

RESONATE | TECHNICAL WHITE PAPER

Resonate Central Dispatch™ An In-Depth Technical White Paper

Resonate's LAN-based traffic management that provides consistent, reliable availability and performance.

Resonate Central Dispatch provides traffic management capabilities that ensure dependable, predictable performance and responsiveness of e-business applications. Moreover, it introduces industry-leading features to extend its intelligent traffic management to XML-based business-to-business applications, wireless communication, and internal client/server applications. This helps e-businesses meet the challenge of assuring a positive online user experiences and satisfying user expectations, by providing preferential services in addition to high availability and performance.



Table of Contents

Resonate Central Dispatch™ Overview.....	3
Scheduling Rules.....	6
Load Balancing Policies.....	10
Resonate Central Dispatch Components.....	11
TCP Connection Hop.....	13
Advanced Features.....	14
Supported Environments.....	20
Scalability Guidelines.....	21

Resonate Central Dispatch™ Overview

Resonate's Central Dispatch allows users to provision and allocate resources among specified classes of users, delivering predictable user experiences based on differentiated service situations. These situations are dependent upon the user, application, class of transaction and specific system environment.

Central Dispatch is a patented traffic management software application that provides *active* Service Level Management (SLM) within a data center. *Active* SLM consists of a number of functions including: traffic management, systems, monitoring, and service level control. Central Dispatch is designed as a true distributed software application, monitoring resources, and managing network traffic to IP-based services residing across a collection of computers distributed across a LAN. This collection of distributed computers is viewed as a virtual machine and referred to as a Central Dispatch site with each computer referred to as a node. A Central Dispatch site is accessible to the rest of the network by one or more Virtual IP addresses (VIPs).

Roles of Nodes

Each node in a Central Dispatch site has one of the following functional roles (see Figure 1). The role of a node can change based upon the configuration of the Central Dispatch site.

Primary Scheduler

The Scheduler acts as the traffic cop for the Central Dispatch site. All inbound client requests to the VIP address are processed initially by the Scheduler. At the start of a connection the Scheduler:

- Determines which nodes have the requested service or resource, as indicated by a set of scheduling rules.
- Determines which nodes are available to serve the requested service or resource, based on the load balancing policies.
- Forwards request to the node that is optimally suited to service the request.

Backup Scheduler

Each Scheduler can have a designated Backup Scheduler that takes over the scheduling responsibility during failure of the primary Scheduler node. The Backup Scheduler ensures there are no single points of failure in a Central Dispatch site. Resonate recommends that a Backup Scheduler always be configured for redundancy.

Managed node

A managed node is the node where the service or specific resource resides, for example a Web content server or application server. A managed node receives

requests via the TCP Connection Hop from a Scheduler and responds to these requests. A managed node can support multiple services, thus allowing for application stacking and provisioning on larger enterprise class computers.

Multi-Function Node

At moderate traffic volume, a node can function concurrently as both a Scheduler and managed node, or Backup Scheduler and managed node. This allows for greater flexibility when designing a Central Dispatch site and reduces cost since no dedicated hardware and rack space is needed for the primary Scheduler and Backup Scheduler.

At higher traffic volumes, a dedicated Scheduler may be more appropriate since the Scheduler will be using more of its resources to schedule requests. Central Dispatch is self-regulating in that the Scheduler will automatically send less traffic to itself as the volume of traffic that the Scheduler is handling increases. A dedicated Backup Scheduler node may also be appropriate to prevent degradation of site performance and capacity should Scheduler failure occur.

Traffic Flow

The distributed software design of Central Dispatch provides for unparalleled scalability. All traffic flows through a Central Dispatch site in a triangular fashion: client connects to the Scheduler via the VIP, the Scheduler forwards the connection to the selected managed node via the TCP Connection Hop, then the managed node sends its response back to the client without passing back through the Scheduler. The client receives the reply from the virtual IP address (VIP) to which the client originally made the request (See Figure 2). Since the client is communicating to the site via a single VIP address, the Scheduler is able to maintain the appearance of a virtual machine and schedule new connections around failed nodes or services the instant a failure occurs.

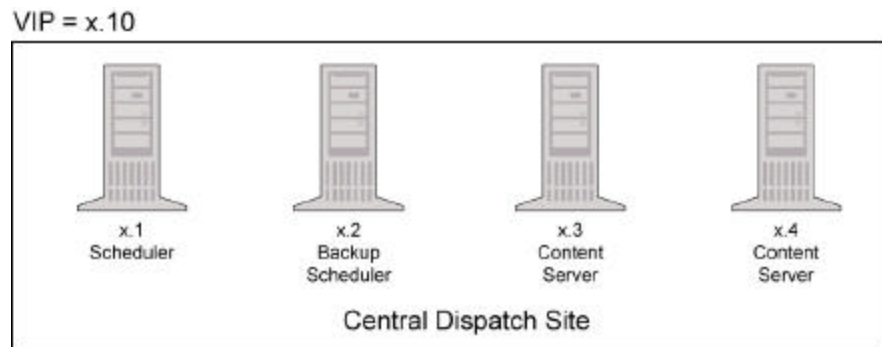


Figure 1 - Basic Central Dispatch site.

When the application running on the managed node (e.g., Web server, email server, LDAP server, custom application) receives the request, it sees the request as if the client had made the request directly. Thus, the application logs all transactions with the correct client identification. The design of this traffic flow allows Central Dispatch to support any application without customization. The

only customization that might be needed is a change in the application's IP address bindings (e.g., the IP address for a Web server would be the VIP address).

