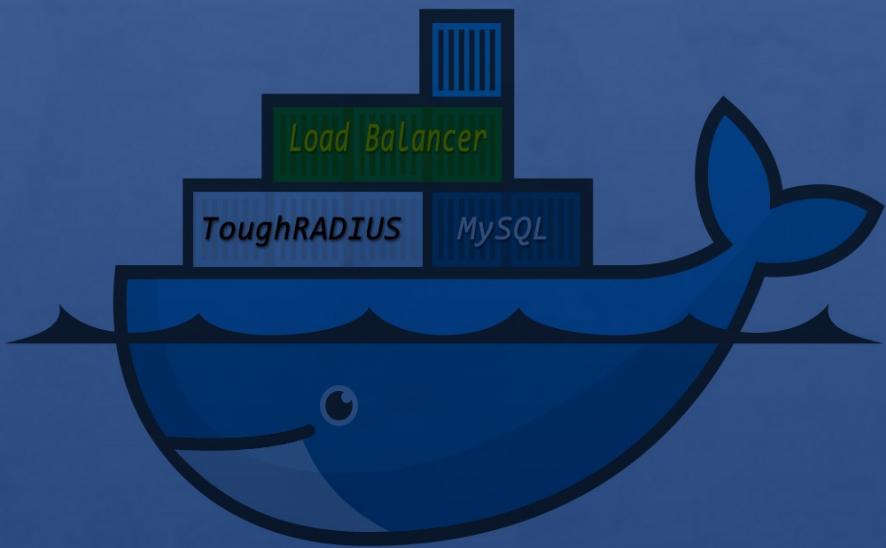


ToughRADIUS

Book



docker

docs.toughradius.net

Table of Contents

1. Introduction
2. ToughRADIUS 用户手册
 - i. 项目简介
 - ii. 快速指南
 - iii. 高级技巧
 - i. 系统服务设置
 - ii. 服务端口映射
 - iv. RouterOS 对接
 - v. Linux PPTP 对接
 - vi. Linux L2TP 对接
 - vii. 操作手册
 - i. 系统维护篇
 - ii. 系统管理篇
 - iii. 营业管理篇
 - iv. 维护管理篇
 - v. 查询统计篇
 - viii. 开发指南
 - i. 定制你的ToughRADIUS专有版本

ToughRADIUS 教程



欢迎阅读ToughRADIUS用户文档。

及时向我们反馈问题是对我们最大的支持。

向我们发送邮件: support@toughstruct.com

加入我们的QQ交流群组: 464025428, 247860313

访问我们的网站: <http://www.toughstruct.com>

ToughRADIUS 项目地址

Github 项目地址: <https://github.com/talkincode/ToughRADIUS>

Github 文档地址: <https://github.com/talkincode/ToughRADIUS-GitBook>

感谢

感谢 ToughRADIUS 开发团队的家人，没有你们在背后默默的支持，我们不可能顺利的前进。

感谢 ToughRADIUS 用户以及各位友人的支持，如果没有你们的支持和反馈，ToughRADIUS 不可能这么快速的推进。

感谢 ToughRADIUS 开发团队的默默奉献，我们是一个棒棒哒的团队。

这是最简单实用的 ToughRADIUS 说明书，面向一般使用者，以及高级使用者。

我们倡导简单，使用，我们不再会教你用传统的方式去折腾lamp(linux + apache+mysql+python)或者lnmp (linux + nginx+mysql+python)。

ToughRADIUS的用户应该将注意力集中到如何使用，而不是淹没在各种技术细节中慢慢绝望。



项目介绍

ToughRADIUS是一个AAA应用软件。

ToughRADIUS支持标准RADIUS协议，提供完整的AAA实现。支持灵活的策略管理，支持各种主流接入设备并轻松扩展，具备丰富的计费策略支持。

ToughRADIUS支持使用Oracle, MySQL, PostgreSQL, MSSQL等主流数据库存储用户数据，支持数据缓存，提供高性能的支持。

ToughRADIUS采用Docker容器模式安装部署，简单方便。

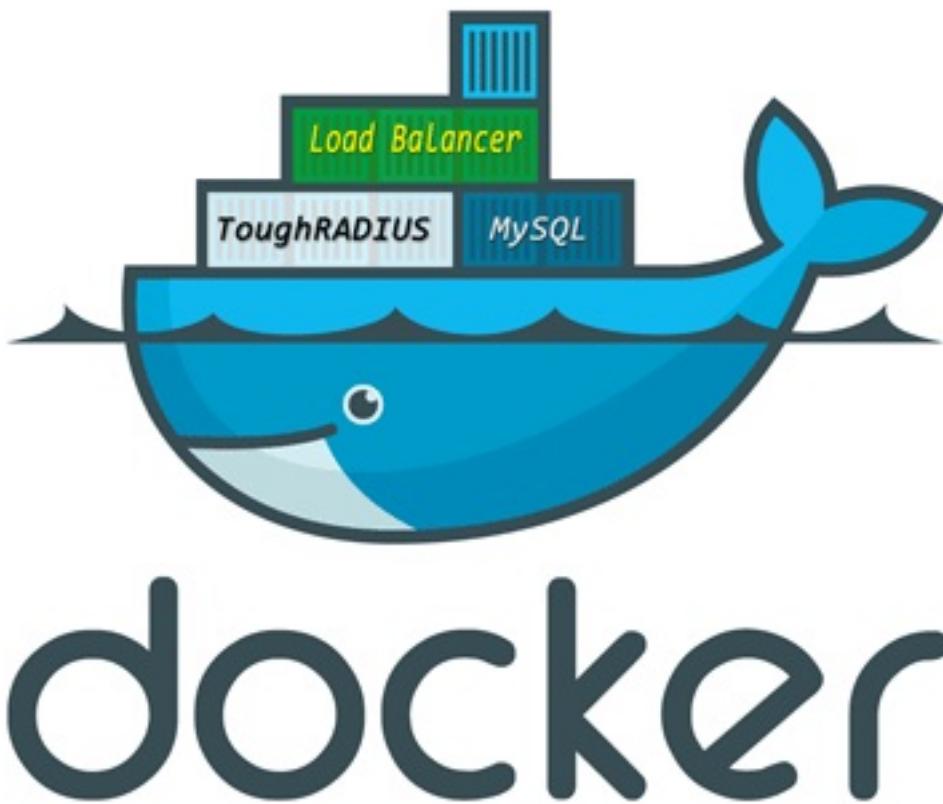
ToughRADIUS提供了RADIUS核心服务引擎与Web管理控制台,用户自助服务三个子系统，核心服务引擎提供高性能的认证计费服务，Web管理控制台提供了界面友好，功能完善的管理功能。用户自助服务系统提供了一个面向终端用户的网上服务渠道。

ToughRADIUS快速指南

准备

一台完整的服务器，或者远程VPS，给服务器安装Linux系统，CentOS6以上，ubuntu13以上，或者其他你自己熟悉的Linux发行版。

你要懂一点技术，比如安装操作系统，会在终端敲命令。



ToughRADIUS 是Docker技术的拥抱者，当你决定使用ToughRADIUS，你也需要去学习关于Docker的知识。

安装部署

ToughRADIUS主要采用了Docker镜像部署的模式，ToughRADIUS的镜像基础是CentOS 7。

我们可以把Docker看作一个软件集装箱，半世纪之前，集装箱发挥了巨大的力量，改变了整个运输产业，也改变了人们的生活。而Docker就类似这样一个集装箱工具，只不过他封装的是软件。

还记得linux安装lamp的经历吗？现在可以对各种安装配置apache，php等繁琐的工作说再见了。

我们把ToughRADIUS相关的配置，运行依赖环境等全部打包在一个“Docker集装箱”里，我们只需要在我们的服务器上简单的安装一个支持运行“Docker集装箱”的环境，那么我们不用去折腾各种运行环境搭建就能简单的让ToughRADIUS跑起来。

通常我们把封装了软件应用的“Docker集装箱”叫做镜像，有点类似你可能了解的ISO文件。

Docker 安装

CentOS 6

```
$ sudo yum install http://mirrors.yun-idc.com/epel/6/i386/epel-release-6-8.noarch.rpm  
$ sudo yum install docker-io  
$ sudo service docker start
```

