CamGrid: Experiences in constructing a university-wide, Condor-based grid at the University of Cambridge

M Calleja[1], **B Beckles**[4], M Keegan[3], M A Hayes[2], A Parker[2] and M T Dove[1, 3]

1. Department of Earth Sciences, University of Cambridge, Downing Street, Cambridge CB2 3EQ
2. Cambridge eScience Centre, Centre for Mathematical Sciences, University of Cambridge, Wilberforce Road, Cambridge CB3 0EW
3. National Institute for Environmental eScience, Centre for Mathematical Sciences, University of Cambridge, Wilberforce Road, Cambridge CB3 0EW
4. University of Cambridge Computing Service, New Museums Site, Pembroke Street, Cambridge CB2 3QH

Abstract

In this article we describe recent work done in building a university-wide grid at the University of Cambridge based on the Condor middleware [1]. Once the issues of stakeholder concerns (e.g. security policies) and technical problems (e.g. firewalls and private IP addresses) have been taken into account, a solution based on two separate Condor environments was decided on. The first of these is a single large pool administered centrally by the University Computing Service (UCS) and the second a federated service of flocked Condor pools belonging to various departments and run over a Virtual Private Network (VPN). We report on the current status of this ongoing work.

1. Introduction

The concept of building a university-wide grid whose purpose is to harness as many idle computational resources as possible is not new. Indeed, this has already been achieved using the Condor middleware [1] at University College London (UCL) as part of the eMinerals project [2]. Condor is a High Throughput Computing (HTC) resource and job management software designed to harness idle CPU cycles from a heterogeneous pool of computers. The approach and solution employed in the current instance are, however, markedly different from that used at UCL. This is mainly due to more complex stakeholder issues as well as to greater technological obstacles, which has led us to implement two distinct, non-interacting environments. We use the term environment to mean a number of flocked Condor pools (or a single Condor pool) that do not interact with any other Condor pools in the University. The aim is to integrate these environments into a single resource when the middleware reaches a sufficient level of functionality and maturity such that it is able to address all stakeholders' requirements

The available resources consist of a wide range of platforms, running different operating systems, and owned by a multitude of different bodies. These bodies consist of individual research groups, entire Departments or Colleges, and the central University Computing Service (UCS). To complicate matters, some (but not all) of these resources are behind firewalls. Whereas this obstacle may be circumvented through the addition of appropriate exceptions in a firewall ruleset (and indeed, Condor has improved in the way it allows the requisite port ranges to be defined), this approach is unattractive from an administrative point of view since all firewall rules would have to be updated every time a machine joins or leaves the environment. This problem may be mitigated to some extent by only allowing jobs to be submitted from a small number of hosts, though this restricts current usage policy in some departments that allow all machines participating in a Condor pool to submit to that pool. As an additional complication, many of these machines have private IP addresses rendering difficult their participation in a Condor environment spanning all of the above resources.

Apart from the technological problems mentioned above, we also had to contend with the security requirements of the different stakeholders, as not all stakeholders have the same requirements. For example, the UCS wished to employ some form of "strong" host-based authentication, e.g. using X.509 certificates or Kerberos authentication, for all participating machines, which was not necessarily compatible with the security model of other interested parties. Also since the UCS machines under discussion would be running Linux when participating in a Condor environment, such a requirement also automatically precludes any machines running

the Windows OS from being used since Condor does not currently offer any "strong" authentication method across all the platforms it supports (and in particular, across the Windows and Linux platforms). Consideration of points such as these led us to a solution that consists of two separate, non-interacting, environments.

Despite this division of the available resources into two non-interacting environments, a key design goal for CamGrid is that this division should be invisible to end-users, i.e. that the end-user should not have to make a conscious decision about which environment their job should run in (unless they wish to be exposed to this level of choice or their requirements can only be satisfied by machines in one particular environment). Provisional plans for achieving this goal are discussed in Section 3.1.

It must be borne in mind that the university-wide grid is still being constructed and so many of the details given here may change in due course.

2. The Two Environments

2.1 Environment 1

The first of these environments comprises a single pool whose execute nodes are drawn from machines owned and managed by the UCS, and which belong to the Personal Workstation Facility (PWF). The PWF is a collection of PCs and Apple Macintosh computers that are centrally managed and have been set up to present a consistent environment to ordinary users. A series of images – snapshots of the operating system (OS) made at a particular moment in time – are installed on these machines periodically throughout the year, and mechanisms are in place to update the machines with patches, updates, virus definitions for their anti-virus software, etc. as necessary. These machines are not operating behind a firewall, and all have globally visible IP addresses.

