tollgate Documentation

Release 3.0.0

Michael Farrell

May 05, 2012

CONTENTS

tollgate is a captive portal system for Linux.

You can find out more at the website or the GitHub project.

Contents:

CHAPTER

ONE

INTRODUCTION

Welcome to tollgate. This is a captive portal system for Linux, designed for operating LAN parties. A lot of the concepts in the software are specific to how a LAN party operates, however you could use it for a sharehouse if you wanted, or something else.

It consists of two parts, connected via dbus:

- A frontend system, which does most of the heavy lifting, including managing users and quota. It is a Django website.
- A backend system. This is only there to insulate the frontend from running programs as root directly. It also abstracts calls to the firewall, and maintains the list of unmetered and blacklisted hosts.

It's important to note that tollgate only manages routing on your network. A typical network will also need a DHCP server and DNS. tollgate simply sets up NAT for you, and manages client access to the internet.

tollgate is a little bit different to most captive portal solutions because it doesn't use RADIUS at all. It is entirely managed within your database, which can run on anything Django can such MySQL or SQLite.

1.1 Requirements

tollgate will only run on Linux. You do not need to have your DHCP or DNS servers on the same machine (though is it recommended).

tollgate has some limited support for running the frontend, for development purposes, on Mac OS X and Windows. However, the backend **will not work** on these systems.

At the time of writing, your author is not aware of any other operating system that supports the needed functionality to make the system work. See *Porting to non-Linux systems*.

The following packages must be installed:

- Python 2.5 or later
- Django 1.2 or later, as well as a supported database (such as MySQL or SQLite3).
- WSGI-compliant web server, such as apache2. It is strongly recommended that you run the site using HTTPS only, so you will also need `mod_ssl`.
- Linux 2.6.28 or later with netfilter (most distributions ship with this).
- iptables 1.4.3 or later.
- xtables-addons 1.22 or later.
- dbus

- django-south
- djangorestframework
- nmap
- python-daemon
- python-dbus
- python-iplib or python-ipy
- python-lxml
- python-simplejson (if using Python 2.5)
- python-tz

Optional dependencies:

- mrab-regex-hg

1.2 Client Platform Support

tollgate should allow any client with an IPv4 stack and a web browser to work with it. The software has been tested with (but support is not limited to):

- Apple Mac OS X 10.4 and later
- FreeBSD
- Linux
- Microsoft Windows 98 and later
- Solaris

Because the software is designed for running LAN Parties, it also has support for gaming consoles, including those without web browsers, such as:

- Microsoft Xbox [1] [2]
- Microsoft Xbox 360 [2]
- Nintendo DS [3]
- Nintendo DSi / 3DS
- Nintendo Gamecube [1] [4]
- Nintendo Wii
- Sega Dreamcast [1] [3] [4]
- Sony Playstation 2 [3] [4]
- Sony Playstation 3
- Sony Playstation Portable [1]

[1] This platform is untested, however should work.
[2] Platform lacks a web browser, so requires you to login from another device which does.
[3] Platform has a web browser application which isn't installed by default or requires purchase. The web browser is not required, but will require you login with another device which has a web browser if you don't.
[4] Platform does not come with an in-built ethernet or wireless adapter as standard.

tollgate allows you to claim ownership of another device remotely, so if the device does not have a web browser, you can use this to login the device from something else.

CHAPTER

TWO

DEPLOYING TOLLGATE

2.1 Deploying tollgate into a Django project

The "proper" way to deploy tollgate is to install the software (using `setup.py`), and then create a Django project with tollgate setup inside of it.

This is achieveable fairly simply, however be aware that **tollgate only manages routing**, it does **not** manage things like DNS and DHCP which you'll need to make your network actually accept clients.

This has the advantage of allowing you to easily customise configuration and templates. Be aware though that all modifications to tollgate **must** be made available, as well as the software itself, to all of your users, as a condition of the license.

2.1.1 Pre-requisites

I'm assuming here that you have:

- Installed and configured an apache2 server with mod_wsgi and mod_ssl.
- Installed and configured a database server, for example, MySQL, as well as installed appropriate Python bindings to allow interaction.
- Installed everything else you need to make your network work – that is, DHCP server, DNS server, multiple network interfaces in your tollgate machine (which will be your router / default gateway).
- Installed and configured DBUS.
- Installed other dependencies.

Installation and configuration of those is outside of the scope of this document. If you're looking up HOWTO documents on the internet, do not do anything with *iptables*, as setting up a NAT and routing itself is part of tollgate.

2.1.2 Install tollgate

Install tollgate, either using an official stable build, git repository, or distribution package. You can install the latest *master* version of tollgate using *pip* with this command:

```
$ sudo pip install git+https://github.com/micolous/tollgate.git
```

This **may** not work though, as the state of *git master* may be in flux.

This will install the entire *tollgate* package into your Python path, and install the captivity and backend daemons.

2.1.3 Configure DBUS

We need to add some configuration files for tollgate to DBUS' configuration in order to allow the web server process to use tollgate's backend.

In `docs/example/dbus/system.d/tollgate.conf` are some example configuration you can use with tollgate. Copy this to `/etc/dbus-1/system.d/`, and modify with the appropriate username that the webserver uses (if it is not `www-data`).

Then reload the DBUS configuration with `/etc/init.d/dbus reload`.

2.1.4 Create a project

Now, you should create a Django project for tollgate to use. This won't have any of tollgate's code in this folder – it will reference a the system-installed copy.

```
$ django-admin startproject mylanportal
```

This will create some boilerplate code for a Django site. Currently, tollgate doesn't support not being at the root of the site, but this may change in the future.

From here on in, I'm going to assume your project name is *mylanportal*.

