



bright cluster



*Performance Simplified™*

## Bright Cluster™ Pulse™ 1.0 White Paper



## INTRODUCTION

Bright Cluster™ Pulse™ technology raises the bar for cluster scalability, manageability, and performance. Pulse™ intelligently provisions clusters and grids, streamlines operator workflow, reduces the burden on and costs of storage, and increases throughput.

Bright Cluster™ Pulse™ is a high throughput data transfer and cluster management technology that provides rapid, easy, and reliable data replication across a cluster.

Rapid data replication is one of the most critical components of data-intensive distributed computing. The same piece of data, whether an operating system, application, or data set, needs to be installed on each cluster node before a job can be started.

Traditionally each node in a cluster makes an individual request or connection to the file server for this identical piece of data. The result is a tremendous amount of file server thrashing as the server tries to comply with all these identical requests, very low network performance as identical bits of data are transmitted separately to each node, and ultimately, cluster nodes that are starved for data and sitting idle. This carries with it a tremendous cost to the scalability and the performance of a cluster.

As the number of systems within a cluster increases, the accompanying network load creates a problem of diminishing returns that can completely negate the additional usefulness of more nodes. Commonly these file server and network bottlenecks are addressed by purchasing very expensive, high bandwidth file servers and networks. These solutions push out the reasonable size of a cluster but their costs rapidly curb the price performance potential of clustering.

Bright Cluster™ Pulse™ solves these problems by replacing individual file transfers with a fast, efficient and reliable multicast based communication and transfer protocol. Very simply, Pulse delivers data just in time for local application access. The local applications then access the data directly from their local hard drive instead of from a network file system.

## RETURN ON INVESTMENT

Pulse delivers a straight forward increase in computational ROI by eliminating the data throughput bottleneck that causes decreasing marginal return of investing in additional compute nodes in a distributed computing environment.

In a standard computational cluster implementation the desired outcome is to achieve as many operational computing cycles as possible for a given budget. Ideally this would mean that the entire budget is spent on cluster nodes and computational software to provide the most raw power possible for the most sophisticated software possible. In reality though for every dollar spent on purchasing more nodes there is corresponding decrease in operational data throughput. This means that the bigger the cluster the more time it spends sitting idle waiting for data. To address this blockage it is common to then spend a very large amount of the cluster budget not on additional nodes but on very expensive networks and high bandwidth file servers. In conjunction,



tedious and error prone data staging processes are implemented to further attempt to allay the throughput constraint on production. In the end only a fraction of cluster investment is actually applied to direct computational resources.

Pulse delivers almost linear throughput scalability and automated workflow that eliminate the need for spending budget on expensive file servers and networks or suffering the inefficiencies of data staging. The result is higher utilization of larger clusters with nearly constant marginal costs. Put simply, Pulse allows the budget to be spent directly on more computational power that in turn can be operated at higher utilization rates. This means that for every dollar spent, Pulse provides more raw computational power to be kept at higher rates of production than other solutions on the market.

## **DATA MINING**

Pulse is ideally suited for distributed (embarrassingly parallel) data mining applications including life science (e.g. genomic research), financial analysis (e.g. monte carlo simulations), geophysical (e.g. gas & oil exploration), homeland security and more. Pulse easily moves data sets directly to the cluster nodes just in time for processing.

A basic workflow model is to create a job series which can be submitted to the existing scheduler environment. The job sequence begins with a script which interacts with the database / data cube to create and save the desired data set to file format. Next a preparation algorithm is applied to the data set and query list. The queries are sliced for optimal distribution across the cluster and the data set is diced into segments which fit entirely into RAM on the target nodes. The entire query list is contained in a single file that includes instructions as to which queries are to be run on each node. Pulse is then instructed to transfer the query list file and the data set to all the nodes in the cluster. Once Pulse has completed and verified that all data and queries are in place the actual command to begin processing on the cluster is given.

## **PERFORMANCE**

### **Removes Storage Bottlenecks**

Traditional cluster provisioning relies on unicast transfer mechanisms that make an individual connection to the file server for every node needing the same piece of data. Such a deluge of connections from even a small cluster can bring a file server to a thrashing, grinding halt. Pulse™ solves this problem by making only a single connection to the source file server (currently NFS, SMB, HTTP and FTP servers are supported) to retrieve the data. Even a modest file server will send data destined for the entire cluster as fast as if only a single node had made a request.

