

A Virtual Connectivity Layer for Grids

Jefferson Tan, David Abramson, Colin Enticott

Caulfield School of I.T.

Monash University

Melbourne, Australia

Jefferson.Tan@infotech.monash.edu.au

David.Abramson@infotech.monash.edu.au

Colin.Enticott@infotech.monash.edu.au

Abstract— Computational grids are now mainstream facilities for e-research worldwide. While enterprise grids exist within organizations, national grids have become common, usually consisting of government as well as academic facilities. Such facilities are not uncommonly lenient with blanket policies to allow inbound and outbound grid traffic. This is far from ideal, from a security perspective, but given the dynamic nature of grid use, it is impractical to keep restrictive firewalls and manually keep up with on-demand firewall reconfiguration. Other solutions are necessary, where security is not sacrificed. Apart from first generation solutions that were mostly not sufficiently generic, standardization work is now ongoing, but exclusively aimed at firewall virtualization. We argue for an architectural solution that encompasses firewall virtualization as well as other methods that can be more appropriate in many environments. This paper describes our notion of the missing layer between grid and fabric, which we refer to as the virtual connectivity layer. We have developed two implementations within this layer and discuss how they fit into a complete and well-defined architectural solution.

Keywords—Grid, firewall virtualization, tunneling

I. INTRODUCTION

The Grid, “a distributed computing infrastructure for advanced science and engineering” [2] is a complex system that brings together scientists, geographically distributed scientific instruments, computers, mass storage devices, data collections and scientific applications for collaborative projects involving otherwise independent organizations. In order for Grid actors and machinery to interconnect across continents, the Grid relies heavily on network technology. This involves, in most cases, the Internet, in order to interconnect the local networks of independent organizations that collaborate on a Grid testbed. Due in part to advances in wide area networking, a Grid testbed can be made operational within a short period of time. Yet even with the sophisticated state of network technology in the 21st century, Grid connectivity is not simple, and conflicts within the infrastructure occur. In particular, conflicts that arise between components of the Grid network infrastructures are the subjects of this paper.

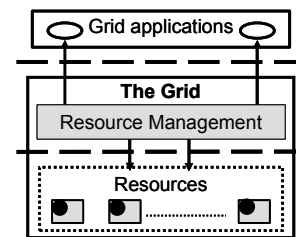


Figure 1. Dashed lines indicate divisions between components where firewalls can hinder necessary connections.

A. Grids over the Internet

Fig. 1 illustrates the typical connectivity between applications and the Grid, as well as between the Grid and the underlying resources. The arrows represent links at two separate levels: between Grid applications and the Grid, and internally, between the Grid resource management system and the resources underneath. The dashed horizontal lines are divisions between layers of software, and it is important to understand that connectivity between Grid applications and resource management systems is usually over the Internet. This is typical: rather than forge new network connections, each participant will make its resources accessible to one another over the Internet, using pre-existing network infrastructures. This is where conflicts occur: the security policies of the underlying network infrastructures address threats from the Internet, not the Grid. The focus is on supporting access to resources from inside the organization, while heavily restricting those coming from outside. On the other hand, the collaborative agenda of Grid participation requires that connections from outside organizations who are Grid collaborators must be widely permitted. Moreover, most commonly used Grid infrastructure, middleware and applications make wide assumptions about the underlying network infrastructure, expecting unrestricted connectivity. We thus have a predominance of e-Science applications that lack built-in mechanisms for network flexibility. Grid infrastructures makes similar assumptions, even those built with the most widely adopted Globus Toolkit. These assumptions cause problems when deploying them over the practical Grid with resources behind enterprise firewalls.