CentOS 7

```
$ sudo yum install docker  
$ sudo service docker start
```

Ubuntu

```
$ sudo apt-get update  
$ sudo apt-get install docker.io
```

```
$ sudo ln -sf /usr/bin/docker.io /usr/local/bin/docker  
$ sudo sed -i '$accomplete -F _docker docker' /etc/bash_completion.d/docker.io
```

Windows

请下载Windows安装文件：<https://github.com/boot2docker/windows-installer/releases/download/v1.8.0/docker-install.exe>

注意：Window下的Docker服务是借助虚拟机来实现的，性能上远不如使用linux主机。

Docker 容器创建

确认Docker服务已经运行，通过以下指令创建ToughRADIUS的容器实例，当看到一串hash打印时，说明容器创建成功了。

```
$ docker run -d --name trserver --net host index.alauda.cn/toughstruct/toughradius
```

这样我们的服务就已经运行了。我们可以通过浏览器来访问我们的应用了。

营业管理：<http://ipaddr:1816> 管理权限 admin/root

自助服务：<http://ipaddr:1817>

系统管理：<http://ipaddr:1819> 管理权限 ctlman/ctlroot

容器的基本管理

启动容器

```
$ docker start trserver
```

停止容器

```
$ docker stop trserver
```

重启容器

快速指南

```
$ docker restart trserver
```

查看容器日志，*Ctrl+c*退出

```
$ docker logs trserver
```

高级技巧



通过 systemd 设置系统服务

目前很多Linux的发行版已经开始采用systemd系统，通过systemd来设置ToughRADIUS的服务是一件很容易的事情。

创建文件 /usr/lib/systemd/system/toughradius.service

内容如下：

```
[Unit]
Description=ToughRADIUS Service
After=docker.service
Requires=docker.service

[Service]
Type=forking
RemainAfterExit=yes
ExecStart=/usr/bin/docker start trserver
ExecStop=/usr/bin/docker stop trserver
ExecReload=/usr/bin/docker restart trserver

[Install]
WantedBy=multi-user.target
```

其中trserver就是你的容器实例的名称。通过以下指令设置开机启动开关。

```
systemctl enable trserver
```

由于我们的Docker容器都依赖于docker服务本身，因此也要保证docker服务开启。

```
systemctl enable docker
```

服务端口映射

默认情况下，ToughRADIUS的Docker容器采用的是host模式的网络配置，注意缺省创建容器指令：

```
$ docker run -d --name trserver --net host index.alauda.cn/toughstruct/toughradius
```

其中 --net host 表示使用服务主机网络配置，这种模式无需任何NAT转换，这将获得最大的网络性能。如果需要灵活的改变端口，我们可以使用bridge模式，这种模式下容器将会创建虚拟的网络配置，并与主机网络进行桥接，我们需要将容器的端口映射到主机端口上：

```
$ docker run -d --name trserver -p 1812:/1812/udp -p 1813:1813/udp -p 80:1816 -p 1815:1815 -p 1819:181
```

如上所示，我们将主机端口与容器端口进行一对一映射，上面将主机的80端口映射到容器的管理控制台默认1816端口上。容器本身服务的端口保持不变，但主机映射端口可以灵活变化。

我们接着访问 <http://主机地址> 即可访问ToughRADIUS管理控制台。

RouterOS对接指南

Linux PPTP 对接

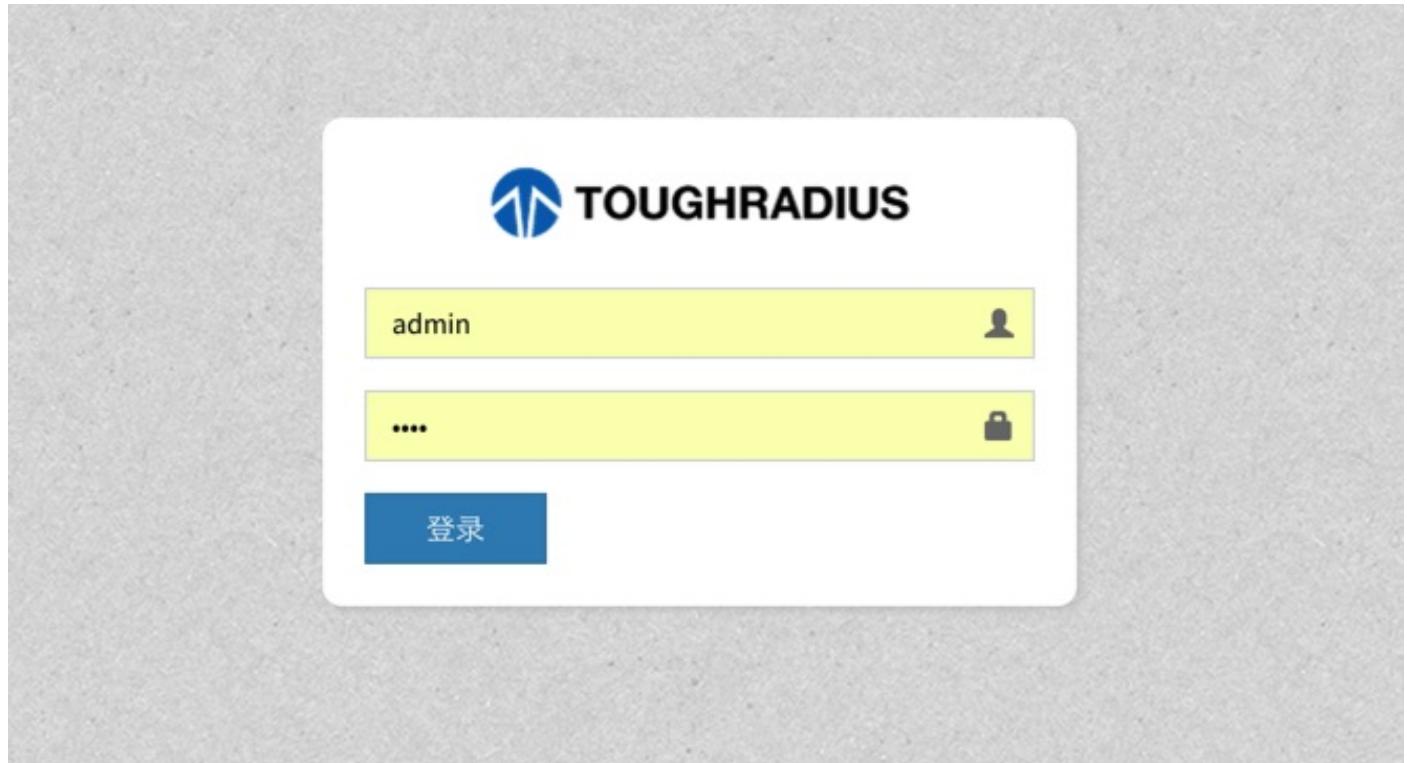
Linux L2TP 对接

操作手册

登录与登出

ToughRADIUS管理控制台系统提供了一个统一的登录界面,通常你可以通过以下URL来 访问:

`http://你的服务器IP地址:WEB端口(默认1816) , 比如: http://192.168.1.1:1816`



输入你的管理用户及管理密码,即可登陆系统,默认登录管理员及密码是admin/root,你应该第一时间修改它。

登陆成功进入系统管理首页,如下图,点击右上角退出链接即可安全退出。

TOUGHRADIUS

功能导航

- 系统管理
- 营业管理
- 维护管理
- 统计分析

控制面板 修改密码 刷新缓存 系统管理 技术支持 (admin) 退出

系统信息

当前登录管理员

登录管理员	admin	登录时间	2015-10-04 00:07:18
登录IP地址	58.20.114.9	系统版本	1.1.5

消息统计

最近15分钟的消息统计

消息量

01:12 01:13 01:14 01:15 01:16 01:17 01:18 01:19 01:20 01:21 01:22 01:23 01:24 01:25 01:26

认证消息
计费消息

认证消息总数	0
认证成功总数	0
认证拒绝总数	0
认证丢弃总数	0
记账消息总数	0
记账上线总数	0
记账更新总数	0
记账下线总数	0
记账丢弃总数	0
记账重发总数	0

在线统计

在线用户统计

2015-10-04 00:00:00 -- 2015-10-04 23:59:59

在线数

0人

01:00

当前在线用户数 0/0

流量统计

流量统计

2015-10-04 00:00:00 -- 2015-10-04 23:59:59

使用流量

0MB

01:00

密码修改

部署系统后，修改密码是首要大事，保证密码的复杂性以及定期修改是必要的。

超级管理员和其他操作员登录后可以修改自己的密码。

The screenshot shows the 'Password Update' page of the ToughRADIUS web interface. The left sidebar has a dark theme with white icons and text. It includes sections for 'System Management', 'Business Management', 'Maintenance Management', and 'Statistical Analysis'. The main content area has a light blue header with the title 'Password Update'. Below the header, there is a checkbox labeled 'Check for password update'. The form fields are as follows: 'Administrator Name' (input field containing 'admin'), 'New Administrator Password' (input field), 'Confirm New Administrator Password' (input field), and a 'Save' button at the bottom. The footer contains copyright information: 'Copyright © 2015 toughradius.net. All rights reserved.' and a support link 'Online Support'.

