Teradata® Data Mover Best Practices Guide

General Implementation and Configuration

Release Date: June 2022

Product ID: B035-6500-000K

Copyright Reference

Product Safety

Safety type	Description
Notice:	Indicates a situation which, if not avoided, could result in damage to property, such as to equipment or data, but not related to personal injury.
Caution:	Indicates a hazardous situation which, if not avoided, could result in minor or moderate personal injury.
Warning:	Indicates a hazardous situation which, if not avoided, could result in death or serious personal injury.

Warranty Disclaimer

Except as may be provided in a separate written agreement with Teradata or required by applicable law, the information available from the Teradata Documentation website or contained in Teradata information products is provided on an "as-is" basis, without warranty of any kind, either express or implied, including the implied warranties of merchantability, fitness for a particular purpose, or noninfringement.

The information available from the Teradata Documentation website or contained in Teradata information products may contain references or cross-references to features, functions, products, or services that are not announced or available in your country. Such references do not imply that Teradata Corporation intends to announce such features, functions, products, or services in your country. Please consult your local Teradata Corporation representative for those features, functions, products, or services available in your country.

The information available from the Teradata Documentation website or contained in Teradata information products may be changed or updated by Teradata at any time without notice. Teradata may also make changes in the products or services described in this information at any time without notice.

Feedback

To maintain the quality of our products and services, e-mail your comments on the accuracy, clarity, organization, and value of this document to: docs@teradata.com.

Any comments or materials (collectively referred to as "Feedback") sent to Teradata Corporation will be deemed nonconfidential. Without any payment or other obligation of any kind and without any restriction of any kind, Teradata and its affiliates are hereby free to (1) reproduce, distribute, provide access to, publish, transmit, publicly display, publicly perform, and create derivative works of, the Feedback, (2) use any ideas, concepts, know-how, and techniques contained in such Feedback for any purpose whatsoever, including developing, manufacturing, and marketing products and services incorporating the Feedback, and (3) authorize others to do any or all of the above.

Revision History

Revision/Version*	Date	Comments
1.0	March 2020	Initial Release
1.1	October 2020	Documentation maintenance
1.2	October 2021	Documentation maintenance
1.3	January 2022	Documentation maintenance
1.4	June 2022	• Added the User Orientation topics to the *Introduction* chapter. • Added the recommended data stream values for copying tables and databases of different sizes using DSA to the *Running Jobs* chapter in the *Streams/Sessions* topic.

Table of Contents

1 Introduction

1.1 Using Teradata® Data Mover Best Practices Guide

This guide recommends the best implementation solutions to plan out your Teradata Data Mover solution, from high level architecture down to fine tuning for performance.

1.2 Why Would I Use this Content?

If you are a new user or a site team, refer to this document where we have captured the advice and know-how from the previous users so that you and site teams will have that information from the get-go.

1.3 How Do I Use this Content?

These recommendations are based on lessons learned from field implementations of Teradata Data Mover at over 200 customer sites. Use this content to be aware of the best practices used while:

- Planning high-level architecture
- Configuring and tuning Teradata Data Mover
- Managing Workload

1.4 How Do I Get Started?

This document collects into one place all the best practice recommendations for Teradata Data Mover solutions. The contents of this guide are divided into three sections:

Section 1 covers high level planning and architecture. We want to first consider if Teradata Data Mover is the right product to meet the needs of your use case. Assuming Teradata Data Mover is the right solution, we next walk through all the architecture considerations. Here we will look not just at Teradata Data Mover but at your whole Teradata ecosystem. This includes example architectures for reference.

Section 2 covers best practices for configuration and tuning of Teradata Data Mover. This section is complimentary to the information already provided in the *Teradata Data Mover Install, Configuration, and Upgrade* guide and in the *Teradata Data Mover User Guide*. Those documents already cover how to setup and configure Teradata Data Mover. Here we will cover best practices for network performance, incorporating and updating information that was previously provided in a separate Data Mover Networking Best Practices guide.

Section 3 covers best practices for workload management. After you have Teradata Data Mover set up, how to best create and execute jobs to not only meet your initial goals but to also provide a usage model that can grow to take on bigger workloads in the future. If you are looking to implement a DR solution with Teradata Data Mover, this section also discusses the challenges and techniques used to keep data synchronized.

Section 1: Use Cases and Architecture

2 Scoping Your Project

In order to determine if Teradata Data Mover will meet your needs and how best to configure, it is best practice to first consider what use cases you have, what systems you plan to copy data between, and the type and amount of data you need to copy. Going through this process will help answer questions in later sections of this best practice guide.

2.1 Scoping Use Cases

Teradata Data Mover can be used as part of the solution for the below use cases. Consider your use cases and see if those cases fall into one or more of these categories:

- **Seeding new system.** Data needs to be copied from an existing Teradata SQL Engine to a new Teradata SQL Engine. Teradata Data Mover does not support copying Teradata users, roles, permissions, and database definitions. So, an outside process needs to be used to first copy those objects. Teradata Data Mover can then be used to copy over the data and objects within databases including tables.
- **Disaster recovery.** Teradata Data Mover can be used to keep data between a primary Teradata SQL Engine and a secondary Teradata SQL Engine synchronized. Teradata Data Mover does not support copying Teradata users, roles, permissions, and database definitions. An outside process needs to be used to first copy those objects. Teradata Data Mover does not provide a change data capture mechanism. An outside process must also be used to keep track of what changes occur on the primary Teradata SQL Engine. Teradata Data Mover can be used to copy over changes in data from primary to secondary systems either by copying the whole object or by providing a WHERE clause defining the rows that have changed. See section 3 for techniques on copying delta.
- **Ad-hoc copying of data between Teradata SQL Engines**. Teradata Data Mover can be used to copy data between Teradata SQL Engines. This could be for a multitude of use cases including copying data from production to development for testing purposes.
- **Copying data between Teradata SQL Engine and Teradata Aster**. Teradata Data Mover can be used to copy tables between Teradata SQL Engine and Teradata Aster.

If your use case was covered in-part or in-full by one of the above scenarios, then Teradata Data Mover is a tool that could be part of the solution you are looking for. Cases that specifically would not be met by Teradata Data Mover are listed below:

- **Active-Active in real time.** Teradata Data Mover can only copy data that has already landed on a Teradata SQL Engine or a Hadoop or Teradata Aster system. If your use case requires data be synchronized between two or more Teradata SQL Engines in real time, consider Teradata Unity for your solution.
- **Copying data between Teradata SQL Engine and Hadoop**. Teradata Data Mover can be used to copy data between Teradata SQL Engine and Hadoop. However, Teradata Data Mover support here is being capped and no support is provided for newer versions of Hadoop. As such, unless you are using older versions of Hadoop that Teradata Data Mover supports (ex. HDP 2.x) and have no plan on using newer Hadoop versions (ex HDP 3.x), it is recommended to use Teradata QueryGrid for your solution.
- **Copying data between Hadoop systems**. Teradata Data Mover requires that Teradata SQL Engine is at least either the source or the target when copying data. Teradata Data Mover cannot be used to copy data between Hadoop systems.
- **Copying data between Teradata Aster systems**. Teradata Data Mover requires that Teradata SQL Engine is at least either the source or the target when copying data. Teradata Data Mover cannot be used to copy data between Teradata Aster systems.
- **Archiving data**. Teradata Data Mover cannot be used to archive data or to land data to any type of storage outside of the Teradata SQL Engine, Hadoop, or Teradata Aster. Consider Teradata BAR solutions instead.

2.2 Scoping Source and Targets

The type of systems you need to copy data between and the location of these systems is essential to determining which Teradata Data Mover architecture will best suite your needs. As the first step in planning your solution, do an inventory of the Teradata systems that you plan to copy data between. For each system, consider the following:

- What type of system. Teradata SQL-E, Hadoop, Teradata Aster.
- What version
- Where the system is located
- Will this system be a source system (i.e. data will be copied from this system to other systems)? If so, what are all of the target systems that will be receiving data from this source. Gather the information in this check list for each of the target systems.
- Will this system be a target system (i.e. data will be copied from another system to this system)? If so, what are all of the source systems that will be sending data to this target. Gather the information in this check list for each of the source systems.