B. Firewalls on the Internet

Using the Internet is intuitive since it provides a massive but convenient and pre-existing worldwide infrastructure. However, Grids built over the Internet also inherit formidable headaches. From 21,756 compromised security incidents reported within the US in 2000, the numbers rise up to 137,529 in 2003 [3]. Some of these activities cost substantial amounts of money, as much as \$80M recorded for a single incident [4]. Even when the number of incidents has fallen slightly of late [5], including targeted attacks, security will continue to be a significant problem. It is thus reasonable for security managers to exercise extreme caution and resort to firewalls. A firewall is a hardware and software device that restricts network access to resources in accordance with the organization's security policies [6]. A firewall is also known as a packet filter as it stands between two zones of a network, examines packets of data as they pass, and allows or disallows their passage as appropriate. Firewalls are legitimate means to defend an organization's resources, but for likewise legitimate reasons, firewalls should not stand in the way of Grid collaborations.

The problems posed by firewalls and similar devices to Grid applications are reported in many studies [7-12]. Ideally, with Grid computing becoming part of an organization's activities, Grid connections should be allowed across firewalls. It is not so simple, however, since Grid computing distributes computations across a large number of computers running from several networks. This may therefore require excessively permissive network access, which will contravene existing security policies of an organization [13]. On the other hand, it is undesirable and expensive to rewrite or customize legacy applications and mature Grid toolkits in order to comply with security restrictions. Ideally, legacy applications should be reusable, and programmers must concentrate on their application rather than on programming for different network situations.

II. SURVEY OF FIREWALL TRAVERSAL FOR GRIDS

There are a number of ways to establish connections despite the restrictions of firewalls. It is possible to tunnel Grid connections through one or a small number of connections using network paths that are allowed by firewalls. However, this tunneling method causes slower data transfer rates, since several connections are crammed into a smaller number of connections. Furthermore, tunneling usually involves an intermediary software component known as a proxy [14, 15], or an application level gateway (ALG) [6], which runs on another machine on the network. Proxies and ALGs are often considered synonymous, but proxies are usually understood to require explicit configuration of the application, whereas ALGs typically do not. In any case, they would either extend the route of network traffic or require additional processing, or both, adding to delays. There are also security implications with tunnels if the transported connections within are encrypted, making it difficult to verify the legitimacy of tunneled data and transactions [6]. Alternatively, it is possible to temporarily open pinholes through the firewall [8, 10,

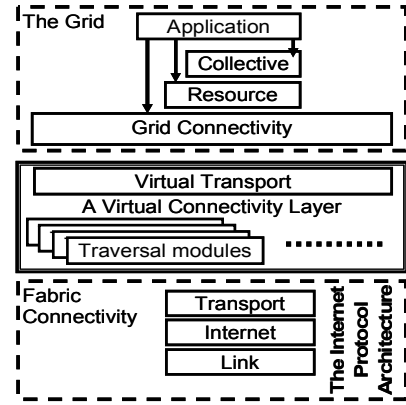


Figure 2. A new grid architecture with a virtual connectivity layer shielding the Grid from Fabric firewall issues.

12, 16, 17]; this is typically referred to as firewall virtualization in a Grid context. Pinholes can be open for the duration of authorized connections, which can then be made directly, and then disabled after some predetermined amount of time, e.g., when a user's certificate is revoked. However, it is not clear that current models of firewalls support the required functionality to receive and validate pinhole requests over the network as Grid transactions. Such features will require mechanisms to recognize and perform operations over Grid-specific security credentials that identify trusted users. This will thus require some heavy customization in order to bring together perhaps proprietary remote firewall management protocols and Grid-specific authentication and authorization mechanisms. We have not found anecdotal evidence from the literature that such a project will be trivial when dealing with commercial, enterprise-grade firewalls from leading vendors such as Cisco and Juniper. Many routers assume infrequent ACL manipulation, and only provide proprietary, non-standard remote management. Nor will such enterprise routers be easy to configure (if at all possible) so they may host custom code for Grid-specific firewall controls. Moreover, not all routers support SNMPv3, whereas older SNMP versions are less than ideal [18]. There are also risks in opening pinholes that, although temporary, must be addressed in order to ensure that security remains intact [8].

Such solutions as described above are suitable for different scenarios, whereas it seems reasonable to make all solutions accessible for selective application, as the situation requires. Even current standardization initiative is limited, focusing as it does on firewall virtualization, e.g., FiTP [19] and Fireblade [17].

