Making Clustering Work with SecureTransport

A Technical Overview of
High-Availability, High-Capacity and Geographically Distributed Clustering Models
from Tumbleweed Communications Corp.

Summary

Enterprise IT managers can reduce risk and increase transaction throughput by implementing Tumbleweed Communications' SecureTransport in clusters. SecureTransport 4.7 supports both **high-availability** clustering to limit risk in disasters and **high-capacity** clustering to cope with traffic spikes.

This paper describes clustering configurations supported in SecureTransport 4.7, limitations on that support, and the advantages and disadvantages of each configuration. Also described are the exposure inherent to clustering with Secure Transport 4.7 and steps for disaster recovery under SecureTransport 4.7.

For more information, Tumbleweed invites you to contact your account manager or technical account manager.

Background and Concepts[1]

Illustration

Whether to limit risk or improve performance, enterprises often configure multiple servers in clusters. A cluster contains one primary server (machine) and one or more secondary servers. The primary server distributes tasks among servers in the cluster. Both primary and secondary servers process requests.

When a server in a cluster goes down, two events occur. First, the load balancer detects the server outage and re-routes incoming tasks to a secondary server. In this case, the original session ceases and the client that initiated the task must reestablish the session. Second, the cluster promotes one of the secondary servers to primary and re-routes tasks from the downed server to other servers in the cluster.

Clusters generally offer either **high availability**, uptime and reliability; or **high capacity**, performance and responsiveness. These objectives and the configuration necessary to support them are mutually exclusive, so enterprises choose the model that best fits their business and support resources.

High-Availability Configuration

Description

A high-availability (or active/passive) cluster relies on a third-party load-balancer to distribute tasks among the different servers. There is no automatic replication of information collected by the primary server to the secondary server; rather, the primary server passes that information along to the database. The secondary server remains in a passive state. When the primary server fails, the cluster fails over to the secondary server. Before becoming active, the secondary server retrieves all of the event, configuration, and state information from the database.

Advantages/Disadvantages

The main advantage in using a high-availability cluster is that, with identical hardware on all servers, performance does not degrade. Also, high-availability configuration minimizes cluster administration and maintenance. For example, administrators can use the passive server to test updates and patches without directly affecting the production servers.

However, during production hours or high usage times, it is not possible to leverage the hardware resources of the secondary servers or use them to balance the load.

[1] Portions abstracted from *Tumbleweed SecureTransport Release 4.7 Administrators Guide.*

High-Capacity Configuration

Description

High-capacity (or active/active) clustering balances the load among the primary and the secondary servers. In this way, all servers are active. The cluster automatically replicates all event information collected by the primary server to the secondary server. If the primary server fails, the cluster knows that the primary went down and it automatically switches control to the secondary server.

Advantages/Disadvantages

The main advantage of a high-capacity cluster is that it balances the load automatically among the different servers in the cluster. Also, the secondary servers are in "hot" mode, meaning that they can take file transfer activities immediately, or nearly immediately, after the primary fails.

However, a high-capacity cluster can suffer from performance degradation in the event of failure. For example, if the secondary server has to take over while both the primary and secondary servers are past 50% usage, and the primary server fails, the secondary server cannot carry the load for both servers. When a secondary server has to take over for a failed primary, there is some performance degradation across the entire cluster, so it is important to consider the load on each server.

Geographically Distributed Clustering

Many enterprises go beyond distributing clustered machines in a single data center to distributing clusters in data centers across a country or around the globe.

This has the advantage of spreading traffic load and risk not only across machines but also across time zones and geography. Geographically distributed clustering allows tuning for minimal response times and optimal disaster recovery on a regional, national or global scale. With geographic dispersal, however, comes the overhead of latency, as more variables (switches, time, wide area network conditions) fall outside of the enterprise's control.

Tumbleweed Communications' SecureTransport

Clustering with SecureTransport

Enterprise customers often deploy SecureTransport in clustering scenarios similar to those described above. SecureTransport supports high-availability and high-capacity configurations, with limitations described in this paper; however, Tumbleweed Communications does not support the use of SecureTransport in conventional, geographically distributed clustering.