2.3 Scoping Workload

Now that you have a list of the systems that you need to copy data between, the next step is to create an inventory of the data you need to copy. For each source/target pair in your list, create an inventory that includes the following:

- Type of objects: Tables, databases, statistics, indices, stored procedures, macros. What needs to be copied. Just data or also schema or statistics?
- Size of tables. For tables, how much data is stored and needs to be copied. For data sizing, consider using the following classifications:
  - Small - under 100 GB
  - Medium - 100 GB to 200 GB
  - Large - 200 GB to 500 GB
  - Extra Large - over 500 GB
- Number of objects in databases being copied. Also consider size of tables that are in the database.
- How often does data change and to what degree: How frequent do ETL jobs update the object. For tables, what is the rate of growth. For tables, does the table have unique keys that can be used to determine just delta?
- How often do you require changes to copied over: Hourly, daily, weekly?
- How critical is it to your business to copy the data over for each object: Ranking the data will help determine workload priority as well as which workloads should be implemented first?

3 Architecture Best Practices

There is no single standard architecture when it comes to Teradata Data Mover. Data Mover has many deployment and configuration options to meet a variety of customer needs. To help you choose the best architecture, this section will first walk you through a list of questions that will drive your architecture choices. Afterwards, we'll walk through a series of example architectures to show what Data Mover might look like in action.

3.1 What Copy Method to Use

Teradata Data Mover supports the following copy methods:

- DSA
- Teradata QueryGrid
- ARC
- TPT
- JDBC

When Teradata Data Mover was first introduced, the choice of which copy method Data Mover used had no impact on the architecture. All legacy copy methods (ARC, TPT, JDBC) function in the same manner with the underlying utility running on the DM Agent server and the data being copied traveling from the source system to the DM Agent server to the target system. In recent years however, modern copy methods have been added (DSA, QueryGrid) that rely on external components and copy data directly from source nodes to target nodes. This difference means that you should first consider which copy method you plan to use with Data Mover when planning your architecture.

3.1.1 Copy Method Recommendations

Choose Modern over Legacy

We recommend choosing one of the two modern copy methods (DSA, QueryGrid) to be your primary copy method. ARC will not support Teradata SQL-E 17.00 or later. Data Mover itself will drop support for using ARC as a copy method for older Teradata SQL-E versions in Data Mover 17.00. Customers currently using ARC with Data Mover will need to migrate to using DSA or another copy method in Data Mover 17.00.

Choose DSA if You Need to Copy Databases

DSA provides the ability to copy at the database level rather than the object level. Data Mover with Teradata QueryGrid does not currently provide the same ability. ARC can also copy at the database level if you are unable to use DSA.

DSA is Challenging to Use for Older Teradata SQL-E Versions

It can be difficult to implement Teradata Data Mover with DSA for Teradata SQL-E versions older than 16.00. Prior to version 16.00, Teradata SQL-E only supports a single DSC. DSC is the primary processing engine of DSA and DSU. If you are not currently using DSA or DSU for your BAR solution, then this not a problem as Teradata Data Mover comes with its own DSC and that can be the single DSC registered to both your source and target Teradata SQL-E. The problem is that most customers are also using DSA or DSU for BAR in which case each Teradata SQL-E system may already have its own DSA/DSU. In order to use Data Mover with DSA, the architecture would need to be reconfigured to use a single DSA/DSU for both the source and target and have Data Mover share that same DSC. This is something that most customers are unwilling to do given that it would require using a single DSA/DSU to handle the BAR workload for both Teradata SQL-E systems.

In such cases, a solution utilizing either Teradata QueryGrid or ARC/legacy is recommended. If you choose ARC, you should still take time to consider how the architecture will evolve to using DSA or QueryGrid once you upgrade to a newer version of Teradata SQL-E.

3.1.2 Planning Based on your Copy Method Choice

DSA

You will need to plan for the following if you choose to use Data Mover with DSA:

- Additional install/configuration.
  - DSA ClientHandler must be installed on the target Teradata SQL-E nodes. Best practice is to also install on source Teradata SQL-E nodes.
  - Network mask or fabric must be defined to tell DSA how to send data between source and target.
- Networking impact.
  - Data copied by DSA travels directly from source to target Teradata SQL-E nodes.
  - Teradata Data Mover host needs to be able to connect to source and target Teradata SQL-E systems. This is to query metadata and create objects only. Data will not be copied along this path.
- Potential impact to existing DSA/DSU.
  - May need to configure Data Mover to share an external DSC if copying between Teradata SQL-E 15.10 or older. If so, Teradata Data Mover needs to be able to connect to the external DSC host and the external DSC host needs to be able to connect to the Data Mover server.

Teradata QueryGrid

You will need to plan for the following if you choose to use Data Mover with Teradata QueryGrid:

- Additional install/configuration.
  - Teradata QueryGrid needs to be setup in the environment.
  - Need to configure Data Mover to work with Teradata QueryGrid Manager
  - Need to setup QueryGrid Teradata Connectors for source and target Teradata SQL-E systems
  - Need to create foreign server on source or target Teradata SQL-E system to use Teradata Connectors to link source and target systems
- Impact to Data Mover job creation.
  - Need to provide name of foreign server when creating Data Mover jobs.
- Network Impact.
  - Data copied by Teradata QueryGrid travels directly from source to target Teradata SQL-E nodes.
  - Teradata Data Mover host needs to be able to connect to source and target systems. This is to query metadata and create objects only. Data will not be copied along this path.
  - Teradata Data Mover host needs to be able to connect to the Teradata QueryGrid Manager host.

Legacy Copy Methods (ARC, TPT, JDBC)

You will need to plan for the following if you choose to use Data Mover with legacy copy methods (ARC, TPT, JDBC):

- Network Impact
  - Data Mover host(s) need to connect to both source and target Teradata SQL-E systems.
  - Data copied by legacy copy methods travels from source Teradata SQL-E nodes to Data Mover DM Agent host to target Teradata SQL-E nodes.

3.2 Where to Deploy Teradata Data Mover

Teradata Data Mover can be deployed in a variety ways on-premises or in the cloud. Technically, it does not matter where Teradata Data Mover is deployed so long as it has network connectivity to the source and target systems and to other Teradata applications like Teradata Viewpoint. However, it is best to locate Data Mover close to the systems it is involved with.

3.2.1 Deployment Recommendations

Deploy Data Mover in Same Environment as Source or Target

When using legacy copy methods, the data travels through the Data Mover host. As such, it is best practice to deploy Data Mover in the same environment as the source or target system. Source and target systems are often located in different geographic environments. Deploying Data Mover in either the source or target environment allows half of the data path to use LAN or Infiniband fabric. The target environment is generally the preferred location for Data Mover as the target side of the data transfer has more work to do.

Even if you choose to use a modern copy method such as DSA or QueryGrid, it is still best practice to deploy Data Mover in the same environment as the source or target system. While data is not traveling through the Data Mover host in these cases, Data Mover still requires network connectivity to the source and target and those connections will be more reliable if Data Mover is close by.

When choosing between the source and target environment, also consider which external Teradata applications that Data Mover will be interacting with. It is best if Data Mover can be co-located with the Teradata applications that it works with to increase the reliability of the network connection to those application hosts. Depending on the configuration, Data Mover interacts with the following applications:

- Teradata Viewpoint (required for using Data Mover portlet and Data Mover security permission features)
- Teradata Ecosystem Manager
- External DSC (If scenario prevents use of embedded DSC on Data Mover host)
- QueryGrid Manager (if using Data Mover with QueryGrid)

Deployment Platform Pros/Cons

On-Premises Deployments: Dedicated TMS

Pros	• Dedicated network cards and CPU/Memory good for legacy copy methods • Can co-locate with target or source system to allow for use of LAN for one half of data transfer when using legacy
Cons	• Requires rack space

On-premises Deployments: Consolidated TMS

Pros	• Can share single server with Teradata Viewpoint and/or Teradata Ecosystem Manager reducing rack space requirements • Can co-locate with target or source system to allow for use of LAN for one half of data transfer when using legacy

On-premises Deployments: VM on Customer VMWare

Pros	• No additional rack space required. • Flexible deployment • Can co-locate with target or source system to allow for use of LAN for one half of data transfer when using legacy
Cons	• Sharing network interface could introduce performance bottleneck if using legacy copy methods

Cloud Deployments

Pros	• No need for on-premises rack space • Flexible deployment
Cons	• Network can be bottleneck if using legacy copy methods and transferring data in or out of the cloud.