III. A VIRTUAL CONNECTIVITY LAYER

We therefore advocate an explicit **virtual connectivity layer**, shown in Fig. 2, providing an abstraction between the Grid and the network infrastructure. It must allow for flexibility and interoperability with current and future Grid systems as well as network infrastructure standards. This makes the environment transparent so that Grid infrastructure, applications and firewalls need not be reprogrammed in order to work together. It also shields against

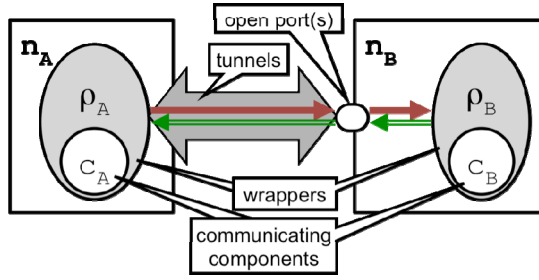


Figure 3. Remus uses wrappers for transparency and automation. Multiple tunnels can be used to mitigate the performance impact of tunnelling.

evolutionary changes to the fabric in terms of the network topology, security policy, and the rest of the network infrastructure. The implementation is a middleware that sits between Grid system middleware and the network beneath it. We have thus far developed two components of this middleware, one that implements tunneling while the other implements firewall virtualization. The glue that holds them together is the design of the virtual connectivity layer itself, a means to standardize connectivity, regardless of the mechanism that provides it. This approach is motivated by a number of principles:

- Participation is voluntary, and one cannot assume the imposition of a uniform firewall traversal strategy.
- The fabric is assumed to be heterogeneous even in the security policies in place.
- It is desirable to minimize the impact of joining a Grid to an organization's topology and security policies.
- It is likewise desirable to *avoid customization of Grid middleware and applications*.

We have thus far focused on firewall issues, whereas there are other problems such as NAT. But they should also be resolved in the same way, via mechanisms abstracted inside the layer.

IV. REMUS AND ROMULUS

Fig. 2 shows two sublayers in what we envision: a *virtual transport* sublayer, which provides standard interfaces for use by Grid middleware and applications, and a sublayer of several *traversal modules*, indicating a library of mechanisms that provide firewall traversal in different ways. It is ideal that the virtual transport sublayer is as unobtrusive as possible. We achieve this using wrappers that automatically handle the traversal, but only in one of our modules, Remus (described shortly). Concerning traversal modules, we have thus far implemented two: Remus and Romulus.

A. Remus

Remus (rerouting and multiplexing system) applies the use of tunnels and/or proxies [20]. It is actually appropriate to consider the Remus proxy as an application level gateway since it provides for transparency using *wrappers* that reroute TCP connections without customizing either the Grid middleware or application. Its features also include the following:

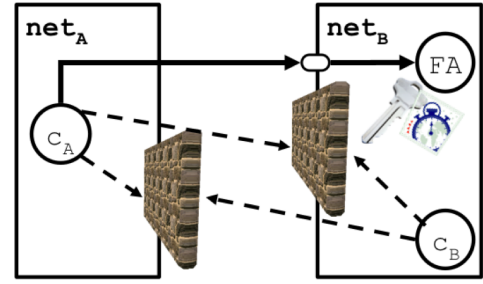


Figure 4. Romulus uses a simple control mechanism to open and shut temporary pinholes for authenticated requests.

- It uses SOCKS, either version 4 or 5, although more work is needed to extend it to use GSSAPI.
- It may use SSH tunnels in addition to SOCKS.
- It has been configured to use multiple tunnels in order to mitigate the encapsulation overhead of tunneling [1].

