**Version 1.0**

**Alan Dekok (aland@networkradius.com)**

*Abstract: A new DHCP server implementation is presented. The software is based on the well known "FreeRADIUS" core. The added DHCP functionality leverages that base to create a GPL'd DHCP server that can store leases in an SQL database. This document describes the motivation for this approach, presents an overview of the implementation, and finally discusses the benefits of this approach. It assumes that the reader is familiar with DHCP, and with administering the ISC DHCP implementation.*

## Motivation

Networks are becoming more complicated. Every network has more devices than a few years ago, and the devices are getting more complex. The number of device vendors and implementations is also increasing. At the same time, however, security and auditing requirements for networks are becoming more stringent.

Something has to give. We cannot maintain complex networks and restrictive security policies using administrative methods from a decade ago. A new solution is needed. This solution must integrate information from a wide disparity of networking protocols into one view of the network. It must provide a global view of the network, rather than a "piecemeal" view of each networking protocol. It must allow for dynamic management and control. On top of those requirements, it must be fast, scalable, feature-rich, and easy to use.

This paper describes the foundations of such a system. We show how replacing existing DHCP systems can provide the benefits outlined above.

The rest of this paper begins by describing common DHCP software capabilities and limitations. It follows with a description of the FreeRADIUS DHCP implementation that uses an SQL database. It then describes the new solution. It outlines a number of additional benefits that can be gained by using the new solution. Finally, it includes a call for participants to help extend the solution to its full potential.

## ISC DHCP

The most commonly used Open Source DHCP server is the ISC DHCP implementation [ISC]. It has been in wide-spread use for well over a decade, and is the default server of choice for many sites. However, there are some well-known issues with the implementation:

- Although it is distributed under a “BSD-like” license, development is closed. The revision control tree is not publicly visible.
- It uses flat-text files to store bulk lease data.
- It uses flat-text files to store all configuration information
- It implements a pseudo-database API to update, delete, insert, etc. information related to leases.
- Any permanent change to leases, ranges, options, etc. requires a restart of the server.
- Implementation issues result in behavior that is quadratic in the number of leases.
- The protocol used to provide primary/secondary server failover has undergone great flux during its development.

This is not to say that the implementation is terrible. On the contrary, it has proven its utility in tens of thousands of sites over the past decade. It is stable, feature-rich, well understood, works on a wide range of platforms, and inter-operates with all known DHCP clients.

## Graphical Examples

The following diagrams graphically show some of the issues presented above. The diagrams are obviously artificial, in that few sites would store willingly store ten million leases in flat-text files. However, they highlight inefficiencies that can affect real-world deployments.