2.1.5 Configure the project

Jump into the `mylanportal/mylanportal/urls.py`. Change it so it includes `tollgate.urls`. `tollgate.urls` will also give you the Django admin site, and the internationalisation configuration app. It should look something like this:

```
from django.conf.urls.defaults import patterns, include, url
urlpatterns = patterns('',
      (r'^', include('tollgate.urls')),
)
```

The next step is to setup `settings.py`.

Near the top, add these lines:

```
from os.path import *
PROJECT_PATH = realpath(dirname(__file__))
```

This is a handy trick because you can use it to setup other paths later.

Setup the location of the database. It is recommended you use MariaDB (MySQL).

You should also setup a `STATIC_ROOT` for where all the static files should be served from, and a `STATIC_URL`. Be aware that if you are deploying on a HTTPS site (which you should!) you need to make your resources also be on a HTTPS site. The purpose of this is that outside of DEBUG mode, you're expected to serve static files external to Django – as it is much faster.

To your `INSTALLED_APPS`, append:

```
'django.contrib.humanize',
'django.contrib.admin',
'djangorestframework',
'south',
'tollgate.api',
'tollgate.frontend',
'tollgate.scripts'
```

You should also add the following extra settings for tollgate and configure appropriately:

```
AUTH_PROFILE_MODULE = 'frontend.userprofile'
LAN_SUBNET='10.4.0.0/23'
LAN_IFACE='eth1'
DEFAULT_QUOTA_AMOUNT=150
RESET_EXCUSE_REQUIRED=True
RESET_PURCHASE=False
ONLY_CONSOLE=False
RESTRICTED_CALLS_KEY=''
LOGIN_URL='/login/'
LOGOUT_URL='/logout/'
```

The final setting to add is a URL where you are hosting the tollgate sources with your modifications, `SOURCE_URL`. You should **never** link back to the official tollgate repository using this method (there is already a link to the official repo on the source page).

Not hosting the source code yourself may expose you to legal liability.

2.1.6 Configure daemons

Install the init scripts and backend configuration:

```
$ sudo cp platform/debian/init.d/* /etc/init.d/
$ sudo cp platform/debian/default/* /etc/default/
$ sudo mkdir /etc/tollgate/
$ sudo cp example/tollgate/backend.ini /etc/tollgate/
```

Modify the scripts (`tollgate-backend` and `tollgate-captivity`) as appropriate to match the path to the tollgate_backend and tollgate_captivity scripts.

Edit `/etc/default/tollgate-captivity` to point to the URL where tollgate is hosted.

To make the daemons start, run:

```
$ sudo update-rc.d tollgate-backend defaults
$ sudo update-rc.d tollgate-captivity defaults
```

Modify the backend configuration as appropriate for your network (`/etc/tollgate/backend.ini`).

We won't start the daemons just yet, though.

2.1.7 Configure cron

tollgate requires a periodic cronjob to refresh the list of hosts in it's database.

An example configuration is given in `docs/example/tollgate.cron`. You will need to adapt it to point to the path of your Django project.

2.1.8 Configure webserver

You'll need to now configure your web server.

If you are using Django 1.3 or earlier, you may wish to copy `tollgate/tollgate.wsgi` and use it in your own project folder. However, be sure to change the `DJANGO_SETTINGS_MODULE` to the name of your project (eg: `mylanportal.settings`), as tollgate itself includes a `tollgate.settings` for use in development deployment.

In Django 1.4 or later, it will create a file named like `mylanportal/wsgi.py` with settings that you should use instead.

There is an example apache2 configuration, including all vhosts, in `docs/example/apache2/tollgate-vhost`.

You will need to modify the path of static items (like the WPAD and WFC vhosts, and aliases for static files) to the appropriate locations, and URLs.

Included in the examples is how to configure a gitweb instance. You could also push code changes to an external repository, however it must be accessible to users at all times (ie: you should mark it as "unmetered").

2.1.9 Start the daemons

The first time you run you'll need to manually start the daemons. They will start automatically on next boot.

2.2 Deploying tollgate in development

In development, you can run and deploy `tollgate` from within a git clone of the repository. This is the "old" way of deploying tollgate in production, and has since been superceeded.

You can run tollgate in development either out of a WSGI-compatible webserver, or using Django's single-threaded development server.

2.2.1 Useful Functions

repair_permissions

```
$ python manage.py repair_permissions
```

Repairs execute permissions on scripts.

setup_settings

```
$ python manage.py setup_settings
```

Creates a `tollgate/settings/local.py` for your local settings, and configures your `SECRET_KEY`.

2.3 Clustering tollgate with CARP

tollgate can run in a clustered configuration with CARP (Common Address Redundancy Protocol). You'll need to also set up redundant DHCP, DNS and database (eg: multi-master MySQL, or a single external database server) for this to work.

tollgate's quota saving procedures are written in such a way that it will work with multiple copies of tollgate simultaneously. No special configuration of tollgate is required in order for it to work (apart from possibly changing database settings).

However, there is a window (between `refresh_hosts` calls, normally every 10 minutes) where you can use all of your quota via one tollgate and still have it available on the other, because the counters aren't synchronised live (and doing so is quite expensive).

In typical deployments however I haven't had this as a real problem, as it hasn't been possible to use more than 50% of the allocated quota in 10 minutes. Doing so would require quite fast internet access, and you're generally competing for that resource with other clients on the network.

Be sure when configuring your network infrastructure for redundancy that:

- Your two tollgate machines have different power sources. This could mean they're supplied via a different mains circuit, or one of them has a battery backup.
- You also provide redundancy for the switch, if you have one.
- You have either a multi-master database server setup, or a single database server with redundant power supplies or battery backup.
- If running with one database server, make sure that if one half of your power goes down, that the database server is still accessible (ie: use two switches and two NICs in your database server).
- Use protocols like Spanning Tree Protocol (STP) on your switches to break routing loops.