As Fig. 3 indicates, Remus assumes that one or more open ports exist, in order for a tunnel to become possible at all. SSH has been a favored protocol to provide us with tunneling, given that it includes the ability not only for SOCKSv4, but also encryption, compression, and multiple SSH sessions that can be configured to distribute the tunneling traffic [1]. An advantage of Remus is that it is unobtrusive, configuration-wise, since Grid middleware and applications are not reprogrammed. However, there is the inescapable impact of tunneling to throughput, due to the effects of extra encapsulation. However, it is most suitable when security policies do not allow for the relaxation of firewall restrictions. The only requirement is for some open port, and SSH is typically allowed to some degree. Even in the use of a DMZ staging node that alone allows for SSH access from the Internet, Remus is able to facilitate Grid connectivity.

B. Romulus

Romulus instead uses firewall virtualization using a *firewall agent* that controls the opening up and closing down of pinholes for authorized users. Fig. 4 shows how explicit pinhole requests are submitted by component (C_A) from network A to the firewall agent (FA) in network B, which must process and either approve or ignore the request. If approved, a pinhole is punched through the firewall that specifically allows the host of the requesting component to access some Grid service (C_B). Pinholes are timed, so that they can be subsequently revoked when authorization times out, e.g., when the user's short-term Grid proxy certificate expires, or if the user's Grid credentials are revoked. Requests are submitted via GridFTP [21] or using HTTPS [22]. Authentication is based on X.509 certificates, and two firewalls have been tested:

- A Linux router uses a kernel-level firewall, and pinholes are added or removed by a firewall agent hosted on the router itself [21, 22].
- A Cisco router uses access control lists that are added or removed using SNMPv2 [22].

An advantage of Romulus is that it is nearly transparent in terms of latency, since traffic is direct if

allowed through pinholes, once the latter have been added. Requests can be processed and decided upon almost instantaneously, although a second version of Romulus we've implemented uses fixed intervals to perform pinhole operations in batches. Another feature of Romulus is that the firewall agent may be local or remote to the actual firewall. It is more secure not to host the agent on the firewall host itself. In two different implementations, a local component on the firewall host will download pinhole operations from the firewall agent host, thus avoiding the need to bind to a socket on the firewall host, which would have become a target for exploits.

C. The Virtual Transport Sublayer

A key ingredient in layered architectures is that of an interface between layers. The virtual transport sublayer is intended to provide this, and is still under development. Unlike that of highly mature work in the OSI 7-Layer reference model for networks, we have not had to define a peculiar "service access point" for service requests from the upper layer, which is the Grid connectivity layer. At present, we implement two different interfaces for Remus and Romulus traversal, with the latter requiring further work.

Remus is the older of the two, and benefits from a completely transparent interface. Given a site or network from which Grid client operations will originate, traversal is facilitated through a *rerouting table*, as we call it. This is a map that must be preconfigured to direct traffic for initiated connections. It decides two things: (1) whether or not connections require a proxy, and (2) *which* proxy is to be used for a given connection. Mapping is defined based on the source IP addresses and ports, and/or the destination IP address and ports. Romulus is still in need of a similarly transparent interface for Grid applications above it. At present, an explicit pinhole request is still necessary, which is trivial to effect but best to hide in order to simplify things for Grid middleware and applications. At present, Globus GridFTP and HTTPS are the supported request mechanisms.

- When GridFTP requests are used, the user's Grid proxy certificate is submitted with the request. This is not the most ideal approach in terms of security, but provides one advantage: the firewall agent can make the pinhole temporary, using the proxy certificate's expiry.
- HTTPS pinhole requests are more convenient, allowing users to make requests via web browser, whereas the request itself is encrypted. However, unless the Grid proxy certificate itself is posted in the HTTPS pinhole request, the firewall agent has no basis for a pinhole expiry, apart from relying on revocation list updates that will indicate when user credentials are revoked or expired.

The "service access point" in the case of Romulus is thus more explicit, requiring as it does a pinhole request operation prior to using Romulus traversal.

