IBM Spectrum Protect
Node Replication

Disclaimer

IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM’s sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
Agenda

• Overview
• Preparing for Replication
• Performing a Replication
• Best Practices
• 7.1.3 Enhancements
• Future Enhancements
What is node replication

1. Initial replication – all objects are copied to the target server
   - Backup, Archive and Space Management Objects
2. Deleted objects are deleted from target server
3. Modified objects are updated on the target server
4. Newly stored objects are copied during next replication

Node replication for disaster recovery

- Spectrum Protect server replicates data to another server
- Ensures completeness and consistency for a Node or File Space
- Native solution – no dependencies on specific storage devices
- Incremental client data transfer with deduplication
- Remote server could be hot standby for improved RTO
Node replication for branch office

Node replication advantages

1. Different Operating Systems on source and target
2. Storage Pool layout can be different on each server
3. Meta-data and data always kept in sync
4. Data can be replicated from any storage pool
5. Objects without data are also replicated
Characteristics

- Highly parallel
  - Can run multiple replications at once
  - Examination of objects in a file space overlaps with batches being sent to target server
- Highly resilient
  - Multiple levels of retry to achieve success
  - Will attempt to get data from any storage pool
- Deduplication aware if deduplicated data is stored to a deduplication-enabled target pool

TSM 7.1 Automatic client redirection

Pre-failure
1. Server A replicates node data and metadata to target server B
2. Server A sends failover server connection information to client

Failover
3. Server A becomes unavailable
4. Based on failover server connection information and policy, client is automatically redirected to Server B for restore/retrieve operations

Failback
5. When server A becomes available, client operations are directed back to server A
TSM 7.1.1 Recovery of damaged files from target

Scenario
1. Server A replicates data to Server B
2. Storage pool volume on Server A is destroyed or data is damaged
3. Inaccessible data is replaced on Server A using replicated data from Server B

- Replication protects against localized failures and large scale disasters
- Potential replacement for local copy storage pool tapes

Recovery of damaged files

- New replicate node parameter RECOVERDAMAGED=Yes|No|Only
- After normal replication process ends, source server checks whether any of the nodes just replicated have files that are:
  - Marked as damaged
  - Stored on a volume that is marked as destroyed
- If the source server finds any damaged files, it notifies the target server to begin a recovery process
- Target server sends files corresponding to those damaged physical files or destroyed volumes
- Source server replaces damaged data with data recovered from the target server
TSM 7.1.1 Dissimilar policies

- Different retention policies can be defined on source and target servers
- Expiration runs independently on each server
  - Objects deleted based on policies in effect for that server

Use case: Branch office with limited storage

Use case: Return only most recent versions at remote site to conserve storage

Flexible management of replicated data at different sites
Optimal use of resources

Dissimilar policies

- Policy/expiration behavior with like policies (default)
  - Retention and versioning of replicated data is determined by policy on the source server
  - Expiration is not performed for replicated data on the target server

- Policy/expiration behavior with dissimilar policies
  - Versions and retention of replicated files on target server will be managed by policy on the target server, not the source server

- How it will work
  - Source server will send deletion requests only for files that have been explicitly deleted by the client or administrator
  - Files marked for immediate deletion on the source server will not be deleted until they have been replicated
Reconsile processing

- Prior to TSM 7.1.1, replication has always done a reconcile
  - Compares complete list of files between the source and target server
  - Used to synchronize the source and target servers
- Reconcile in TSM 7.1.1: Examines entire list of files in a file space (much like pre 7.1.1)
  - Used during the initial replication between 7.1.1 servers
    - Once reconcile completes, change tracking processing takes over during the next replication
    - Restartable – remembers where it left off if cancelled or after some catastrophic server event
    - Automatically runs following a database restore on the source or target server
    - Can run manually using `REPLICATE NODE FORCERECONCILE=NO=YES`
      - Synchronize source/target files - used like an audit
- Change Tracking in TSM 7.1.1 eliminates need to query target server for its list of files
  - New and change files are assigned a change identifier, when it is stored and when metadata is updated
  - Replication only processes files with a change identifier – incremental replication
  - Replication picks up where the last replication left off
  - Improves performance for fs with lots of files
- Showed a 2-3x improvement. ~200 GB/hr → ~500 GB/hr
File deletion processing