This triangular data flow was implemented and optimized for transactions of IP-based services. Clients generally make lightweight requests while the servers respond with heavy weight replies. Since the replies go from the managed nodes directly back to the client, Central Dispatch is able to scale performance to levels beyond that of other solutions. Central Dispatch's triangular data flow allows for complete utilization of network capacity, eliminating the Scheduler as a bottleneck.

Central Dispatch's distributed software design also allows for maximum network design and implementation flexibility. Because the TCP Connection Hop is a TCP-based encapsulation protocol, it can be passed through routers and firewalls. This allows the nodes to be connected to different subnets, VLANs, or even physical networks, providing an architecture that prevents most physical network devices from becoming a bottleneck or single point of failure. Nodes in a Central Dispatch site may also use non-routable IP addresses while the VIPs are public IP addresses. This type of configuration provides pseudo NAT functionality and enhanced security for the site without placing a potential bottleneck in front of the site like a firewall or other NAT device.

Primary Schedulers and their configured Backup Schedulers must reside on the same network broadcast domain (i.e., subnet or VLAN) since failover of the scheduling role utilizes gratuitous ARPs. If the Scheduler and its Backup Scheduler are located on different broadcast domains, the Scheduler failover process will not function properly.

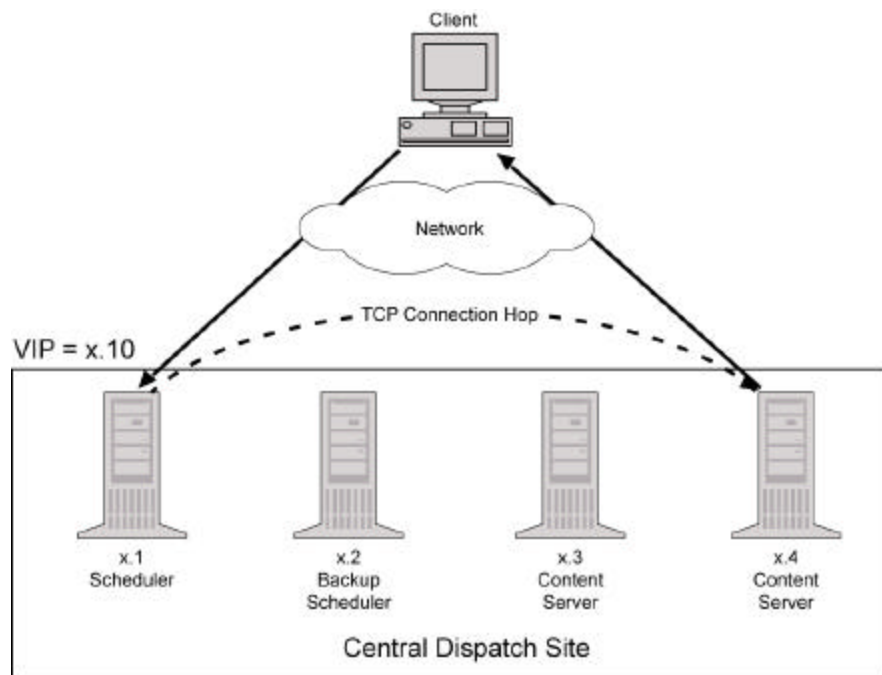


Figure 2 - Central Dispatch data flow diagram.

Scheduling Rules

Scheduling rules are the user-defined rules that tell the Scheduler how to determine which nodes are capable of handling a particular service or resource. The core parts of a scheduling rule include the following: a VIP to accept traffic on, a port number to accept traffic on, a set of servers to direct the traffic to, and the port on those servers to direct the traffic. The following example is for a simple scheduling rule that schedules Web traffic to a set of three servers:

```
10.10.10.1:80 —→    10.10.10.100:80
                   10.10.10.101:80
                   10.10.10.102:80
```

Scheduling rules can be constructed in an overlapping fashion. The Scheduler evaluates the scheduling rules in the order from most specific to least specific. This overlapping allows for the consolidation of multiple services on a shared set of systems and increased flexibility for effective site design (see Figure 3).

Once the Central Dispatch site is running, the only type of traffic that the Scheduler will accept on the VIP is an ICMP echo. Connection requests to ports with no defined rules will be rejected. Thus, the Scheduler provides a level of security for the servers and the VIP by accepting only connections for services defined by the scheduling rules.

Service-Based Scheduling Rules

Service-based scheduling rules are the simplest type of rules. When configured, these rules schedule traffic based solely on the service (LDAP, SMTP, etc.) that the client is requesting. The VIP and port number combination define the service.

Generic TCP Scheduling Rules

The generic TCP rules are used to schedule any TCP connection to a service. Generic TCP rules require fewer resources by the Scheduler to make the scheduling decision. Generic scheduling rules may be used for any TCP-based protocols, including Web, email, streaming media, gateway, directory.