At the moment, tollgate doesn't support running multiple instances of itself managing *different* subnets. That's a plan for down the track.

2.4 Running on large subnets (bigger than /24) or with more than 128 hosts

You may encounter performance issues and hosts dropping out "randomly" when running the software on subnets larger than a /24. This is because of the size of the ARP table in Linux is effectively limited to 128 hosts, and the software will automatically send large amounts of ARP requests to see who currently holds each IP address on the network.

2.4.1 Reality Check!

It is at this point you should seriously consider the size of your subnet. If you have less than 200 hosts on your network, then you really only need a /24. If you have a proper network plan in place, with DNS and static DHCP entries setup, you can still segment your network a lot more tightly. You can use hostnames to provide memorable names to services, rather than wanting `10.0.13.37` when all your other hosts are in `10.0.1.0/24`.

When you're planning for a LAN party, I generally do the math based on:

```
hosts = (maximum_attendance * 2) + static_hosts
```

You should only be using a `/16` if you're expecting in excess of 30,000 people attending your LAN. And even then you should consider slicing it up into subnets, because most operating systems have an ARP cache limit of about 1024 hosts, and you'll have problems with broadcast packets. Even something as simple as a Master Browser Election could knock out your network (though you should be *Using WINS* at this point).

With dynamic DNS assignments by DHCP and routing in place, you can even keep it so that hostnames across subnets can still talk to each other by name. Without this, you'll end up with a lot of "noise" on your network from all sorts of multicast protocols.

At this point of time though, you'll need to setup multiple copies of tollgate: one to service each network. However, each instance should be able to share a single database provided the IP addresses are unique.

There are, of course, some applications and games which simply won't work because they require multicast or link-local packets. But it is also those games which become increasingly unreliable on large networks.

2.4.2 Tweaking Linux's ARP table

You can tweak the behaviour of the ARP cache on Linux to let you have a bigger ARP table. But this comes at a price – it uses more memory, and the cron job for tollgate's refresh process will take much longer.

Linux provides three settings in `/proc/sys/net/ipv4/neigh/default/`:

- `gc_thresh1`: 128 hosts. This is the minimum number of entries to keep in the ARP cache. The garbage collector will not run if this amount isn't exceeded, and will reduce the number of entries every 30 seconds by default.
- `gc_thresh2`: 512 hosts (gc_thresh1 * 4). This is the soft-maximum number of entries to keep in the ARP cache. The garbage collector will allow this to be exceeded for 5 seconds.
- `gc_thresh3`: 1024 hosts (gc_thresh2 * 2). This is the hard-maximum number of entries to keep in the ARP cache. It will always run if there are more entries in the cache.

You should keep those ratios if you adjust it, but gc_thresh needs to be able to handle the base amount of hosts on your network.

`tollgate-backend` will automatically set this for you if you set the `arp_table_size` option in `backend.ini`.

This will automatically set all three garbage collector thresholds appropriately according to the ratios above.

You absolutely require this value to be set to the number of hosts in your subnet, with a little bit of leeway for your WAN ethernet interface. Which means if you have a `/23` (512 IPs) on your LAN side, and about 10 machines on your WAN side, you should set the value to about 530 (enough for both sides with some leeway):

```
arp_table_size = 530
```

If you set it to exactly 512, then the non-result ARP table entries will push out legitimate ones, and also entries from your WAN side will push out entries from your LAN size.

2.5 Windows Clients

While this isn't a core issue inside of tollgate, there's a pretty strong chance when running LAN Party events that you will have a large amount of Microsoft Windows hosts.

There are many things that Windows doesn't handle properly, which will require some manual tweaking to sort out. Most of these problems you will be blamed "for breaking it", despite there being problems in the Windows OS.

Note: These issues are not caused by tollgate. They are simply included in this guide because they are problems not often documented in a single place.

Here are some problems your author has encountered in the past:

2.5.1 Multiple search domains do not work

In DHCP options, you can offer multiple DNS search domains. On Windows, only the first search domain will be used.

You should separate your static (official) hosts and dynamic (user) hosts into two subnets still:

```
css01.example.lan
openttd1.example.lan
irc.example.lan
jimmy-pc.dhcp.example.lan
janes-macbook-pro.dhcp.example.lan
```

You should then specify the resolution order as follows:

```
example.lan      (Windows will only use this one)
dhcp.example.lan
```

You can work around this bug, however it is an "opt-in" and requires some manual configuration in Windows:

1. Open Network and Sharing Centre.
2. Select the adapter to modify that is connected to the local network.
3. Click `Properties`.
4. Click `Internet Protocol Version 4 (TCP/IPv4)`.
5. Click `Properties`.
6. Click `Advanced`.
7. Click the `DNS` tab.
8. Select `Append these DNS suffixes (in order):`.
9. Add entries for each DNS suffix your network uses.
10. Click `OK`.
11. Click `OK`.
12. Click `Close`.
13. Click `Close`.

Then this brings us to the next bug in Windows' DNS resolver:

2.5.2 Dotted-domain lookups are never recursive

On a non-Windows machine, say you have a search domain set to `example.lan`. If you lookup `jimmy-pc.dhcp`, it will look up `jimmy-pc.dhcp.example.lan.` then `jimmy-pc.dhcp.`.

