat6/tomcat6.conf，可以看到：
- CATALINE_HOME的值是：
/usr/share/tomcat6



```
# Where your java installation lives
#JAVA_HOME="/usr/lib/jvm/java-1.5.0"

# Where your tomcat installation lives
CATALINA_BASE="/usr/share/tomcat6"
CATALINA_HOME="/usr/share/tomcat6"
JASPER_HOME="/usr/share/tomcat6"
CATALINA_TMPDIR="/var/cache/tomcat6/temp"
```

直接部署Tomcat的应用

□ 在/usr/share/tomcat6目录下再看一下：

```
[root@v-9m4Dzq7YBNL tomcat6]# ls -all
total 12
drwxrwxr-x.  3 root root  4096 Jun 29 14:32 .
drwxr-xr-x. 120 root root  4096 Jun 29 16:41 ..
drwxr-xr-x.  2 root root  4096 Jun 29 14:32 bin
lrwxrwxrwx.  1 root tomcat  12 Jun 29 14:32 conf -> /etc/tomcat6
lrwxrwxrwx.  1 root root   23 Jun 29 14:32 lib -> /usr/share/java/tomcat6
lrwxrwxrwx.  1 root root   16 Jun 29 14:32 logs -> /var/log/tomcat6
lrwxrwxrwx.  1 root root   23 Jun 29 14:32 temp -> /var/cache/tomcat6/temp
lrwxrwxrwx.  1 root root   24 Jun 29 14:32 webapps -> /var/lib/tomcat6/webapps
lrwxrwxrwx.  1 root root   23 Jun 29 14:32 work -> /var/cache/tomcat6/work
[root@v-9m4Dzq7YBNL tomcat6]# _
```

□ 实际的webapps在/var/lib/tomcat6目录下

□ 再去看看：终于找到了。

```
[root@v-9m4Dzq7YBNL tomcat6]# cd /var/lib/tomcat6
[root@v-9m4Dzq7YBNL tomcat6]# ls -all
total 12
drwxrwxr-x.  3 root root  4096 Jun 29 14:32 .
drwxr-xr-x. 29 root root  4096 Jun 30 08:49 ..
drwxrwxr-x.  7 root tomcat 4096 Jun 29 14:41 webapps
[root@v-9m4Dzq7YBNL tomcat6]# _
```

直接部署Tomcat的应用

□ 在/var/lib/tomcat6目录下再看一下：

```
[root@v-9m4Dzq7YBNL tomcat6]# cd webapps
[root@v-9m4Dzq7YBNL webapps]# ls -all
total 28
drwxrwxr-x. 7 root tomcat 4096 Jun 29 14:41 .
drwxrwxr-x. 3 root root 4096 Jun 29 14:32 ..
drwxrwxr-x. 5 root tomcat 4096 Jun 29 14:41 examples
drwxrwxr-x. 5 root tomcat 4096 Jun 29 14:37 host-manager
drwxrwxr-x. 5 root tomcat 4096 Jun 29 14:37 manager
drwxrwxr-x. 3 root tomcat 4096 Jun 29 14:41 ROOT
drwxrwxr-x. 5 root tomcat 4096 Jun 29 14:41 sample
[root@v-9m4Dzq7YBNL webapps]# _
```

□ 已经有几个应用实例了。看一下examples的例子。

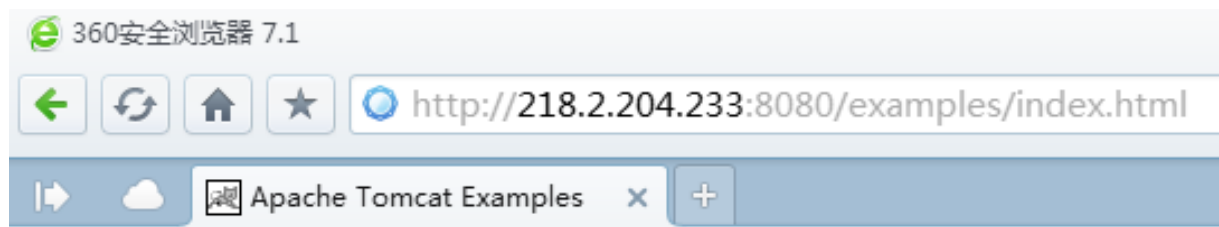
直接部署Tomcat的应用

□ 看一下examples的例子：

```
[root@v-9m4Dzq7YBNL webapps]# cd examples
[root@v-9m4Dzq7YBNL examples]# ls -all
total 24
drwxrwxr-x.  5 root tomcat 4096 Jun 29 14:41 .
drwxrwxr-x.  7 root tomcat 4096 Jun 29 14:41 ..
-rw-r--r--.  1 root tomcat 1127 May 13 02:07 index.html
drwxrwxr-x. 21 root tomcat 4096 Jun 29 14:41 jsp
drwxrwxr-x.  3 root tomcat 4096 Jun 29 14:41 servlets
drwxrwxr-x.  7 root tomcat 4096 Jun 29 14:41 WEB-INF
[root@v-9m4Dzq7YBNL examples]# _
```