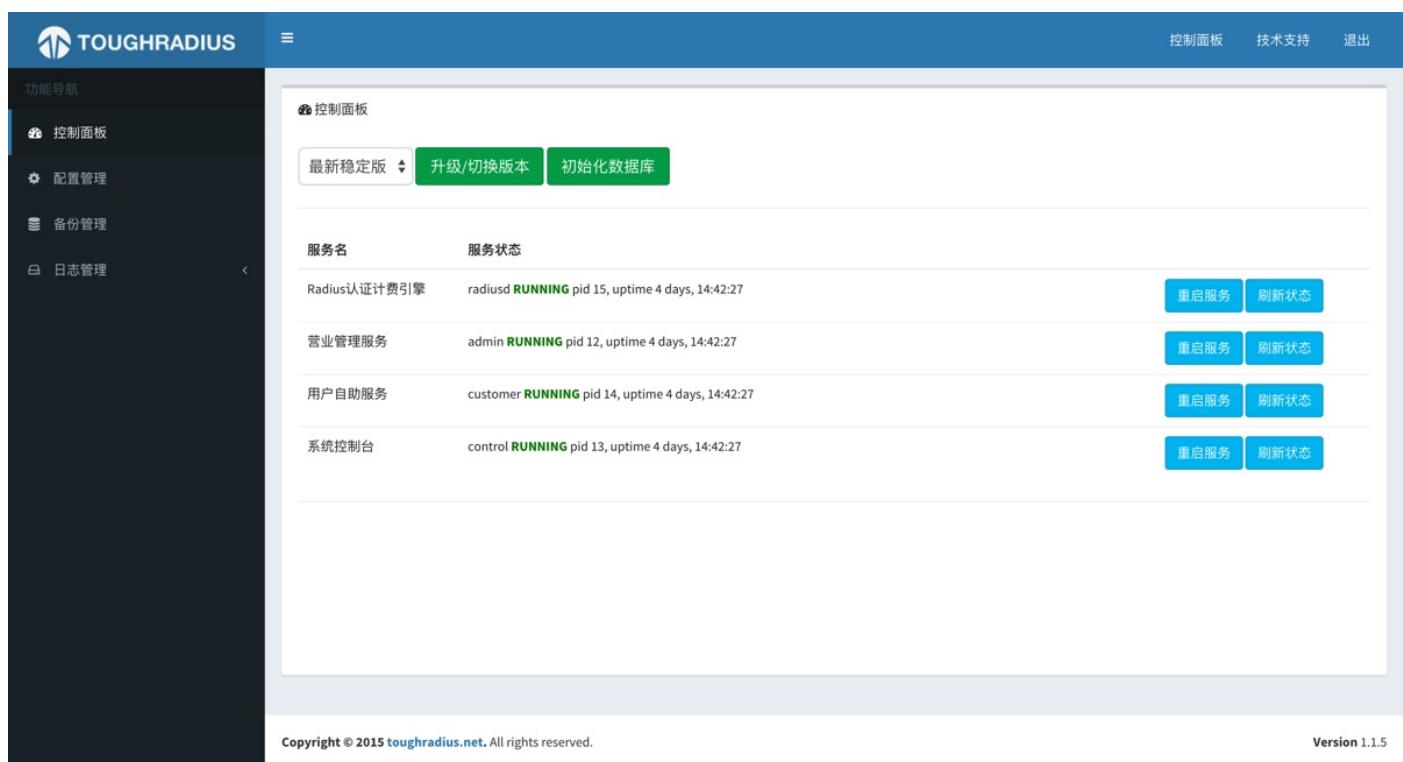
系统维护篇

对于系统的管理维护，ToughRADIUS提供了一个专门的子系统，与运营管理子系统分离，它主要提供了系统级别的维护功能，如系统的升级，服务的启动，停止，数据库备份，恢复，系统日志查看等功能。他是纯粹针对系统管理员的。

缺省访问地址是 <http://server:1819/>，默认的登录账号密码是 ctlman/ctlroot。

控制面板

在控制面板界面，可以对当前系统进行升级，和数据库初始化。版本升级后会自动重启动服务，数据库初始化功能是针对系统初次创建，迁移和升级而设的，数据库初始化会删除已经存在的表重新建立，因此，如果不是新数据库，在操作时尽量对数据库先进行备份。在升级版本时应该总是这样做。



The screenshot shows the ToughRADIUS Control Panel. The left sidebar has a dark theme with white text and icons. It includes links for '功能导航' (Function Navigation), '控制面板' (Control Panel) which is selected and highlighted in blue, '配置管理' (Configuration Management), '备份管理' (Backup Management), and '日志管理' (Log Management). The main content area has a light gray background. At the top, there's a header with the ToughRADIUS logo, a menu icon, and navigation links for '控制面板', '技术支持' (Technical Support), and '退出' (Logout). Below the header is a section titled '控制面板' with three buttons: '最新稳定版' (Latest Stable Version), '升级/切换版本' (Upgrade/Switch Version), and '初始化数据库' (Initialize Database). The main table lists four services with their names, current status, and process ID (pid). Each row also has '重启服务' (Restart Service) and '刷新状态' (Refresh Status) buttons. The services listed are:

服务名	服务状态		
Radius认证计费引擎	radiusd RUNNING pid 15, uptime 4 days, 14:42:27	重启服务	刷新状态
营业管理服务	admin RUNNING pid 12, uptime 4 days, 14:42:27	重启服务	刷新状态
用户自助服务	customer RUNNING pid 14, uptime 4 days, 14:42:27	重启服务	刷新状态
系统控制台	control RUNNING pid 13, uptime 4 days, 14:42:27	重启服务	刷新状态

At the bottom of the page, there are copyright information 'Copyright © 2015 toughradius.net. All rights reserved.' and a version note 'Version 1.1.5'.

在控制面板中可以分别针对4个子系统进行服务重启和状态刷新。

配置管理

提供针对系统核心配置进行修改的功能。

The screenshot shows the 'System Settings' tab in the configuration management section. It includes fields for enabling DEBUG mode (set to 'Yes'), selecting the time zone (Asia/Shanghai), entering a security key (LpWE9AtfDPQ3ufXBS6gJ37WW8TnSF920), and specifying SSL certificate paths for private key and certificate.

Copyright © 2015 toughradius.net. All rights reserved. Version 1.1.5

关于更多的配置技巧在高级技巧篇幅中说明。

备份管理

ToughRADIUS的在线数据备份采用了json格式，以及gzip压缩，支持在线备份，下载，删除，上传，在线恢复功能。

数据备份排除了上网日志，记账日志，在线用户信息，以免备份数据过大导致备份失败。

The screenshot shows the 'Database Backup Management' section. It allows users to set a backup directory (set to /var/toughradius/data) and click '立即备份' (Immediate Backup). Below this, a table lists existing backup files with options to delete, download, or restore them.