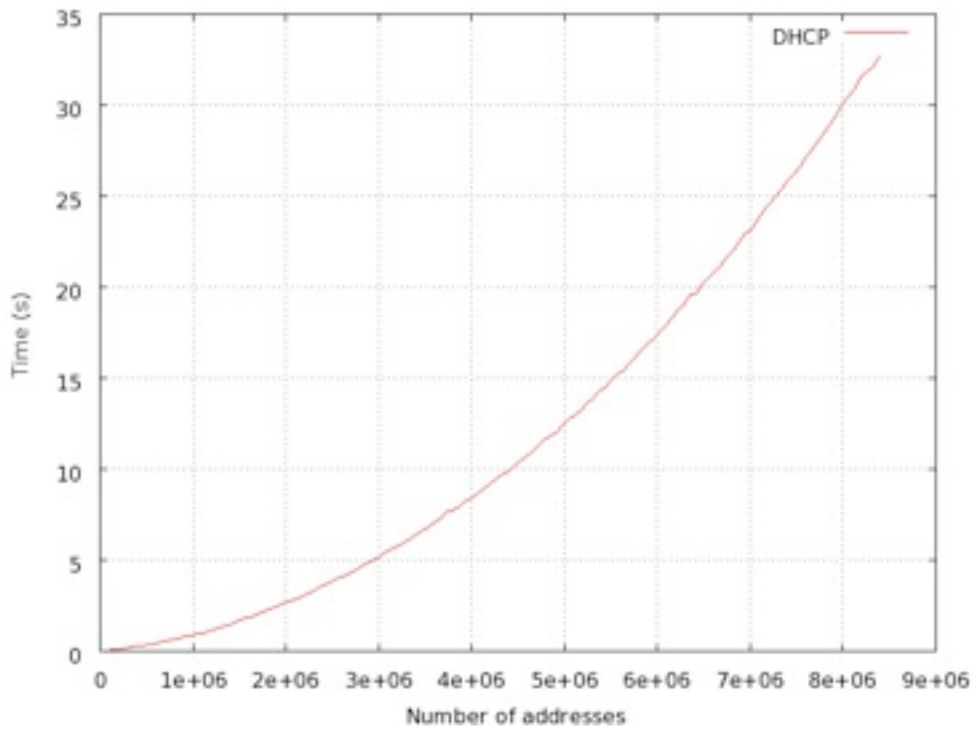

*Figure 1: Start time of dhcpd versus number of leases in a pool.*

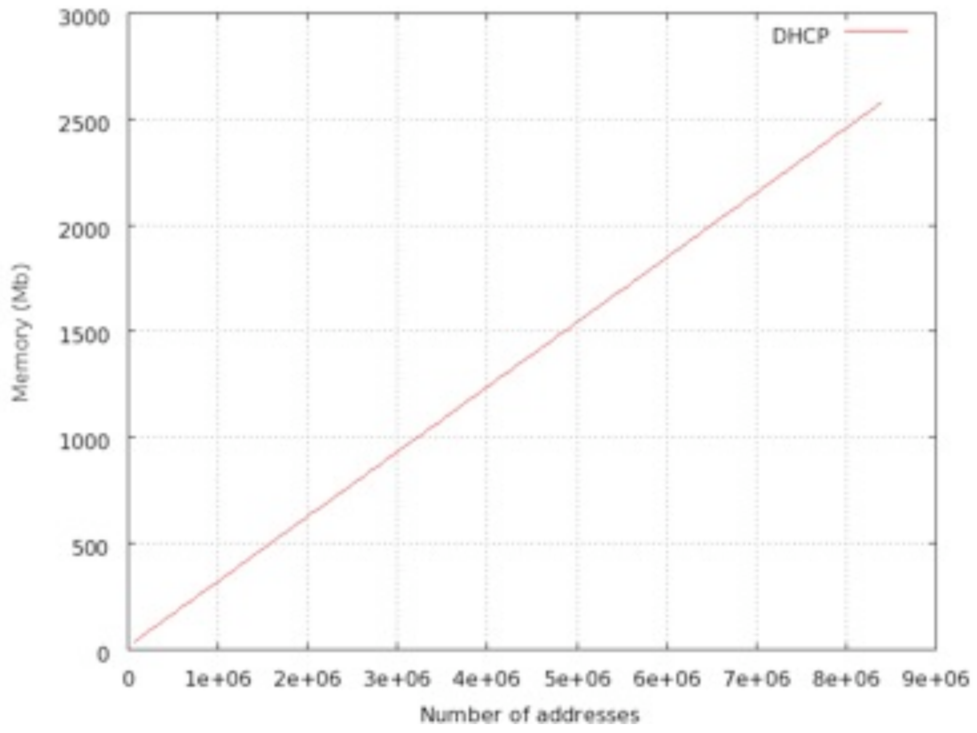

*Figure 2: Memory consumption versus number of leases in a pool.*

The first graph shows a start time that is nearly quadratic in the number of leases.  This behavior is due to the fixed-size hash table implementation in the software.  As number of hash buckets is fixed at about 10,000, the hash table quickly devolves to a linked list when the number of leases grows significantly past 10,000 addresses.

The second graph shows memory consumption that is linear in the number of leases.  This behavior is to be expected, as all of the leases are cached in memory for speed of operation.

If the software did not implement its own database, the following benefits would be seen:

- Start time would be nearly instantaneous, as the database would take care of accessing the leases when required

- Depending on the implementation of the database, lease lookups would be O(lg(n)) or O(n), instead of O(n^2).

- Administrators could use their own preferred database, rather than relying on the protocol-specific database in the DHCP server.

This last point is important.  Giving administrators choice over their preferred data store means that the software fits more easily into any wider network management system.  This capability goes a long way to addressing the goals outlined at the start of the paper.

## Existing Work-Arounds

A number of projects try to work around these issues.  They usually involve scripts that move data back and forth between an SQL database and flat-text lease files.  These solutions have been proven to be useful, but they do not address the issues demonstrated by the graphs above.  They also do not solve the problem that many configuration changes require a restart of the server.

Gluff [GLUFF] patches the ISC code to log leases to an SQLite database.  A separate program periodically reads these leases, and writes them to a MySQL database.  This process places the leases into SQL, but provides no way to manage leases in SQL, and write that information back to the leases file.