□ 我们知道，在IE上直接运行就 O K！

□ 结果如下：



Apache Tomcat Examples

- [Servlets examples](#)
- [JSP Examples](#)

把自己的应用搬到云上

□ 我们自己的WEB应用搬到云上的例子

□ 方法2：【利用Tomcat自动部署】

□ 将应用程序直接复制到Tomcat的webapps路径下，
Tomcat启动时将自动加载。

□ 复制文件：通常使用scp命令（涉及SSH）

□ 另一个简单的工具：

□ 安装lrzsz：

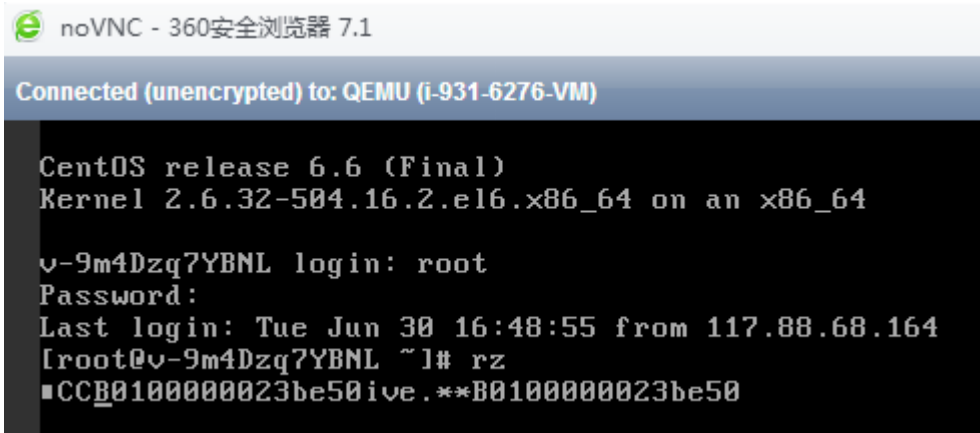
yum -y install lrzsz

□ 现在就可以正常使用rz、sz命令上传、下载数据了，rz是传到linux，sz是从linux下载到windows。

□ 输入：rz，出现本地（windows）的文件选择菜单

把自己的应用搬到云上

- Lrzs需要配合SecureCRT使用，因为后者是图形仿真
- 在noVNC上运行rz，出现错误如下：



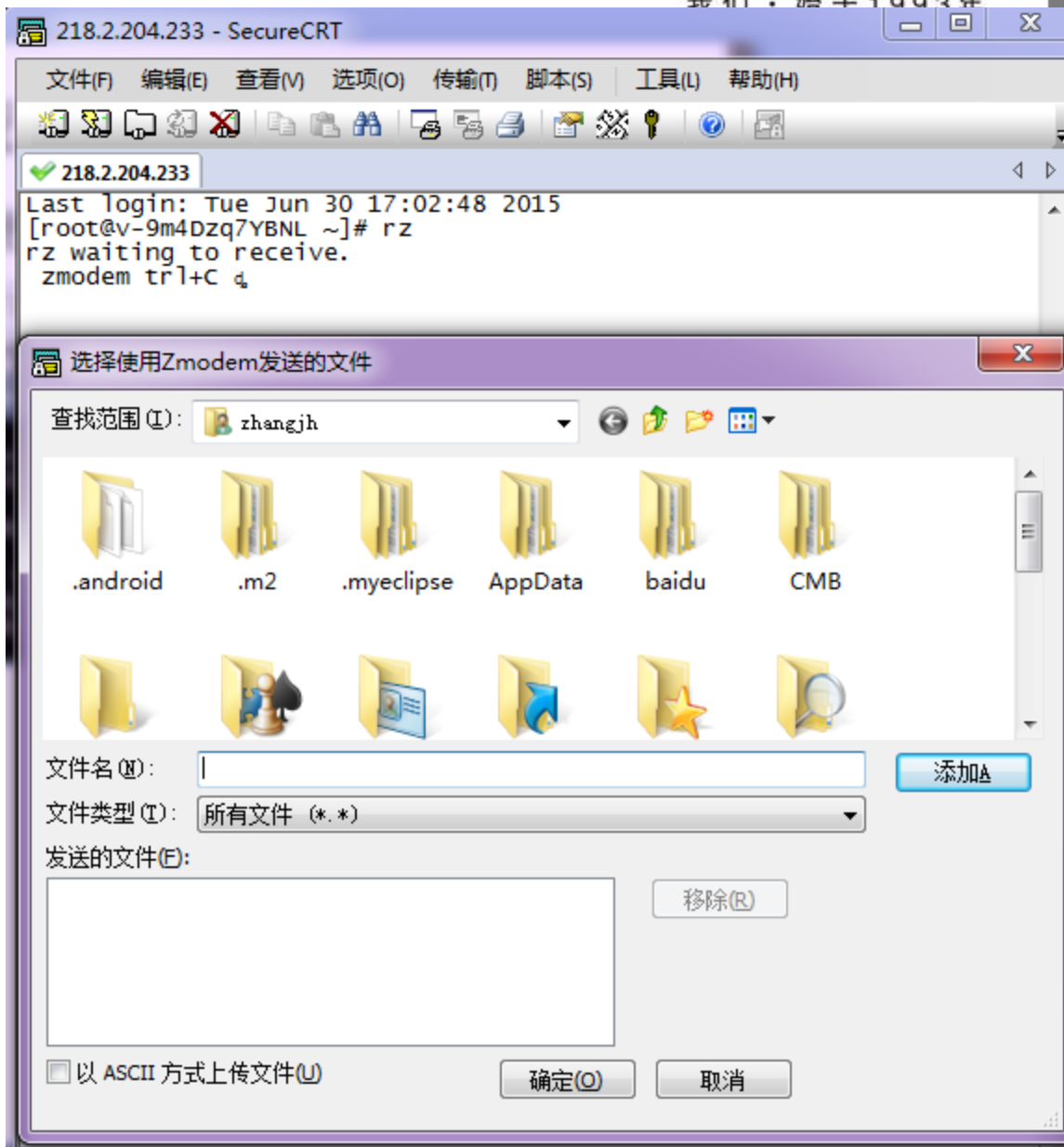
```
noVNC - 360安全浏览器 7.1
Connected (unencrypted) to: QEMU (i-931-6276-VM)

CentOS release 6.6 (Final)
Kernel 2.6.32-504.16.2.el6.x86_64 on an x86_64

v-9m4Dzq7YBNL login: root
Password:
Last login: Tue Jun 30 16:48:55 from 117.88.68.164
[root@v-9m4Dzq7YBNL ~]# rz
■CCB0100000023be50ive.**B0100000023be50
```


把自己的应用搬到云上

□ 而在SecureCRT运行rz，则出现本地文件选择界面：



把自己的应用搬到云上

❑ 如果需要把一个项目放到webapps时，可以将该项目文件压缩为一个zip包，用rz方式，送到云服务器的当前目录下。

❑ 在当前目录下，执行：

```
mv ****.zip /var/lib/tomcat6/webapps
```

❑ 为该项目创建一个目录