备份名称	操作
toughradius_db_20151005_002820_0002.json.gz	删除 下载 恢复
toughradius_db_20151005_002813_0001.json.gz	删除 下载 恢复

Copyright © 2015 toughradius.net. All rights reserved. Version 1.1.5

日志查看

可以查看最近的系统日志信息，对于系统出现故障时，这些日志信息非常有用，当你需要更多详细信息时，请打开debug开关。

The screenshot shows the ToughRADIUS web interface. The top navigation bar includes the logo, 'TOUGHRADIUS', and links for '控制面板', '技术支持', and '退出'. The left sidebar, titled '功能导航', has sections for '控制面板', '配置管理', '备份管理', and '日志管理', with 'Radius系统日志' currently selected. The main content area is titled '/var/log/radiusd.log' and contains the following log entries:

```

-09-16 22:31:40+0800 [AdminServerProtocol,3,127.0.0.1] Client connecting: tcp4:127.0.0.1:42229
2015-09-16 22:31:40+0800 [AdminServerProtocol,3,127.0.0.1] WebSocket connection open.
2015-09-16 22:32:06+0800 [-] Received SIGTERM, shutting down.
2015-09-16 22:32:06+0800 [autobahn.twisted.websocket.WebSocketServerFactory] (TCP Port 1815 Closed)
2015-09-16 22:32:06+0800 [autobahn.twisted.websocket.WebSocketServerFactory] Stopping factory
2015-09-16 22:32:06+0800 [AdminServerProtocol,3,127.0.0.1] WebSocket connection closed: connection was closed uncleanly (peer dropped the TCP connection without previous WebSocket closing handshake)
2015-09-16 22:32:06+0800 [AdminServerProtocol,2,127.0.0.1] WebSocket connection closed: connection was closed uncleanly (peer dropped the TCP connection without previous WebSocket closing handshake)
2015-09-16 22:32:06+0800 [CoAClient (UDP)] (UDP Port 38497 Closed)
2015-09-16 22:32:06+0800 [CoAClient (UDP)] Stopping protocol
2015-09-16 22:32:06+0800 [RADIIUSAccounting (UDP)] (UDP Port 1813 Closed)
2015-09-16 22:32:06+0800 [RADIIUSAccess (UDP)] (UDP Port 1812 Closed)
2015-09-16 22:32:06+0800 [RADIIUSAccess (UDP)] Stopping protocol
2015-09-16 22:32:06+0800 [-] Main loop terminated.
[36;2m[INFO] - use config:/etc/radiusd.conf
running radiusd server...
2015-09-16 22:32:08+0800 [-] Log opened.
2015-09-16 22:32:08+0800 [-] server listen 0.0.0.0
2015-09-16 22:32:08+0800 [-] RADIUSAccess starting on 1812
2015-09-16 22:32:08+0800 [-] Starting protocol
2015-09-16 22:32:08+0800 [-] RADIIUSAccounting starting on 1813
2015-09-16 22:32:08+0800 [-] Starting protocol
2015-09-16 22:32:08+0800 [-] WebSocketServerFactory starting on 1815
2015-09-16 22:32:08+0800 [-] Starting factory
2015-09-16 22:32:09+0800 [AdminServerProtocol,0,127.0.0.1] Client connecting: tcp4:127.0.0.1:42236

```

系统管理篇

TOUGH RADIUS

功能导航
控制面板 修改密码 刷新缓存 系统管理 技术支持 (admin) 退出

系统信息

当前登录管理员			
登录管理员	admin	登录时间	2015-10-04 00:07:18
登录IP地址	58.20.114.9	系统版本	1.1.5

消息统计

最近15分钟的消息统计

刷新数据

认证消息总数	0
认证成功总数	0
认证拒绝总数	0
认证丢弃总数	0

记账消息总数	0
记账上线总数	0
记账更新总数	0
记账下线总数	0
记账丢弃总数	0
记账重发总数	0

在线统计

在线用户统计

2015-10-04 00:00:00 -- 2015-10-04 23:59:59

刷新数据

当前在线用户数 0/0

流量统计

流量统计

2015-10-04 00:00:00 -- 2015-10-04 23:59:59

刷新数据

系统参数配置

ToughRADIUS提供了一些全局参数配置，可以通过系统参数配置界面统一管理。

系统设置

注意在生产环境中，一般选择关闭DEBUG选项，以免造成日志文件膨胀占用过多存储空间。

系统参数管理

系统设置 自助服务 到期提醒 邮件服务器 Radius设置

管理系统名称	ToughRADIUS管理控制台	?
自助服务系统名称	ToughRADIUS自助服务中心	?
自助服务系统网站地址	http://forum.toughradius.net	
开启在线支持功能	是	?
开启DEBUG	否	?
更新		

自助服务设置

系统参数管理

系统设置 自助服务 到期提醒 邮件服务器 Radius设置

激活邮箱才能自助开户充值	否	?
微信公众号二维码图片(宽度230px)	http://img.toughradius.net/toughforum/jamiesun/1421820686.jpg!230	
客户服务电话	000000	
客户服务QQ号码	000000	
充值卡订购网站地址	http://www.tmall.com	
更新		

到期提醒设置

系统参数管理

系统设置 自助服务 **到期提醒** 邮件服务器 Radius设置

到期提醒提前天数:

到期提醒邮件模板:

```
账号到期通知
尊敬的会员您好:
您的账号#account#即将在#expire#到期, 请及时续费!
```

到期通知触发URL:

到期用户下发最大会话时长(秒):

到期提醒下发地址池:

更新

邮件服务器设置

系统参数管理

系统设置 自助服务 **到期提醒** **邮件服务器** Radius设置

SMTP服务器:

SMTP用户名:

SMTP密码: 

更新

Radius设置

系统参数管理

- 系统设置
- 自助服务
- 到期提醒
- 邮件服务器
- Radius设置

Radius认证模式	强制密码认证
是否允许查询用户密码	是
Radius服务地址	127.0.0.1
Radius服务管理端口	1815
Radius记账间隔(秒)	120
Radius最大会话时长(秒)	86400
拒绝延迟时间(秒)(0-9)	0

更新

区域信息管理

系统通过区域提供了对用户进行逻辑分类的支持，界面如下图，支持区域信息的新增，修改和删除。注意：如果区域下已经有用户信息，该区域不允许删除。

区域列表

区域名称	区域描述	操作
default	测试区域	修改 删除

增加区域

查询记录数 1 首页 ← 上一页 1 下一页 → 尾页

BAS信息管理

ToughRADIUS系统支持多BAS接入管理，BAS设备信息必须在系统中登记才能和RADIUS进行通信。

BAS设备对应着实际的物理接入网关设备，是具备PPPOE Server功能的路由器或交换机等设备；BAS设备以IP地址为唯一标识，每个BAS与Radius配置对应的共享密钥。

系统支持标准的Radius协议接入，通常大多数BAS设备可以以标准类型配置。但是对于很多厂家的BAS设备，在标准协议下并不能工作的很完美，比如MAC地址，VLAN往往以各自的私有协议封装，Radius为了准确的处理这些信息，必须对这些BAS设备进行类型标识，并在消息处理时进行解析。

ToughRADIUS对各个厂家私有协议处理是可扩展的，通过编写插件可以实现更多的厂家扩展支持。

BAS列表						增加BAS
BAS名称	BAS地址	BAS类型	共享密钥	CoA端口	时区	操作
ROS	192.168.88.1	RouterOS	123456	3799	标准时区，北京时间	<button>修改</button> <button>删除</button>

CoA端口通常用来下发用户强制下线消息，以及支持一些动态授权管理。

操作员管理

系统提供了简洁实用的二级权限管理，通过操作员管理很方便的管理二级操作员权限。

操作员权限管理是一项重要的工作，应该有管理级别的系统管理员来完成，在实际操作过程中，注意不要对普通的操作员赋予过高的权限，这样会带来潜在隐患。

操作员的每次操作都会有系统日志记录，如非必要，不要删除操作员，如果不希望某个操作员再登陆系统管理，可选择停用此操作员。

操作员列表					增加操作员
操作员名称	操作员姓名	操作员类型	操作员状态	操作	
admin	admin	系统管理员	正常		

操作员表单：

TOUGH RADIUS

功能导航

- 系统管理
 - 系统控制面板
 - 系统参数管理
 - 区域信息管理
 - BAS信息管理
 - 操作员管理
 - 客户经理管理
 - 资费信息管理
 - 黑白名单管理
 - 充值卡管理
- 营业管理
- 维护管理
- 统计分析

增加操作员

操作员名称: opr

操作员姓名: 运维

操作员密码:

操作员状态: 正常

关联区域(多选): 测试区域

关联资费(多选)