3.3 Assessing High Availability Needs

How long of a downtime in Teradata Data Mover services are you able to accept?

3.3.1 High Availability Recommendations

At the very least, Do Regular Backups

If you don't need to ensure that Teradata Data Mover is always available, but you want to protect your workloads to failure, best practice is to perform regular backups of the Teradata Data Mover repository. Refer to the Teradata Data Mover User Guide for instructions on performing a backup. Backups are initially stored on local storage on the Teradata Data Mover host. These backups should then be transferred to external storage to avoid loss if the Teradata Data Mover host goes down. In the event of a failure, these backups can be used to setup a new Teradata Data Mover server or restore the original server once it is back online.

Create a Data Mover High Availability Cluster if you always need Data Mover Services

If you need to ensure that Teradata Data Mover is always available, the best practice is to setup a high availability cluster. This requires at least two Teradata Data Mover servers, one for active and one for standby.

on-premises deployments do not need to be paired with the same type of on-premises deployment. You can pair two TMS' or you can pair a TMS with a CTMS or a TMS with a VM. The best practice is to make sure the active and standby servers are separated as much as possible to prevent a single failure from bringing down both servers.

The additional standby server can be utilized as an additional agent so that its resources can be used actively when not being used as the active server. Consider however if the active and standby server are located in different labs or regions. Performance may vary when using the agent on the active server versus the agent on the standby server.

The Teradata Data Mover high availability configuration includes a failover monitor process. It is best practice to locate the active and standby failover monitor processes on two Viewpoint servers. This helps protect the failover monitor from failures that occur to the Teradata Data Mover servers.

3.3.2 Planning Based on Your High Availability Choice

If You Choose to Do Regular Backups

- External storage requirement.
  - Need to identify a storage area outside of the Data Mover host where backups can be stored
- Impact to Data Mover jobs
  - Best to do backups when no Data Mover jobs are running.

If You Choose to Have Full High Availability

- Additional host requirement.
  - Need at least two Data Mover hosts, one for active, one for standby.
  - Need two hosts external from the Data Mover hosts where active and standby DM Failover processes can run. Recommend running on Teradata Viewpoint hosts.

3.4 Scaling Strategy

A single Data Mover instance is capable of handling multiple jobs in parallel and can copy data between multiple pairs of sources and targets. That being said, your workload may grow to the point where you will need to consider how to scale to keep up. In section III, we will discuss methods to tune Data Mover and your jobs to better handle larger workloads. Here we will discuss architectural modifications that can be made.

3.4.1 Scaling Recommendations

Add Additional DM Agents if using Legacy Copy Methods

Adding additional Teradata Data Mover servers to act as additional agents can increase Teradata Data Mover's workload capacity in the following cases:

- When using legacy copy methods. Additional server provides more CPU, memory, and network bandwidth for running legacy copy methods.

Agents Not as Important for Modern Copy Method Performance

For modern copy methods, additional agents do not necessarily increase workload capacity. Since data is not traveling through Teradata Data Mover servers, modern copy method tasks use little local resources when running. If only using modern copy methods, best practice is to modify existing agents to increase task limit rather than adding additional agents.

Consider Independent Teradata Data Mover Environments

In certain cases, it may be beneficial to have separate independent Teradata Data Mover instances to process different workloads. These cases include:

- Scenarios where you have multiple source and targets, some of which are located in the cloud and some on-premises. If copying data between source and targets in the cloud, it may make sense to have a Teradata Data Mover deployment in the cloud to use for that purpose and then a separate on-premises Teradata Data Mover deployment to handle workloads going between on-premises source and targets and those going from on-premises to cloud.
- High workload environments where a single Teradata Data Mover instance is becoming a bottleneck. The Teradata Data Mover Daemon will eventually become a limiting factor when the number of jobs run per hour becomes high enough. The limit is dependent on the size of the jobs being run and how quickly they finish. If the Teradata Data Mover Daemon is the limit and not the source or target system, then separating workloads between two or more different Teradata Data Mover instances can increase workload capacity.

There are challenges to operating independent Teradata Data Mover instances. Consider the following:

- Each Teradata Data Mover instance is unaware of what the other instance is doing. There is no built-in coordination of resource usage.
- Each Teradata Data Mover instance needs its own set of Teradata users with proper permissions. This will help avoid contention between instances.
- Cannot easily use the same Teradata Data Mover Command Line instance to communicate with multiple independent Teradata Data Mover servers. Best to have separate command line instances.
- Best practice is to ensure the workloads processed on each Teradata Data Mover instance is completely independent to avoid contention.
- Best to not use two independent Data Mover instances against the same source and target.

3.4.2 Planning Based on Your Scaling Choices

If Adding Additional Agents

- Need Additional Host(s)
  - Need separate host per DM Agent
  - Cloud templates include way to increase number of agents when deploying
- Network Impact
  - Each DM Agent host will need to be able to connect to source and target system. If using legacy copy methods, network should be optimized for data transfer performance.
  - If using external DSC, each DM agent needs to be able to connect to external DSC host

If Adding Additional Independent Data Mover

- Need separate credentials for each Data Mover if copying data from or to same source or target as another Data Mover instance
- Don't need separate Viewpoint for each Data Mover
  - Can monitor multiple Data Mover instances using same Viewpoint Data Mover portlet as long as all Data Mover instances match the version of the Data Mover portlet.

3.5 Development and Testing Requirements

Many customers have a Data Mover instance dedicated to development and test purposes. A common use case is testing a new Data Mover version in a development environment prior to deploying that version in production.

3.5.1 What to Plan for If Adding a Development Data Mover Instance

If Using Data Mover Viewpoint Portlet

- Need additional Viewpoint instance for development environment
  - Assuming Data Mover versions will be different in development environment versus production environment
  - Cannot use same Viewpoint instance to monitor both development and production as can only have one version of Data Mover portlet deployed on a Viewpoint instance

4 Example Architecture

The following are example potential Data Mover solution architectures for various customer cases. The solution architectures below are not the only potential solutions as there are many possible variations. These examples are provided to help give you an idea of what the process looks like and to highlight important factors to consider when choosing your architecture.

4.1 Disaster Recovery with Data Mover and DSA (Example)

The following is an example of an architecture where the customer has two sites: one for their production systems and one for their DR systems. Below is the configuration they currently have:

4.1.1 Starting Architecture

Site A	Site B

• Production TD SQL-E System
  ○ TD 16.20
• DSA/BAR Media Servers
  ○ Configured for backup of production system
• Customer VMWare Servers A
  ○ VP Primary
	• DR TD SQL-E System
	  ○ TD 16.20
	• DSA/BAR Media Servers
	  ○ Configured for backup of DR system
	• Customer VMWare Servers B
	  ○ VP Secondary

4.1.2 Use Case

Customer wants to synchronize data from production to DR system. Data to synchronize includes databases and tables.

4.1.3 Architecture Choices

Architecture Question	Customer Choice
What copy method to use?	DSA. Customer wants to make use of full database copy. BLC is also in use and, while source/target differently configured, DSA will maintain BLC across network. Both source and target are SQL-E 16.20, so embedded DSC on DM host will be used.
Where to deploy Data Mover?	DM will be deployed on-premises in production and DR environment. DM will be deployed in customer VMWare in both cases.
Is high availability needed?	High availability is desired. DM will be deployed in full HA cluster. Active DM will be deployed in customer VMWare in production environment. Standby DM will be deployed in customer VMWare in DR environment.
What is scaling strategy?	A single DM instance is all that is needed at this time. Since using DSA as primary copy method, will not add additional DM Agents.
Is development/test setup required?	Not at this time.