Evolution of support for clustering with SecureTransport

As of version 4.6, SecureTransport supports the entire infrastructure required for high-capacity, active/active clustering.

- With load balancers (at least two) and shared storage, the SecureTransport servers become virtualized and the cluster routes connections to the least loaded or currently available server.
- User account information, configuration data and the event queue containing information about current tasks reside in a relational database.
- The cluster synchronizes all account and event information among nodes in the cluster.

SecureTransport v. 4.6 supports high availability through an active/active implementation, but it does not support true active/passive clustering. SecureTransport Client can reconnect and retransfer a failed session (auto-retry and auto-restart) and perform file integrity checks, resulting in the same high availability typical of active-passive clustering.

SecureTransport version 4.7 features several improvements for clustering:

- High availability with true active/passive implementation.
- Diagnostic tools for gauging more accurately the current state of the cluster, including heartbeat among the nodes, making it easier to perform functions normally associated with clustering, such as adding and removing a primary or secondary server.
- XML import/export utilities for generating a snapshot of each server in the cluster (configuration, user accounts, file routing data).
- Scheduler and Folder Monitor services always running on the current primary server in the cluster. If the primary server goes down, the secondary server that replaces it as primary takes over the Scheduler and Folder Monitor services. To this end, the cluster replicates the Scheduler configuration from the primary server to all the machines in the cluster.
- Account Manager for distributing dynamic information to all servers in the cluster when a user logs in or changes password.
- Transfer Status Manager to consolidate the transfer log on the primary server. All transfer log entries reside in the database on the primary server.

These improvements in version 4.7 take greater advantage of the SecureTransport architecture.

Architecture

Inside the data center (Figure 1), each SecureTransport node consists of two separate products running on two separate machines:

- SecureTransport Edge, a file transfer gateway for secure DMZ streaming, prevents the storage of sensitive information on unsecured network equipment.
- SecureTransport Server, the file transfer platform, provides a centrally managed system for monitoring and managing secure file transfer activity across multiple file transfer sites or applications.

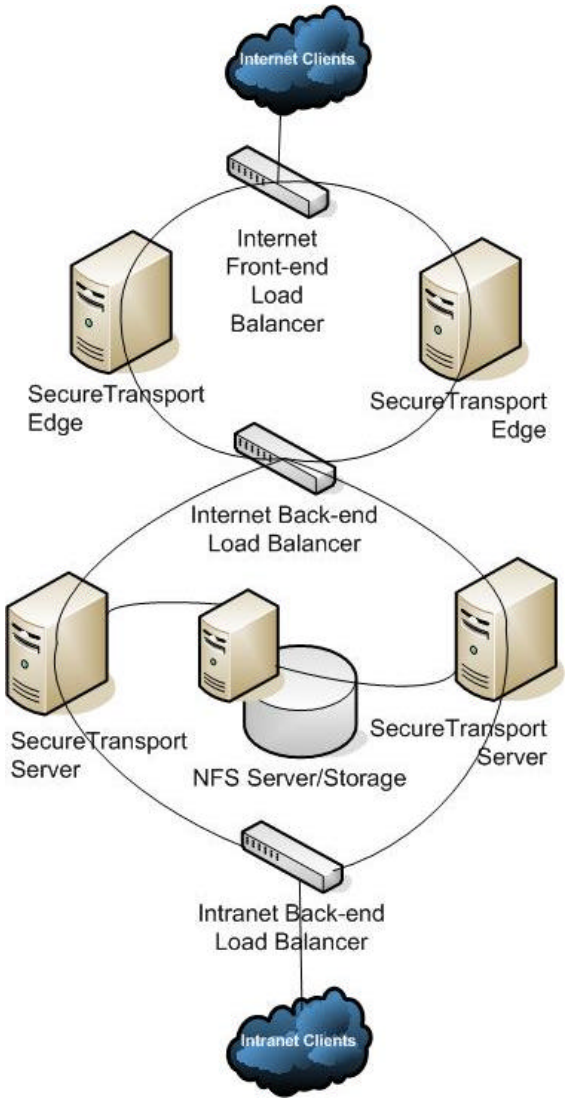

Figure 1 - Clustering with SecureTransport

Thus, each SecureTransport node is in fact two discrete components. Aside from these nodes, SecureTransport requires other elements common to clustering configurations, which are separate from the SecureTransport platform:

- Load Balancer (absolute requirement). SecureTransport in a cluster requires an appliance for failover intelligence at the Internet front-end of SecureTransport Edge and another to balance among back-end SecureTransport Servers. Depending on internal volume, another appliance can also balance intranet traffic among back-end SecureTransport Servers.
- File Storage (absolute requirement). A storage tier for the repository connects to the back-end Servers using Network Attached Storage technology such as NFS (Network File System).
- Backup/Replication. Successful recovery from equipment failure or disaster will depend on a recent backup.
- Monitoring Tools. Adding or removing servers from the configuration depends on accurate information about the state of the cluster, which these tools report.

Communication among the nodes

The improvements in version 4.7, along with the solution architecture, require SecureTransport Servers to exchange status information constantly, at the same time that they are performing the principal business of securely transferring files. SecureTransport Servers are also constantly exchanging and therefore replicating data on any changes in the Admin UI – including records tracked by the Admin UI – and communication from the Transaction Manager. This constant communication accounts for the high degree of "chattiness" in a cluster running SecureTransport and underlies its differences with conventional clustered applications.

How Clustering in SecureTransport is Different

Certain characteristics of clustering with SecureTransport distinguish it from clustering in other systems (e.g., database and application servers) by configuration, best practices and disaster recovery. As a result of these differences, SecureTransport supports high-availability and high-capacity configurations, but does not support conventional, geographically distributed clustering. Furthermore, enterprises deploying SecureTransport in clusters should understand their exposure to data loss and take steps to limit it.

1. SecureTransport-Edge-to-SecureTransport-Server coupling

In multi-tier architecture, SecureTransport Edge acts as a gateway to prevent the storage of sensitive information in unsecured network zones. SecureTransport Edge terminates a client-initiated file transfer session on any protocol, but the session state resides in the SecureTransport Server to which it is connected. Thus, SecureTransport Edge is effectively coupled to and configured to make connections to a single SecureTransport Server.[2]

Consider inbound tasks to the primary SecureTransport Edge, which the cluster then routes to the primary SecureTransport Server (Figure 2). When the primary SecureTransport Edge goes down, the load balancer re-routes those tasks to the secondary SecureTransport Edge, which can route them only to the secondary SecureTransport Server. This leaves the primary SecureTransport Server still operational yet unavailable for processing tasks, because of the coupling to the disabled primary SecureTransport Edge. The primary SecureTransport Server can still process internal tasks originating behind SecureTransport Edge, but not inbound tasks from the Internet.

