little information, we either retrieve a log or request it from the user who owns node. The log typically provides some useful information. Additionally, as with all multithreaded applications, deadlocks happen, we found it useful to add liveness states to threads to assist in finding deadlocks.

Other information we monitor includes peer count, memory, and CPU usage. Node count can be quite difficult to keep track of in Planet-Lab as machines at a rate of 5 to 20 per day are restarted and our software is not automatically restarted on these machines, thus the case to watch for is non-linear loss of nodes. Planet-Lab also places challenges on memory, as the systems can often be I/O starved causing what appears to be memory leaks as Brunet's internal queue can grow without bound. In these cases, we have the node disconnect from the overlay and sleep before returning. The advantage of Planet-Lab as a test ground is that it presents so many unique situations that can be very difficult to reproduce in a lab controlled test system. It is our belief that any system that uses large scale Planet-Lab deployments as a testing ground will be quite reliable.

We are still actively seeking better ways to check and verify the state of our system. For example, the cost of doing a crawl can take $O(N \log(N))$ time, since we have to communicate with every single node with an average routing time of $O(\log(N))$. Future work in the arena is focused on Brunet's MapReduce [9] framework, which can be used to provide system wide searches and status checks in $O(\log(N))$ time.

7 Related Works

Our work is not the first to propose using a group like mechanism for regulating members in a VPN. Hamachi presently offers the ability to create and join a network from either a VPN client or through their Web site. To create a group, a user can either form a private invite only VPN or a password protected public VPN. Users must exchange out of band both the password and the VPNs name and both registered and unregistered users (guest) can join Hamachi VPNs. Also, Hamachi uses a model similar to a key distribution center (KDC) as opposed to a PKI, thus it is quite easy for Hamachi to do man-in-the-middle attacks. Hamachi's server is not redistributable and all users must use LogMeIn's (Hamachi's owner) KDC and relays. Our model ensures that each user is traceable and has to be authenticated by the group administrator. The groups is further secured by giving each user a unique key to retrieve signed certificates. Most importantly our PKI is open and redistributable, so users can self-host the group VPN web server, and our relays are built into the VPN and require no management.

Our group system is not the first to provide an automatic PKI. Previous work in this field includes RobotCA [1]. A RobotCAs receives request via e-mail, verifies that the sender's e-mail address and embedded PGP key match, signs the request, and mails it back to the sender. RobotCAs are only as secure as the underlying e-mail infrastructure and provide no guarantees about the person beyond their ownership of an e-mail address. In certain cases, this model could be used in our use cases, such as a SMB or for universities if it enforces that all users use university e-mail addresses, then the RobotCA would only sign e-mails if they come from a specific domain. Our experience suggests that is not rare for an academic to use a non-university e-mail for university purposes. Another concern is that the RobotCA would require management to limit allowed users to members of a class or an organization whose e-mail addresses does not contain domain names.

VINI [3], a network infrastructure for evaluating new protocols and services, uses OpenVPN along with Click [23] to provide access from a VINI instance to outside hosts, as an ingress mechanism. OpenVPN only supports a single server and gateway per a client and does support distributed load balancing. VINI may benefit from using a VPN that uses a full tunnel model similar to ours, as it lends itself readily to interesting load balancing schemes.

Our work is not the first to suggest using a P2P infrastructure to enable the discovery of physically close TURN-like relays. Skype [21] queries super nodes in an attempt to find physically close relays. The primary difference in our work is that our model could easily be configured to let users create and select their own relays.

7.1 P2P VPN in Other Structured Overlays

The purpose of this work is to develop a P2P VPN model that can easily be applied to other structured P2P systems. In this section, we focus on the portability of our platform to other structured P2P systems, namely Pastry and Chord by analyzing FreePastry and NChord respectively. FreePastry can easily reuse our C# implemented library through the use of IKVM.NET, which allows the porting of Java code into the CLR. NChord is a Chord implementation written in C#. In Table 5, we compare the features of the structured P2P systems as they apply to the use as a VPN. The specific focus of this section is to understand how discovery and VPN connections would work. Our discovery model for mapping node ID to IP works through the use of a DHT. During the IP address allocation, the VPN client will place in the DHT a key, value pair mapping a virtual IP to a node ID, as described in [41].

The bootstrapping of a connection in NChord begins by finding the owner of the DHT key containing the mapping of virtual IP address to node ID through