4.1.4 Solution Architecture

Site A	Site B

• Production TD SQL-E System
  ○ TD 16.20
  ○ BAR ClientHandler installed on nodes
  ○ LAN/WAN connection to DR system nodes at site B
  ○ LAN connection to VMWare Servers A
  ○ LAN/WAN connection to VMWare Servers B
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
• Customer VMWare Servers A
  ○ VP Primary
  – DM Portlet
  – DM failover setup in active mode
  ○ DM Active
  – Embedded DSC configured for DM DSA jobs
  – 1 DM agent configured to work with both active and standby
	• DR TD SQL-E System
	  ○ TD 16.20
	  ○ BAR ClientHandler installed on nodes
	  ○ LAN/WAN connection to production system nodes at site A
	  ○ LAN connection to VMWare Servers B
	  ○ LAN/WAN connection to VMWare Servers A
	• DSA/BAR Media Servers B
	  ○ Configured for backup of DR system
	• Customer VMWare Servers B
	  ○ VP Secondary
	  – DM Portlet
	  – DM failover setup in standby mode
	  ○ DM Standby
	  – Embedded DSC configured for DM DSA jobs (only used if failover occurs)
	  – 1 DM Agent configured to work with both active and standby

4.1.5 Alternate Architecture

In this alternative solution, TMS' are used instead of Customer VMWare.

Site A	Site B

• Production TD SQL-E System
  ○ TD 16.20
  ○ BAR ClientHandler installed on nodes
  ○ LAN/WAN connection to DR system nodes at site B
  ○ LAN connection to DM TMS A
  ○ LAN/WAN connection to DM TMS B
• DM TMS A
  ○ Configured as Active
  ○ DSC configured to allow for DM DSA jobs
  ○ 1 DM Agent
  ○ LAN/WAN connection to both production and DR TD SQL-E systems
• VP TMS A
  ○ Primary VP server
  ○ DM Portlet configured
  ○ DM Failover setup in active mode - monitors active and standby DM servers
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
	• DR TD SQL-E System
	  ○ TD 16.20
	  ○ BAR ClientHandler installed on nodes
	  ○ LAN/WAN connection to production system nodes at site A
	  ○ LAN connection to DM TMS B
	  ○ LAN/WAN connection to DM TMS A
	• DM TMS B
	  ○ Configured as Standby
	  ○ DSC configured but inactive
	  ○ 1 DM Agent
	  ○ LAN/WAN connection to both production and DR TD SQL-E systems
	• VP TMS B
	  ○ Secondary VP server
	  ○ DM Failover setup in standby mode
	• DSA/BAR Media Servers B
	  ○ Configured for backup of DR system

Notes

- Since both prod and DR TD SQL-E systems are version 16.20 and later, multiple DSC's can interact with the SQL-E systems. This allows DM to utilize the DSC embedded on the DM TMS while also allowing for external DSC's to operate at each site for BAR activity. The BAR DSA will have BAR ClientHandler configured on BAR

media servers to use for BAR activity. DM will use BAR ClientHandler installed directly on the prod and DR TD SQL-E nodes for copying data between the two sites.

- Data copied by DM using DSA will travel directly from prod TD SQL-E nodes to DR TD SQL-E nodes. The network connections should be optimized for best performance. See network best practices section.

4.2 Newer/Older SQL-E (Example)

In this scenario, the user has multiple version of SQL-E in their environment. The production and DR systems are the same version but there is an additional system that is currently using an older version.

4.2.1 Starting Architecture

Site A	Site B	Site C

• Production TD SQL-E System
  ○ TD 16.20
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
• QueryGrid Manager TMS A
	• DR TD SQL-E System
	  ○ TD 16.20
	• DSA/BAR Media Servers B
	  ○ Configured for backup of DR system
	• QueryGrid Manager TMS B
		• TD SQL-E System C
		  ○ TD 15.10
		• DSA/BAR Media Servers C
		  ○ Configured for backup of system C

4.2.2 Use Case

Customer wants to synchronize data from production to DR system. Data to synchronize includes databases and tables. The customer also has an older system at site C and they need to periodically copy data between their production system and system C.

4.2.3 Architecture Choices

Architecture Question	Customer Choice
What copy method to use?	DSA and QueryGrid. DSA will be used when copying data from production to DR. QueryGrid will be used when copying data from production to system C since DSA cannot be used to copy from a newer SQL-E version to an older version. QueryGrid will also be used if need to copy from system C to production.
Where to deploy Data Mover?	Active DM will be deployed on-premise in production and DR environment. DM will be deployed using TMS.
Is high availability needed?	High availability is desired. DM will be deployed in full HA cluster. Active DM will be deployed in production environment. Standby DM will be deployed in DR environment.
What is scaling strategy?	A single DM instance is all that is needed at this time. Since using DSA and QueryGrid as primary copy methods, will not add additional DM Agents.
Is development/test setup required?	Not at this time.

4.2.4 Solution Architecture

Site A	Site B	Site C

• Production TD SQL-E System
  ○ TD 16.20
  ○ BAR ClientHandler installed on nodes
  ○ LAN/WAN connection to DR system nodes at site B
  ○ LAN/WAN connection to system C nodes at site C
  ○ LAN connection to DM TMS A
  ○ LAN/WAN connection to DM TMS B
• DM TMS A
  ○ Configured as Active
  ○ DSC configured to allow for DM DSA jobs
  ○ 1 DM Agent
  ○ LAN/WAN connection to production, DR, and system C TD SQL-E systems
• VP TMS A
  ○ Primary VP server
  ○ DM Portlet configured
  ○ DM Failover setup in active mode - monitors active and standby DM servers
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
• QueryGrid Manager TMS A
	• DR TD SQL-E System
	  ○ TD 16.20
	  ○ BAR ClientHandler installed on nodes
	  ○ LAN/WAN connection to production system nodes at site A
	  ○ LAN/WAN connection to system C nodes at site C
	  ○ LAN connection to DM TMS B
	  ○ LAN/WAN connection to DM TMS A
	• DM TMS B
	  ○ Configured as Standby
	  ○ DSC configured but inactive
	  ○ LAN/WAN connection to production, DR, and system C TD SQL-E systems
	• VP TMS B
	  ○ Secondary VP server
	  ○ DM Failover setup in standby mode
	• DSA/BAR Media Servers B
	  ○ Configured for backup of DR system
	• QueryGrid Manager TMS B
		• TD SQL-E System
		  ○ TD 15.10
		  ○ LAN/WAN connection to production system nodes at site A
		  ○ LAN/WAN connection to DR system nodes at site B
		  ○ LAN/WAN connection to DM TMS A
		  ○ LAN/WAN connection to DM TMS B
		• DSA/BAR Media Servers C
		  ○ Configured for backup of system C

Notes

- Since both prod and DR TD SQL-E systems are version 16.20 and later, multiple DSC's can interact with the SQL-E systems. This allows DM to utilize the DSC embedded on the DM TMS while also allowing for external DSC's to operate at each site for BAR activity. The BAR DSA will have BAR ClientHandler configured on BAR media servers to use for BAR activity. DM will use BAR ClientHandler installed directly on the prod and DR TD SQL-E nodes for copying data between the two sites.
- DSA's restriction on copying data from newer to older versions prevents us from using DSA for copying data from production to system C. DSA does support copying older to newer so we could technically use DSA to copy from system C to production or DR. However, system C is using TD SQL-E 15.10 and TD SQL-E 15.10 only supports a single DSC. The user currently has a DSA/BAR setup in site C to backup system C and this DSA/BAR setup is using system C's single DSC spot. To allow DM to use DSA with system C would require us disconnecting the DSA/BAR at site C which is not desirable. The simpler solution is to use QueryGrid.
- Using QueryGrid for copying data to and from system C means that we cannot copy at the database level. We can still copy tables and other non-database objects.
- Data copied by DM using DSA and QueryGrid will travel directly from prod TD SQL-E nodes to DR TD SQL-E nodes. The network connections should be optimized for best performance. See network best practices section.