### **Efficient Multicast File Transfers Over Ethernet**

Pulse™ provides extremely fast and easy transfers of files across a cluster without generating excessive burden to the network. It does this by using UDP-based multicast to send data to multiple systems simultaneously. When the nodes in the cluster are ready for data, the Pulse™



Server makes a single connection to the file server and transmits the data to the nodes via a single multicast stream. This method allows for extremely high performance over inexpensive, commodity Ethernet.

### **Extremely High Throughput**

Bright Cluster™ Pulse™ delivers data as fast as a cluster can absorb. The Data Absorption Rate of a cluster is simply the speed that a node can accept data. Small files can be absorbed at full network speed because they fit in a node's memory buffer. Files or directories of files larger than the memory buffer though must begin writing to the local hard drive before the entire file transfer can occur. As the write speed to a hard drive is much slower than the network transfer speed the result is a data backup that causes thrashing and packet loss in the data stream. Pulse regulates the data flow below the absorption threshold resulting in a smooth data stream that delivers the maximum throughput possible to the cluster.

The Data Absorption Rate (DAR) is primarily determined by the speed that the local nodes can write data to disc. Modern drives have a typical maximum write speed of 25MB/s to the outer portion of the disc platters, far less than the 120MB/s max speed of Gigabit Ethernet. This rate decreases dramatically for data being written to the slower rotating inner tracks of the hard drive platters.

Data sets are typically larger than the memory buffer on a node. Once the buffer is full a node can not absorb any more data until space is made by writing data to disc. Data streams reaching a full node buffer are lost and must be resent. Multiple target systems though tend to be slightly out of sync and each loses different data packets. Each node then begins requesting different packets of data be resent which again results in network and storage thrashing that drastically slows data throughput.

### **Performance That Scales**

Pulse™ scales to hundreds and thousands of nodes delivering near linear throughput performance with minimal overhead. Eliminating the throughput bottleneck means eliminating the most significant constraint to the size and productivity of a cluster. Pulse™ is ideally suited for the largest clusters.

### **High Production Clustering**

Until now, throughput bottlenecks have constrained High Performance Throughput Clusters. The larger the cluster, the greater the cost of data storage and staging yet the processors continue to sit idle, waiting for data. Pulse™ enables the operation of larger clusters that return linear productivity performance without increasing marginal costs.



## EFFICIENCY

### **Automated Workflow**

Pulse commands are easily scripted for automated operation and integration with existing scheduler environments.

### **End-to-End Data Transfer.**

Pulse moves data directly from the primary storage system to the cluster nodes. No need for intermediate data staging.

### **File Server Proxying**

Simply enter the network location of any file, and Pulse™ will automatically transfer it to the cluster (currently NFS, SMB, HTTP and FTP servers are supported). Use of the web interface allows the user to access files on their local system.

### **Flexible Targeting**

The user specifies the targets of each Pulse™ command as part of the command. This allows great flexibility in transferring files or executing commands on only a subset of nodes quite easily. Targets can be specified as resolvable hostnames, Pulse™ group aliases, individual IP addresses and IP address ranges.

### **Utilizes Standard Local File Systems**

One of the benefits of Pulse™ is that one can make use of the fast access speed of data saved on the local node hard drive; improving performance and eliminating the network burden and administrative complexities of network mounted and/or parallel file systems. Pulse™ eliminates the need to NFS mount file systems on each node in a cluster.

### **Transfer Individual Files or Directories of Files**

Pulse™ makes it as easy for a user to transfer a file or directory full of files to any number of remote systems, as it is to transfer a single file to a single system.

### **Grouping**

Pulse's integrated grouping capability allows for flexible and dynamic organization of the cluster. Administrators can easily designate subsets of the cluster as distinctly named groups. Administering a group is as easy as administering a single system.

### **Compatibility**

Pulse™ works on standard network protocols and is compatible with any file type or file system supported by the standard Linux kernel and system utilities. Pulse™ operates completely in user space and supports the most popular Linux distributions including Red Hat, Debian and SuSE.

### **Low Memory Footprint**

Pulse™ uses less than 4KB of memory on each node.



## SECURITY

### **Secure Remote Access**

Pulse™ can be remotely accessed via secure http login to the Pulse™ Server from anywhere on the network.