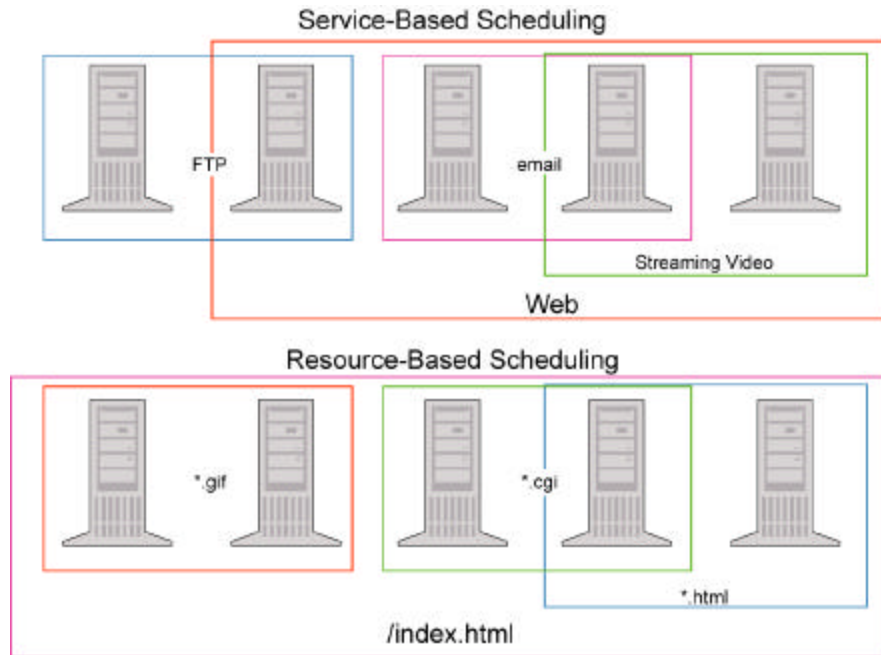


Figure 3 - Example of scheduling rule overlap.

Generic IP Scheduling Rules

Generic IP scheduling rules are used to schedule IP-based connections, such as UDP, to a service. Both persistent and non-persistent connections can be made on any IP-based protocol.

FTP Scheduling Rules

FTP scheduling rules are used for the scheduling of traffic for the FTP service. The FTP service is different from most other services in that it uses dual TCP channels (connections); a control channel and a data channel. Because of the multiple connections, FTP connections require special handling by the Scheduler.

IP Persistence Scheduling Rules

IP persistence scheduling rules are used for generic TCP connections that require persistence. This rule ensures the Scheduler remembers the client (based on the client's IP address), the time the client last connected, and what server the client was scheduled to. If the client connects to the site again, within the user defined rolling persistence window, the Scheduler sends the client back to the same server they used the previous time.

Persistence is used for applications that store user state information, cache data locally, or that do extra work during the initial client connection (i.e., SSL sessions that use encryption). Persistence allows these applications to work correctly and optimize the overall site by reducing application workload.

Persistent scheduling causes the Scheduler to store persistence information in memory for quick lookups. The use of persistence scheduling rules will increase the amount of memory that a Scheduler uses based upon how many unique clients connect to the service and how long the persistence window is. The nature of persistence is that it will modify the nature of load balancing to some degree and may result in non-uniform traffic distribution.

Resource-Based Scheduling Rules

Resource-based scheduling rules are more flexible than service-based rules, and provide a higher degree of control over the scheduling of traffic. Resource-based scheduling allows the Scheduler to direct traffic based not only on the service the client is requesting, but also on the specific resource the client has requested. Today Central Dispatch supports a number of different resource-based scheduling rules for Web-based applications.

HTTP URL-Based Scheduling Rules

HTTP URL-based rules require the Scheduler to look at the URL requested in order to make its scheduling decision. A simple wildcard search expression is defined that describes the path or document that each rule will match a URL against. HTTP URL-based scheduling rules allow Web content to be segregated either functionally or structurally across multiple servers. Segmenting Web content can dramatically increase performance by allowing servers to be optimized for the type of content that they provide, resulting in more efficient use of server resources.

HTTP Cookies-Based Scheduling Rules

HTTP cookie-based scheduling rules cause the Scheduler to look at the cookie passed to the server in order to make its scheduling decision. A simple expression is defined that describes the attribute-value pair that each rule will match a cookie against. HTTP cookie scheduling may be used to direct traffic from distinct types or classes of users to different servers. This allows clients to be scheduled to different servers providing a unique user experience to customers (e.g., customers who are purchasing or who have previously purchased are sent to less utilized servers versus customers who are simply browsing).

CGI Parameter-Based Scheduling Rules

CGI parameter scheduling rules cause the Scheduler to look in the CGI parameters passed to the server via the URL in order to make a scheduling decision. A simple expression is defined that describes the attribute-value pair that each rule will match the CGI parameters against. CGI parameter scheduling rules offer the same flexibility and control as the cookie-based scheduling rules. CGI parameters are also necessary for customers that disable cookies on their local browsers.

SSL3 Persistence Scheduling Rules

SSL3 Session ID-based persistence rules are used for SSL 3.0 connections. SSL3 persistence allows the Scheduler to identify the clients using a cascading set of identifiers. If the client is using SSL 3.0, they are identified by their SSL session ID, which is unique for each client and passed through Web proxies. If the client is not using SSL 3.0, they are identified by their IP address (however, this requires additional Scheduler resources).

SSL3 scheduling rules allow for fine-grained control over the identification of the client. When Web proxies are deployed between the client and the server, multiple clients are viewed as a single client based on the proxies IP address. Likewise, if there are multiple Web proxies, the client may go through a different proxy on subsequent connections. Using SSL3 scheduling rules for persistence allows the Scheduler to identify each client uniquely, regardless of potential problems caused by the proxies.

HTTP Cookie Persistence Scheduling Rules

HTTP cookie persistence is used to provide persistence-based on a cookie. This rule allows the Scheduler to identify the client-based on the value of a specified HTTP cookie. This provides another persistence method for Web sites using HTTP cookies.

CGI Parameter Persistence Scheduling Rules

CGI parameter persistence scheduling rules are used to provide another means of persistence, based on a CGI parameter. Similar to the HTTP cookie persistence scheduling rule, CGI parameter persistence rules allows the Scheduler to identify the client, based on the value of a specified CGI parameter. CGI parameters are also necessary for customers who disable cookies on their local browsers.