4.3 Hybrid Cloud DR (Example)

The following is an example where the customer has added a DR cloud site.

4.3.1 Starting Architecture

On-Premise Site	AWS US-East 1

• Production TD SQL-E System
  ○ TD 16.20
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
	• DR TD SQL-E System
	  ○ TD 16.20
	• DR DSU
	  ○ Configured for backup of DR system

4.3.2 Use Case

Customer wants to synchronize data from production to DR system. Data to synchronize includes databases and tables.

4.3.3 Architecture Choices

Architecture Question	Customer Choice
What copy method to use?	DSA. DSA will be used when copying data from production to DR.
Where to deploy Data Mover?	DM will be deployed on-premise in production on TMS. Alternatively, DM could be deployed in the cloud alongside the DR system.
Is high availability needed?	Customer decides full HA is not required right now. Regular backups of DM will be performed.
What is scaling strategy?	A single DM instance is all that is needed at this time. Since using DSA as primary copy methods, will not add additional DM Agents.
Is development/test setup required?	Not at this time.

4.3.4 Solution Architecture

On-Premise Site	AWS US-East 1

• Production TD SQL-E System
  ○ TD 16.20
  ○ BAR ClientHandler installed on nodes
  ○ Connection to DR cloud system via Direct Connect
  ○ LAN connection to DM TMS A
• DM TMS A
  ○ DSC configured to allow for DM DSA jobs
  ○ 1 DM Agent
  ○ LAN connection to Production SQL-E
  ○ Connection to DR SQL-E via Direct Connect
• VP TMS A
  ○ DM Portlet configured
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
	• DR TD SQL-E System
	  ○ TD 16.20
	  ○ BAR ClientHandler installed on nodes
	  ○ Connection to Production system via Direct Connect
	• DR DSU
	  ○ Configured for backup of DR system

4.3.5 Alternate Architecture

On-Premise Site	AWS US-East 1

• Production TD SQL-E System
  ○ TD 16.20
  ○ BAR ClientHandler installed on nodes
  ○ Connection to DR cloud system via Direct Connect
  ○ Connection to DM via Direct Connect
• DSA/BAR Media Servers A
  ○ Configured for backup of production system
	• DR TD SQL-E System
	  ○ TD 16.20
	  ○ BAR ClientHandler installed on nodes
	  ○ Connection to Production system via Direct Connect
	• DR DSU
	  ○ Configured for backup of DR system
	• DM
	  ○ DSC configured to allow for DM DSA jobs
	  ○ 1 DM Agent
	  ○ Connection to Production SQL-E via Direct Connect
	  ○ LAN connection to DR TD SQL-E system
	• VP
	  ○ DM portlet configured

Notes

- Since DSA is being used as the primary copy method, there is no advantage to having DM/VP on-premise or in cloud in terms of performance. Data will be copied directly from production TD SQL-E node to DR TD SQL-E nodes in either scenario.

Section 2: Configuration

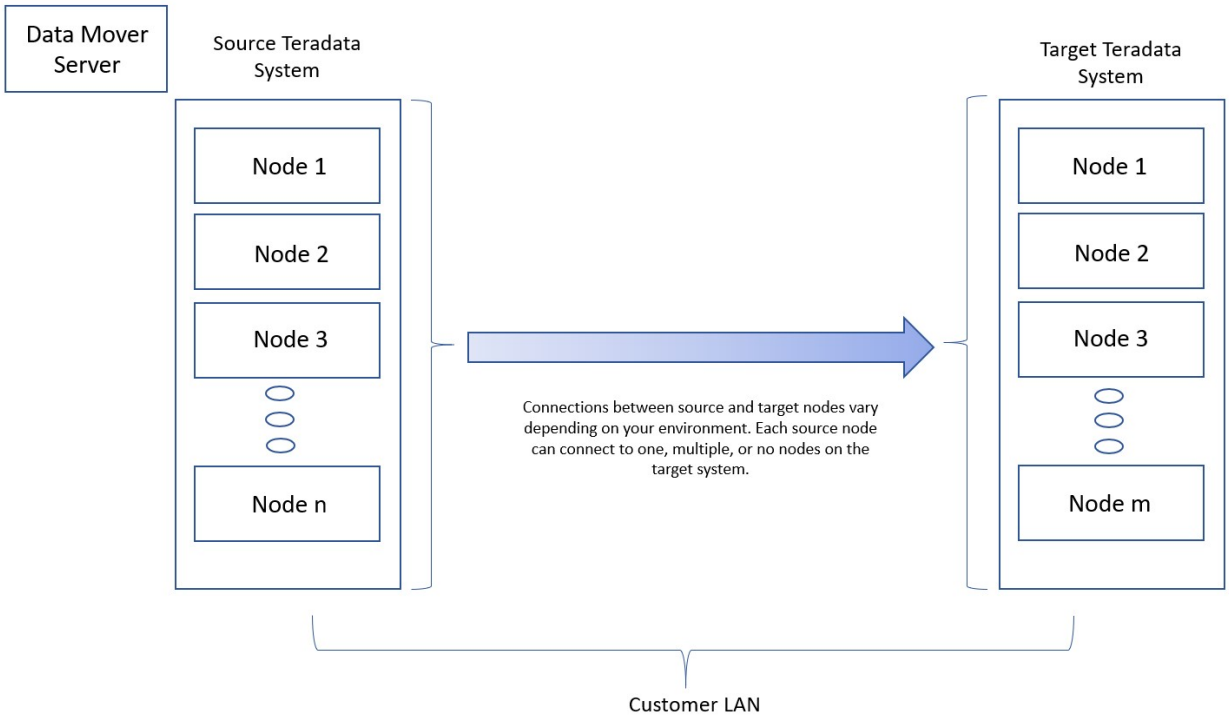
5 Networking

Origins of Poor Performance

The path that data travels changes depending on the copy method Data Mover utilizes. For modern copy methods such as DSA and QueryGrid, data is sent directly from source Teradata nodes to target Teradata nodes. For legacy copy methods such as ARC, TPT, and JDBC, data travels from the source Teradata nodes to the Data Mover servers and then to the target Teradata nodes. This distinction is important as it will determine which network paths need to be optimized to improve Data Mover performance. The below discusses the challenges with legacy copy methods.

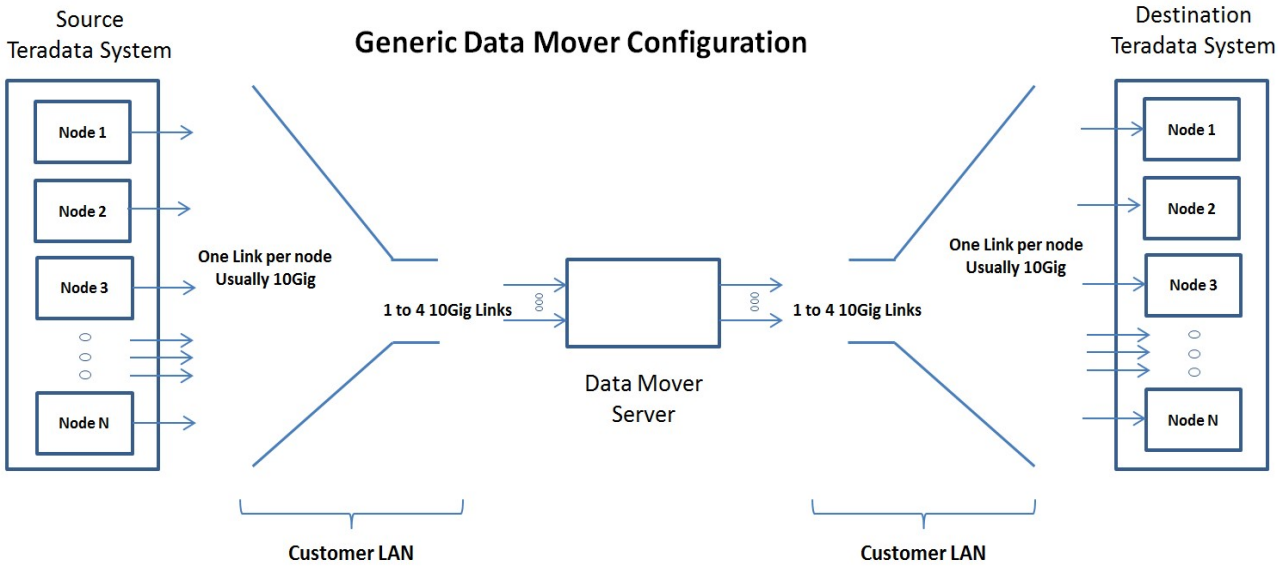
When using legacy copy methods, the Teradata utility (arcmain, tpt, or jdbc) on the Data Mover server opens connections to multiple TPA nodes of the source system to request table rows from those nodes. Collectively, the bandwidth of these nodes to the customer network is usually much greater than the available bandwidth of the receiving interfaces on the Data Mover server. Further, the transmissions from the TPA nodes are “bursty”, as requests are put on the wire when disk I/O completes, which is somewhat random. Sometimes many I/Os will complete at close to the same time. This can cause congestion in the customer’s network, resulting in lost packets.