The PCs that comprise the PWF are either dual boot Windows/Linux machines or else only run the Windows OS. At the time of writing the image about to be deployed on the PWF PCs will upgrade them to run Windows XP and SuSE Linux 9.0, or Windows XP only, as appropriate. The Linux installation on PWF PCs is known as PWF Linux and the Windows installation on PWF PCs is known as PWF Windows. The Apple Macintosh computers that are part of the PWF are known as PWF Macs and currently run Mac OS X v10.2 – this is about to be upgraded to Mac OS X v10.3.

Eventually it is planned that all machines that are part of the central UCS-owned PWF will be Condor execute nodes in this environment (Environment 1). However, a phased implementation has been adopted which has begun with PWF Linux, and will probably then incorporate the PWF Macs and finally PWF Windows.

The standard setup for a PC which has both PWF Windows and PWF Linux installed is for the PC to boot into Windows by default, and to reboot to Windows if no user is logged in and the machine is left idle in Linux. It is proposed that this behaviour will be changed so that when the PWF machines in a particular area are not in use (e.g. overnight) they will (re)boot into Linux and remain there, unless a user wishes to use them under Windows, until those machines are likely to be again in demand (e.g. the next morning). The details of this change of policy are still being negotiated, but it seems likely that this will leave most of those PCs which support PWF Linux booted into Linux for at least 8 hours or so a day.

So Environment 1 will initially consist of about 400 PWF Linux machines (in this initial phase, when machines are not running PWF Linux they will not run any of the Condor daemons and so will not take part in Environment 1). This means that most of the machines in this Environment will only be available for Condor jobs during those periods when the PWF receives very little use – overnight during the University term, probably for longer (perhaps even all the time for some collections of PWF machines) outside of term. Users who wish to submit Condor jobs to this Environment will need to take this into account when deciding whether it is worth running their jobs in this Environment. This Environment will clearly favour jobs with shorter execution times and jobs that can checkpoint.

There will be a single dedicated central manager, which may also act as Kerberos domain controller for the machines in this Condor pool, and a small number of submission nodes (initially one), with 1TB of attached short-term disk storage. Access to the submission nodes is via the SSH protocol. No checkpoint server will be provided. (This model is similar to the UCL model described in [2], though in that model the central manager is also the submission node, and the execute nodes run almost exclusively under Windows.)

All daemon-daemon communication in this environment is authenticated via Kerberos, but

Condor does not perform any user authentication. Condor's network transmissions will not be encrypted. Access to this pool is administered by the UCS, but it is currently envisaged that any user entitled to use the PWF will be entitled to run jobs in this Condor pool – indeed, the user authentication performed by the SSH server on the submission node(s) may authenticate via the same service which authenticates normal interactive logins to the PWF.

Thus the user authentication is performed at the 'point of entry' to the pool (i.e. the submission node), and not thereafter. All communication between the Condor daemons, i.e. communication using Condor's protocols to any machine in the Condor pool, is authenticated using a "strong" authentication method, so that it should not be possible for either an unauthenticated user to submit a job, or for a job to be submitted from a machine that has not been officially designated as a submission node.

By providing a large amount (1 TB in total) of short-term file storage, most users' jobs will not need to access any external file storage as there should be sufficient short-term storage attached to the submission node for any files required by the job. Condor can then transfer these to the execute node if necessary. Similarly, there should be sufficient space for the output, including any checkpoint files, of the most users' jobs. A policy regarding the length of time that users' files can be stored on this short-term file storage is still to be decided.

Initially the job and user priorities will be left at the Condor supplied default values or very slightly adjusted. The START expression for the execute nodes will almost certainly be adjusted so that jobs start immediately (although this may vary depending on the activity of interactive users in that area of the PWF). This situation will be kept under review and may well change depending on the usage pattern of this Condor pool and of interactive usage of the machines by users.

As well adding the different OS platforms on the execute nodes to the pool in a phased manner, 'official' support for Condor's different universes will also be added in a phased manner. Note that, with the exception of the PVM universe (where it is envisaged that the relevant module will not be installed), little or no attempt will be made to prevent users running jobs in a particular universe – 'official' support for a universe merely indicates that users can expect support for running jobs under that universe from the relevant UCS members of staff.

Initially the vanilla universe, and probably the Java universe will be 'officially' supported. 'Official' support will then be added for the standard universe. It is very unlikely that there will ever be 'official' support for the Globus or PVM universes, or for user jobs which attempt to use the scheduler universe directly, although the use of DAGMan will be eventually 'officially' supported. It is also unlikely that there will be 'official' support for the MPI universe – such support would require a Condor-supported implementation of MPI to be installed on the execute nodes, and this would be a non-trivial process for PWF Linux.