提交

全部选择 取消选择

系统管理

<input type="checkbox"/> 系统控制面板	<input type="checkbox"/> 系统参数管理	<input type="checkbox"/> 区域信息管理	<input type="checkbox"/> BAS信息管理
<input type="checkbox"/> 客户经理管理	<input type="checkbox"/> 客户经理创建	<input type="checkbox"/> 客户经理修改	<input type="checkbox"/> 客户经理删除
<input type="checkbox"/> 客户经理删除	<input type="checkbox"/> 资费信息管理	<input type="checkbox"/> 资费详情查看	<input type="checkbox"/> 新增资费
<input type="checkbox"/> 修改资费	<input type="checkbox"/> 删除资费	<input type="checkbox"/> 新增资费扩展属性	<input type="checkbox"/> 修改资费扩展属性
<input type="checkbox"/> 删除资费扩展属性	<input type="checkbox"/> 黑白名单管理	<input type="checkbox"/> 新增黑白名单	<input type="checkbox"/> 修改黑白名单
<input type="checkbox"/> 删除黑白名单	<input type="checkbox"/> 充值卡管理	<input type="checkbox"/> 充值卡导出	<input type="checkbox"/> 充值卡生成
<input type="checkbox"/> 充值卡激活	<input type="checkbox"/> 充值卡回收		

营业管理

<input type="checkbox"/> 用户信息管理	<input type="checkbox"/> 用户信息导出	<input type="checkbox"/> 用户详情查看	<input type="checkbox"/> 删除用户信息
<input type="checkbox"/> 修改用户资料	<input type="checkbox"/> 用户快速开户	<input type="checkbox"/> 增开用户账号	<input type="checkbox"/> 修改用户上网账号
<input type="checkbox"/> 用户数据导入	<input type="checkbox"/> 用户账号停机	<input type="checkbox"/> 用户账号复机	<input type="checkbox"/> 用户账号续费
<input type="checkbox"/> 用户账号充值	<input type="checkbox"/> 用户资费变更	<input type="checkbox"/> 用户账号销户	<input type="checkbox"/> 用户受理查询
<input type="checkbox"/> 用户受理导出	<input type="checkbox"/> 删除用户账号	<input type="checkbox"/> 用户计费查询	<input type="checkbox"/> 用户计费导出
<input type="checkbox"/> 用户交易查询	<input type="checkbox"/> 用户交易导出	<input type="checkbox"/> 用户工单管理	<input type="checkbox"/> 用户工单详情
<input type="checkbox"/> 创建用户工单	<input type="checkbox"/> 处理用户工单	<input type="checkbox"/> 转派用户工单	<input type="checkbox"/> 删除用户工单

维护管理

<input checked="" type="checkbox"/> 用户账号查询	<input checked="" type="checkbox"/> 用户消息跟踪	<input checked="" type="checkbox"/> 账号详情	<input checked="" type="checkbox"/> 用户释放绑定
<input checked="" type="checkbox"/> 在线用户查询	<input checked="" type="checkbox"/> 上网日志查询	<input checked="" type="checkbox"/> 操作日志查询	

统计分析

<input type="checkbox"/> 用户流量统计	<input type="checkbox"/> 在线用户统计
---------------------------------	---------------------------------

Copyright © 2015 toughradius.net. All rights reserved.

控制面板 修改密码 刷新缓存 系统管理 技术支持 (admin) 退出

返回

资费信息管理

资费是对计费规则的定义，决定了对每个上网账号如何收费，如何扣费的规则。

系统资费类型支持以下几种：

1. 预付费包月：按自然月定义资费单价，用户可以按需要订购月数。
2. 预付费时长：按小时定义资费单价，用户只需要充值，在实际使用时周期性扣费。
3. 买断包月：预付费包月的一种打包形式，将多个月打包为一个资费，定义资费总价，是一种优惠策略资费形式。
4. 买断时长：时长计费打包的一种方式，一次性将时长打包为一个套餐，是一种优惠策略资费形式。
5. 预付费流量：按流量(MB)定义资费单价,用户只需充值，在实际使用时按使用的流量进行扣费。
6. 买断流量：流量计费打包的一种方式，一次性将流量打包为一个套餐，是一种优惠策略资费形式。

资费管理界面如下：

资费列表								
资费名称 资费策略 价格(元) 并发数 是否绑定MAC 是否绑定VLAN 上行速率 下行速率 状态 添加资费								
50元包月4M套餐	预付费包月	50.00	0	否	否	4194304.000 Mbps	4194304.000 Mbps	正常 <button>修改</button> <button>删除</button>
4M包年500元	买断包月	500.00	1	否	否	4194304.000 Mbps	4194304.000 Mbps	正常 <button>修改</button> <button>删除</button>
查询记录数 2 首页 ← 上一页 1 下一页 → 尾页								

资费表单：

增加资费

资费名称	50元包月4M套餐
计费策略	预付费包月
每月单价(元)	50
并发数控制(0表示不限制)	0
是否绑定MAC	否
是否绑定VLAN	否
最大上行速率(Mbps)	4194304
最大下行速率(Mbps)	4194304
资费状态	正常

提交

Copyright © 2015 toughradius.net. All rights reserved.

Version 1.1.5

在资费中定义了一些基本的策略，比如绑定，限速，不同BAS类型会将标准限速策略进行转换适配，同时也支持进行策略定制，在资费详情界面，点击增加策略属性可进行定制。

资费信息

资费名称	50元包月4M套餐	资费策略	预付费包月	资费状态	正常
资费价格	50.00	计费时段		买断月数	
绑定MAC	否	绑定VLAN	否	并发数	0
最大上行速率	4194304.000 Mbps	最大下行速率	4194304.000 Mbps	授权时长	0.00 小时
创建时间	2015-10-04 01:32:28	更新时间	2015-10-04 01:32:28	授权流量	0.00 MB

增加策略属性

策略名称	策略值	策略描述

资费策略配置界面：

增加策略属性

返回

策略名称	Mikrotik-Rate-Limit	?
策略值	4m/4m	
策略描述	客户的速率限制。字符串表示，格式为 rx-rate[/tx-rate] 如 256k/256k 或	
提交		

黑白名单管理

黑白名单是一个有用的策略管理功能，黑白名单以MAC地址为标识，同时有一个有效范围时间。被加入黑名单的MAC地址进行认证时，在数据校验之前就会被拒绝，在白名单的MAC地址用户在认证时将不受绑定策略限制。

黑白名单列表

[增加黑白名单](#)

MAC地址	黑白名单类型	开始时间	结束时间	
A2:B3:C4:00:00:D3	白名单	2015-10-04	2015-10-31	修改 删除

查询记录数 1 首页 ← 上一页 **1** 下一页 → 尾页

黑白名单配置界面：

修改黑白名单

返回

MAC地址	A2:B3:C4:00:00:D3
开始时间	2015-10-04
结束时间	2015-10-31
类型	白名单
更新	

充值卡管理

充值卡可以提供给用户用来充值或续费，通过系统可以发行充值卡销售，用户购买充值卡自行到自助服务系统进行充值或续费。

充值卡分为资费卡和余额卡两种：资费卡与余额卡。

资费卡：与系统中指定的资费进行绑定，用户可以使用资费充值卡开通该卡绑定的资费的账号，也可以为自己已经订购该资费的账号进行充值。

余额卡：只能开通预付费时长和预付费流量的账号，以及为预付费流量和预付费时长类型账号续费。

充值卡状态：分为未激活，已激活，已使用，已回收四个状态，未激活的卡不可使用，必须通过激活才能销售给用户使用，每个充值卡号码只能使用一次，已使用的卡不能再次充值续费，过期的充值卡会被自动回收或人工回收，回收后的卡不可再激活销售使用。

充值卡管理界面：

The screenshot shows the ToughRADIUS system management interface. On the left is a sidebar with a logo and a navigation tree. The main area is titled '充值卡管理' (Recharge Card Management) and contains search filters and a table of card records.

Search Filters:

- 资费 (Plan): 50元包月4M套餐 (50 yuan monthly package 4M plan)
- 充值卡类型 (Recharge Card Type): 资费套餐卡 (Fee Plan Card)
- 批次 (Batch): (empty)
- 卡状态 (Card Status): (empty)
- 创建开始时间 (Creation Start Time): 2015-10-04
- 创建结束时间 (Creation End Time): (empty)

Table Headers:

类型 (Type)	批次 (Batch)	卡号 (Card Number)	面值/售价 (Face Value/Cost)	时间(月) (Time Month)	时长(小时) (Duration Hours)	流量(MB) (Traffic MB)	过期 (Expiry)	状态 (Status)	创建时间 (Creation Time)	操作 (Operations)
-----------	------------	------------------	-------------------------	--------------------	-------------------------	---------------------	-------------	-------------	----------------------	-----------------

Table Data:

资费卡	20151004	2015100410000	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410001	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410002	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410003	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410004	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410005	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410006	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410007	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410008	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410009	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410010	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410011	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410012	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410013	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410014	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410015	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410016	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410017	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410018	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>
资费卡	20151004	2015100410019	50.00	1	N/A	N/A	2016-04-30	未激活	2015-10-04	<button>激活</button> <button>回收</button>

Pagination:

查询记录数 21 | 首页 | ← 上一页 | 1 | 2 | 下一页 → | 尾页

Copyright © 2015 toughradius.net. All rights reserved.