Modern Copy Methods



Legacy Data Mover Configuration

The legacy drawing below intends to show the “funneling” effect of the network from the TPA nodes to the Data Mover server.



Addressing the congestion and some other areas of networking inefficiency can, potentially, greatly improve throughput.

We will, for the time being, assume that the “fan-out” of the traffic from the Data Mover server to the destination system is not subject to congestion problems, although later we’ll introduce WAN delays and congestion to the discussion.

In an ideal world, all of the TCP connections from the TPA nodes would be transmitting at an equal and constant rate, and the sum of these rates would be equal to the input bandwidth of the Data Mover server interfaces. The speed of the flow can be limited without loss by the customer’s network of switches via the Ethernet hardware flow control. This is effective if there is congestion in a single switch or in the case where an over-subscribed fabric extender is connected to a switch. However, customers usually refrain from extending Ethernet flow control throughout the network since a single misbehaving machine can lock up the entire network.

Consequently, the TCP stack in each TPA node must figure out what the correct transmission bandwidth is for each connection to the Data Mover server by using a “congestion algorithm”. The congestion algorithm tries transmitting a given bandwidth and if every packet is acknowledged by the receiver, it tries sending faster until loss occurs, whereupon it slows down. Over the years a number of different TCP congestion algorithms have been developed.

The bursty nature of the transmissions from the TPA nodes subjects the traffic flows to the possibility of dropped packets as they converge on the Data Mover server. Without requesting changes to the customer’s network, there are a number of things that we can do to reduce the packet drop rate and to mitigate the effects of the drops. Most of the rest of this document addresses these items.

5.1 Recommended Network Settings

The following items will be discussed in this section as well as reason for the changes. How to implement the change will be given in the **Implementing Best Practices** section.

1. Setting MTU to 9000 (if supported) – TPA nodes and Data Mover servers
2. Setting TCP Segmentation Offload to `off` – TPA nodes and Data Mover servers

3. Setting RX and TX Descriptors to 4096 – TPA nodes and Data Mover servers
4. Establishing explicit host routes to distribute traffic across multiple interfaces with separate IP address on the same subnet – Data Mover servers only
5. Setting ARP configuration parameters to support multiple IPs on the same subnet – Data Mover server only
6. Set `tcp_slow_start_after_idle` to 0 – TPA nodes and Data Mover servers
7. Set `active/active` with L3/L4 hashing when using bonding

5.1.1 Small MTU

Use MTU 9000 (TPA nodes and Data Mover servers)

Most customer 10Gig networks support “Jumbo Frames”. This allows sending packets with a size up to 9000 bytes instead of being limited to 1500 bytes. Sending larger packets has a number of positive effects on bandwidth:

- 85% fewer packets transmitted, reducing interrupt rate. Reduced header overhead, increasing the maximum theoretical data rate from 9.45 gigabits/second to 9.90 gigabits/second per 10Gig link
- Reduced queue lengths in the output queues of the switches connecting the TPA nodes to the Data Mover server. Observed behavior seems to indicate that some networks limit queue lengths rather than number of queued bytes
- More rapid convergence on the correct transmission speed

Support for jumbo frames are specified in the `/etc/sysconfig/network/ifcfg-eth<n>` file or in the `/etc/sysconfig/network/ifcfg-bond<n>` file by changing the `MTU=` parameter. There is an automated procedure for applying the best practice for this parameter which will be discussed in the Implementing Best Practices section.

5.1.2 TCP Segmentation Offload (TSO)

Turn TCP Segmentation Offload (TSO) “OFF” (TPA nodes and Data Mover servers)

TCP Segmentation Offload is a feature of the Ethernet card that tries to improve performance by lying to the TCP/IP stack about the actual MTU resulting in a larger MSS (Maximum Segment Size) than the network actually supports. The Ethernet card’s firmware receives very large packets from the IP layer (up to 64K) and splits the packets up for transmission at the actual network MTU. On reception the interface tries to reassemble the incoming packets into larger packets for delivery to the IP layer. In the absence of dropped packets, this lowers CPU load, although our current crop of processors are so fast that we don’t need to depend on this offload feature to maintain wire line rates.

In the presence of lost packets on the network, the TCP Segmentation Offload feature has a very negative effect on the network performance. If a packet on the wire is lost, the recovery may resend all of the packets associated with the larger “fictitious” packet associated with the larger fictitious MTU reported to the sending TCP. This results in a large burst of redundant packets on the LAN, which compete for bandwidth on the LAN and consume unnecessary space in the destination queues of the switches along the way. The problem is magnified in some cases if these packets have to transit over a WAN.

The `ethtool -k <ethn>` command can be used to check the TSO status of an interface. There is an automated procedure for applying the best practice for this parameter which will be discussed in the **Implementing Best Practices** section.

5.1.3 Insufficient RX Descriptors

Increase RX and TX Descriptors to 4096 (TPA Nodes and Data Mover Servers)

The number of incoming packets that can be received via DMA without servicing an interrupt is determined by the parameter RxDescriptors. Modern Linux systems have a two-stage interrupt processing. The first stage is the actual hardware interrupt. The first stage hardware interrupt cannot allocate memory nor do anything else that might block. So it kicks off a second stage interrupt, which is a “software interrupt”, to do the real work. When the systems are

under heavy load, the second stage interrupt, which refreshes the input DMA buffers, can be delayed resulting in unnecessary assertion of Ethernet flow control to the switch. Depending on the switch's configuration and level of congestion, packet loss may result with its associated negative impact on performance. It is therefore beneficial to increase the number of receive descriptors to minimize the chance that the switching network will throw away packets headed towards the node.

A slight improvement in output performance has been observed in some cases by increasing the number of transmit descriptors. The current best practice is to change both TxDescriptors and RxDescriptors from default (512) to 4096. The `ethtool -g <ethn>` command will read out the current values.

5.1.4 Ineffective Traffic Distribution

Apply Traffic Distribution Using Multiple Interfaces with Multiple IP Addresses On A Common Subnet

It is common to connect a Data Mover server to the customer's network using multiple Ethernet interfaces with different IP addresses on a single subnet. This technique has been described in the *Teradata Data Mover Configuration and Upgrade Guide.* This technique often fails to produce the anticipated improvement in performance. The configuration should be audited for potential issues as described in this section.

5.1.4.1 Explicit Host Routes

Use Explicit Host Routes (Data Mover Servers Only)

Explicit host routes are often used to divide traffic up amongst multiple interfaces on the same subnet. See the *Teradata Data Mover Configuration and Upgrade Guide*. Although this configuration technique may feel clumsy, the results can improve performance even when it would seem that a single interface should be sufficient for the anticipated bandwidth. The improvement comes, in part, from the additional queue space in the customer's switching network because there are more inbound paths through the switch. This reduces the number of dropped packets during transmission bursts. The retransmissions required by dropped packets cause direct delays in transmission, kick in the sender's congestion algorithm which unnecessarily slows down transmission and may cause additional congestion during the retransmissions. Therefore, multiple input queues can be a big win.