On a Windows machine, it assumes any name being resolved with a dot in it is actually being resolved as a root object (ie: `jimmy-pc.dhcp` internally becomes `jimmy-pc.dhcp.`), so it will never try to look up `jimmy-pc.dhcp.example.lan.`

We can work around this with a DNAME zone for `dhcp` similar to this:

```
dhcp. IN SOA ns1.example.com. root.example.com (
      2010012301 ; serial
      60         ; refresh (1 minute)
      60         ; retry (1 minute)
      3600       ; expire (1 hour)
      60         ; minimum (1 minute)
      )
      NS      tollgate.example.lan.

dhcp. IN DNAME dhcp.example.lan.
```

2.5.3 Web Proxy Auto-Discovery Vulnerabilities

Internet Explorer on Windows will try to discover a proxy server by doing NetBIOS lookups for the server called `WPAD` by default. As a result, a local network user may intercept all traffic from a vulnerable computer by specifying proxy settings that redirect traffic.

Included in tollgate's source repository is a site at `/www/wpad/`. This should be hosted at the server named `wpad.example.lan.` and `wpad.` (where `example.lan.` is your search domain).

Likewise, you should send DHCP option 252 to indicate an absolute path to the WPAD configuration. In ISC DHCPd, you can do this with:

```
option auto-proxy-config code 252 = string;
subnet 10.4.0.0 netmask 255.255.255.0 {
  # ... some other configuration here

  option auto-proxy-config "http://10.4.0.1/wpad.dat";
}
```

See also:

- CVE-2009-0094, 2009-03-11
- MS09-008: Vulnerabilities in DNS and WINS Server Could Allow Spoofing (962238), 2009-04-12
- MSDN Blogs: We know IE: WPAD detection in Internet Explorer, Aurthur Anderson, 2008-12-18
- Perimeter Grid: WPAD: Internet Explorer's Worst Feature, Grant Bugher, 2008-01-11
- SkullSecurity: Pwning hotel guests, Ron Bowes, published 2009-11-19

2.5.4 Using WINS

In an effort to help reduce the master browser election traffic, and assist in NetBIOS name resolution, you should setup a WINS server.

In ISC DHCPd, this is done with the following configuration option:

```
option netbios-name-servers 10.4.0.1;
```

You'll also need to run an actual WINS server too. Samba 3 provides a WINS server, but it is not enabled by default. In the `[global]` section of `/etc/samba/smb.conf`, you can enable this functionality with:

```
wins support = yes
dns proxy = yes
```

After this, reload your Samba and DHCP daemon.

2.5.5 Mass-mailing Worms

It's pretty much a given you will have problems with infected Windows hosts. One major thing you will want to consider is blocking external SMTP traffic to at least prevent your network from becoming a spam hub, and angering your ISP (as well as other internet users). You can do this with an entry in `backend.ini`, under the section `blacklist`:

```
externaldns = 0.0.0.0/25
```

Normally you only have to block port 25 traffic. SMTP over SSL is generally never used by such worms, and mail servers running on SSL generally also require authentication (which the spam bots won't have).

It will also allow legitimate senders of mail on your network to be able to continue sending mail.

Unfortunately, there isn't a simple way at this time to exempt blocking of SMTP over TLS (which uses port 25 and `STARTTLS` command). Additionally, many ISPs do not offer encrypted SMTP servers – until they are lobbied by users. ;)

2.6 Nintendo Consoles / WFC

Warning: Nintendo DS and DS Lite, as well as any DS games on the DSi and 3DS will **only** connect to wireless networks that are either unencrypted or encrypted with WEP. Additionally, they will only connect to 2.4GHz 802.11b networks.
Because of the additional radio bandwidth that 802.11b clients require, it is recommended that you run a seperate 802.11b-only network for those devices.

Note: On the Nintendo DSi and 3DS, connection profiles 1 - 3 do not support WPA or WPA2 encryption (for compatibility with DS games), only the profiles 4 - 6 support it.

All of Nintendo's gaming consoles, with the exception of the Gamecube, will probe a site called `conntest.nintendowifi.net` during connection setup.

If this site is inaccessible or does not return a "200 OK" response, the console will assume it cannot connect to the internet, and refuse to save the connection profile.

Included in tollgate's source repository in `/www/wpad/` is a website you can host at `conntest.nintendowifi.net`, with a DNS record pointing to your server. This must be accessible inside of your LAN.

2.7 Playstation Portable (PSP)

Warning: Playstation Portable will only connect to 2.4GHz 802.11b networks, and does not support WPA2 encryption.
Because of the additional radio bandwidth that 802.11b clients require, it is recommended that you run a seperate 802.11b-only network for those devices.

Warning: Playstation Portable E-1000 does not have WiFi.

PSP System software v2.00 includes a web browser. Earlier versions of the system software do not include a web browser.

If you wish to sign earlier versions of the PSP into tollgate, you will need to do it from another device with a web browser.

2.8 Consoles without web browsers

The general process for logging a system into tollgate when the device does not have a web browser is:

1. Set the hostname of the device to be something uniquely and easily identifiable.
2. Connect the device to the network.

3. Attempt a connection test (this will fail).
4. Find the device in tollgate's *login other computers* screen, and sign it in.
5. Reattempt the connection test (this should succeed).

After this, the device will be registered with that user's account. Whenever they are signed into the event they will automatically grant access to the internet for all of their devices.

2.9 Rogue DHCP / DNS Servers

There have been several instances at events your author has administed where Windows worms propegating on the network will send out rogue DHCP server responses, attempting to either route traffic through the infected machine, or replace DNS with a third-party server that will redirect traffic to popular websites through an attacker's server.

There are two major mitigation steps you should take:

2.9.1 Block external DNS servers

This can be done in `backend.ini`, by adding a blacklist line like:

```
externaldns = 0.0.0.0/53
```

This will only allow your DNS server, and any whitelisted / unmetered servers to have DNS traffic passed through to them.