Version 1.1.5

充值卡管理支持查询，导出，卡生成，激活，回收操作，支持按指定条件进行查询，并导出Excel格式的数据（包括密码）。

充值卡生成界面：

The screenshot shows the ToughRADIUS system management interface. On the left, there is a sidebar with a logo and navigation links for '功能导航' (Function Navigation), '系统管理' (System Management), '营业管理' (Business Management), '维护管理' (Maintenance Management), and '统计分析' (Statistics Analysis). The '系统管理' link is currently selected and highlighted.

The main content area is titled '充值卡生成' (Recharge Card Generation). It contains several input fields for generating a recharge card:

- 充值卡类型: 资费卡 (Fee Card)
- 批次号(年+月+2位序号, 如: 20150201): 20151004
- 资费: 4M包年500元
- 开始卡号(最大5位): 20000
- 结束卡号(最大5位): 20099
- 密码长度(最大为16): 8
- 面值/销售价(元): 500.00
- 授权时间(月): 12
- 总时长(小时): 0.00
- 总流量(MB): 0.00
- 过期时间: 2017-03-03

A blue '提交' (Submit) button is located at the bottom of the form.

At the bottom of the page, there is a copyright notice: 'Copyright © 2015 toughradius.net. All rights reserved.' and a version information: 'Version 1.1.5'.

营业管理

用户定义

在系统内已登记注册并可以拥有上网账号使用权的终端用户，称之为用户，每个用户可以拥有多个认证账号。

每个用户的认证账号是相互独立的，一个用户可以有多个不同资费的认证账号。

用户可以由营业操作员在营业系统开户，或通过自助服务渠道注册。

用户授权有效期

用户授权有效期因资费而不同。

预付费包月与买断包月由过期时间来控制用户账号的使用有效期。

预付费时长与预付费流量靠余额来控制用户账号的使用权限。

买断时长和买断流量靠用户剩余时长与剩余流量来控制用户账号的使用权限。

用户订单

当用户受理产生相关缴费退费费用时，比如开户，销户，续费，充值操作，营业系统会为高操作生成订单，订单记录了用户名，上网账号，资费，费用，支付状态，受理渠道，受理时间信息。

用户计费账单

订购预付费时长，和预付费流量的用户认证账号采用短周期扣费的方式，比如1分钟或5分钟一次，在用户使用账号的过程中会对每一次扣费记录，生成计费账单。

维护管理

维护管理模块提供了一些专为运维人员设计的管理功能。

查询统计

开发指南



Git是一个分布式的版本控制系统，最初由Linus Torvalds编写，用作Linux内核代码的管理，现在它已经成为最流行的版本管理工具。Github是一个开源代码库以及版本控制系统，同时GitHub还是一个开源协作社区，ToughRADIUS的代码库正式托管在此。如果你不喜欢Github和Docker，我们是很难在一起愉快的玩耍了。

定制你的ToughRADIUS专有版本

想拥有自己的ToughRADIUS专有版本吗，想保持与官方版本的同步同时加入自己的定制特性而不冲突吗，这当然是可以的。

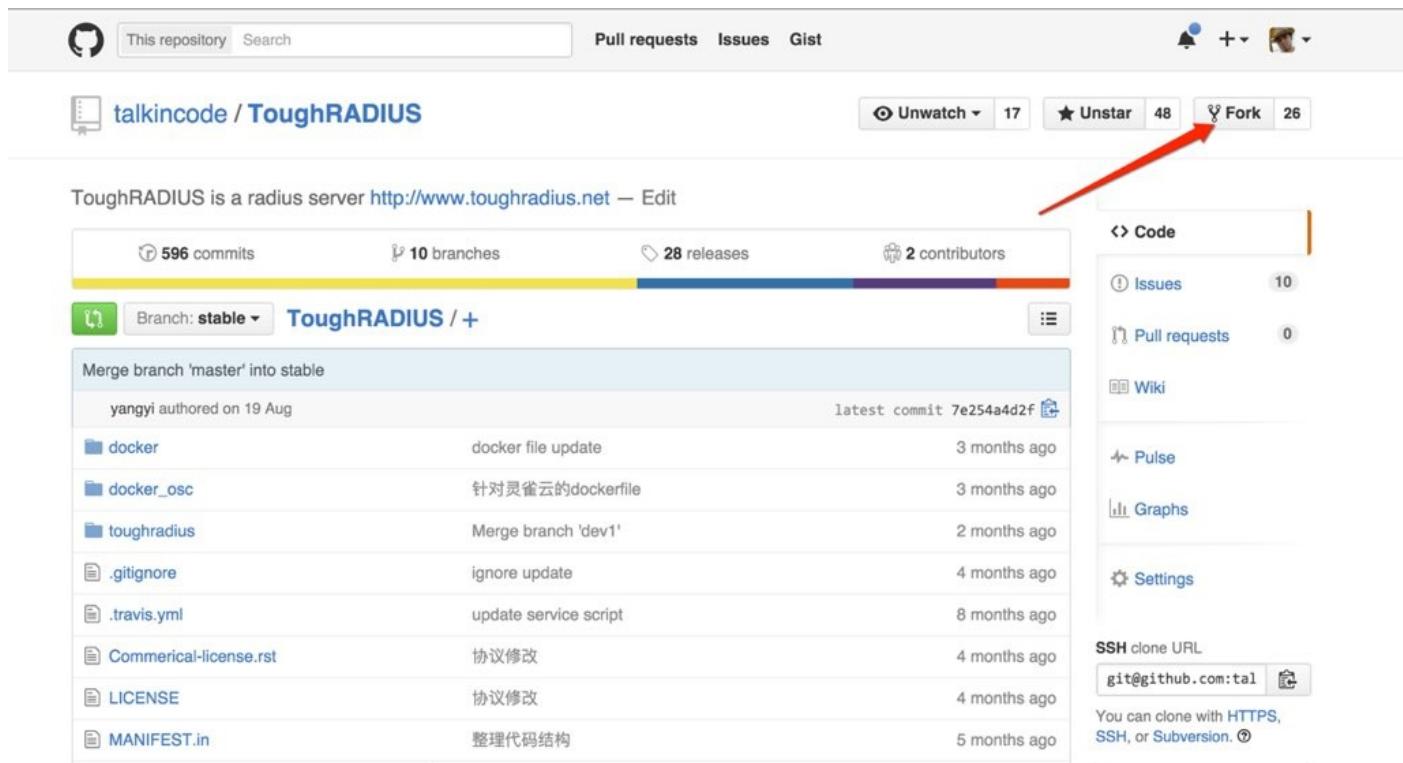
准备

你是一个程序猿或接近猿类。

拥有一个自己的Github帐号，<https://github.com/>

拥有一个Docker容器云平台的帐号，推荐国内灵雀云，<http://www.alauda.cn/>

fork一个ToughRADIUS仓库



The screenshot shows the GitHub repository page for `talkincode / ToughRADIUS`. The header includes a search bar, navigation links for 'Pull requests', 'Issues', and 'Gist', and a notification bell icon. The main content area displays repository statistics: 596 commits, 10 branches, 28 releases, and 2 contributors. Below this is a list of recent commits, all authored by `yangyi` on August 19. On the right side, there's a sidebar with links for 'Code', 'Issues' (10), 'Pull requests' (0), 'Wiki', 'Pulse', 'Graphs', 'Settings', and an 'SSH clone URL' section. A red arrow points to the 'Fork' button in the top right corner of the header.