Note: HTTP cookies and CGI parameters are encrypted when clients connect to a Web site using SSL. In this case, HTTP cookie and CGI parameter persistence and scheduling cannot be used since these values are encrypted.

Custom Scheduling Rules

Data Stream Processing occurs when custom scheduling rules are used to parse for the inbound data stream for any data pattern within the entire packet. These are used to schedule persistent and non-persistent connections. Complex scheduling rules can be created based upon port numbers, TCP/IP header fields, expressions, or other parts of the inbound data stream.

Custom rules can also be applied to the content server's outbound data reply. The outbound data stream can be scanned for error codes, specific content, or any other data pattern that indicates success or failure of original request. Several actions can be taken if conditions are met.

- Redirect the request to another server
- Increment or decrement the server weight, or
- Disable the server from the cluster

Note: This subject is covered in more detail in the application note, "Central Dispatch: Data Stream Processing."

Load Balancing Policies

Once the Scheduler has matched an inbound connection against a scheduling rule, it has a set of one or more target nodes that can respond to the connection. If the set contains more than one node, the Scheduler uses its load balancing policy to choose which node is currently least loaded and optimally suited to handle the connection. Each policy uses one or more of the following load balancing metrics:

- CPU workload
- Open connections
- Network latency
- Server weight (externally set)

Central Dispatch Supported Load Balancing Policies

CPU Load and Connections (Basic)

This policy takes into account the current CPU workload of a node and the current number of open connections to that node. The number of open connections is given a higher preference. Server weights are also taken into consideration to adjust the traffic distribution. This policy works well for most configurations.

CPU Load and Connections (Enhanced)

This policy takes into account the current CPU workload of a node and the current number of open connections to that node. CPU workload is given a higher preference. Server weights are also taken into consideration to adjust the traffic distribution.

Round-Robin (Basic)

This policy chooses the best node by simply alternating through the set of nodes provided from the scheduling rule.

Round-Robin (Enhanced)

This policy chooses the best node by rotating alternately through the set of nodes provided from the scheduling rule. Server weights skew the distribution of traffic so that high weighted servers receive proportionally more traffic.

Custom

Allows for a customized load balancing policy in which both a primary and secondary metric is chosen from CPU load, open connections, and network latency. The primary metric is given a higher preference while the secondary metric receives a lower preference. Server weights are also used to manually adjust the load balancing.

Global vs. Per-rule Load Balancing Policy

Load balancing policies can be set globally, where all policies, by default, can operate on one of the aforementioned policies. For finer control, a load balancing policy can be set on each specified rule. In addition, a server weight can also be specified for all servers specified in the rule.

Load Balancing API

The load balancing API allows users that desire sub-second, programmatic control over the Central Dispatch scheduling policy. Resonate provides a C-header file for applications to reference the load balancing API. Users can influence the server weight in three ways:

- Set server weight
- Decrement server weight, or
- Increment server weight

This can be applied site wide to all nodes, one single node, or to all of the schedulers known by a single node. The changes can also be applied on a server or rule basis.

Resonate Central Dispatch Components

Central Dispatch consists of four separate components. The RXP device driver and Central Dispatch agent are required on each node in the Central Dispatch site, while CDMaster and CDaction are administrative and can be installed on any computer connected to the network.

RXP Device Driver

The RXP device driver is installed on each node in a Central Dispatch site and is responsible for scheduling inbound client traffic and sending/receiving TCP Connection Hops (see Figure 4). On the Scheduler, the RXP device driver accepts inbound connections addressed to the VIPs and uses the scheduling rules and load balancing policies to determine the optimal node for servicing the connection. The RXP device driver on the Scheduler then initiates the TCP Connection Hop process to the RXP device driver on the managed node.

Central Dispatch Agent

A Central Dispatch agent is installed on each node in a Central Dispatch site. The agent is responsible for monitoring and communicating site status, including node availability, health, and performance. The agent transmits site status information to the other nodes in the site through a heartbeat sent at a configurable interval. When the specified number of heartbeats are missed from a node's agent, Central Dispatch determines that the node has failed and marks it unavailable.

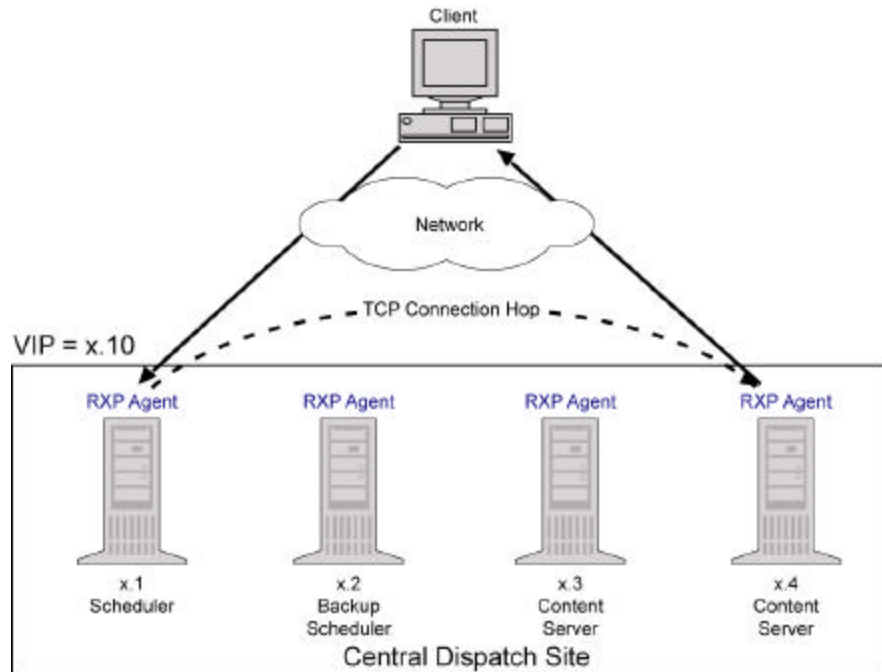


Figure 4 - Central Dispatch Site with node components.