Sauron [SAURON] is a web-based management system for DNS and DHCP. It can generate configuration files for the ISC server, but it has no way to read leases assigned by the server back into SQL.

There exist few other projects that are active in this space. For example, a search of Freshmeat and Sourceforge showed a number of related projects, most of which had not been updated in two to three years. Off-line discussions with system administrators indicates that many sites “roll their own” solution to this problem. This duplication of work is inefficient, and leads to our proposed solution.

## FreeRADIUS as a DHCP Server

FreeRADIUS is the well-known GPLv2 implementation of the RADIUS authentication protocol. While it started off as a RADIUS server, it has been extended since then to other protocols. The implementation of the server is based around a core processing engine, which is protocol agnostic. The core is sends and receives packets through protocol-specific packet handlers. The core then consults its configuration files to determine how to process the packet, and then calls a sequence of one or more modules to perform the actual processing.

Each module is also protocol agnostic, and implements one logical piece of functionality. The core engine calls the modules in a configurable order, with fail-over, load-balancing, etc. This design is critical to the servers ability to support new protocols.

This architecture is outlined in the diagram below:

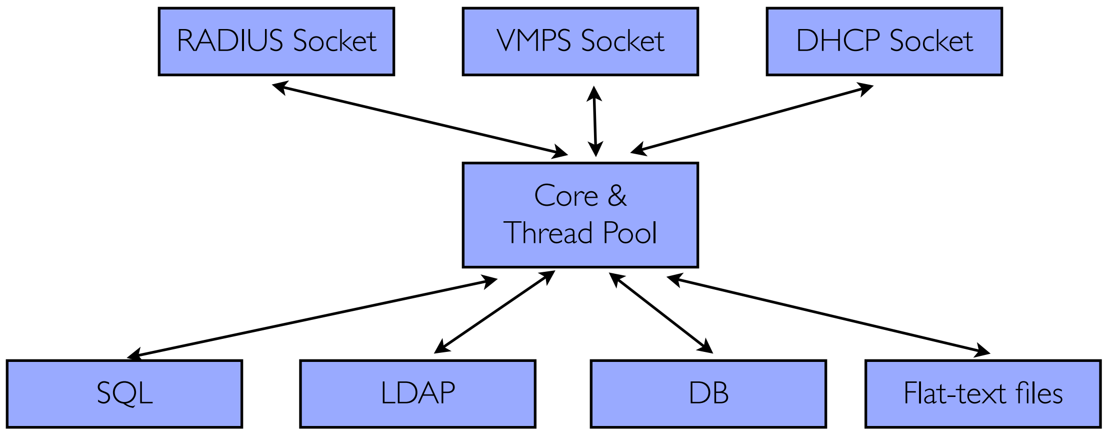

*Figure 3: Outline of the FreeRADIUS architecture*

This design enabled the easy addition of the VLAN Management Policy Server (VMPS) protocol in 2007, followed by DHCP in 2008.

## Configuration

The Git tree for the project is available on GitHub:

http://github.com/alandekok/freedhcp-server/tree/freedhcp/

The standard process is sufficient to install it:

```
$ git clone git://github.com/alandekok/freedhcp-server.git
$ cd freedhcp-server
$ ./configure --with-udpfromto
$ make
$ make install
```

Then run it in debugging mode via:

```
$ freedhcpd -X
```

It will print out a large amount of test, followed by the ports where it accepts packets. As we do not want to interfere with an existing DHCP server on the network, the default DHCP port is 6700.

At this point, the server is ready to accept packets. When it receives packets, it will process them through the virtual server policy contained in the file `raddb/sites-enabled/dhcp`. The default policy there is to do nothing, so it will need to be customized.

This paper is too short to go into the full details of configuring the server. Please see the source code, web site (http://freedhcp.net), and mailing list (http://lists.freedhcp.net) for more assistance.

## Benefits

This solution not only puts leases into an SQL database, but also pools, options, networks, and related configuration. The benefits of this approach are that few, if any, configuration changes require a restart of the server. The diagram below shows the suggested schema.