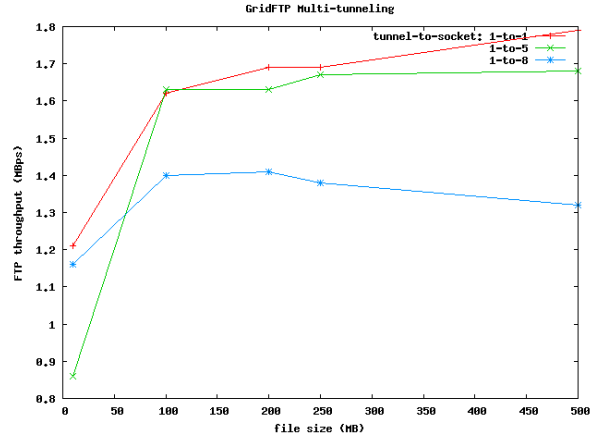


Figure 5. There is improvement when the ratio between tunnel and socket connections is generous. The top graph above uses a ratio of 1:1, while the worst graph at the bottom uses 1:8 [1].

V. TESTS, RESULTS AND OBSERVATIONS

Remus redirects Grid traffic automatically and transparently via rerouting tables preconfigured like maps. The current Remus implementation uses the **Dante** SOCKS package [23], which includes a simple utility to create a "socksified" wrapper environment in order to automatically reroute TCP traffic as configured to do so. Remus thus makes it unnecessary to define new interfaces for the wrapped process to use in order to traverse via SOCKS. A requirement of this interface is that Dante be installed on the same machine as the Grid client. Apart from that, most compiled binaries or interpreted scripts will be rerouted transparently without modification. Remus has been shown to work without modification for WS or pre-WS GRAM, GridFTP, as well as Nimrod/G [24], a parameter sweep meta-scheduler that sits on top of Globus as a Grid application. Remus will also work for general-purpose network utilities, e.g., *iperf*. However, some environments such as those of Java VMs will ignore the Remus rerouting.

Network throughput is impacted negatively due to the extra encapsulation involved, from 10% to 20% [1, 20] in throughput degradation when one tunnel is used. We improve upon this when using multiple tunnels, shown in Fig. 5, where we by which we can achieve higher throughput [1]. Other tests in controlled environments have shown results of up to just 6% degradation compared to direct connections for short, burst-type traffic [25]. Tests were conducted for large file transfers using GridFTP and for burst traffic using *iperf*. Degradation of up to 39% was observed in our tests, with a 5:1 socket-to-tunnel ratio for three tunnels, but may yet be reasonable, given that the traffic is encrypted and *does not require additional port openings on the firewall*. Parameter sweep applications, which our group is familiar with using Nimrod/G, do not exhibit constant streaming traffic, and will mostly suffer only at the beginning and the end of parameter sweeps, when possibly large input and output files are staged in or staged out. Otherwise, the burst-type traffic between Nimrod/G remote agent and local server components are typically only lightly impacted during the course of

sweeps. In terms of security, Remus requires a user account at the SSH tunnel “bridgehead” on the other side. This can be considered a security mechanism, but can be unwieldy if too many users require traversal via Remus.

Romulus uses firewall virtualization, thus allowing for direct connections when pinholes are in effect. We implemented two means of managing pinhole requests:

- The firewall agent immediately performs pinhole operations, either locally if it is deployed on the firewall machine, or remotely otherwise. In our tests, the turnaround time for adding a pinhole to the firewall can be well within one second when using a Linux router and a local firewall agent that uses **iptables** to add pinholes immediately after validating the request [21]. More delays can be expected in two fronts: (1) If the firewall agent is not on the same machine as the firewall, delays will include the remote invocation of pinhole operations; (2) More delays are expected during the authentication of requests, which involves X.509 certificate validation either by GridFTP or HTTPS.
- Another implementation uses **cron** to batch pinhole operations by intervals of one minute, which makes that also the worst-case turnaround time between reception of an authenticated request and the opening of the pinhole [22].

We implemented two different kinds of firewalls for Romulus. One is a Linux router using **iptables**-based packet filtering. The other is a Cisco 2612 router that uses access control lists (ACLs). The firewall agent could not be hosted on the router itself, so it was hosted on a separate Linux machine. Pinhole operations were conducted using SNMPv2 and TFTP, the first protocol being used to prompt a TFTP upload of configuration commands to add or remove ACLs. The SNMPv2 and TFTP components of this implementation are its least secure aspects. However, many router and firewall models do not support SNMPv3, and/or cannot accept ACL operations via SNMP.