在windows平台，你可以使用 [GitHub Desktop](#) 来进行本地仓库管理。

遵循git版本管理的规范进行开发是你要自己把控的，保证自己的定制特性，同时保持对官方版本的同步合并。

选择开发工具

通常pycharm这样的ide是不错的选择。

```

FROM centos:centos7
MAINTAINER jamiesun <jamiesun.net@gmail.com>

ADD docker_osc/radiusd.conf /etc/radiusd.conf
ADD docker_osc/supervisord.conf /etc/supervisord.conf
ADD docker_osc/toughrad /usr/bin/toughrad

RUN chmod +x /usr/bin/toughrad

RUN mkdir -p /var/toughradius/data

ADD docker_osc/privkey.pem /var/toughradius/privkey.pem
ADD docker_osc/cacert.pem /var/toughradius/cacert.pem

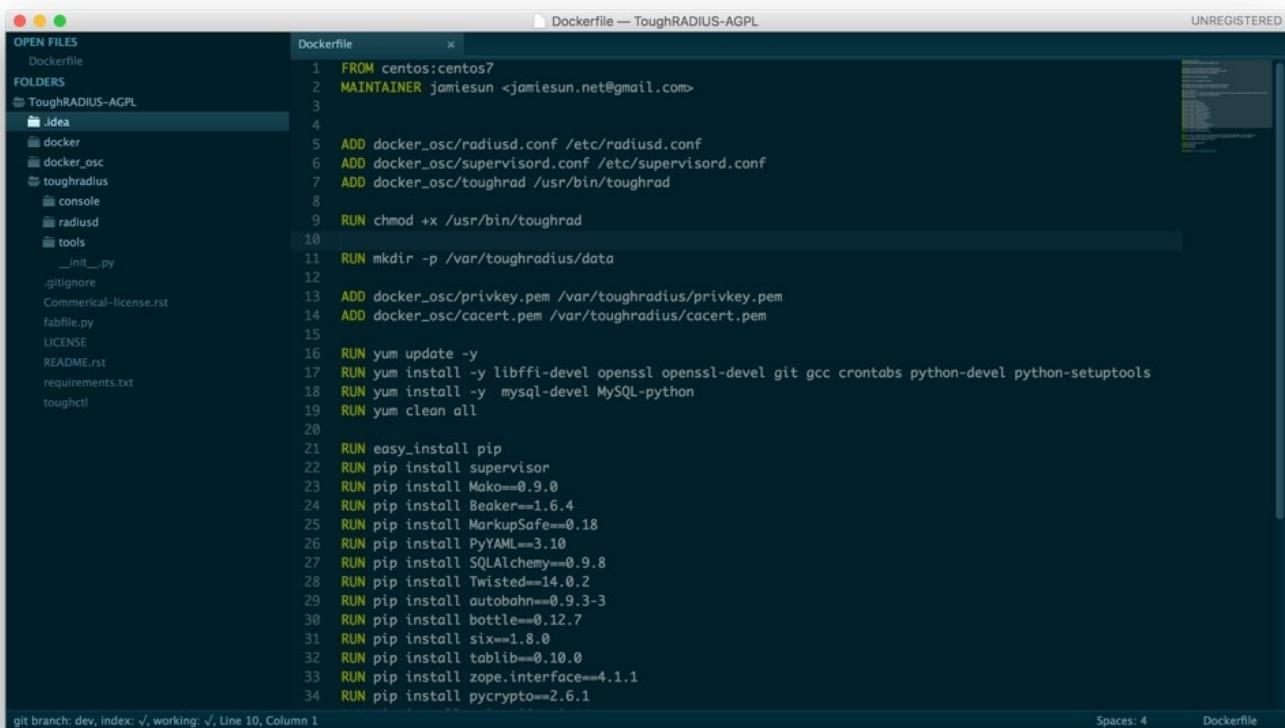
RUN yum update -y
RUN yum install -y libffi-devel openssl-devel git gcc crontabs python-devel python-setuptools
RUN yum install -y mysql-devel MySQL-python
RUN yum clean all

RUN easy_install pip
RUN pip install supervisor
RUN pip install Mako==0.9.0
RUN pip install Beaker==1.6.4
RUN pip install MarkupSafe==0.18
RUN pip install PyYAML==3.10
RUN pip install SQLAlchemy==0.9.8
RUN pip install Twisted==14.0.2
RUN pip install autobahn==0.9.3-3
RUN pip install bottle==0.12.7
RUN pip install six==1.8.0
RUN pip install tablib==0.10.0
RUN pip install zope.interface==4.1.1
RUN pip install pycrypto==2.6.1
RUN pip install pyOpenSSL==0.14
RUN pip install service_identity

RUN git clone -b stable https://github.com/talkincode/ToughRADIUS.git /opt/toughradius
RUN ln -s /opt/toughradius/toughctl /usr/bin/toughctl && chmod +x /usr/bin/toughctl
RUN /opt/toughradius/toughctl --initdb

```

但ide并非必须的，一个编辑器足以胜任，尤其是像sublime text 这样的精品。



```
FROM centos:centos7
MAINTAINER jamiesun <jamiesun.net@gmail.com>

ADD docker_osc/radiusd.conf /etc/radiusd.conf
ADD docker_osc/supervisord.conf /etc/supervisord.conf
ADD docker_osc/toughrad /usr/bin/toughrad

RUN chmod +x /usr/bin/toughrad
RUN mkdir -p /var/toughradius/data

ADD docker_osc/privkey.pem /var/toughradius/privkey.pem
ADD docker_osc/cacert.pem /var/toughradius/cacert.pem

RUN yum update -y
RUN yum install -y libffi-devel openssl-devel git gcc crontabs python-devel python-setuptools
RUN yum install -y mysql-devel MySQL-python
RUN yum clean all

RUN easy_install pip
RUN pip install supervisor
RUN pip install Mako==0.9.0
RUN pip install Beaker==1.6.4
RUN pip install MarkupSafe==0.18
RUN pip install PyYAML==3.10
RUN pip install SQLAlchemy==0.9.8
RUN pip install Twisted==14.0.2
RUN pip install autobahn==0.9.3-3
RUN pip install bottle==0.12.7
RUN pip install six==1.8.0
RUN pip install tablib==0.10.0
RUN pip install zope.interface==4.1.1
RUN pip install pycrypto==2.6.1
```

定制Docker构建脚本

修改 Dockerfile 构建脚本（针对灵雀云的在 docker_osc 目录），修改以下内容：

```
RUN git clone -b stable https://github.com/talkincode/ToughRADIUS.git /opt/toughradius
```

修改为：

```
RUN git clone -b master <你的仓库地址> /opt/toughradius
```

修改服务脚本 toughrad（针对灵雀云的在 docker_osc 目录）修改以下内容：

```
upgrade()
{
echo 'starting upgrade...'
cd ${appdir} && git pull origin stable
git checkout stable
supervisorctl restart radiusd
supervisorctl restart admin
supervisorctl restart customer
supervisorctl status
```

```
echo 'upgrade done'  
}'
```

修改为：

```
upgrade()  
{  
echo 'starting upgrade...'  
cd ${appdir} && git pull origin master  
git checkout master  
supervisorctl restart radiusd  
supervisorctl restart admin  
supervisorctl restart customer  
supervisorctl status  
echo 'upgrade done'  
}
```

开始定制你的专有版本

现在可以开始加入你自己的代码了。

广告：欢迎加入ToughRADIUS的QQ开发群组讨论：487229323

可以修改一个自己的酷炫版本号：

文件在 `../toughradius/init.py`

```
#!/usr/bin/env python  
  
__version__ = 'ohmyradius_v1.2.0.1'  
__license__ = 'AGPL'  
__author__ = 'jamiesun.net@gmail.com'
```

完成之后就提交你的代码吧。

在灵雀云建立自己的构建镜像库

使用你的帐号进入灵雀云控制台，选择镜像仓库，创建镜像构建仓库。

ToughRADIUS 教程



The screenshot shows the LinqCloud dashboard. At the top, there's a navigation bar with links for Home, Products, Prices, Activities, Documentation, Demonstrations, Blogs, and Mirror Center. On the right side of the header, there's a user profile icon for 'tougttech'. Below the header, there are several service icons: Services, Mirror Library, Build, Storage Backup, and Dedicated Host. To the right of these are more icons for Organization Settings, Free Acceleration, Service Status, Work Orders, and Bills. The main content area has a breadcrumb path 'Home / Mirror Library'. It displays a message 'You haven't created any mirror library yet' and a large blue button labeled '+ Create Mirror Library'. A dropdown menu from this button shows two options: 'Mirror Library' and 'Mirror Build Library', with 'Mirror Build Library' being highlighted with a red box. At the bottom of the page, there's a footer with copyright information: 'Copyright © 2015 - LinqCloud Technology Beijing ICP Registration No. 15011102' and an email address 'info@mathildetech.com'.