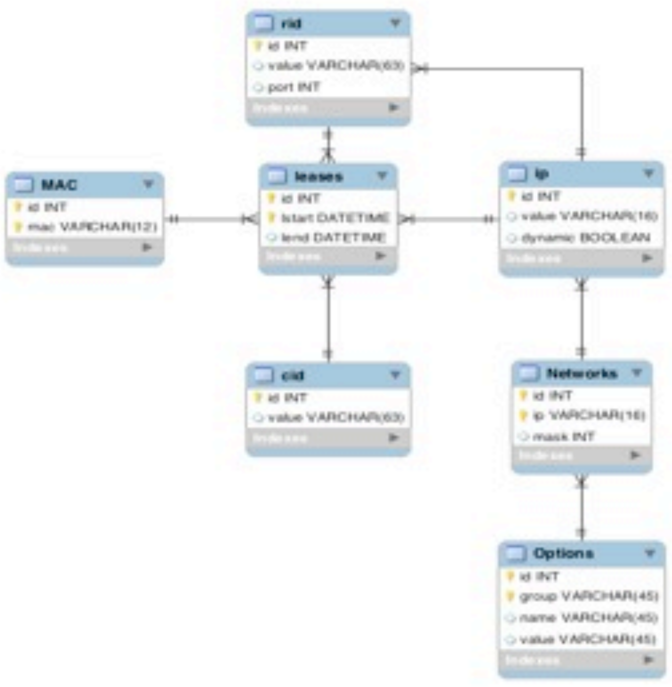

*Figure 4: Suggested SQL schema*

The operation of the server is simple. Nearly all logic that was part of the DHCP lease allocation strategy has been moved to SQL statements. These statements are configurable, and can be adapted to many different schemas.

The schema above has a number of tables. The MAC and IP address table are separate, and are referenced from the leases table. The leases table also references the client identifier (cid) and remote identifier (rid) tables.

## Limitations

All software has its limitations, and this project is no different. The main limitations currently are:

- Functionality is still “beta”, and needs more testing
- Validation against more clients is needed
- The failover protocol is not supported
- migration from the ISC configuration format and leases file is still a manual process
- More example configurations for different SQL databases are needed.

## Future Work

The work outlined here is only the beginning of the capabilities of this project. At present, there is no common SQL schema that is shared by DHCP servers, DNS servers, and RADIUS servers. We believe that schema has significant value for a network administrator. A unified schema enables the following kinds of functionality:

- RADIUS will tie user name to MAC address (and VLAN)
- DHCP will tie MAC address to IP address
- DNS will tie IP address to DNS address

These connections enable an administrator to perform simple queries that are difficult or impossible to do today. For example:

- Which person is using IP address 192.168.10.24?
- Which IP address is assigned to user “bob”?
- Which switch & port is user “bob” on?
- Which switch & port is IP address 192.168.10.25 on?

In addition, as RADIUS is the “authorization” protocol, recent extensions to RADIUS enable dynamic “Change of Authorization”, or “Disconnect”, by leveraging the security of RADIUS. These changes can be done without resorting to potentially insecure SNMP. For example, when the above information is stored in a queryable database, RADIUS allows the following control statements to be made about the network:

- Disconnect user “bob”
- Place user “bob” into the quarantine VLAN
- Place all of the Windows machines into the remediation VLAN

We believe that these benefits are large enough to warrant developing a new DHCP implementation. This implementation should galvanize administrators into believing that there is something more than the “status quo”. The capability to easily control the network, and also to have up to date visibility on network activity is now available.

## Call For Participants

As with any Open Source project, the speed of development is based on the number of active participants.  This project has proven that a GPL'd DHCP server with leases in SQL is possible.  Further extension of the project beyond that basic functionality is ongoing.

Many of the benefits presented above are in progress, but not yet finished.  Finishing them requires additional effort to finalize the schemas, examples, integrations with other protocols, web UI, etc.

We believe that the flexibility of the basic server makes these additions straightforward.  The modular design of the server means that the system is not tied to any one SQL database, or even any one particular schema.  It should also be possible to quickly adapt this solution to custom schemas.

The solution we have presented here relies heavily on the belief that administrators not only can, but should know what is happening in their network.  It relies on the belief that administrators should be able to easily configure their network, without editing a number of unrelated flat-text files.

We believe that this project has tremendous potential.  But that potential can only be realized by working in conjunction with other projects, and with the contributions and feedback from the people who need it the most.

## References

[FREEHCP] “The FreeDHCP Project”, http://freedhcp.net

[FREERADIUS] “The FreeRADIUS Project”, http://freeradius.org

[GLUFF] “A DHCP Lease Logger for ISC DHCPd”, http://hans.liss.pp.se/software/gluff

[ISC] Internet Systems Consortium, *“ISC DHCP”*, https://www.isc.org/software/dhcp

[SAURON] “A free DNS & DHCP Management System”, http://sauron.jyu.fi/