Based on our work thus far, applications will not typically even see the virtual connectivity layer as proposed. This is particularly true for firewall virtualization solutions such as Romulus, where the firewall is simply opened for application socket connections. Even for Remus, however, this is largely true. In a GT4 implementation where the Globus services are rerouted effectively as we have done, job manager environments inherit the rerouting feature. We have thus far tested Linux compiled binaries that are quietly rerouted without issues. Java applications, however, resist the rerouting environment. The vast majority of e-science applications we have encountered do not use Java.

The question of whether to tunnel or to pinhole is relative. The Open Grid Forum favors the pinhole approach, despite fairly straightforward objections that security managers will likely make. But with middlebox initiatives in the IETF, and advocacy from leaders such as Cisco and IBM, it is likely to remain a valid option. On the other hand, others would prefer to avoid the

remote control capabilities required in the pinhole solution, despite such voices getting behind the firewall virtualization paradigm. In such cases, tunneling becomes the best option, where the firewall remains tight except for the tunneling port such as 22/tcp for SSH, or some other port if preferred, e.g., web services via 8443/tcp. This makes our proposed virtual connectivity layer appropriate, as it allows both approaches under the same architectural solution.

VI. CONCLUSIONS

In this paper, we presented our work on a virtual grid connectivity layer, which we propose as an architectural resolution to connectivity problems for Grids due to firewalls. We discussed the advantages of a single framework that uses multiple firewall traversal mechanisms. We also presented the benefits of having a two-sublayer approach with a virtual transport sublayer for interfacing with Grids above, and a sublayer of multiple traversal mechanisms for dealing with connectivity problems from the network infrastructure below. We illustrated the benefits of a transparent virtual transport interface through the Remus approach that unobtrusively reroutes via wrappers and SOCKS. In this regard, both Grid middleware (Globus) as well as a Grid application above it (Nimrod/G) are given a transparent virtual transport interface. We also presented experiences in terms of the impact on performance in using the tunneling/rerouting approach of Remus, but presented multi-tunneling as a means of mitigating the throughput degradation due to extra encapsulation.

Several follow-up projects are in order. GSSAPI for SOCKSv5 should be supported when using Remus, in order to provide authentication prior to traversal being granted. This will align Remus more closely with the security infrastructure of Globus, which includes GSSAPI support. This will also allow the use of a set of Remus proxies administered at the Grid services site, where security administrators can implement accounting without the necessity of individual SSH accounts for authorized users. For example, the tunneling may instead proceed via web services, although the performance impact is likely to be considerable. Either way, further study is required. Romulus also requires a major improvement in terms of virtual transport interface: the explicit pinhole requests should be replaced by better firewall agents that can make decisions in real time when normal Grid transactions start arriving. We are also in the process of engaging more recent enterprise-grade firewalls, e.g., Cisco's ASA firewall models, which support *security contexts* as well as SNMPv3. Such devices are expected to provide more firewall virtualization capabilities. Also, other connectivity issues can be dealt with in the proposed layer, such as the involvement of NAT-enabled networks on either the client or server side. Finally, many existing and past endeavors in firewall traversal can be supported in the proposed layer. Foremost among these are those with ongoing standardization work, e.g., OGF's FiTP [19] and Globus' Fireblades [17].

ACKNOWLEDGMENT

The thesis projects of Y. Chang, K. W. Kim and M. Albathan, and that of J. Alshammari, Q. Zhang and A. Gargote were instrumental in more recent developments in Remus and Romulus, respectively.