```
mkdir **** （假定为file123）
```

❑ 把该压缩文件移动到新目录下

❑ 在新目录下，使用解压缩命令：

```
unzip ****.zip
```

❑ 现在需要重启tomcat。简单的方法就是重启服务器。

❑ 在IE上看看是否能看到这个项目？

❑ 注意：项目地址： http:

```
//218.2.204.233:8080/file123/zhuce.html
```

把自己的应用搬到云上

萬和® IT教育
因专业而精彩
我们，始于1999年

□ OK!



商务九州

www.shangwujiuzhou.com.cn

因为用心，所以专业

首 页 | 电 脑 城 | 通 讯 城 | 数 码 港 | 女 人 街 | 家 居 城 | 汽 车 城 | 影 视 区

会员登录

会员名:

密 码:

请按[这里](#)取回密码

分类检索

【 电脑城 】

笔记本	台式机
二手电脑	DIY区
配 件	掌上电脑

【 通讯城 】

GSM手机	CDMA手机
对讲机	小灵通

【 数码港 】

您的位置: 会员注册

请如实填写以下信息，您的资料将严格保密。(带*号的为必填内容)

会员用户名:	<input type="text"/>	<input type="button" value="检测用户名"/>	*
密 码:	<input type="password"/>		*
再次输入密码:	<input type="password"/>		*
电子邮件:	<input type="text"/>		*
确认电子邮件:	<input type="text"/>		*
真实姓名:	<input type="text"/>		*
性 别:	<input type="radio"/> 男 <input type="radio"/> 女		*
证件类型:	<input type="text" value="- 请选择 -"/>		*
证件号码:	<input type="text"/>		*
详细地址:	<input type="text" value="- 请选择 -"/>	<input type="text"/>	*
邮政编码:	<input type="text"/>		*
联系电话:	<input type="text"/>		*