2.9.2 Use layer 3 managed switches with DHCP filtering

Layer 3 managed switches offer various filtering options. You can limit the spread of a rogue DHCP server by:

1. Only allowing DHCP to be served from the tollgate server(s) port(s) on the backbone switch.
2. Only allowing DHCP to be served from the port(s) connecting to the backbone switch for leaf switches.

If you are low on budget, there's a good chance that you will not be able to afford all Layer 3 managed switches. In this case, save the money for at least one on your backbone, so any rogue DHCP server issues will be limited to one leaf switch, and you'll be able to quickly determine which host is compromised.

CHAPTER

THREE

DEPLOYING ON FEDORA

3.1 Before you start

- You must have a computer with two network interfaces - One to the internet, the other to your LAN.
- You must have rpmfusion-free enabled.
- You must be running Fedora 16 or higher.
- You must have updated your system (`yum update`)
- You should run SELinux in permissive mode (`/etc/selinux/config`). While we have an SELinux policy package, at this time, it is not 100% guaranteed to work on a live system. If you feel brave, run in enforcing mode and notify us of encountered errors with attached AVC messages.

3.2 Installing the software

Add the tollgate repository, or grab the rpms manually.

You can install the tollgate repository by running:

```
yum localinstall --nogpgcheck 'http://repo.tollgate.org.au/pub/fedora/tollgate-repo.noarch.rpm'
```

Install tollgate by running:

```
yum install tollgate
```

- All docs and examples are provided in the tollgate rpm. They can be found in `/usr/share/doc/tollgate/`

3.3 Core network

Make sure your networking is set to start on boot.:

```
systemctl enable network.service
systemctl start network.service
```

We need to allow certain traffic into our system. This should be configured by iptables. An example set of rules can be found in `example/fedora/iptables` and should be placed into `/etc/sysconfig/iptables`. Once you have configured these rules, reload iptables with:

```
systemctl restart iptables.service
```

We can setup the network either with DNSMASQ or ISC-DHCP and BIND9. This will document how to install ISC-DHCP and BIND9.

Install the packages:

```
yum install dhcp bind bind-utils
```

Setup your LAN facing network device with a static IP address. There is an example of this in example/fedora/ifcfg-lan, and the file you want to edit will be /etc/sysconfig/network-scripts/ifcfg-DEVICENAME.

Additionally, ensure that your internet facing device is set to ONBOOT="yes"

Once configured run.:

```
ifup DEVICENAME
```

Next, we setup ISC-DHCP. This will provide DHCP addresses to your LAN network. Make sure you get this right, else you will have a DHCP conflict on your Internet side. There is an example config in example/fedora/dhcpd.conf.

Before you can start DHCP, you must create the rndc key that will be shared with named. Run the command:

```
rndc-confgen -a -r keyboard -b 256
chown named:named /etc/rndc.key
```

Now ISC-DHCP can be started:

```
systemctl enable dhcpd.service
systemctl start dhcpd.service
```

Check systemctl status dhcpd.service and /var/log/messages if you encounter issues.

Next named.conf needs to be configured. There is an example of this in example/fedora/named.conf. This is a modification of the default named.conf.

Additionally, you must configure the forwards and reverse zones to match for ISC-DHCP. There are example zones in example/fedora/named/. These should go into /var/named/dynamic/.

Please note, we have provided a zone for conntest.nintendowifi.net. This is also aided by a component in HTTPD (Documented later). This is to allow the Nintendo DS, Nintendo DSi and Nintendo Wii wireless connection test to complete, so that the Access point can be associated with. If this is not avaliable, Nintendo devices will be unable to join the wireless access point.

Now BIND9 is picky about permissions, but afterwards, can be started:

```
chown named:named /etc/named.conf
chown named:named /var/named/dynamic/*
systemctl enable named.service
systemctl start named.service
```

You can check that bind it working from the server, by running a query against localhost. In this case, we also try zone transfers (axfr):

```
dig @127.0.0.1 example.lan A
dig @127.0.0.1 example.lan axfr
dig @127.0.0.1 dhcp.example.lan axfr
dig @127.0.0.1 1.0.4.10.in-addr.arpa PTR
dig @127.0.0.1 0.4.10.in-addr.arpa axfr
dig @127.0.0.1 conntest.nintendowifi.net A
```

From a client connected to the LAN side, you should NOT be able to carry out a zone transfer, but you should see the A and PTR records returned:

```
dig @10.4.0.1 1.0.4.10.in-addr.arpa PTR
dig @10.4.0.1 tollgate.example.lan. A
dig @10.4.0.1 conntest.nintendowifi.net A
dig @10.4.0.1 example.lan axfr
```

When a client connects you should see messages in `/var/log/messages` like:

```
tollgate dhcpd: DHCPREQUEST for 10.4.0.10 from 00:00:00:00:00:00 (Franky) via p1p1
tollgate dhcpd: DHCPACK on 10.4.0.10 to 00:00:00:00:00:00 (Franky) via p1p1
tollgate dhcpd: Added new forward map from Franky.dhcp.example.lan. to 10.4.0.10
tollgate dhcpd: Added reverse map from 10.0.4.10.in-addr.arpa. to Franky.dhcp.example.lan.
```

If you see messages like:

```
tollgate dhcpd: Unable to add forward map from Franky.dhcp.example.lan. to 10.4.0.10: not found
```

Then you have made a mistake somewhere. Check that the rndc-key permissions are set to named:named, that dhcpd and named have been reloaded, that you have the correct control statements in named.conf and that in dhcpd.conf you have the primary option either as an ip or a resolvable hostname - We recommend this be the same as the IP in the named.conf control statement.

3.4 SQL

Django supports a number of SQL servers for it's operation. We have extensively tested MariaDB (Formerly MySQL) with Tollgate. However, PostgreSQL and SQLite are also valid options.