REFERENCES

- [1] D. A. J. Tan, C. Enticott, "Optimizing Tunneled Grid Connectivity across Firewalls," in *7th Australasian Symposium on Grid Computing and e-Research* Wellington, New Zealand: Australian Computer Society, 2009.
- [2] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the Grid: enabling scalable virtual organizations," *International Journal of Supercomputer Applications*, vol. 15, 2001.
- [3] CERT, "Computer Emergency Response Team coordination centre (CERT/CC) Statistics 1988—2006," 2007.
- [4] U. D. o. Justice, "Computer Crime Cases, Computer Crime & Intellectual Property Section," United States Department of Justice 2008.
- [5] R. Richardson, "2008 CSI Computer Crime and Security Survey," Computer Security Institute 2008.
- [6] R. Niederberger, W. Allcock, L. Gommans, E. Grünter, T. Metsch, I. Monga, G. L. Volpato, and C. Grimm, "Firewall Issues Overview," Open Grid Forum Aug 16, 2006 2006.
- [7] D. Abramson, R. Sasic, J. Giddy, and B. Hall, "Nimrod: A tool for performing parametrised simulations using distributed workstations," in *4th IEEE Symposium on High Performance Distributed Computing* Virginia, 1995.
- [8] D.-G. I. (DGI), "Dynamic Firewalls Configuration, Ver. 1.0," D-Grid Integrationsprojekt (DGI) Feb 2006 2006.
- [9] S. Graupner and C. Reimann, "Globus grid and firewalls: issues and solutions in a utility data center environment," Hewlett-Packard Company Oct 2, 2002 2002.
- [10] M. L. Green, S. M. Gallo, and R. Miller, "Grid-Enabled Virtual Organization Based Dynamic Firewall," in *5th IEEE/ACM International Workshop on Grid Computing (GRID'04)*, 2004, pp. 208-216.
- [11] J. Quittek, M. Stiernerling, and P. Srisuresh, "Definitions of Managed Objects for Middlebox Communication," IETF 2006.
- [12] S. Son, B. Allcock, and M. Livny, "CODO: Firewall Traversal by Cooperative On-Demand Opening," in *14th IEEE Symposium on High Performance Distributed Computing*, 2005.
- [13] A. Denis, O. Aumage, R. Hofman, K. Verstoep, T. Kielmann, and H. E. Bal, "Wide-Area Communication for Grids: An Integrated Solution to Connectivity, Performance and Security Problems," in *13th IEEE International Symposium on High Performance Distributed Computing (HPDC-13 '04)*, 2004, pp. 97-106.
- [14] Y.-D. Lee, "SOCKS: A protocol for TCP proxy across firewalls," NEC 2005.
- [15] J. Tan, D. Abramson, and C. Enticott, "Bridging organizational network boundaries on the Grid," in *6th IEEE/ACM International Workshop on Grid Computing (Grid 2005)* Seattle, Washington, 2005.
- [16] M. Stiernerling, J. Quittek, and T. Taylor, "Middlebox Communications (MIDCOM) Protocol Semantics," IETF 2005.
- [17] "Incubator/Fireblade (Wiki page)," Globus Alliance, 2009.
- [18] M. Barnes, "Middlebox Communications (MIDCOM) Protocol Evaluation," June 2005 2005.
- [19] R. Niederberger, "FiTP - A protocol draft for dynamic opening of Firewalls," in *25th Open Grid Forum (OGF25)* Catania, Italy: Open Grid Forum, 2009.
- [20] D. A. J. Tan, C. Enticott, "REMUS: A Rerouting and Multiplexing System for Grid Connectivity Across Firewalls," *Journal of Grid Computing*, vol. 7, pp. 25-50, March 2009 2009.
- [21] J. Alshammari, "Prototyping firewall virtualization for grids," Thesis, Master of Network Computing: Monash University, 2009.
- [22] Q. Zhang, "Firewall virtualization over grids," Thesis, Master of Network Computing: Monash University, 2009.
- [23] "Dante- A free SOCKS implementation," Inferno Netzwerk A/S, 2005.
- [24] R. Buyya, D. Abramson, and J. Giddy, "Nimrod/G: an architecture of a resource management and scheduling system in a global computational Grid," in *HPC Asia 2000* Beijing, China, 2000.
- [25] M. Albathan, "Performance study of multi-tunneling grid transactions," Thesis, Master of Network Computing, submitted August 2009: Monash University, 2009.