Central Dispatch agents also communicate configuration changes across the site. Thus, an administrator is able to make a change to the configuration of the site through any agent, and that change is automatically propagated to all nodes within the site.

CDMaster

CDMaster is the graphical user interface (GUI) for performing administration and monitoring of Central Dispatch. It can be installed on a computer system that is either internal or external to the Central Dispatch site. CDMaster is a Java-based application that communicates to the Central Dispatch site in a client/server fashion. CDMaster communicates with a Central Dispatch site by connecting to a CDAdapter, which is an agent used to collect statistics. While connected to a site, it receives real time statistics and status information from the site.

CDMaster operates in two different password protected modes: Monitor and administration. In monitor mode, CDMaster displays real-time graphical status information from the site. Monitor mode also displays the current message log for the site since connection time. Monitor mode allows operations staff, management, or clients to view the current status of a site while preventing any changes to the site configuration.

In administration mode, CDMaster allows both monitoring of the site status as well as modifying the configuration, e.g., start or stop the site, disable nodes for maintenance, or purge the current message log. The configuration of the site can be changed by adding nodes, removing nodes, adding VIP addresses, removing VIP addresses, modifying load balancing policies, modifying message alerts, and modifying scheduling rules. Most configuration changes to Central Dispatch can

be made while the site is live and operational, without disrupting the flow of traffic into the site.

CDaction (Command Line Interface)

CDaction is a Java-based command line administration interface for Central Dispatch. It can be installed on a computer system that is either internal or external to the Central Dispatch site. CDaction has the functionality of CDMaster with the exception of the real-time, graphical traffic monitoring. CDaction can be used to integrate Central Dispatch with other monitoring scripts, application packages, and tools allowing them to make changes to the configuration of the Central Dispatch site.

TCP Connection Hop

The TCP Connection Hop is the core of the patented technology of Central Dispatch. The TCP Connection Hop allows Central Dispatch to have no tradeoffs in features or performance. The Primary Scheduler uses the connection hop to transfer requests to managed nodes. The TCP Connection Hop is a TCP-based encapsulation protocol that the RXP device drivers on each node use to talk to each other. The RXP driver on the Primary Scheduler receives the original connection from the client, encapsulates that connection within the TCP Connection Hop and sends it to the target managed node. The RXP driver on the managed node removes the encapsulation to recover the original connection from the client and then passes that connection up to the application to be handled.

The real power of the TCP Connection Hop comes from the level of intelligence contained within the process. The TCP Connection Hop contains a level of in-band testing that provides a second level of intelligence with regards to server and service availability. If the node that the TCP Connection Hop is sent to has failed, and the Scheduler hasn't detected that failure yet, the TCP Connection Hop will fail. The Scheduler will detect that failure, penalize that particular node so that it is less likely to be chosen again, and re-hop the connection to another managed node that is available and contains the requested service or resource. If another TCP Connection Hop fails in the future, the server will be further penalized, eventually reaching a point where a test connection is sent to the server occasionally to see if the server is back up. When a TCP Connection Hop succeeds to the server, all penalties are removed. A TCP Connection Hop failure, as described above, can also occur if the application on the target node is not running (e.g., no application is bound to the specified target port).

Note: this subject is covered in more detail in the Resonate Central Dispatch TCP Connection Hop white paper.

Advanced Features

Affiliated Servers

Affiliated servers are nodes that do not have the Central Dispatch software running on them, yet a Scheduler can still direct traffic to them. The uniqueness of affiliated servers is that they do not have the RXP device driver or the Central Dispatch agent installed on them. Affiliate nodes allow for greater design flexibility by allowing any computer or network attached device, regardless of the operating environment, to be included under the traffic management umbrella of Central Dispatch. The same Primary Scheduler that schedules traffic to normal managed nodes can also schedule traffic to affiliated servers.

Administrators still gain the fine grain control and monitoring that Central Dispatch provides.

There are three modes for affiliated servers.

- **Half-NAT (Network Address Translation)**
The scheduler traffic is accepted on the VIP, and then is forwarded to the affiliate node server. The key point is that the source IP address remains the same. This is used in situations when the client IP address must be retained.
- **Full-NAT**
The scheduler traffic is accepted on the VIP, and then is forwarded to the affiliate node server by changing the source IP address to be the IP address of the scheduler. This is used in situations when affiliate nodes do not reside on the same subnet as the scheduler.
- **MAT (MAC Address Translation)**
The scheduler traffic is accepted on the VIP, and then is forwarded to the affiliate node server by changing the MAC address of the data frame and retransmitting it on the network. This is used in situations for router-like resources, such as, transparent firewalls and transparent caches. This is also used to overcome the limitation in Network Address Translation by sending the outbound reply directly from the content server.

There are a couple of caveats to remember with affiliated servers. First, half-NAT and full-NAT affiliated servers are not able to return their response traffic directly back to the client. All response traffic from affiliated servers must pass back through the Scheduler (See Figure 6). MAT affiliated servers are able to maintain triangular data flow.

Second, Central Dispatch is not able to measure the actual CPU load of affiliated servers. This load balancing metric is not taken into consideration when a Scheduler schedules traffic to affiliated servers.