Note that these decisions, as indeed are most of the decisions concerning this environment, are dictated by a combination of end-user and stakeholder requirements, resource constraints and best practice. As these change, so too will the relevant policies and implementation details of this environment – in particular, the UCS seeks to provide a service that is responsive to the reasonable requirements of its users and potential users, and as new or changed end-user requirements are made known to the UCS the service provided will adapt accordingly.

2.2 Environment 2

The second environment comprises all other machines belonging to participating Departments and Colleges. These may or may not be operating behind firewalls, and may or may not be using private IP addresses. Potentially, machines in any participating institution may need to communicate with machines in any other. This "many-to-many" communication model potentially conflicts with efforts to establish network choke-points in the form of firewalls, whose design tends to be aimed at an asymmetric "one-to-many; many-to-one" model of interaction between network hosts. Unless the benefits of firewalls are to be sacrificed or the cost of maintaining firewall rulesets is to rise with the number of machines protected, the second environment must channel inter-machine communications in such a way as most closely to resemble the model encouraged by the firewalls involved. Communications may be switched at the application, connection or network level.

The Condor project is currently developing extensions to the middleware [3] in order to facilitate the inclusion of such resources – Dynamic Port Forwarding (DPF) and Generic Connection Brokering (GCB) – but these are not yet ready to be deployed in a stable release.

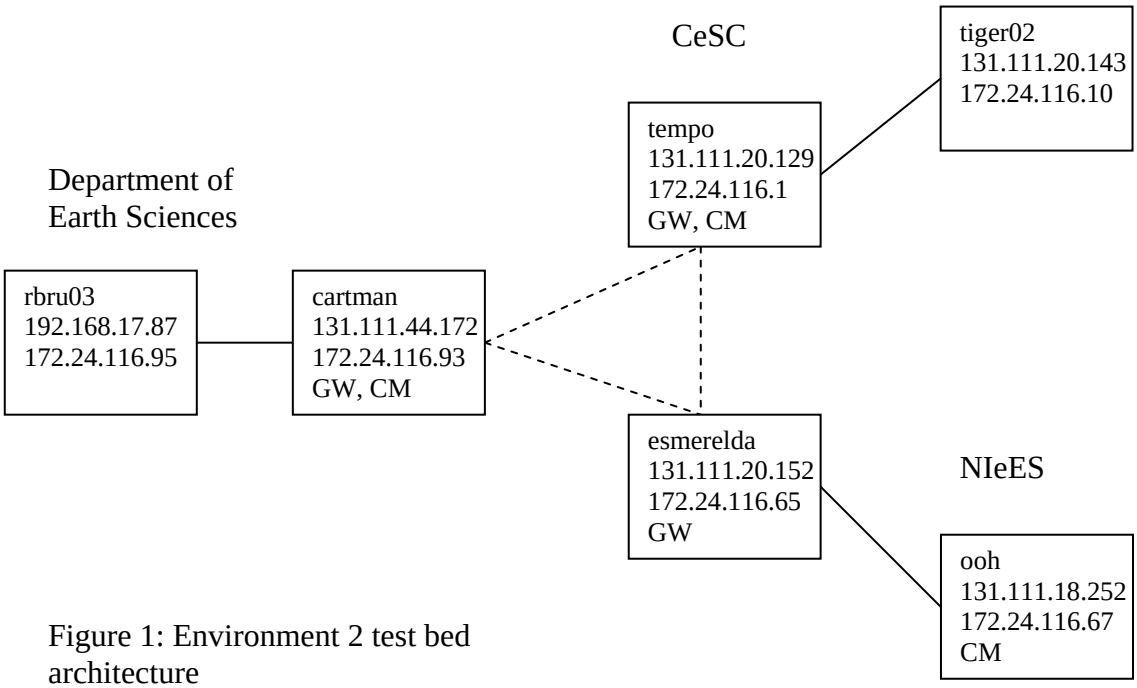

Figure 1: Environment 2 test bed architecture

Hence, we have implemented our own experimental solution external to Condor – specifically, we have constructed a dedicated Virtual Private Network (VPN) based on secnet [4], which is freely downloadable and developed within the Department of Computer Science at the University of Cambridge. All participating machines are given an (additional) IP address in this VPN. The aim was to deduce how effective such a solution would be, and whether it was a feasible production environment once security considerations were taken into account.