### **Agent Persona Switching**

All remotely executed commands and file accesses are performed only after the agent process has switched its effective uid and gid to match that of the user executing the Pulse™ client application (users with sudo capability granted by the administrator can specify an alternate uid/gid to switch to). When no matching user exists or when the agent is awaiting a command, it operates as a completely unprivileged user ('nobody' or equivalent).

### **Utilizes Existing User Authentication Methods**

User authentication takes place between the Pulse™ server and client using existing user authentication systems such as Kerberos, LDAP, and NIS; or even by replicating static password files to the Pulse™ server itself. This ensures users do not have to learn new passwords or userids, but can access the Pulse™ server via their existing username. Once logged into the server as a given user, all remote commands (e.g. file transfers) are executed as that user on each remote target with no additional password entry required.

### **User Level Capabilities**

Once logged into the Pulse™ server, the user can only perform those Pulse™ commands, which have been allowed by the system administrator. By default, users are only permitted to transfer and delete remote files (subject to the standard filesystem ownerships and permissions on the remote systems). The administrator must specifically enable additional capabilities, such as managing group memberships, rebooting/halting remote systems or installing system images, before a user can execute them.

### **Digitally Signed Protocol Messages**

As Pulse™ makes extensive use of multicast UDP to efficiently communicate both data and commands to remote systems, there is no direct one-to-one connection between a remote system and the server through which traditional authentication and authorization can take place. Instead each Pulse™ command sent from the server to the agents is digitally signed using a private RSA key stored on the Pulse™ server. Each remote agent then uses its copy of the server's public key (securely transferred via scp as part of the agent install process) to validate each command message prior to processing it.

### **Secure File Server Access**

Pulse™ accurately handles NFS and SMB share mounting permissions as well as password accessible HTTP and FTP servers.



## RELIABILITY

### **Built In Data Integrity Checks**

As Pulse™ operates on a nominally unreliable multicast UDP transfer mechanism, it implements built in checks on data integrity and completeness for all data transfers. In addition, the user can request automatic MD5 checksum verification on any file transfer via NFS or SMB without the need for a precomputed hash.

### **Fault Tolerant Transfers**

If a node target loses one or more data packets during a transfer it will automatically request resends of lost data to ensure all data is received properly. Packets lost to one node are re-sent via multicast to all nodes. As any packet that is lost to one node is likely lost to multiple nodes, Pulse™ rapidly fills in missing sections of the data stream across the entire cluster.

### **Subscription Based Ownership & Support**

Bright Cluster™ Pulse™ is as easy to own, as it is to operate. Pulse™ is available on an annual per node subscription basis that includes all software updates and standard support from Bright Cluster™ engineers. Pulse™ comes delivered preinstalled on a Pulse™ Server appliance, or as part of a pre-built cluster.