Note: Schedulers for a Central Dispatch site cannot be affiliated servers.

Class of Service Thresholds

The class of service (COS) threshold feature allows multiple thresholds to be configured for each node, where each threshold represents a different class of service for the same or different applications. The thresholds are based on the number of open connections and help ensure that an application never receives

more traffic than it can handle. When a scheduling rule is created, one of three different classes of service can then be optionally chosen for that rule. The Scheduler will never send more connections to a node than what is specified by that node's COS threshold. When all COS thresholds have been reached, then the Scheduler will by default reset new connections for the application.

By employing COS thresholds on a site that has an application whose scalability limits have been characterized, users are ensured a higher class of service when interacting with applications. Central Dispatch will ensure that each application configured with a COS threshold will never receive more traffic than it can handle. So, during times of unexpected high traffic volume, an application will remain stable and available to address the needs of users currently requesting services.

Failover Servers

The failover servers feature allows one or more nodes to be specified within each scheduling rule to act as failover servers. The Scheduler directs traffic to the failover server nodes if all standard nodes were to fail or to have met their COS threshold. If multiple nodes are specified, then the Scheduler will continue to use the selected load balancing policy to determine which server is to receive traffic.

By using failover servers with the COS thresholds, users experience a higher level of service. If an unforeseen spike in user demand were to cause all node's thresholds to be reached, the failover servers can be used to direct excess users to a "busy signal" server. This server can inform users that you are experiencing high demand, and might even offer them a something for their patience, such as a gift.

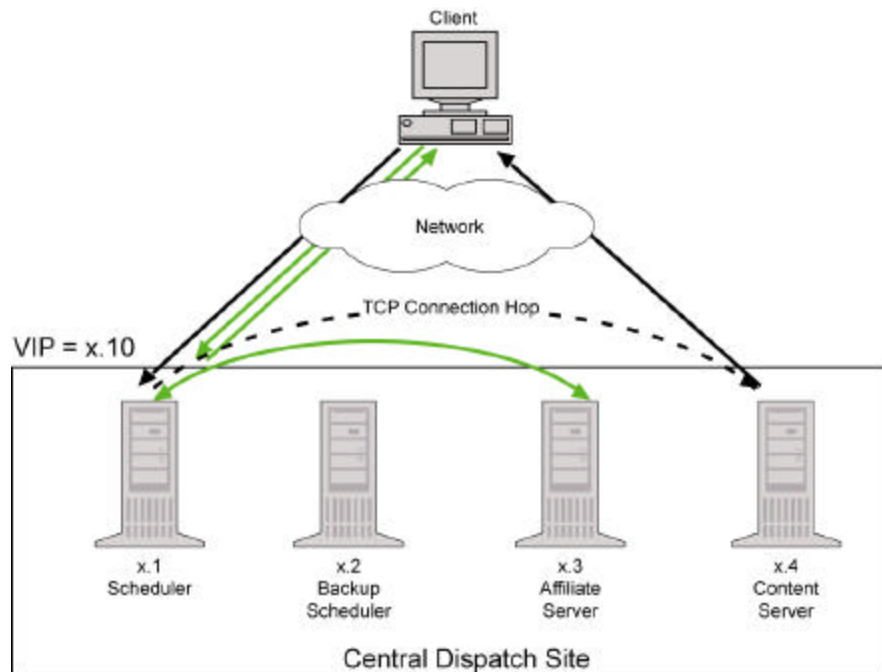


Figure 6 - Central Dispatch site data flow with affiliated server.

Firewall Load Balancing

Users are able to optimize traffic to router-like devices, including transparent firewalls. Central Dispatch operates in affiliate mode to schedule to the firewalls; a Scheduler is required on each side of the firewalls. Because firewalls ensure stateful communication, i.e., to ensure that spoofing does not occur, it is necessary to direct the outbound reply to the same firewall that handled the inbound request. The schedulers communicate to the firewall using MAT (MAC address translation).

Port Mapping

The port mapping functionality allows Central Dispatch to accept traffic on one port while scheduling that traffic to the server nodes on a different port. As an example, the Scheduler could accept Web traffic on the standard port 80, but then schedule that traffic to Web servers, which are running on port 8000.

Port mapping allows for increased security by masking the true port that an application is listening on. Port mapping also allows multiple instances of the same service to be supported on the same node, by running each instance on a different, non-standard port and then having a separate VIP for each service.

Port Load Balancing

A Scheduler may distribute traffic across multiple servers, but also across multiple ports on the same server. This feature is called port load balancing, where multiple ports are given for each managed node in the scheduling rule and the Scheduler then evenly distributes the connections across each port.

Port load balancing provides administrators with a solution to help overcome applications that have scalability limits and are not able to take full advantage of larger multi-processor enterprise class servers. The same application can be run on multiple ports on the same server, and Central Dispatch will distribute traffic across all the ports. This allows enterprise class servers to be fully utilized or provides a mechanism to cover the time until application scalability issues can be overcome. Clients accessing the site still interact with the primary application port so no changes are necessary, easing potential administrative issues.

Port load balancing can be combined with port mapping to different ports and ranges. For example, traffic coming into ports 100-105 can be redirected to ports 200, 202, 221-224.

IP Load Balancing

IP load balancing is used when users have multiple instances of the same application bound to different local physical addresses. BEA WebLogic is once such application. IP load balancing allows users to create a mapping table of IP:Port combinations to specify to a VIP.