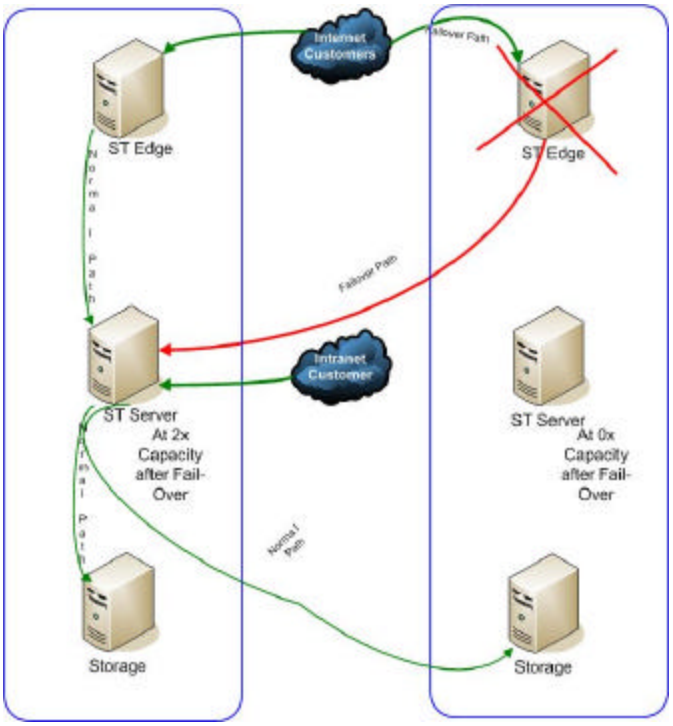

Figure 2 - Primary Fails Over to Secondary

Note that SecureTransport Server can maintain simultaneous connections to multiple storage networks, so both storage networks would remain available to the secondary SecureTransport Server in this scenario.

So, in contrast to a typical cluster, SecureTransport Edge and SecureTransport Server in a cluster configuration do not behave as completely independent, clustered failover machines.

2. Communication overhead complicates geographic-distribution model

As described above, the role of geographic distribution is important to conventional, high-availability clustering. However, the overhead associated with SecureTransport's method of file transfer often results in greater latency than is typical for geographically distributed clustering. The delays in processing may still be acceptable when the average message is extremely

[2] This "single Server" may be virtualized through a load balancer.

large, but they usually consume an unacceptably high percentage of total traffic as message size decreases. As such, SecureTransport does not support conventional, geographically distributed clustering.

3. Protocol

One of the technologies currently used to support the constant exchange of information among SecureTransport Servers depends on multicast IP packets. Because most gateways block these from going outside of a VLAN (virtual local area network),. the packets are not optimal for conventional, geographically distributed clustering.

SecureTransport supports other protocols, but no single protocol meets the needs of a high-availability, geographically distributed configuration.

4. Automatic replication vs. manual synchronization

Besides its normal work, every cluster must allow for certain landmark events, notably the addition of servers, the removal of servers, failover in case of a downed server, and disaster recovery for rebuilding/replacing a downed server. These landmark events would go smoothly if all necessary data were kept up to date in real time and automatically propagated across the cluster, but there are high costs associated with attaining those goals, so most configurations entail some element of compromise.

To allow for these events, SecureTransport distinguishes between **automatic replication**, in which the primary server is continually exchanging data with secondary servers in the background, and **manual synchronization**, in which the administrator launches the process of exchanging those data. Replication is ongoing and spontaneous; synchronization entails a service outage of 1 to 5 minutes in the cluster, during which SecureTransport can execute existing tasks but cannot accept new ones.

Table 1 lists types of data and how SecureTransport deals with them. Note that in high-capacity clustering with SecureTransport, all machines are actively replicating these data in real time, whereas the high-availability objectives of supporting failover and avoiding a single point of failure require manual synchronization.

Data	**High-Capacity Clustering**	**High-Availability Clustering**
Account changes made in the Admin screen	Replicated	Synchronization required
Account runtime changes	Replicated	Synchronization required
Configuration changes	Replicated	Synchronization required
Session runtime changes	Replicated	Not available
Work, Event and Transfer Queues	Replicated	Synchronization required

Table 1 - Replication and Synchronization

5. Cluster composition and changes

The difference between replication and synchronization affects the flexibility of clustering with SecureTransport. Because of the manual step involved in taking servers on and off line, the composition of a cluster running SecureTransport is generally less dynamic than that of conventional clusters.

6. Partial Synchronization

The high-capacity model in SecureTransport 4.7 provides for full synchronization among nodes. The high-availability model provides for synchronization of user account and file routing information among machines in the cluster, but does not allow synchronization of in-flight file transfer work items in the work queue. So, if the Server in a high-availability configuration begins a job and goes down before completing it, the task will be lost because the work queues containing the entry for that task are not on any other machine. Once the cluster has resumed or restarted the file transfer itself, there is no way to recover other information about the work item (e.g., how and where to deliver it, next steps).

In the high-availability model, therefore, it is necessary to synchronize the user account and file routing information among the Servers, and to replicate the files in the user account home directory tree through the

storage network. (Other than the use of shared storage within a site, SecureTransport does not provide any built-in facilities for replicating account files between sites. This function is generally up to the storage network.)