This approach greatly simplifies negotiating firewalls, since now only a single machine (the VPN gateway) from a firewalled institution needs to have external access, and then only via a single UDP port for a relatively small number of machines (the other VPN gateways). All other machines belonging to that institution now tunnel their 'Condor traffic' through that gateway, regardless of whether they have private IP addresses. An added bonus is that traffic between different gateways is automatically encrypted, adding a layer of security to the model. However, running such a VPN raises its own security issues, since institutional firewalls are effectively bypassed by this mechanism, so extreme care needs to be taken both in administering the gateway and in formulating an appropriate security model for this environment.

In Environment 2, each institution runs its own Condor pool, which is then flocked with any number of the other pools in the environment. The VPN gateway can be the central manager for that pool, but this is not a requirement. Policy as to how to administer and trouble-shoot this cross-institutional resource is non-trivial, and is generally developed in regular stakeholder meetings. Our intention is that once DPF becomes standard within Condor it will replace the VPN solution, and will make further integration with the UCS environment more likely.

Figure 1 shows the architecture of our Environment 2 test bed. There are currently three participating departments, each of which contributes a small Condor pool (Condor versions 6.6.3 and 6.6.4) consisting of a Central Manager and another node capable of both submitting and servicing jobs. The departments are the Department of Earth Sciences (ES), the National Institute for Environmental eScience (NIEeS) and the Cambridge eScience Centre (CeSC): note that ES lies behind a departmental firewall. The boxes in the Figure contain (in descending order) the host name, the host's real IP address, its VPN address and an indication of whether it's a Central Manager (CM) or VPN gateway (GW). Each gateway is configured as a secnet gateway [4], which creates a second virtual interface for that machine within the private IP range dedicated for this purpose:

172.24.116.0/24. The gateway also needs to support IP forwarding, e.g. on a Linux machine one needs to set the contents of /proc/sys/net/ip4/ip_forward to 1 and load the TUN/TAP module, which is usually already resident on most Linux distributions as tun.o. For RedHat Linux this is usually located in /lib/modules/<kernel #>/kernel/drivers/net.

Leaf nodes on the VPN do not run any special software, but simply have an additional (virtual) interface added (e.g. using ifconfig under Linux) and a route added to their routing table to nominate the relevant gateway for all VPN traffic. Hence note that in the figure, solid lines indicate intradepartmental traffic between leaf/leaf and leaf/gateway nodes, while the dashed lines indicate interdepartmental traffic between gateways. The latter is encrypted, which is a feature of secnet.

Each host on the VPN also has an additional identity within a new domain, grid.private.cam.ac.uk, which is recognised by the DNS servers across the university. Hence each CM allows flocking from other machines in this domain by setting FLOCK_FROM = grid.private.cam.ac.uk in the condor_config file. A CM nominates which other pools it wants to flock to by listing the other CMs in FLOCK_TO in the order it wants to try flocking in. For Condor to use the VPN one must also set the value of NETWORK_INTERFACE in the condor_config file to point at the correct virtual interface on that machine. Hence, for cartman this would be 172.24.116.193 (see Figure 1). Obviously all other references to host and domain names within the Condor configuration files must refer to entries in the grid.private.cam.ac.uk domain.

3. Conclusions and future work

3.1 Environment 1

Environment 1 is still in the design and testing phase, and additional user requirements continues to be gathered and stakeholder and user feedback continues to be sought for this Environment. There are a number of technical issues that still need to be addressed in the development of Environment 2. In addition the deployment team for Environment 1 still have a number of unresolved security concerns with the Condor middleware which they are currently actively investigating.

One of the most significant technical issues is the requirement for a “strong” authentication method for Condor that is supported under Linux, Windows and Mac OS X. It is thought that this may be solved by the addition of Kerberos support to Condor under Windows and Mac OS X, which members of the Condor Team currently suggest may be available in Condor 6.7.2. Were this to come about, a decision would then have to be made whether to wait for the next stable release of Condor (provisionally due around May 2005) or to deploy an experimental release of Condor in a production environment.

As mentioned is Section 1, one of the key design goals of CamGrid is that the separation into two separate environments should not adversely affect the end-user – in particular, the grid should be presented to the end-user as a single resource with which they then interact. Since the authentication requirements for Environment 1 are greater than those for Environment 2, it is probably most sensible to require that the user satisfy those requirements and then provide a transparent mechanism for the user to submit their jobs to the most appropriate environment.