For example, for a VIP (10.0.0.10, in the example), an application may have the following instances:

	Server A	Server B
Application 1	10.0.0.21:130	10.0.0.31:130

Application 2 10.0.0.22:130 10.0.0.32:130

However, the applications will not be listening as they are not bound to the VIP address. This is handled by creating a mapping table of IP aliases to ensure that each mapped port corresponds to a single application instance on each server. An example of the table above included with IP aliases:

	<u>Server A</u> (Mapped IP:Port → Real IP:Port)	<u>Server A</u> (Mapped IP:Port → Real IP:Port)
Application 1	10.0.0.10:131 → 10.0.0.21:130	10.0.0.10:131 → 10.0.0.31:130
Application 2	10.0.0.10:132 → 10.0.0.22:130	10.0.0.10:132 → 10.0.0.32:130

This will ensure proper scheduling to each application instance regardless of the IP:port combination. This is administered with the command line utility, `minrxpctl`.

Shadow Scheduling

The shadow scheduling functionality can be turned on to automatically synchronize the persistence information between Scheduler and Backup Scheduler. A configurable synchronization period can be set at which time the changes to the persistence information on the Scheduler is replicated to the Backup Scheduler. Periodically, the Scheduler will do a full table synchronization to the Backup Scheduler to ensure that no entries are lost. Shadow scheduling ensures that upon Scheduler failure the Backup Scheduler is able to accurately maintain all persistent sessions. Shadow scheduling also works with multiple Schedulers; each Scheduler synchronizes its persistence table with its associated Backup Scheduler.

Multiple Schedulers

In certain high traffic volume situations, the Scheduler for a Central Dispatch site can become resource constrained. This is the point at which the Scheduler is doing nothing but scheduling traffic—this point keeps moving up as CPU speeds keep increasing, refer to the Scalability Guidelines section for this exact number using today’s highest speed processors. To eliminate the Scheduler as a bottleneck in such a scenario, Central Dispatch has the flexibility of allowing multiple Schedulers to be running in parallel scheduling traffic to a single pool of managed nodes.

Each Scheduler maintains its own set of VIP addresses and should be assigned a Backup Scheduler (see Figure 7). If a single service is receiving high level of traffic, then the traffic to that service is split between two different VIPs—one on each Scheduler—using either Resonate Global Dispatch™, or round robin DNS, depending on the site configuration. If multiple services are receiving an aggregate high level of traffic, then the VIPs for these separate services are redistributed across the multiple Schedulers in order to increase scalability.

CDMaster also condenses the statistics for the multiple Schedulers so the Central Dispatch site is still viewed as a single virtual hostname. The hits and open connections displayed for the Central Dispatch site in the Site View screen are the aggregate value from each Scheduler.

Multiple Schedulers allow for continued scalability of a Central Dispatch site under extremely high traffic volumes and provides increased design and implementation flexibility. By having Schedulers in two different locations belong to the same Central Dispatch site, the Schedulers can take advantage of managed nodes located in both locations.

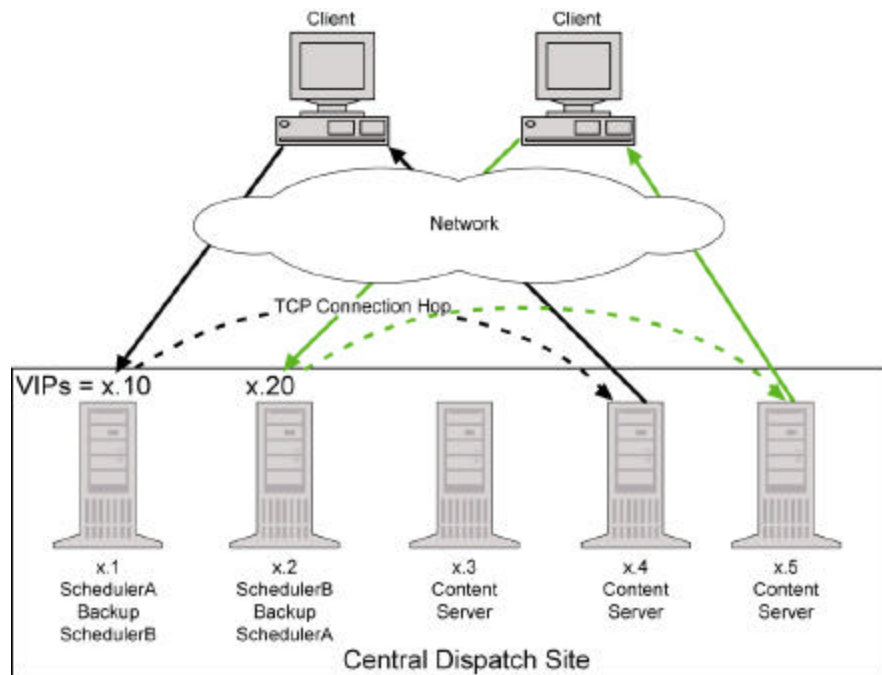


Figure 7 - Multiple scheduler Central Dispatch site.

Shopping Cart Persistence

Normal persistence scheduling rules store separate persistence information for each scheduling rule. However, there are times when shopping carts are employed on a site when users will be scheduled to a server using an IP persistence rule (port 80) to browse a catalog and add items to their shopping cart. When users want to check out and pay for their items, they are then connecting to the site using SSL and are scheduled using a SSL persistence-scheduling rule (port 443). Since normally both of these persistence rules keep their information separate from each other, the user may not be sent back to the same server where their state information is stored, when they connect using SSL. However, when the shopping cart persistence option is turned on, both persistence-scheduling rules (ports 80 and 443) use the same persistence information, thus ensuring that users are always sent back to the correct server with their state.