There are several considerations when using this that should be checked for when using this technique.

Optimum Host Route Assignment

According to the *Teradata Data Mover Configuration and Upgrade Guide* (B035-4012), the IP addresses of the TPA nodes are statically assigned to each of the interfaces on the customer's LAN. Since inbound and outbound traffic can be present on the same interfaces without interfering with each other*, the best assignments of traffic to interfaces is a round-robin assignment rather than segregating each Teradata system to one interface. The distribution of IP addresses should be round-robin across the ACTIVE nodes and then round-robin across the hot spares. Consider these two scenarios for a Teradata system with ten 2+1 cliques and the Data Mover server has three interfaces on the LAN connecting to these TPA nodes:

Typical Desirable Data Mover Routing Distribution

Iface 1	Iface 2	Iface 3
TPA1	TPA2	TPA4
TPA5	TPA7	TPA8
TPA10	TPA11	TP13
TPA14	TPA16	TPA17
TPA19	TPA20	TPA22
TPA23	TPA25	TPA26
TPA28	TPA29	HS3

Poor Data Mover Routing Distribution

Iface 1	Iface 2	Iface 3
TPA1	TPA2	HS3
TPA4	TPA5	HS6
TPA7	TPA8	HS9
TPA10	TPA11	HS12
TP13	TPA14	HS15
TPA16	TPA17	HS18
TPA19	TPA20	HS21

HS6	HS9	HS12
HS15	HS18	HS21
HS24	HS27	HS30

TPA22	TPA23	HS24
TPA25	TPA26	HS27
TPA28	TPA29	HS30

The example on the left distributes the traffic evenly. The example on the right will have no traffic on interface 3 unless there is a hot spare active!

Input and output traffic do not interfere with each other*, so input and output traffic do not need to be segregated onto separate interfaces. Continue the round robin distribution of active destination nodes using all of the interfaces available on the data transfer subnet. In the example below, we'll assume a 3 clique 2+1 source system (9 nodes) and a 4 clique 1+1 system (8 nodes) with a Data Mover server with just two interfaces:

Good Distribution	
round robin by active nodes	
iface 1	**iface 2**
SRC-TPA01	SRC-TPA02
SRC-TPA-04	SRC-TPA-05
SRC-TPA-07	SRC-TPA-08
SRC-HS-03	SRC-HS-06
SRC-HS-09	
DST-TPA-01	DST-TPA-03
DTS-TPA-05	DST-TPA-07
DST-HS-02	DST-HS-04
DST-HS-06	DST-HS-08

Poor Distribution	
all src	**all dst**
iface 1	**iface 2**
SRC-TPA01	DST-TPA-01
SRC-TPA02	DST-HS-02
SRC-HS-03	DST-TPA-03
SRC-TPA-04	DST-HS-04
SRC-TPA-05	DTS-TPA-05
SRC-HS-06	DST-HS-06
SRC-TPA-07	DST-TPA-07
SRC-TPA-08	DST-HS-08
SRC-HS-09	

Poor Distribution	
ignoring HS locations	
iface 1	**iface 2**
DST-TPA-01	DST-HS-02
DST-TPA-03	DST-HS-04
DTS-TPA-05	DST-HS-06
DST-TPA-07	DST-HS-08

Examples of files to implement static host routing can be found in the document under Setting Explicit Host Routes.

*Discussion of simultaneous transmit/receive on the same interfaces

The discussion above suggests that if you only have two interfaces for Data Mover traffic that you should configure the routing so that inbound traffic from the source system and outbound traffic to the destination system share those interfaces, rather than committing one interface to receive from the source and one interface to send.to the destination.

An even better configuration is to have 2 (or more) inbound ports and 2 (or more) outbound ports as shown in the diagram. Having separate receive and send interfaces can improve performance for a not-so-obvious reason. The issue is not that an interface can't send and receive at "full rate" at the same time, but that the acknowledgements going back to the sender (which are very short packets) get delayed by the traffic being sent to the destination (which are very long packets). If there are 4 or more ports available, then segregate the source system to one set of

interfaces and the destination system to another set of interfaces. It is still important to apply the round-robin rules discussed above so you don't end up with one interface just having hot spares.

If you only have 2 ports available, use both interface ports to talk to both systems. This will work better than segregating the traffic as long as the ARP Flux issue is addressed.

5.1.4.2 ARP Flux

Observation of traffic on the interfaces of Data Mover servers often show the traffic "correctly" distributed across the Data Mover server's multiple interfaces at the beginning of a job but later all of the inbound traffic will be coming in just one interface. We want the inbound traffic for the IP address assigned to a specific interface to come in on that interface. Under some conditions the remote systems (or router) can get the MAC address of a different interface on the same subnet. Typically, at the beginning of a job, the correct MAC address for each interface will be sent out in ARP requests from the Data Mover server. These correct entries will be cached for a period of time in the remote machine or router's ARP table and consequently incoming traffic to the Data Mover server will arrive on the correct interface. When the remote machine or router's ARP cache entry times out (20 minutes is common) it will send an ARP request to the Data Mover server to refresh the ARP cache entries. This time the remote machine (or router) will receive the wrong MAC address – usually the MAC address of the Data Mover server's lowest numbered interface on that subnet. This often causes dramatically reduced performance because all the traffic now arrives on one interface.

Three network parameters have to be changed to allow the Data Mover server to respond with the MAC address corresponding to the interface with that IP address:

```
net.ipv4.conf.all.rp_filter=2
net.ipv4.conf.all.arp_filter=0
net.ipv4.conf.all.arp_ignore=1
```

This problem is widely discussed on the 'Net under the term "ARP Flux". Unfortunately, the fixes discussed are incomplete for Linux systems which have additional IP interfaces on other subnets. Since Teradata systems are typically connected to multiple subnets (Pri SM3G, Sec SM3G, BYN0, Customer LAN, and possibly BYN1, BAR, etc) all 3 values must be changed using the instructions in the ARP Flux Settings section.

5.1.4.3 TCP Window Size

Adjustment of TCP memory parameters to increase the default TCP window has been proposed by some to support other applications. These changes are not needed for Data Mover performance because of the inherent small size of the application buffer. The window size changes should be considered optional. However, these changes are included automatically in recent versions of teradata-linux-config and may improve the performance of other applications:

- SLES11: teradata-linux-config 03.01.07.03-1 6/12/2013

TCP Slow Start After Idle

For SLES 11 SP3, the option `tcp_slow_start_after_idle` is set to 1 by default. This causes communication to start or resume gradually after an idle period.

This setting has been shown to cause performance issues for Data Mover, in particular when using TPT. As mentioned, the DM data traffic is "bursty" resulting in idle periods that will trigger the tcp slow start after idle setting. Using too many sessions may also result in idle periods.

The recommendation is to set `tcp_slow_start_after_idle` to 0 to disable the feature. This needs to be done on both ends of the network. For legacy copy methods, this means disabling this feature on the source Teradata system nodes, the target Teradata system nodes, and on the Data Mover servers. For modern copy methods, need to disable on the source and target Teradata system nodes.

5.1.5 Bonding

There is at this time no general recommendation as to whether or not to use bonding. However, if you choose to use bonding, the recommendation is to use an "active/active" configuration with "L3/L4" hashing.

Many enterprise configurations include two sets of switches for redundancy. When active/active bonding is used, using L3+L4 Hash ensures that the outbound traffic is evenly distributed to the two links. Within the datacenter, if the two bonded links land on two separate switches (as is often the case), the interfaces the data arrives on elsewhere in the data center may be influenced by what interfaces were used at the sending end. Therefore, it can be beneficial to have the TPA nodes bonded with Active/Active using L3+L4 has as well.

For example, in the `/etc/sysconfig/network/ifcfg-bond<n>` file, you could have:

For LACP:

```
BONDING_ODULE_OPTS='mode=802.3ad xmit_hash_policy=layer3+4 miimon=100'
```

For Static Port Channel:

```
BONDING_MODULE_OPTS='balance-xor xmit_hash_policy=layer3+4 miimon=100'
```

6 Implementing Network Best Practices

This section describes how to implement the changes discussed in the previous section.