8. Transfer log vulnerability

In the high-availability configuration, when the primary server recovers from an outage, all transfer log updates posted during the outage are lost. When the old primary server comes back up again and resumes its role as primary server for the cluster, SecureTransport does not preserve the transfer log information from the temporary primary server and restore it to the original primary server. By default, the secondary server overwrites the transfer log on the primary server when the cluster comes back up.

8. Monitoring Tools

SecureTransport does not include monitoring tools or hooks to monitoring tools for cluster status. Administrators cannot easily monitor, for instance, when a node has failed, when a server in a node has failed, that a secondary has assumed a primary role, or that an old primary has resumed its role as primary.[3]

To iterate these important differences between conventional clusters and SecureTransport's clustering model:

- SecureTransport does not support conventional, geographically distributed clustering.
- enterprises deploying SecureTransport in clusters should understand their exposure to data loss and take steps to limit it.

Limiting Exposure to Data Loss

Every cluster architecture has some areas of exposure. In clusters running SecureTransport, here are areas of potential data loss, with remedies for limiting that exposure:

Cluster synchronization

SecureTransport's Administration Tool limits some exposure by synchronizing the following types of configuration data from the primary server to secondary servers:

- All configuration files listed in the <FILEDRIVEHOME>/conf/sync.conf file
- All database tables listed in the <FILEDRIVEHOME>/conf/sync_tables.conf file

Note that the Administration Tool does not copy the files listed in the sync.excl from the primary to the secondary server.

Vulnerable files

End-user and account files

Exposure: SecureTransport does not protect or synchronize end-user or account files (i.e., the actual files to be transferred securely) in the cluster.

Remedy: Implement a shared storage architecture and store the files on a RAID or shared file system. SecureTransport appliances use a RAID controller for some protection and recovery purposes, but that forms part of the platform, not the SecureTransport application per se.

Transaction data

Exposure: These are server-initiated work items inferred from other data in SecureTransport, especially from queue entries. SecureTransport Server makes routing decisions based on transaction data such that, for example, it will handle a file queued for delivery to an AS2 partner in a specific manner.

[3] See Appendix for details on planning for additional servers when clustering with SecureTransport.

Remedy: SecureTransport protects against loss of transaction data in high-capacity configurations. In high-availability configurations, however, it is necessary to re-initiate the work items and resulting routing decisions, even at the risk of some redundancy.

Transaction logs

Exposure: Transaction Logs contain the history of all events, attempts and failures in SecureTransport. As noted above, these logs are subject to irretrievable loss during recovery from an outage.

Remedy: Implement regular backup jobs to capture transaction log files and copy them to another server. (Note also that if, after an outage, the recovering primary server starts **without** the SecureTransport service, it may be possible to recover the logs before the secondary – temporary primary – server has the opportunity to overwrite them.)

Event Queue

Exposure: SecureTransport's Persistent Event Queue service stores certain events and replicates them to all cluster servers, performs recovery operations when a machine from the cluster fails during the processing of an event, and spreads the execution of event-related operations to all servers in the cluster. SecureTransport relies on the event queue when resuming or restarting file transfers, so if the cluster should fail during its last attempt at transferring a file, that transfer will not take place automatically.

Remedy: Resubmit the failed file transfer manually. The File Tracking screen in the Admin UI displays all attempted, failed and in-progress transfers, so it is possible to identify transfers that did not take place automatically and then resubmit them manually.

Backup

While SecureTransport features live database backup to reduce some exposure to data loss, the ultimate remedy is robust system backup, administered regularly. Please see the *SecureTransport Best Practices Guide*, pp. 60-62, and the Tumbleweed Cummunications Technical Support paper, "Cluster Configuration and Setup" for more details on system backup.

SecureTransport Implementation Choices and Downtime

This section describes the effects of downtime (server disabled or taken offline for maintenance) on a cluster running SecureTransport. The downtime scenarios consider a SecureTransport Edge/Server combination as an individual node in the cluster, and they apply within a data center, not among conventional, geographically distributed data centers.