• Processing of files deleted on the source server (prior to 7.1.1)
  • With current implementation, files that have been deleted on the source server are deleted on the target server during replication
  • Locking issues can cause delays, especially for deduplicated files
• Processing of files deleted on the source server (7.1.1)
  • During replication, source server sends list of files that have been deleted on source (does not include expired files if dissimilar policies enabled)
  • During replication, target server updates its database to indicate deleted files
  • Expiration processing deletes indicated files on the target, outside the replication window
• Processing of files deleted on the target server (7.1.1)
  • During replication, target server sends list of files that have been explicitly deleted on the target
  • During replication, source server resends explicitly deleted files to the target

Agenda

• Overview
  ➢ Preparing for Replication
    • Performing a Replication
    • Best Practices
    • 7.1.3 Enhancements
    • Future Enhancements
Hardware requirements

- CPU/RAM minimum recommendations
  - With deduplication, 8 CPU cores, 64GB RAM
    - Best practice: 8 CPU cores, 128GB RAM
  - Without deduplication, 4 CPU cores, 32GB RAM
    - Best practice: 4 CPU cores, 64GB RAM
  - Best practices assume complete server replication
    - Requirements less if replicating less

Log and DB requirements

- Active Log
  - At least 64GB Active Log
  - Reconcile changes in TSM 7.1.1 greatly reduced log requirements
- Database
  - A 300GB DB on a source server will require an additional 300GB of DB on the target server
    - In addition to current size of target DB
    - Plan DB size and growth appropriately
Tasks

- Create/Verify the server definitions
- Set the replication target for the server
- Determine which nodes, file spaces, and data types are to be replicated
  - Assign appropriate rules, or use defaults
  - Enable replication for the nodes
- Determine whether dissimilar policies will be used
- Replicate

Populate target server

- Two basic methods to populate target server
  - Method 1 – Replicate from scratch
    - Best if source and target are in close proximity
    - All eligible data is sent
    - Could take a long time
  - Method 2 – Synchronize and Replicate
    - Best for large distances or if bandwidth is limited
    - Use media-based Export/Import to populate target
    - Replication with SYNC links the source and target objects
Replication terms

Mode
- The replication mode indicates the role for a node (source/target)
- Normal modes
  - SEND – the node is the source of replication
  - RECEIVE – the node is the target of replication
    - Cannot be set directly
- SYNC modes
  - SYNCSEND – the node is a synced source
  - SYNCRECEIVE – the node is a synced target

State
- The replication state indicates whether replication is enabled
  - Used to temporarily stop replicating

Policies
- Replication does not replicate the policy
  - Use EXPORT/IMPORT or Enterprise Configuration
- If using like policies, you should ensure the policies on each server are the same, important for the case the replication is disabled
  - If a policy construct is missing on the target server, the default construct is used
- If using dissimilar policies, you must
  - Validate the policies with the command VALIDATE REPLPOLICY
  - Enable the function with the command SET DISSIMILARPOLICIES
Restrictions on target server

- A replicated node is Read-Only
  - Cannot store new data from a client or application
  - Cannot rename the node
- Data can be deleted from target with:
  
  ```
  DELETE VOL DISCARDD=YES
  AUDIT VOL FIX=YES
  DELETE FILESPACE
  ```
- Data deleted from target will be sent during next replication

Removing replication

- REMOVE REPLNODE <nodename>
  - Deletes all replication information from the DB
  - Can be run on source, target, or both
  - Sets the REPLSTATE and REPLMODE to NONE
  - Does not delete any data
Planning

- Need to plan for the daily change rate
  - Are your RAM, CPU, and disks sufficient?
- How much data needs to be initially replicated to get to the steady state?
  - Do you have the time and bandwidth to replicate it from scratch?
  - Would it be better to use Export/Import?

Agenda

- Overview
- Preparing for Replication
  - Performing a Replication
- Best Practices
- 7.1.3 Enhancements
- Future Enhancements
Performing a replication

- The REPLICATE NODE command accepts:
  - Multiple nodes and/or node groups
  - Specific file spaces belonging to a node
  - The data type to replicate
  - The priorities to include in the replication
- REPLICATE NODE starts a single process
  - Process ends when ALL nodes and file spaces are complete
  - Can be scheduled as part of daily maintenance

Replication processing

- Each node and file space specified is examined
- Source and target exchange information
- For each node being processed
  - Target node is registered, if necessary
  - Target file spaces are created, if necessary
  - Replication State and Mode are verified
    - Verify the node and/or file space is enabled for replication
    - Verify source server is in SEND mode, and the target is in RECEIVE mode
  - Target node is synchronized to source
    - Attributes, including passwords, are replicated
Agenda