在云上开发一个小应用

□ 把树莓派的CPU温度传到云上，用手机或IE可以监控。

□ 需求：

- 1、在树莓派上，将CPU的温度上传到云上；
- 2、在Web服务器端，接收上传的数据；
- 3、在Web服务器的HTML网页中，实现将上传的温度数据，用图标曲线的形式画出来。

有关数据上传与接收

□ Web Socket的方式有很多种，目前比较流行、简单的传输协议和数据格式是

□ 发送方：

```
HttpClient client = new HttpClient();
```

```
PostMethod method = new PostMethod(URL); //具体method里面还可以设置一下编码，header之类的
```

//1. 第一种方式，基于Content-Type='multipart/form-data'形式的表单

```
Part[] parts = ...; //FilePart和StringPart都可以放进去
```

```
method.setRequestEntity(new MultipartRequestEntity(parts,  
method.getParams()));
```

//2. 第二种方式，普通表单

```
NameValuePair[] pairs = ...; //纯参数了，键值对
```

```
method.addParameters(pairs);
```

```
client.executeMethod(method);
```



有关数据上传与接收

接收方:

```
private void parseRequest(HttpServletRequest request) throws Exception {  
    boolean isMultipart = ServletFileUpload.isMultipartContent(request);  
    if (isMultipart) {  
        DiskFileItemFactory factory = new DiskFileItemFactory();  
        ServletFileUpload upload = new ServletFileUpload(factory);  
        List items = upload.parseRequest(request);  
        for (int i = 0; i < items.size(); i++) {  
            FileItem item = (FileItem) items.get(i);  
            if (!item.isFormField()) {  
                //文件数据  
            } else {  
                //普通表单数据  
            }  
        }  
    }  
}
```



有关数据上传与接收

接收方:

```
} else {  
    Enumeration en = request.getParameterNames();  
    while (en.hasMoreElements()) {  
        String paramName = (String) en.nextElement();  
        String paramValue = request.getParameter(paramName);  
    }  
}  
}
```

有关在服务器上建立的一个简单网站

□ 建立网站的方法很多，不外是HTML5+CSS+JS

应用程序框架结构：

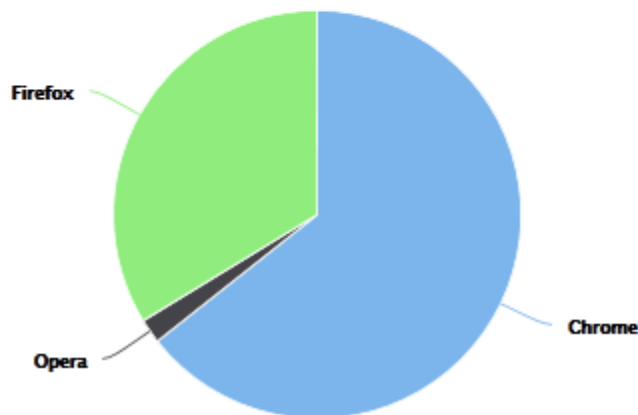
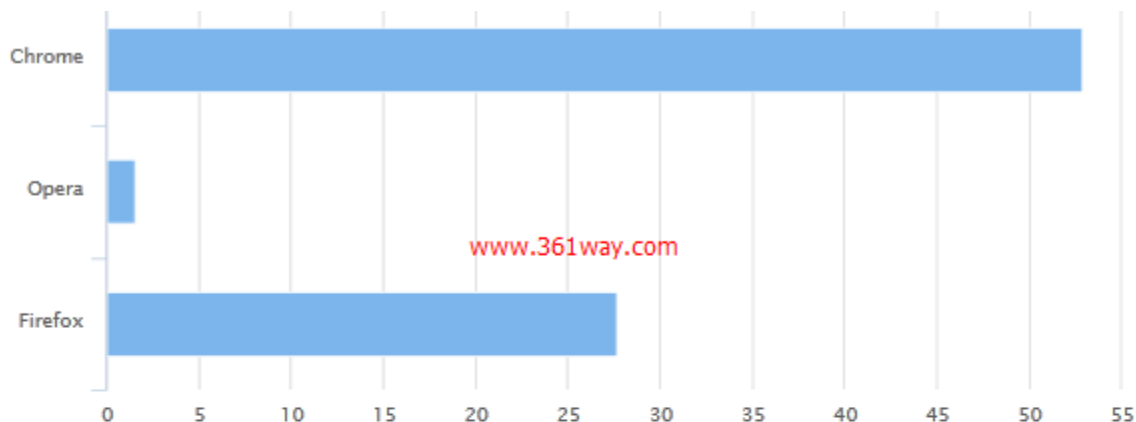
[root@361way chartkick]# tree

```
.
├── run.py
├── static
│   ├── chartkick.js
│   ├── highcharts.js
│   └── jquery.min.js
├── templates
└── index.html
```