3.4.1 MySQL / MariaDB

We have extensively tested Tollgate with MySQL and MariaDB. Additionally, they support replication features which allows for retrospective conversion to a clustered setup.

First install the mysql packages.:

```
yum install MySQL-python mysql-server mysql
```

Now you need to setup the database. We advise you to remove the anonymous users and test tables, as well as setting a strong root password.:

```
systemctl start mysqld.service
mysql_secure_installation
```

Now we need to login to mysql, to create the database and tollgate user.:

```
mysql -u root -p
mysql> create database tollgate;
mysql> create user 'tollgate'@'localhost' identified by 'password';
mysql> grant all privileges on tollgate.* to 'tollgate'@'localhost';
mysql> flush privileges;
```

Keep these details for when you configure the settings.py - You will need to remember the `USER`, `NAME` and `PASSWORD`. The `HOST` setting will be `localhost`.

3.5 HTTPD

Apache HTTPD is what provides the majority of Tollgate functionality. We highly recommend that you install mod_ssl, mod_nss or mod_gnutls, since tollgate requires user authentication's to be sent via the HTTP channels. Our examples below will cover the usage of mod_ssl.

We create certificates for use with Tollgate.:

```
cd /etc/pki/tls/private/
openssl genrsa -out tollgate.key 2048
openssl req -new -key tollgate.key -out tollgate.csr
```

It is CRUCIAL at this step, that when asked, you put in your servers hostname in the Common Name field.:

```
Common Name (eg, your name or your server's hostname) []:tollgate.example.lan
```

Either you can send this CSR to be signed by another CA, or you can self sign. Either way, your resultant certificate should be tollgate.crt. Below is how you self sign your certificate:

```
openssl x509 -req -in tollgate.csr -days 365 -signkey tollgate.key -out tollgate.crt
```

Now you should reconfigure the ServerName and ServerAlias parameters in /etc/httpd/conf.d/tollgate.conf. Please note the VirtualHost for conntest.nintendo.net. Do not modify this VirtualHost.

Next you must edit /var/www/tollgate/tollgate_site/settings.py. Fill in the DATABASE section with your SQL server information. Finally, at the bottom of the settings.py fill in your LAN details as needed. Check to make sure all values seem sane for your environment.

Additionally, you should configure the SOURCE_URL parameter to ensure that you uphoad your AGPL obligations. If you DO NOT modify the tollgate source code (With the sole exception of the configuration files) this obligation can be met by sharing the source RPMs to the package. The source url parameter you can use is SOURCE_URL='https://tollgate.example.lan/source/' To retrieve these, run the following commands. The HTTPD configuration doesn't need alteration to support this configuration.:

```
yum install yum-utils
yumdownloader --source tollgate --destdir /var/www/tollgate/source
```

NOTE: If you are using mysql, you must add to your settings.py USE_TZ = False

Finally, we need to sync the database, and collect the static components ready for deployment.:

```
cd /var/www/tollgate/tollgate_site
python manage.py syncdb --noinput
python manage.py migrate --noinput
python manage.py collectstatic --noinput
python manage.py createsuperuser
```

Now you should start httpd.:

```
systemctl enable httpd.service
systemctl start httpd.service
```

3.6 Tollgate backends

You should configure /etc/tollgate/backend.ini with your site details. Additionally, you should configure /etc/sysconfig/tollgate with the correct DNS name of your tollgate.

You can now start the tollgate backends.:

```
systemctl enable tollgate-backend.service
systemctl enable tollgate-captivity.service
systemctl start tollgate-backend.service
systemctl start tollgate-captivity.service
```

CHAPTER

FOUR

HACKING ON TOLLGATE

Tasks that I've got in mind at the moment:

4.1 IPv6 support

I've started implementing IPv6 support in the backend of tollgate in a seperate branch. At the moment it's got some basic backend support but I haven't implemented the frontend code just yet (or the scanning module for ipv6). There's some stuff to consider though:

- **Done, implemented in IPv4** `-t nat -j REDIRECT` doesn't work in ip6tables at all. They have `-t mangle -j TPROXY` instead, which requires setting some "interesting" socket options (ie: I don't think Apache will work any more for the first stage of the captivity landing).
- IPv6 privacy extensions will mean you have a lot of constantly changing IPs. We can handle this pretty easily though (at least for the outgoing connections) by having a browser window open that updates the IP address periodically (treating it similar to an IPv4 change).
- Incoming connections I want to have a flag on to say whether a host may be accessible from the outside world, defaulting to no. As part of this getting the "permanent" IPv6 RA address may be important. This can be done by a multicasted ping for most operating systems, or sniffing router advertisements.

4.2 Improve the documentation

This is a fairly general task, anyone can pick this up. Particularly a step-by-step installation guide would be helpful, or some sort of bootstrap script so you can put that on a system, it pulls deps and then the git repo.

This has been now started in the form of the `tollgatebuilder` project: https://github.com/micolous/tollgatebuilder

4.3 Testing framework

There's no test cases, we probably should have some.

4.4 Internationalisation / localisation

Started an Esperanto translation of the project a while ago, haven't really touched it since. Many strings aren't in the translation files, and the setup of where the strings actually are could be improved a lot, because it's a mess.

4.5 Porting to non-Linux systems

I'd like to see this software ported to non-Linux systems. I've done some preliminary research that's come up empty as yet as to operating systems with firewalls that meet all of the following requirements for tollgate's backend to work properly.

The majority of the work you'd need to do in porting the software is in the backend. There's a little bit of Linux-specific frontend code to print out the contents of the ARP table. All the backend is setup in a way that it calls from the frontend are abstracted away from iptables.

Before you start work porting it to a new operating system, please consider the following list of requirements. If you can't get it to do **everything** in this list, then tollgate won't work.