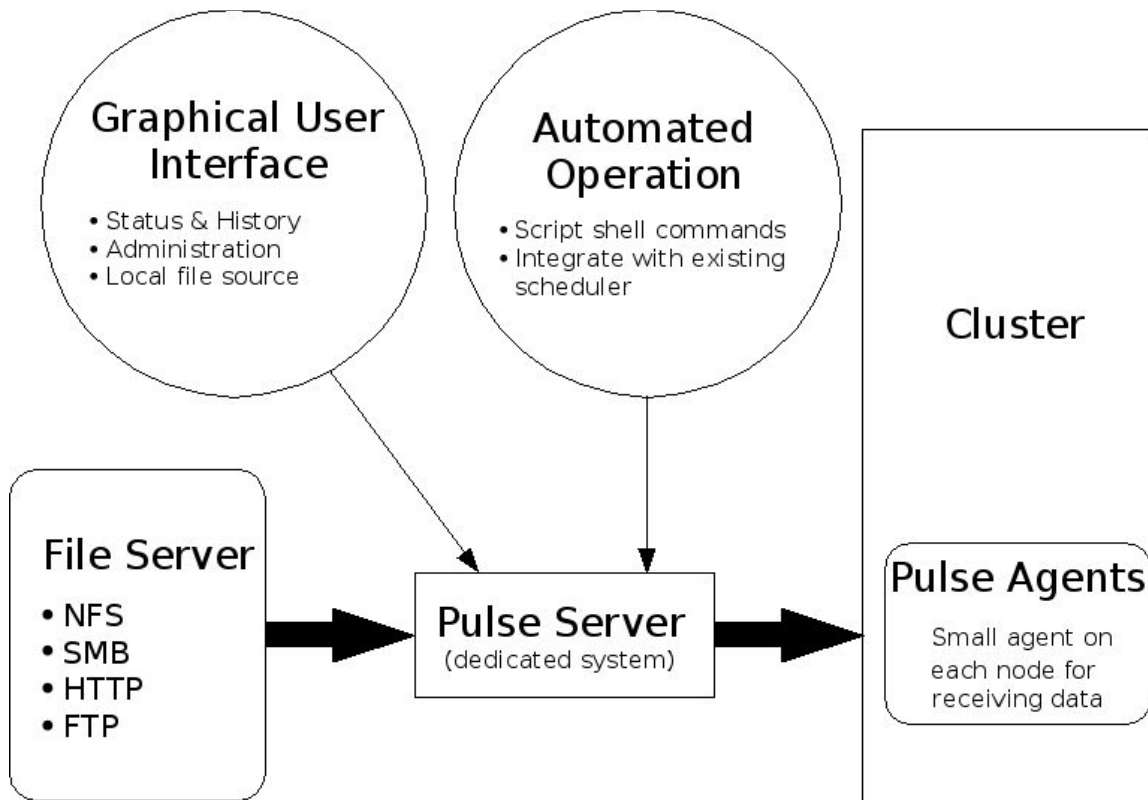
### **Simple Rugged Design**

Bright Cluster™ Pulse™ is designed to take the complexity out of cluster data management while providing fantastic throughput speed and rock solid reliability.



## OPERATION

Bright Cluster™ Pulse™ operates in a two-stage client-server model. Individual clients connect to the Pulse™ server to provide the user interface for users and administrators. When executing a command (for example, a file transfer) the server, in turn, communicates with agent daemons running on each remote system. The Pulse™ server is a dedicated system with network access to the cluster.



The user initiates commands via a graphical web based UI or scriptable commands from the command line. Almost all commands are of the form:

`<command> <options and parameters> <source><target>`

where `<targets>` is a space separated list of individual IP numbers, hostnames, IP ranges (xxx.xxx.xxx.xxx-yyy.yyy.yyy.yyy), or Pulse™ group names (aliases). After some syntax checking on the client side, the command is forwarded to the Pulse™ server where it is assigned a unique request ID.

Before processing the request, the Pulse™ capabilities of the uid of the user executing the client program are checked to see if the user is allowed to execute the command (commands such as changing group memberships are disallowed to ordinary users by default). If allowed, the target



list is processed, with hostnames resolved using standard DNS and group aliases replaced by their multicast IP address equivalents. At this point some command specific processing is done prior to contacting the remote systems. For file transfers, this pre-processing includes mounting the necessary network shares (if not already mounted), downloading the remote file (if the source file is on an HTTP or FTP server), and computing the MD5 checksum for the file(s) if requested.

After completing all server-side processing (provided no fatal errors occurred, in which case they would be reported back to the client and the request processing would be stopped), the server generates the agent-bound message, digitally signs it, and transmits it to the agent(s) via the Pulse™ multicast protocol.

Upon receipt, the Pulse™ agent first verifies the received message is valid by checking its digital signature using the locally stored public key for the Pulse™ server (installed automatically as part of the agent install process) as well as by verifying that the source address is known and that the message timestamp is accurate (messages with a timestamp older than 5 minutes are ignored). Once verified, the command is executed using the same persona (uid and gid combination) of the user executing the client program. This ensures no local files are overwritten, deleted, or stored unless the local filesystem privileges allow. Any local errors are reported back to the server and then forwarded to the client.

The actual transfer of files occurs via a multicast protocol in which each agent listens to a particular multicast address and port (dynamically assigned on a per-transfer basis by the Pulse™ server) for data. During the transfer, each target is queried for any missing packets, which are then also sent to the entire target group via multicast, allowing any number of nodes which also missed the same datagram to receive it without requiring a separate request. (this is critical as it is operating on the nominally unreliable UDP protocol).



**Bright Cluster, Inc.**

**408.829.4029**

**[www.brightcluster.com](http://www.brightcluster.com)**