SYN Attack Protection

SYN attacks are a type of denial of service (DOS) attack that can be launched at an Internet site to block other users from accessing the site. SYN attacks work by sending a continual stream of SYN packets to the site, for each SYN packet the server returns a SYN/ACK packet and then waits for a return ACK packet. Servers classically have a limited amount of memory available to store state for

pending connections. If large volumes of SYN packets are sent to a site with no following ACK packet, the server's available memory will fill up and not be able to accept any new, legitimate connections, thus denying all access to the site.

Since all VIP traffic first contacts the Scheduler, and the Scheduler is doing delayed binding, all SYN attacks are isolated on the Scheduler. The RXP device driver on the Scheduler is then able to protect the site against SYN attacks because it is able to dynamically allocate additional memory to use for holding the SYN packets. The RXP device driver uses less memory to store each SYN packet than typical TCP stacks. These design features, along with the device driver's ability to actively flush SYNs that have been unacknowledged for too long, help protect a Central Dispatch site from SYN flood attacks. As long as the SYN attack is not filling up the site's network connection to the Internet, legitimate traffic can still reach the site and be scheduled to servers during an active SYN attack.

Graceful Server Shutdown

There are occasions when administrators desire to remove or disable a server from service. Normally, shutting down the server is abrupt and causes TCP resets to be sent to the connected clients. This situation can be alleviated by either of the following two methods.

- **Graceful connection shutdown**

When the environment variable `RES_GRACEFUL_AGENT_SHUTDOWN` is set, a server disable will immediately prevent any new requests from being sent to the server. Existing persistent connections will continue to be serviced. Connections will be removed as each reaches the scheduling rule timeout. This permits the administrator to monitor the number of open connections prior to disabling their applications.

- **Soft- & hard-timer shutdown**

This method is bounded by two environment variables: `RES_SOFT_DISABLE_TIMER` and `RES_HARD_DISABLE_TIMER`. When these variables are set and the server is disabled, new connections are prevented from being sent to the server. Existing persistent connections will continue to be serviced. Connections will be removed as each reaches the soft timer or scheduling rule timeout, whichever is less. When the hard timer expires, all connections are removed. This permits the administrator the most flexibility in controlling when to disable servers and applications.

Client Latency for Global Dispatch

Central Dispatch can also optimize the performance of Resonate's WAN-based DNS product, Global Dispatch. When Global Dispatch chooses to use Central Dispatch's latency feed, Central Dispatch collects the TCP SYN to SYN-ACK latency for each client. This is passed to Global Dispatch, which replaces the latency cache information in order to help decide which POP to go to. This is the true measure of the client latency versus latency obtained from reverse DNS lookup, which is actually to the DNS server, not the client. There is an added side benefit: No additional ports need to be opened through the firewall since it is not necessary to contact DNS servers through reverse lookup.

Session Latency Statistics

The client transaction time, or session latency, is an important measure of ensuring that service level objectives (SLO) are being met. Session latency is the time from the first byte (TCP SYN packet) received from the client to the last byte transmitted in the reply (TCP FIN packet). This end-to-end measurement, unique to Resonate, is a weighted moving average applied to all servers in a rule.

Silent installation

Central Dispatch ease of installation is extended to software distribution products through parameterized installation. Installation choices can be provided as command line parameters or configuration file. This allows software distribution packages to automatically install Central Dispatch on any node.

SNMP Agent & MIB

Central Dispatch is instrumented with its own SNMP MIB. This allows SNMP monitors and network management systems to access Central Dispatch statistics directly for trending and display purposes.

Supported Environments

Central Dispatch supports the following in either a homogenous or heterogeneous environment.

Platforms

- Sun SPARC and ultraSPARC (Solaris)
- Intel Pentium Family (Windows 2000, Windows NT)
- IBM PowerPC (AIX)
(see www.resonate.com for updates on supported platforms and operating systems)

Operating Systems

- Solaris 2.6, 7 (32-, 64-bit), 8 (32-, 64-bit)
- Windows 2000 Server and Advanced Server with Service Pack 1 or 2
- Windows NT 4.0 with Service Pack 5 or 6a
- AIX 4.3.1, 4.3.2, 4.3.3

Network Architectures

Operating Environment	Ethernet	Fast Ethernet	Gigabit Ethernet
Solaris	X	X	X
Windows 2000	X	X	X
Windows NT	X	X	X
AIX	X	X	

Scalability Guidelines

Central Dispatch has been tested and certified under the following scalability guidelines for a single Central Dispatch site (certain guidelines can be exceeded with minimal performance degradation—contact Resonate for further details):

Maximum number of nodes	64
Maximum number of schedulers	16
Maximum number of VIP addresses	512
Maximum number of concurrent requests per second per scheduler	Determined by speed and architecture of the Scheduler node's processor. Independent testing by the Tolly Group and Resonate benchmarks have shown 6,300 hits per second on a 700 MHz Intel-based or a 440 MHz UltraSPARC-based Scheduler.
Maximum number of concurrent inbound requests within a site	Determined by the Scheduler capacity and number of Schedulers per site.
Maximum number of concurrent open connections within a site	Determined by the capacity of each node computer system. 100,000+ observed in current customer production environments.
Maximum data throughput	Determined by the network architecture design. There is no theoretic limit based on Central Dispatch's architecture. Benchmarks have shown up to 1 Gbps.
Y2K compliance	Central Dispatch is certified as Year 2000 compliant.

Resonate is a registered trademark, the Resonate logo, Keeping E-Business Open for Business, Resonate Central Dispatch, Resonate Global Dispatch, and Resonate Commander Solutions are trademarks of Resonate, Inc. All other trademarks are the property of their respective owners. Copyright © 2001 Resonate, Inc. 10/001 TWP013.