- Ability to filter traffic by IP and MAC address.
- Port redirection, so captivity can work. (ie: When you have no quota it redirects HTTP requests to a web server, resets all connections)
- Ability to filter based on quota remaining.
- Ability to do both positive and negative accounting. Positive accounting is where you count upwards continuously the amount you've used, and negative accounting is where you decrement the amount you've used. Many firewalls I've looked at only do positive accounting.
- Ability to have shared/named accounting labels. So I can say "decrement counter X", and multiple rules can share that counter.
- Expose the counters via some `procfs` or `sysfs`-like interface. For example, `xt_quota2` (the module we use in *iptables* for quota) exposes it's counters in `/proc/net/xt_quota/LABEL_NAME`. With that you can read and write counters like files.
- Ability to access the ARP table.
- Doing all of the above stuff in kernel space. Running an extra daemon isn't really an acceptable solution to me.

4.6 Flow diagram (Linux)

TODO: Finish writing this.

This is how a packet is handled inside tollgate when running on Linux.

1. If it's a new connection, it hits the NAT table rules.

4.7 Known Issues

- **`xt_quota2` doesn't always show current quota data in iptables command. It should not be relied on for accurate display**
  - Why the in-kernel quota system doesn't work for us: http://bugzilla.netfilter.org/show_bug.cgi?id=541
- Port forwarding doesn't work correctly when the internal and external ports are different.
- There's no way to deregister a console from an account that hasn't signed in to the current event. (ie: Previous event the console is marked as being owned by user X, next event user Y can't sign it in because user X hasn't attended)

CHAPTER

FIVE

CREDITS / THANKS

Michael Farrell (2008-) Website (for email contact): http://micolous.id.au/

Started the project, wrote almost everything.

Ben Christian and Darren Mullighan (2009-) Provided valuable feedback and user experience testing.

David B (2010-) Reports security bugs and fixes bugs.

Rebecca Irving (2009) Contributed portal2 logo.

Thomas Woolford (2012-) Does UI prettyness.

William Brown (2009-) Testing and feedback. Deployment and packaging improvements.

CHAPTER

SIX

CHANGELOG

6.1 3.x series

All releases in the 3.x series are named after types of toothpaste.

6.1.1 3.0.0 "Cavity Protection" (May 2012)

This represents the first public stable release of tollgate (formerly portal2). Changes from 2.8.3 (September 2010):

- Added basic IPv4 port/protocol forwarding ability.
- Application migrated to `setuptools`-based deployment, and can be hosted inside of another Django project in typical deployment. (Issue #7)
- Implemented proper rollover handling, so when there is no current event or the event changes, access is revoked appropriately. (Issue #13)
- Improved documentation.
- License changed to Affero GPL v3.
- Major repository shuffle and cleanups.
- `api`: NetworkHost objects now report a bit more information about the vendor (not just the type of console), match many non-console items. Consoles are now identified by a new `is_console` field.
- `backend`: Tollgate *backend* daemonised, renamed files. Created init scripts.
- `backend`: Configuration file absence now handled better. (Issue #10)
- `backend`: Default configuration location is now `/etc/tollgate/backend.ini`.
- `backend`: New `TPROXY`-based captivity handler backported from experimental IPv6 branch.
- `builder`: Added new `tollgatebuilder` script for experimental deployment documentation. ;)
- `frontend`: Absence of `python-dbus` is handled more gracefully, allowing testing and development of the frontend on non-DBUS systems (Windows).
- `frontend`: Added system for automatically downloading and parsing MAC OUIs for system identification.
- `frontend`: All StreetGeek and SAGA-specific authentication code has been removed, as well as all external authentication code.
- `frontend`: All sign-ins and events are now handled locally, and locally administerable with sign-in wizard.
- `frontend`: Platform-specific code has been abstracted out, and moved into seperate modules, with a dummy fallback module for non-supported platforms (non-Linux).

- `frontend`: Remove several redundant (non-minimised) and unused Javascripts.
- `frontend`: Use django.contrib.staticfiles. (Issue #8)
- `frontend`: When `iplib` is not available, also attempt to use `IPy`. (David B)
- `frontend`: **Security**: Fixed issue where arbitrary protocols would be included on the captive landing page, leading to XSS issue. (Reported by David B)

6.2 2.x Series (portal2)

These are changes which happened before the public source release of tollgate, when the project was still named "portal2".

6.2.1 2.8.3 (September 2010)

- Updated internal documentation.
- Removed documentation that isn't used anymore in favour of the wiki.
- Server-side graph generation replaced with client-side (javascript) one for peformance reasons.
- "My devices and quota" now only updates the information from the kernel space if it hasn't happened in the last 2 minutes.
- API allows cookie-based authentication for faster authentication and better browser integration.
- Clustering support.
- UserProfile objects returned by the API now return the user's Forum UID, so username changes can be handled by API callers, and better integrate with the website.
- API now implements new `python` output method, which is the output from the repr function.

6.2.2 2.8.2 (July 2010)

- A user's first and last name is no longer returned by any API call except for `whoami()`. Other methods which request a *UserProfile* object will have empty strings instead of the user's name.

6.2.3 2.8.1 (May 2010)

- Clarified the "no api" login error message reasons, because there are more reasons why it can occur than were listed.
- Added a test version of the 'modern' and 'platinum' themes. These are incomplete.
- Out-of-subnet error page was not added to version control, now it is.
- `libiptc-python` removed, as it is no longer required.
- Fixed captivity bug where a user with unlimited quota would be forever stuck captive.