Downtime in a High-Availability Cluster

High availability involves an active primary node and a passive secondary node that is alive but not processing tasks. Failover from the disabled primary node to the secondary node entails a manual synchronization.

In this scenario, SecureTransport's focus is on replicating and making quickly available those files required to resume client-initiated transfers on the temporary primary SecureTransport Server, such as end user and account files. SecureTransport does not protect other files which would take longer to recover and restore.

If transactions are not in progress, end user and account files, including login and password information, would be available for recovery. Transaction logs would not be available to the secondary node (unless they had been backed up elsewhere) because they are all local to the disabled machine. SecureTransport features special algorithms for replicating data such as end user and account information, and it supports live database backup of the active machine, but for performance reasons, it does not allow for efficiently replicating all data necessary for disaster recovery, such as transaction logs, system logs and data files stored for transfer.

If client-initiated transactions are in progress, end-users are logging in and getting/putting files from/to SecureTransport. When the primary server goes down, existing sessions cease; SecureTransport does not support failover of active sessions. In a client-initiated scheduled (or batch) transaction, SecureTransport Client automatically reconnects to the server and retries. Once the temporary primary machine is online, the load balancer routes the file accordingly, so SecureTransport provides for receiving the tasks after simple reconnection/retry. In some interactive sessions, there may be an interruption in the file transfer, obliging the user to restart the session manually from SecureTransport Client (log on, navigate to folder, click Upload, click Send). Upon reconnection, SecureTransport will resume the transfer from the point of interruption.

If server-initiated transactions are in progress, the transactions are in a queue in the database of the primary machine and are subject to loss. SecureTransport does not provide for backing up queued transactions; however, a live database backup restored to the temporary primary server is the path to recovery in this scenario. Transaction schedules are part of the configuration data, so if the transactions are scheduled, and processing has not yet begun, the schedules are not lost, since the cluster replicates that configuration data to the secondary server.

Downtime in a High-Capacity Cluster

High capacity involves two or more nodes, all processing transactions to achieve high throughput. As shown in Table 1 above, the cluster replicates much more information in real time among nodes than in the high-availability configuration. Failover from the disabled primary node to the secondary node takes place automatically.

If transactions are not in progress, conditions are the same as in the high-availability scenario described above: end user and account files are available, but transaction logs are not available because they reside only on the primary SecureTransport Server.

If client-initiated transactions are in progress, sessions cease. Failover is automatic, so the temporary primary server will be available for reconnection almost immediately.

If server-initiated transactions are in progress, the tasks will not be lost. The load balancer will re-route inbound tasks to the failover server and SecureTransport will reassign outbound tasks within the cluster.

Conclusion

This paper has described conceptual differences between high-availability and high-capacity clusters, the advantages of each configuration, the implementation of SecureTransport in clusters, differences between traditional clusters and clusters running SecureTransport, and ways of minimizing exposure to data loss with SecureTransport configured in a cluster.

Related technical resources from Tumbleweed Communications include:

- *SecureTransport Administrators Guide*
- *SecureTransport Suite Best Practices Guide*
- Tumbleweed Technical Support paper "Cluster Configuration and Setup"

For More Information

Tumbleweed Communications invites you to contact your account manager or technical account manager for additional details on the technical topics contained in this paper.

Among the resources in the Tumbleweed Customer Portal are a searchable knowledgebase, FAQ section, Solution Finder, support forum, and archive of product bulletins. Tumbleweed also encourages you to subscribe to its technical bulletin/newsletter, available in the Customer Portal.

Appendix

Deployment Monitoring Applications and Tool

Administrators can monitor performance and trigger points for SecureTransport with the following procedure:

1. Within the SecureTransport transfer log, export the lines with "Transfer Time" entries into a spreadsheet.
2. Chart the transfer times over desired intervals (days, weeks, months, etc.). Take particular note of increases in transfer times.
3. Calculate the standard deviation for each transfer time.
4. When the standard deviation is within 20% of the maximum transfer time in 80% of transfers, the cluster is nearing peak capacity and additional servers will soon be necessary.