有关在服务器上建立的一个简单网站

- [Flask](#)是一个轻量级的Web应用框架，使用Python编写。基于 WerkzeugWSGI工具箱和 Jinja2模板引擎。使用 BSD 授权。
- flask + chartkick 需要先安装[chartkick.py](#) 模块。

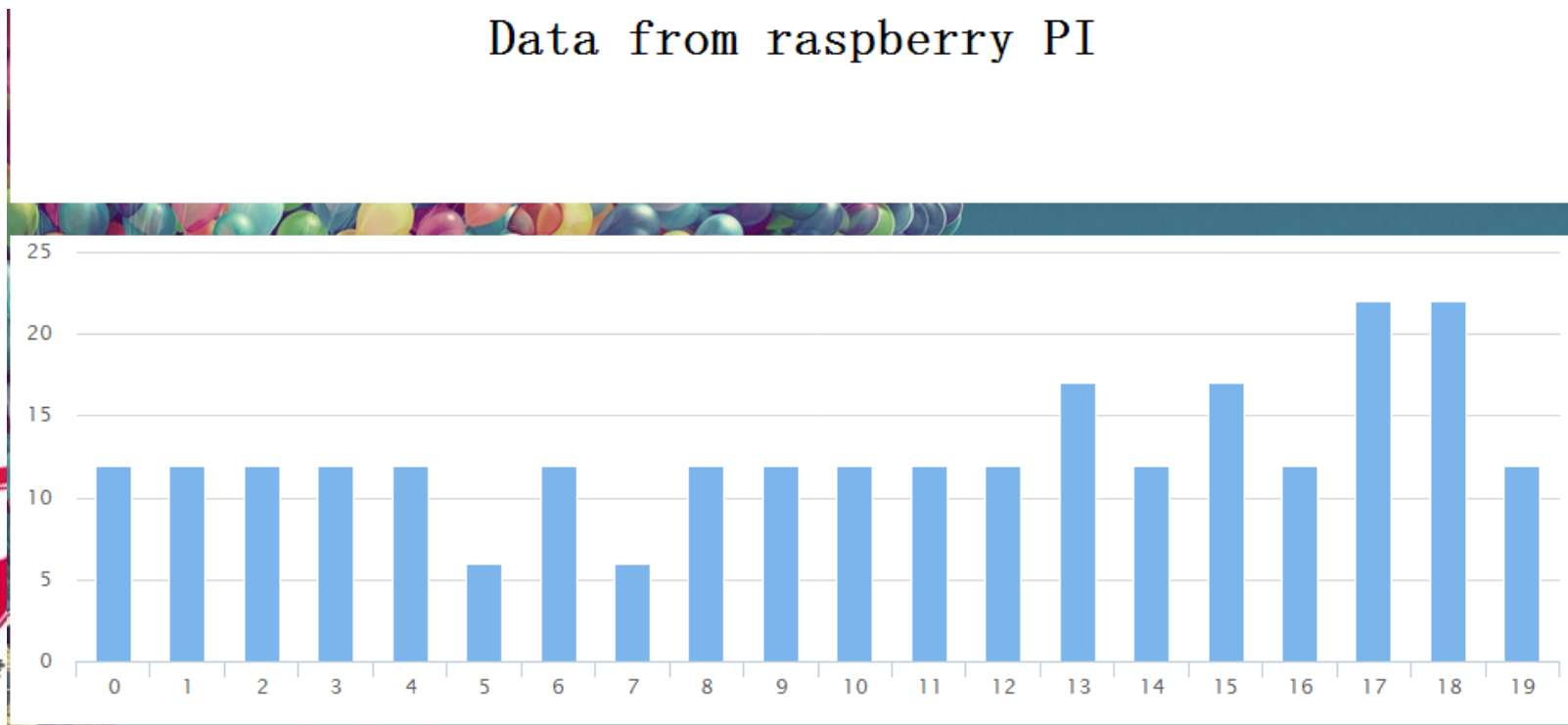
- 这是一个demo实现的图形界面：



有关在服务器上的绘图工具

- 画图的工具很多，我们需要的是一个可以作为js“嵌入”到HTML中的画图工具，这一般选择“框架”。
- [Chartkick](#)是一个图表绘制工具，特点是UI美观、使用简单，并且支持IE6在内的大多数浏览器。chartkick 可以画 javascript 报表，界面比较美观，其支持加载Google Charts 和 Highcharts图形库，而且支持集成 Django, Flask/Jinja2框架