• Overview
• Preparing for Replication
• Performing a Replication

➤ Best Practices
• 7.1.3 Enhancements
• Future Enhancements

Best practices (maintenance plan)

• If not using container pools (7.1.3)
  • Allow sufficient time for IDENTIFY to process all data before replicating
    • Allows replication to benefit from deduplication
  • Replicate the nodes
    • If migrating data to tape, wait for replication to finish before migrating
  • Expire the inventory
  • If migrating from disk to tape with autocopy, migrate the storage pools
  • Back up the storage pools
  • Reclaim the storage pools
Best practices

- Be sure to test replication throughput
- Adjust MAXSESSIONS
  - Network, CPU, and RAM will impact throughput
- Make sure sufficient mount points are available for replication
  - For FILE device class, set mount limit to at least the product of NUMOPENVOLSALLOWED and MAXSESSIONS
- Don’t run all nodes in a single replication
  - Replicate nodes with large number of objects by themselves
    - With a smaller value for MAXSESSIONS (1 - 3)

More sessions doesn’t necessarily mean better performance
- It usually does, but lock contention on the target server can slow it down
- Replication batches have 2 phases:
  - Phase 1: Sending the data to the target server for the new objects
  - Phase 2: Updating the database with the new objects
- File space locks on the target server occur in phase 2
  - When the amount of time spent in phase 1 is large relative to the time spent in phase 2, use more sessions
  - When the amount of time spent in phase 1 is relatively small, use fewer sessions
- Increased distances will add time to phase 1 without affecting phase 2
  - Generally speaking, this will benefit from having more sessions
Best practices

- Replicating from tape can be slow
  - The server will allow only 1 session to read from a tape at a time
- If replicating large amounts of data from tape, turn on collocation by NODE
  - Avoid collocation NONE or GROUP
  - If each node has its own set of tapes, contention among nodes is reduced

<table>
<thead>
<tr>
<th>Bad for replication</th>
<th>Good for replication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume X00011L5</td>
<td>Volume X00011L5</td>
</tr>
<tr>
<td>Node1  Node2  Node1</td>
<td>Node1 Node1</td>
</tr>
</tbody>
</table>

Agenda

- Overview
- Preparing for Replication
- Performing a Replication
- Best Practices
- 7.1.3 Enhancements
- Future Enhancements
7.1.3 Container pool overview

- **Container storagepools**
  - Storage is handled in an automatic fashion such that no direct management of the storage is required
    - Philosophy - write once and don’t fuss with it
    - NO reclamation, migration, copies, backups, shredding, LAN-free, ….
    - NO device classes or volumes like legacy random or sequential access storagepools
  - Dynamic creation & deletion

- **Next generation deduplication (NextGen dedupe) uses container storagepools**
  - Allows for the deduplication of all data from both the client and server
    - With the exception of files that have been encrypted on the client side
  - Deduplication occurs in-line
    - As opposed to legacy TSM server dedupe being done in a separate process following ingest

- **Goals of container and NextGen dedupe are to provide**
  - Easier management
  - Faster performance
  - More scalability

Container pools attributes

- Primary pools only
- Many attributes of device class storage pools are disabled:
  - Migration
  - Copy storage pools
  - Collocation
  - Reclamation
  - CRC checking
    - Replaced by individual extent integrity checks
- The “Nextpool” is used for initial space allocation only
  - if the storage pool is full, or
  - if the MAXSIZE limit is exceeded
Directory pool compared to File pool

- At the directory level, they are deployed in a similar fashion
- Storage Pool Directories are managed individually
  - Access Control (ReadOnly, ReadWrite, Unavailable, Destroyed)

Protecting data in container pools

- Two ways to protect data
  - Node Replication
  - Storage Pool Protection (new)
Node replication for directory based container pools

- Node Replication works as before
  - Can replicate to any container pool (Directory or Cloud)
  - Uses inline deduplication where applicable

Storage pool protection (new function)

- Directory storage pools are protected with a new command
  - PROTECT STGPOOL

- Based on replication technology but different than node replication
  - Target server is the same server as used for node replication
    - SET REPLSERVER
  - Target storage pool is an attribute of the source storage pool
    - Must be a Container Directory pool