The infrastructure which will enable this will be developed along the following lines. A job submission tool, camgrid_submit, will be developed (based on the condor_submit command). camgrid_submit will run on a submission node for Environment 1 and will work out which is the best environment for the user’s job based on the job’s requirements and the currently available resources in the two environments. As it will run on a submission node for Environment 1 it will be easily able to submit a job to the Environment 1 when necessary.

If the job submission nodes for Environment 1 are also submission nodes for the flock that constitutes Environment 2, then camgrid_submit will also be able to submit a job to Environment 2 when necessary. Job results will be returned to the submission node for Environment 1 on which camgrid_submit ran, and the user will be able to collect their job’s output as normal.

A web portal using HTTPS as its transport mechanism will also be developed which will allow users to easily transfer files to the submission nodes for Environment 1 and then launch camgrid_submit remotely via the portal, which will submit the job to the appropriate environment. Upon completion of the job the user will be able to retrieve their results via the web portal.

There is also another possible benefit to this development of a centralised submission point for CamGrid, which may resolve some of the problems faced by Environment 2 (as described

in Section 3.2). By making use of a single job submission node, or an extremely small number of job submission nodes, the requirement for "many-to-many" communication described in Section 2.2 is reduced to a "one-to-many" or "few-to-many" requirement. Such a reduced requirement will place comparatively little administrative burden on firewall administrators and so may well be acceptable.

In addition a solution to the problem of private IP addresses may also be possible. All private IP addresses in the University must be behind firewalls, except for those that have been designated "CUDN-wide" – the CUDN is the Cambridge University Data Network, i.e. the network which connects the separate College and Department networks in the University.

Therefore, one solution to the problem of IP addresses would be to give resources with private IP addresses so called CUDN-wide IP addresses, either as additional IP addresses as described in Section 2.2 or as replacements for their existing IP addresses. This would allow the submission nodes of Environment 1 to communicate with such machines, and those machines would still not be accessible from outside the University (by virtue of having CUDN-wide private addresses) and access from within elsewhere within the University could be controlled by the firewall behind which the machine must lie.

However there may be resource owners or firewall administrators who are unhappy with this solution, and so it is still under investigation.

3.2 Environment 2

We have successfully flocked three small Condor pools across a Virtual Private Network. The VPN enables us to tunnel through departmental firewalls and encrypt traffic across inter-departmental links, allowing nodes with private IP addresses are able to join a Grid that crosses institutional boundaries. Jobs migrate seamlessly across the flocked Condor pools and there is no noticeable degradation in performance due to the overhead of running across the VPN. We clearly need to stress-test our work in at least three directions: enlarge the size of each pool, increase the number of pools and the number of submitted jobs.

Use of a VPN raises its own security concerns. As mentioned previously, the secnet gateways effectively bypass any firewall present, and we are currently investigating methods for mitigating this potential security risk, e.g. by filtering packets on the gateways and by limiting the number of submit nodes within each department. These issues have not been fully resolved at the time of writing and further deployment of Environment 2 will depend on whether suitable solutions can be formulated to meet these concerns.

We have experimented with the use of digital user certificates for Environment 2. (Condor permits certificates to be used for authentication in a number of different contexts.) Setup proved non-trivial, but appeared to work adequately in operation. Certificates can also be employed internally in the communication between the Condor daemons. These uses of digital certificates may mitigate the potential problems of untrusted users gaining unauthorised access to the pool via flocking, but at an increased administrative cost in obtaining and distributing certificates, and may well also decrease the usability of the grid for end-users. Also, this certificate mechanism is not supported by Condor under Windows, and hence cannot be seen as a complete solution to these security concerns.

Some of the resources within CamGrid may be parts of other grids, provided the owners of these resources consent to jobs originating from grids external to the university, e.g. LHC Computing Grid (LCG) and the UK e-Science Level 2 Grid, running on their resources.

4. Acknowledgements

MC, MK and MTD thank NERC for financial support.

5. References

[1] J. Basney, M. Livny, and T. Tannenbaum, "High Throughput Computing with Condor", *HPCU news, Volume 1(2)*, (1997)

[2] P. Wilson, M. Calleja, J. Brodholt, M. T. Dove, M Alfreddson, Z Du, N. H. de Leeuw, A. Marmier and R. Tyer, "A Grid approach to Environmental Molecular Simulations: Deployment and Use of Condor pools within the eMinerals Mini Grid", *Proceedings of GGF 10* (2004)

[3] S. Son and M. Livny: "Recovering Internet Symmetry in Distributed Computing." *Proceedings of the 3rd International Symposium on Cluster Computing and the Grid* (2003)

[4] http://www.chiark.greenend.org.uk/~secnet/release/current/