Data from raspberry PI



有关在服务器上建立的一个简单网站

□ 这是用python写的数据载入部分的程序：

```
from flask import Flask //导入Flask
from flask import request //导入request
from flask import render_template //导入render_template模板
app = Flask(__name__)
a = []
@app.route('/')
def show():
    return render_template('index.html', datalist=a)
@app.route('/', methods=['POST'])
def getDate():
    a.append(request.json['value'])
def store(data):
    pass
def get(data):
    pass
if __name__ == '__main__':
    app.run(host='192.168.1.188',port=5000)
```

有关在服务器上建立的一个简单网站

□ Html文件-头部分

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <title>Anx's cloud</title>
    <meta name="viewport" content="width=device-width,initial-scale=1">
    <link rel="stylesheet" href="../static/css/normalize.css">
    <link rel="stylesheet" href="../static/css/main.css">
    <script src="../static/jquery.min.js"></script>
    <script src="../static/highcharts.js"></script>
    <script src="../static/chartkick.js"></script>
  </head>
```



由于测试主机上不能上外网，所以这里将js文件下载到了本地，并配置为url_for路径。

python run.py运行后，在浏览器中打开

http://127.0.0.1:5000就可以看到上面截图的效果了。

有关在服务器上建立的一个简单网站

□ HTML文件-显示层部分

```
<body>
  <div class="container">
    <header>
      <h1>Data from raspberry PI</h1>
    </header>
    <section>

<div id="chart-12" style="height: 300px; text-align: center; color: #999;
  line-height: 300px; font-size: 14px;
  font-family: Lucida Grande, Lucida Sans Unicode,
  Verdana, Arial, Helvetica, sans-serif;">

  Loading...
</div>
```



由于测试主机上不能上外网，所以这里将js文件下载到了本地，并配置为url_for路径。

python run.py运行后，在浏览器中打开
<http://127.0.0.1:5000>就可以看到上面截图的效果了。

有关在服务器上建立的一个简单网站

□ HTML文件—数据提取部分

```
<script>
//
new Chartkick.ColumnChart(document.getElementById("chart-12"), {0: 12.0, 1:
12.0, 2: 12.0, 3: 12.0, 4: 12.0, 5: 6.0, 6: 12.0, 7: 6.0, 8: 12.0, 9: 12.0, 10: 12.0, 11:
12.0, 12: 12.0, 13: 17.0, 14: 12.0, 15: 17.0, 16: 12.0, 17: 22.0, 18: 22.0, 19: 12.0},
{"library": {}});
//]]&gt;
&lt;/script&gt;
&lt;/section&gt;
&lt;/div&gt;
&lt;/body&gt;
&lt;/html&gt;</pre></div><div data-bbox="264 547 913 625" data-label="Text"><p>由于测试主机上不能上外网，所以这里将js文件下载到了本地，并配置为url_for路径。</p></div><div data-bbox="264 627 835 705" data-label="Text"><p>python run.py运行后，在浏览器中打开<br/><a href="http://127.0.0.1:5000">http://127.0.0.1:5000</a>就可以看到上面截图的效果了。</p></div><div data-bbox="22 787 215 951" data-label="Page-Footer"><img alt="Logo for Gao He IT Education, featuring a stylized 'GH' in red and yellow, with the text '1993-2013' and '专注IT教育二十年' below it." data-bbox="22 787 215 951"/><p>1993-2013<br/>专注IT教育二十年</p></div>
```

有关在服务器上建立的一个简单网站

□ 树莓派的数据上传部分：

```
import serial
import os
import time, subprocess, datetime
import json
import requests
import string

ser = serial.Serial('/dev/ttyACM0', 9600, timeout=2)
ser.open()

try:
    while 1:
        subprocess.call("clear")
        i = os.system('clear')
        response = ser.readline()
        print datetime.datetime.now()
        print time.strftime('%H:%M:%S')
        print response
```



有关在服务器上建立的一个简单网站

□ 树莓派的数据上传部分:

```
try:
    k=string.atoi(response)
except ValueError:
    k=0

ser.write('{0:5}'.format(time.strftime('%H%M')))
# time.sleep(.5)
apiurl = 'http://api.yeelink.net/v1.0/device/181662/sensor/199944/datapoints'
apiheaders={'U-Apikey':'2b70796ee9b6d3264653395bc5d4c60d','content-
type':'application/json'}
payload={'value':k}
r = requests.post(apiurl,headers=apiheaders,data=json.dumps(payload))
print r
#time.sleep(.5)
except KeyboardInterrupt:
    ser.close()
```


有关在服务器上的绘图工具

□ 实现效果:

Data from raspberry PI



7、在云上直接控制树莓派点亮LED

□需求：另一个更像Yeelink的例子，是建立一个自己开发的Web服务器，用户通过浏览器访问一个Web服务器地址，就可以直接控制树莓派上的LED灯：亮还是不亮。

□用户的IE显示和实际效果如下图：



7、使用flask进行python web开发

- 在云服务器上进行一个web开发，可以用Tomcat，当然也可以选其他所谓轻量级的web开发框架
- Flask就是用来在Web服务器上进行web开发的、基于python的轻量级web开发框架。

一、首先在服务器上安装flask开发环境：

假定服务器系统的OS是debian，其他Linux系统的命令可能稍有差异：

ssh登陆服务器后：

```
sudo apt-get install python-pip
```

```
sudo pip install flask
```

然后环境就搭好了。

□

7、使用flask进行python web开发

二、测试

在服务器上写了一个小型的web blog: emdlog,放在github上了,这里可以直接拿来进行测试:

```
git clone https://github.com/embbnux/emdlog.git
```

```
cd emdlog/emdlog
```

```
sudo apt-get install sqlite3
```

```
sqlite3 db/flaskr.db schema.sql
```

```
cd ../
```

```
python runserver.py
```

然后浏览器访问: raspberry_ip:2000, 应该就会看到一个很简易的blog站

Runserver.py的代码:

```
# welcome to my blog : Blog of Embbnux
```

```
# url:http://www.embbnux.com/
```

```
# author : Embbnux Ji
```

```
from rpcloudmanager import app
```

```
app.run(host='0.0.0.0',port=2000)
```



7、使用flask进行python web开发

- 前面介绍了在服务器上使用flask开发web框架
- 在下面，用树莓派可以很容易的通过python来操作gpio，然后，我们可以通过web来控制树莓派的gpio
- 用户可以通过IE浏览器，访问web页面，直接操作raspberrypi的gpio底层，或者也可以通过手机app发送post或者get等请求，来控制树莓派的gpio,这样岂不是很妙！

7、使用flask进行python web开发

一 在树莓派上安装python gpio库（如果已安装，则可跳过此步）

SSH或者终端下：

```
mkdir gpio
```

```
cd gpio
```

```
wget https://pypi.python.org/packages/source/R/RPi.GPIO/RPi.GPIO-0.5.7.tar.gz
```

#或者到这里下载最新版本：<https://pypi.python.org/pypi/RPi.GPIO>

```
tar xvzf RPi.GPIO-*.tar.gz
```

```
cd RPi.GPIO-*/
```

```
sudo python setup.py install
```

安装时如出现如下错误：

```
source/py_gpio.c:23:20: fatal error: Python.h: No such file or directory
```

是因为缺少Python.h文件,没安装python编译环境：

```
sudo apt-get install python-dev
```

再次安装：

```
sudo python setup.py install
```

没问题就安装好了。

二 使用python操作gpio

先测试下输出,新建个led.py文件:

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11,GPIO.OUT)
while True:
    GPIO.output(11,True)
    time.sleep(1)
    GPIO.output(11,False)
    time.sleep(1)
```

注意: 这里使用GPIO.BOARD模式, 所以对于引脚号的排序, 是按26个pin的顺序, 不是gpio1这样的。也就是说pin1就是板子上的3V3. 把led的负极接到板子上的pin11。正极接一个3K3的电阻, 再接到3V3上, 防止烧坏。终端下运行:

```
sudo python led.py
```

如果LED出现一闪一闪就表示成功了。

7、使用flask进行python web开发

再添加个按钮:

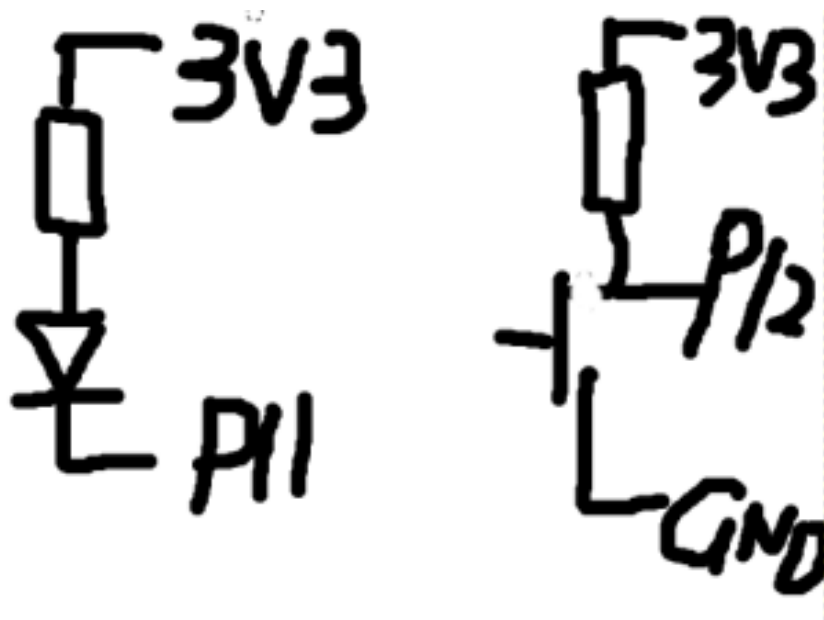
这里使用复位按键,一段接到GND,一段加到PIN12,再接10k电阻到3v3上拉
程序:

```
#!/user/bin/env python
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11,GPIO.OUT)
GPIO.setup(12,GPIO.IN)
while True:
    in_value= GPIO.input(12)
    if in_value ==False:
        GPIO.output(11,False)
        time.sleep(1)
        GPIO.output(11,True)
    while in_value == False:
        in_value = GPIO.input(12)
```