- All extents in the storage pool are protected
  - Not necessarily all objects in the server
    - Some objects (like directories) may have no data stored in a pool
Protect STGPOOL for directory based container pools

- Spectrum Protect source server replicates chunks from a directory-based container pool to a directory-based container pool on target server
- No replication of inventory metadata regarding nodes, file spaces, or the objects which reference chunks
- Chunk-level repair of damaged data in the source pool using chunks in the target pool, without needing to transfer the entire object
- Protect storage pool complements node replication by seeding chunks on the target server
- Protect storage pool exhibits excellent performance and improves the throughput of subsequent node replication operations

Repair storage pool

- During restore or retrieve operations, if an extent cannot be read or is corrupt, the extent is immediately marked “Damaged”
  - All logical files using that extent are also marked “Damaged”
- Repair Stgpool is used to fix damaged chunks
  - Uses the data stored by Protect Stgpool
- Subsequent backup of same extent will also fix a damaged extent
## Comparison of replication and protect stgpool

<table>
<thead>
<tr>
<th>Replication approach</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
</table>
| Node replication     | • Data and metadata  
 |                      | • Restore directly from target  
 |                      | • Can replicate a subset of data  
 |                      | • Supports dissimilar policies  | • No extent level repair (repair stgpool) |
| Protect stgpool      | • Extent level replication can be faster for certain workloads  
 |                      | • Support repair stgpool  | • Must replicate all data  
 |                      |                           | • No dissimilar policies  
 |                      |                           | • No restores directly from target server  
 |                      |                           | • DR requires database restore and reverse replication? |
| Node replication + Protect stgpool | • Supports restore directly from the target  
 |                      | • Supports repair stgpool  | • Must replicate all data  
 |                      | • All recovery options with only sending the data once  | • Cannot use dissimilar policies  
 |                      |                           | • Total elapsed time for replication can be longer |

## Agenda

- Overview
- Preparing for Replication
- Performing a Replication
- Best Practices
- 7.1.3 Enhancements
  - Future Enhancements
2016: Unified replication

- Simplified administration of replication

- Node replication in 7.1.3
  - Complete replication of data and metadata for node/file space
  - No dependence on storage pool
  - Client re-direction
  - Recovery of damaged data in source device-class pools
  - Discriminates policies

- Storage pool protection in 7.1.3
  - Replication of chunks in directory-based container pool
  - No replication of inventory metadata
  - Chunk-level repair of damaged data in source container pool
  - Highly efficient

Unified replication
- One replication method
- Benefits of node replication and storage pool protection

2016: Metadata-only node replication

- Reduces data transferred by Spectrum Protect replication
- Facilitates use of replication by cloud storage or hardware

With replication of data:
- Server A replicates metadata, including storage locations, to Server B
- Server B can access data at indicated storage location
- Requires that data be accessible to both servers via replication or sharing

If data is replicated, administrator would need to ensure consistency with replicated metadata

© 2015 IBM Corporation
Future: Node replication with automatic failover/failback

Pre-failure
1. Server A replicates node data and metadata to target server B.
2. Server A sends failover server connection information to client, which stores information locally.

Failover
3. Server A becomes unavailable.
4. Based on failover server connection information and policy, client is automatically redirected to Server B to retrieve and (optionally) store operations.

Failback
5. When server A becomes available, server B replicates any new data back to server A.
6. Client operations are directed back to server A.

Future: Always-on node replication

Scenario
1. Client sends files to Server A.
2. Newly ingested files (or chunks) are immediately queued for replication to Server B.
3. Client sends additional files to Server A.
4. Additional files (or chunks) are immediately queued for replication to Server B.

Always-on replication might be for:
- Chunks and inventory metadata
- Chunks only
Future: Node replication to multiple target servers

- Greater replication flexibility
- Replication to multiple target sites for additional disaster protection

Capabilities might include:
- Replication of same node to multiple servers
- Replication of different nodes to different servers
- Cascaded replication (Server A → Server B → Server C)
- Replication of chunks without inventory metadata to specified target

Future: Node replication with transparent client access

- Client accesses data on secondary server if unavailable on primary

Scenario:
1. Server A replicates data for client node X to Server B
2. Due to media failure on Server A, file f becomes inaccessible and is marked damaged
3. Client node X attempts to restore file f from Server A
4. Client request is automatically redirected to Server B
5. File f is automatically restored from Server B to client node X
Questions?

Thank You