你可以在灵雀云关联绑定你的Github仓库，也可以选择快速创建。



The screenshot shows the 'Create Mirror Library' form. At the top, there are tabs for 'Select Repository' and 'Quick Create'. The 'Quick Create' tab is selected, indicated by a green border. Below the tabs, there's a note: 'Note: When creating a mirror build library via quick creation, you need to manually trigger the build of the mirror library.' There are two bullet points: 'When using SSH download address to create a mirror build library, we will generate a deployment public key, and the repository access permission can be private or public.' and 'When using HTTPS download address to create a mirror build library, we will not generate a deployment public key, so please ensure that the repository access permission is public.' The form fields include 'Repository Name*' with the value 'tougttech / ohmyradius' and 'Repository Download Address*' with the value 'https://github.com/jamiesun/ToughRADIUS.git'. At the bottom right are 'Create' and 'Reset' buttons. The footer at the bottom of the page is identical to the one in the previous screenshot.

修改构建配置，在构建节点的选择上注意，如果选择国际节点，在构建速度上会有优势，因为 ToughRADIUS 依赖的很多开发库在国外，但是如果你的部署服务器国际带宽不够大，那么在初次部署时会比较慢。选择国内节点则反过来，如何选择请根据自己的情况判断，可以随时重新调整。

The screenshot shows the configuration interface for a Docker image named 'ohmyradius'. It includes fields for build service nodes (set to 'International'), Dockerfile directory ('/ docker_osc'), and Git repository ('https://github.com/jamiesun/ToughRADIUS.git'). A note indicates that持续构建 (Continuous Build) is not supported for this type of build.

如果您需要开启持续构建，选择“开启持续构建”并保存构建配置，我们会为您生成Web钩子URL。由于您使用的是快速构建方式创建此镜像构建仓库，所以我们无法将Web钩子URL自动添加到您的代码仓库设置中，请您手动添加。

代码仓库: <https://github.com/jamiesun/ToughRADIUS.git> (Simple)

构建服务节点: 国际

完成时邮件通知:

镜像版本: latest

代码分支: master

构建上下文目录: /

Dockerfile 所在目录: / docker_osc

添加时间版本: 开启持续构建:

备注: 分支和镜像仓库必须不同，并且版本名称也必须唯一。当您开启持续构建时，我们会为您创建push钩子，当代码更新时，会触发自动构建。此外，我们还会创建deploy keys以便构建时下载代码。请确保您对该代码仓库有相应的权限。

保存 取消

点击开始构建按钮，等待构建完成。

ToughRADIUS 教程

The screenshot shows the Alauda Cloud Platform interface. At the top, there's a navigation bar with links for Home, Products, Prices, Activities, Documentation, Demonstrations, Blogs, and Mirror Center. On the right side of the header, there's a user profile icon for 'toughtech' and a sign-out button. Below the header, there are several icons for different services: 服务 (Services), 镜像仓库 (Image Registry), 构建 (Build), 存储卷备份 (Storage Volume Backup), and 专属主机 (Exclusive Host). To the right of these are more icons: 机构设置 (Organization Settings), 免费加速器 (Free Accelerator), 服务状态 (Service Status), 工单 (Work Order), and 账单 (Bill).

The main content area shows a project named 'ohmyradius'. It has a green '镜像构建仓库' (Image Build Repository) button. Below it is a table with details:

ohmyradius			
属性	公共的		
创建时间	2015-10-05 02:25:35	更新时间	2015-10-05 02:25:35
构建服务节点	国际	代码仓库	https://github.com/jamiesun/ToughRADIUS.git
分享链接	https://hub.alauda.cn/repos/toughtech/ohmyradius		

Below this is a navigation bar with tabs: 信息 (Information), 版本 (Version), and 构建 (Build). The '构建' tab is selected. A table below shows build configurations:

镜像版本	代码分支	构建上下文目录	Dockerfile 所在目录	
latest	master	/	/docker_osc	<button>开始构建</button>

At the bottom of the build section, there's a summary row:

进行中	2d92b34d	toughstruct	toughtech/ohmyradius	2015-10-05 02:32:29
			latest, v20151004.183229	

In the footer, there's a copyright notice: 版权 © 2015 - 云雀科技 京ICP备15011102号-2 联系我们: info@mathildetech.com

可以实时查看构建过程：

ToughRADIUS 教程

The screenshot shows the Alauda Cloud Platform's Docker build interface. At the top, there are navigation links for Home, Products, Prices, Activities, Documentation, Demonstrations, Blogs, and Mirror Center. On the right, there are user profile, search, and account management links. Below the header, there are service-related icons: Service, Mirror Library, Build, Storage Volume Backup, and Dedicated Host. To the right of these are settings, acceleration, status, and bill links.

The main content area shows a build progress for a Docker image named `toughtech/2d92b34d`. The status is "进行中" (In Progress). The build ID is `2d92b34d-9e45-46d1-b94c-49c9fba27fb7`. The build details section includes the repository (`toughtech/ohmyradius`), tag (`latest, v20151004.183229`), URL (`https://github.com/jamiesun/ToughRADIUS.git`), branch (`master`), and committer (`0dec0965`). A "Delete" button is also present.

The "Build Progress" section displays the terminal logs of the Docker build process:

```
2015-10-05 02:34:07 =====
2015-10-05 02:34:07 Updating:
2015-10-05 02:34:07 coreutils           x86_64      8.22-12.el7.1.2      updates      3.2 M
2015-10-05 02:34:16 --> Processing dependency: perl-libs@ 4:5.16.3-285.el7.1.2 for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Socket) >= 1.3 for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Scalar::Util) >= 1.10 for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl-macros for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl-LibXS for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(threads::shared) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(threads) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(constant) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Time::Local) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Time::HiRes) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Storable) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Socket) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Scalar::Util) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Pod::Simple::XHTML) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Pod::Simple::Search) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Filter::Util::Call) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: perl(Carp) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 --> Processing Dependency: libperl.so()(64bit) for package: 4:perl-5.16.3-285.el7.x86_64
2015-10-05 02:34:16 -->> Package perl-Error.noarch 1:0.17020-2.el7 will be installed
2015-10-05 02:34:16 --> Package perl-Exporter.noarch 0:5.68-3.el7 will be installed
2015-10-05 02:34:16 --> Package perl-File-Path.noarch 0:2.09-2.el7 will be installed
2015-10-05 02:34:16 --> Package perl-File-Temp.noarch 0:0.23.01-3.el7 will be installed
2015-10-05 02:34:16 --> Package perl-Getopt-Long.noarch 0:2.40-2.el7 will be installed
2015-10-05 02:34:16 --> Processing Dependency: perl(Pod::Usage) >= 1.14 for package: perl-Getopt-Long-2.40-2.el7.noarch
2015-10-05 02:34:16 --> Processing Dependency: perl(Text::ParseWords) for package: perl-Getopt-Long-2.40-2.el7.noarch
2015-10-05 02:34:16 --> Package perl-Git.noarch 0:1.8.3.1-4.el7 will be installed
2015-10-05 02:34:16 --> Package perl-PathTools.x86_64 0:3.40-5.el7 will be installed
2015-10-05 02:34:16 --> Package perl-TermReadKey.x86_64 0:2.30-20.el7 will be installed
2015-10-05 02:34:16 --> Package python-backports-ssl_match_hostname.noarch 0:3.4.0.2-4.el7.noarch will be installed
2015-10-05 02:34:16 --> Processing Dependency: python-backports for package: python-backports-ssl_match_hostname-3.4.0.2-4.el7.noarch
2015-10-05 02:34:16 -->> Package rsync.x86_64 0:3.0.9-15.el7 will be installed
```

构建完成后，就可以在你的服务器上使用Docker模式部署了，只是Docker仓库地址是你自己的了。

```
$ docker run -d --name trserver --net host index.alauda.cn/toughtech/ohmyradius
```

是不是酷毙了，你甚至可以把你得意的定制版本向更多朋友分享。

继续保持对你专有ToughRADIUS版本的更新维护吧。