6.2.4 2.8.0 (January 2010)

- Fixed an exploit that would allow an attacked user to gain unlimited internet quota through an issue with external authentication.
- Fixed an issue where calls to the Django-side API would not convert the user_id to a string. This is now done in the API, so these calls will now (implicitly) succeed. This fixes an issue where quota wasn't automatically being recorded as part of the crontab job.
- Quota data is now automatically recorded every 10 minutes with history.
- There is a bandwidth graph showing a 10-minute average of metered internet usage over time.
- 'cake' and 'terminal' themes now have text boxes fully enclosed, rather than just an underline.
- Removed some duplicate code relating to quota reporting to backend.
- Reworked backend to use `xt_quota2` instead of the normal iptables quota module.
- tollgate is now finally captive! YAY!
- Fixed an error in the "internet login success" page where it would either not display at all or still show the survey banner on some browsers.
- Fixed an issue where external IP addresses could be logged into tollgate.
- Fixed an issue where IP changes might not be taken into account because expired entries in the ARP cache were not ignored.
- Admin: Internet usage report now defaults to being sorted by username alphabetically instead of by user ID.
- Admin: Internet usage report includes current speed of user's traffic.
- i18n: Started adding internationalisation hooks.
- API: Added HTTP GET API with json, pickle and csv output modes.
- Removed support for `libiptc-python` in backend.

6.2.5 2.6.6 (November 2009)

- LANdit backend also grabs whether a user has ordered unlimited coffee.
- `coffee_ip` API call added.
- Added option to manually change whether a user is allowed to use the coffee notification system, and extra ACL added to determine whether an administrator is allowed to change that value.
- Internet connectivity is no longer switched on on login **if** you have previously disabled internet connectivity and haven't selected to sign the current computer on in your name.
- Backend not running will no longer cause EventAttendance migration failure on login.
- Clarified the meaning of "structure" in the API help to mean a dict(ionary).
- `*_mac` versions of the API calls were removed.

6.2.6 2.6.5 (October 2009)

- ACL fixes.
- New version of the reset lecture.
- Warning added that the "logout" button logs you out of the web interface, not internet access.

- You can now "disown" a host.
- Host scanning changed from `nbtscan` to `nmap`.
- Hosts names are now grabbed from DNS rather than NetBIOS.

6.2.7 2.6.4 (September 2009)

- You can now only reset your quota once you have used 70% of it.
- Reset lecture added.
- Reset logging implemented.
- Network host changes now logged.
- You can now choose different themes, including using the old (green) 'terminal' theme. The default theme is the same as from 2.6.2, the 'cake' theme.
- The 'cake' theme now has underlines on submit buttons.
- `libiptc-python` created (a libiptc module for python)
- Backend ported to allow the use of libiptc-python. Currently disabled due to bugs.
- The automated host scan now also synchronises kernel-level counters with the database at that time.

6.2.8 2.6.3 (July 2009)

- Internal organisational changes to program structure.
- Backend API framework changed from XMLRPC to DBUS.

6.2.9 2.6.2 (June 2009)

- New backend authentication API for LANbru.
- Improved administration interface.
- New theme.
- Better error handling system.

6.2.10 2.6.1 (May 2009)

- Fixed whoami() API call so that it works.
- Added usage() API call.
- Fixed an issue where ownership would not be reassigned locally where it should have been allowed to be.

6.2.11 2.6.0 (April 2009)

- Resynced the two versions of v2.5 of the code in use.
- When there is an external authentication failure (such as attendance not registered, or forum password change) on an already-migrated account, you are no longer kept logged in.
- Offline hosts are now marked as being offline properly.

- Added API for interacting with tollgate.
- Version numbering changed

6.2.12 2.5 (March 2009)

- Fixed an issue where an automated task to find active hosts was failing and not marking offline ones as offline.

6.2.13 2.4 (February 2009)

- Added additional administrative controls.
- Added standalone portal mode.
- Menu links are now much clearer.
- Security: Improved handling of offline hosts that could allow a user to gain additional quota.

6.2.14 2.3 (January 2009)

- Lots more error handling code

6.3 Ancient Changes

First versions 2.0 - 2.2 were from October - December 2008. These were often pulled shortly after the start of the LAN due to bugs. It was later found that many of these problems were related to faulty networking equipment. The equipment has since been replaced.

The system was implemented due to issues with the previous WiFiDog-based setup (GLaDOS).

- Quota limits are now done kernel level so it is much more accurate and cut-offs are instant (previously a 10 minute window).
- Can now log in to more than two consoles at once.
- Logout timeouts removed.

CHAPTER

SEVEN

LICENSING

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

A full copy of the license is available in the file `/LICENSE`.

Please be specifically aware of Section 13 "Remote Network Interaction; Use with the GNU General Public License." This requires you to make modifications to this software available to network users. The easiest way you can do this is by making a publicly available Git repository for your users. (GitHub has a nice 'fork' option, which makes it easy for me to track this stuff...)

A notice will appear on all pages unless you set the SOURCE_URL setting to a location where the source code is stored. This is enforced as some members of the LAN community in my experience are "lax" about following licensing obligations. Removing the code that enforces this *does not* exempt you from the agreement - it is only there as a reminder and to assist you in license compliance.

Pushing your changes upstream is a great thing to do – it not only assists other users of the software, but also assists you as internally-developed features don't have to be patched in every time there is a new release.

7.1 flot

This software uses flot, a jQuery library for generating charts, copyright 2007-2009 IOLA and Ole Laursen. It's terms are as follows:

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION

OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

flot itself also includes:

- excanvas: Copyright 2006 Google Inc., licensed under the terms of the Apache License v2.0.
- jQuery: Copyright 2009 John Resig and The Dojo Foundation, dual-licensed under the MIT and GPL.

CHAPTER

EIGHT

INDICES AND TABLES

- *genindex*
- *search*