对按钮进行下降沿检测.按下按钮后led闪亮一下.

7、使用flask进行python web开发

电路:



没有加开关前和加了开关后的电路图

7、使用flask进行python web开发

三 处理web请求

在服务器上使用flask进行web开发很方便，这里使用post来处理gpio操作请求：

```
@app.route('/gpio/<int:id>',methods=['POST']) def  
gpio_led(id): if request.method == 'POST':  
GPIO.setmode(GPIO.BOARD) if id<100:  
GPIO.setup(id,GPIO.OUT) GPIO.setmode(GPIO.BOARD)  
GPIO.setup(id,GPIO.OUT) GPIO.output(id,False) else:  
GPIO.setup(id-100,GPIO.OUT) GPIO.output(id-100,True)  
return redirect(url_for('show_index'))
```

7、使用flask进行python web开发

四、网页控制按钮的HTML文件

有了web请求处理，还需要写一个用来给IE用户显示操作按钮view的HTML文件：

View.html

```
<form action="/gpio/11" method=post>
```

```
<input type=submit value="led on" />
```

```
</form>
```

```
<form action="/gpio/111" method=post>
```

```
<input type=submit value="led off"/>
```

```
</form>
```

7、使用flask进行python web开发

五、运行web程序

web工程代码我已经上传到github上了，需要的同学可以clone下来,运行测试

```
$git clone git@github.com:embbnux/RpiCloudManager.git  
cd RpiCloudManager sudo python runserver.py
```

代码如下：

```
# welcome to my blog : Blog of Embbnux  
# url:http://www.embbnux.com/  
# author : Embbnux Ji  
from rpicloudmanager import app  
app.run(host='0.0.0.0',port=2000)
```

7、使用flask进行python web开发

通过浏览器访问http://your_raspberry_ip:2000就可以了，效果见下图：



8、云服务器+树莓派可以做什么？

□ 利用云服务器+树莓派，我们可以做什么？

□ 定位：

□ 云服务器的定位：

□ 定位的依据：

□ 角色的作用：

□ 基于角色的应用：

□ 树莓派的定位：

□ 依据：下端、小型、便携、移动、廉价、基本功能

□ 角色：MVC中的C

□ 应用：基于C的应用

□ 个人专有的、定制化的媒体管理器

□ 与一般媒体播放器的区别（定制化、可控制管理）

□ 资源获得与保存、播放控制

□ 在自己的鱼池中抓鱼（海、养鱼池、餐桌）

8、云服务器+树莓派可以做什么？

□ 海、养鱼池、餐桌

□ 海：

□ 养鱼池（云服务器）：

□ 从海到鱼池：在云上实现自动抓取、下载、整理、存储

□ 与传统方式比较：没有鱼池，需要的时候直接到海里捞

□ 餐桌（树莓派）：

□ 与鱼池配合：在自己的库里运用资源

□ 与传统方式比较：不是简单的播放器

□ 应用实例：

□ 美剧追踪：

□ 传统方式：

□ 在线视频网站看（收费、群播的网速限制等）

□ 手动下载：每集下载

□ 云方式：云自动追（下载）、随时看（网速是一对一）、分享

8、云服务器+树莓派可以做什么？

□ 开发项目内容

□ 云上

□需求：在云上实现自动抓取、下载、整理、存储

□对上：订制任务、自动抓取、收集整理、有序存储

□对下：响应树莓派的请求（播放、查询、获取等）

□ 树莓派上

□需求：

□对上：与云对接

□对下：实现播放、查询、用户界面控制等功能

□注意：苏宁云对带宽无限制（也没有收费限制）