6.1 MTU=9000

The performance of Data Mover can be enhanced by changing the MTU from the default of 1500 bytes to 9000 if the customer's network is properly configured to support this MTU. This parameter allows sending large packets, sometimes referred to as "Jumbo Frames" across the Ethernet.

Prerequisites

To be effective, both the TPA nodes and the Data Mover servers need to have this parameter set. In a properly configured network, it is possible to have some equipment connected to the network which has an MTU of 1500, in which case an "ICMP message too large" protocol message from a router can negotiate the size down for WAN segments that don't support the larger MTU. Please discuss the MTU change with the customer's networking team before proceeding with this step. If a firewall in the customer's network blocks ICMP messages, setting MTU to 9000 on the interfaces facing the Data Mover server may have significant side effects with communication between the TPA nodes and customer equipment not supporting jumbo frames.

Implementation

The value of the MTU is specified by the "MTU=" line in the interface's configuration file: /etc/sysconfig/network/ifcfg-eth<n>. Normally this line's value is defined to be the NULL string:

```
MTU=''
```

This means the MTU of the interface is not changed when the interface is brought up. In most cases this implies the driver's default of 1500 bytes will be used. If the entire path of the customer's network supports "jumbo frames", this value should be explicitly set to 9000.

```
`
MTU=9000
```

The apostrophes are optional as there are no spaces in the parameter.

In the case of a bonded interface, the /etc/sysconfig/network/ifcfg-bond<n> file specifies the MTU. The MTU value in the bonded interface files referenced by the ifcfg-bond<n> file is ignored.

Applying this change can be automated by using the "ifedit" command which is part of teradata-gsctools.

Regardless of which procedure is used to edit the /etc/sysconfig/network/ifcfg-eth<n> file, it is recommended that if it is necessary to change the MTU back to 1500 that it be specified explicitly:

```
MTU=1500
```

Unless it is set explicitly, a reboot is required to change the speed back to 1500.

6.2 TSO "off"

The TSO (TCP Segmentation Offload) configuration of an interface is changed with the "ethtool –K" command. Since the release of gsc-tools 02.01.01.01-1 this can be specified on an interface-by-interface basis by lines in the profile selected by the /etc/sysconfig/network/ifcfg-eth<n> file. The profile is specified by the line

```
TERADATA_NETWORK_PROFILE=<profile name>
```

The lines in the profile that set (or clear) the TSO values are:

```
TSO='off'
GSO='off'
GRO='off'
LRO='off'
```

A new set of properly configured profile files which support turning this feature on and off are included with gsc-tools 02.01.01.01-1 and later. Applying this change and saving a backup can be automated by using the “ifedit” as described previously.

6.3 RxDescriptors=4096 & TxDescriptors=4096

These values are set in the profile selected by the `/etc/sysconfig/network/ifcfg-eth<n>` file. The profile is specified by the line

```
TERADATA_NETWORK_PROFILE=<profile name>
```

The default profile file, “default.ixgbe” (in `/etc/sysconfig/network/profiles`) does not set the RXDescriptors which leaves the driver’s default of 512 in place. Normally we increase the Transmit descriptors at the same time. To increase the number of descriptors it is necessary to make a copy of profile file with a new name, and add these lines

```
RxDescriptors=4096
TxDescriptors=4096
```

Then you change the `TERADATA_NETWORK_PROFILE=<profile name>` line in the `ifcfg-eth<n>` file to point to the new profile. Finally, you restart the interface.

Applying this change and saving a backup can be automated by using the “ifedit” command which is part of teradata-gsctools.

6.4 Automated Procedure for Setting MTU, TSO, and Descriptors

The previous items can be implemented using the “ifedit” command which is part of teradata-gsctools. The minimum version is teradata-gsctools.02.01.01.01-1. Regarding ethernet profiles, the current gsctools package offers this newer profile: `gro_on_tso_off_4096.ixgbe`.

This version enables coalescing of adjacent packets belonging to the same TCP stream and can greatly reduce TCP/IP stack overhead. This is especially valuable if the MTU is 1500 but beneficial for MTU=9000 as well. At this time the recommendation is to make the changes to the interfaces involved with Data Mover traffic only.

The automated procedure is to use the “ifedit” command. The command syntax for the ifedit command is:

```
ifedit interface [name1=value1 ...] [--bounce] [-v]
```

A typical invocation of this command which would change the MTU, the number of descriptors, and the TSO setting all at once and restart the interface would be:

```
ifedit eth<n> MTU=9000 TERADATA_NETWORK_PROFILE=gro_on_tso_off_4096.ixgbe --bounce
```

This command can be run on all TPA nodes at once using the “pcl” command. The “pcl” command does not set the $PATH variable to include the gsctools package, so run it this way:

```
pcl –s /opt/teradata/gsctools/bin/ifedit eth<n> \
 MTU=9000 \
 TERADATA_NETWORK_PROFILE=gro_on_tso_off_4096.ixgbe \
 --bounce
```

There is an excellent MAN page, `ifedit(8),` that is installed with gsctools.

Note that this command and the procedures using them supersede SB573. The SB573 procedure made some changes global to all 10Gig interfaces even if they weren’t intended to be global. Ifedit allows making changes on an interface-by-interface basis. Even if you know how, do not use the SB573 procedure anymore. Update to teradata-gsctools 02.01.01.01-1 (or greater) and use ifedit.

6.5 Setting ARP Configuration Parameters

Setting ARP Configuration Parameters (Data Mover Servers Only) – aka: ARP FLUX

Observation of traffic on the interfaces of these systems often show the traffic “correctly” distributed at the beginning of a job but later all of the inbound traffic will be coming in just one interface.

Although we want the inbound traffic for the IP address assigned to a specific interface to come in on that interface, under some conditions the remote systems (or router) can get the MAC address of a different interface on the same subnet. Typically, at the beginning of a job, the correct MAC address will be used, but then later, when the remote machine’s ARP table entry needs to be refreshed, it will receive the wrong MAC address – usually the MAC address of the lowest numbered interface on that subnet. This causes often dramatically reduced performance.

This problem is widely discussed on the ‘Net under the term “ARP Flux”. Unfortunately, the fixes discussed are incomplete for Linux systems which have additional IP interfaces on other subnets. Since Teradata systems are typically connected to multiple subnets (Pri SM3G, Sec SM3G, BYN0, Customer LAN, and possibly BYN1, BAR, etc.) the instructions below must be used.

Three network parameters have to be changed to allow the Data Mover server to respond with the MAC address corresponding to the interface with that IP address:

```
net.ipv4.conf.all.rp_filter=2
net.ipv4.conf.all.arp_filter=0
net.ipv4.conf.all.arp_ignore=1
```

To read the current values, use these commands from the command line:

```
sysctl net.ipv4.conf.all.rp_filter
sysctl net.ipv4.conf.all.arp_filter
sysctl net.ipv4.conf.all.arp_ignore
```

To set them, use these commands from the command line:

```
sysctl –w net.ipv4.conf.all.rp_filter=2
sysctl –w net.ipv4.conf.all.arp_filter=0
sysctl –w net.ipv4.conf.all.arp_ignore=1
```

To arrange for these values to get applied during a reboot, the values should be appended to `/etc/sysctl.conf` as follows:

```
# the next lines added by Teradata under change control xx/xx/xx
# to correct ARP Flux problem
net.ipv4.conf.all.rp_filter=2
```

```
net.ipv4.conf.all.arp_filter=0
net.ipv4.conf.all.arp_ignore=1
# end of change
```

Some previous documentation suggested putting sysctl commands in `/etc/init.d/boot.local`. Putting sysctl commands in `/etc/init.d/boot.local` is incompatible with Teradata's PUT.

Use of ARP Configuration Parameters When Bonded Interfaces Are Present

The value of "rp_filter" needs to be zero for the ARP Flux patch to work correctly. In this special case, use these settings:

```
# the next lines added by Teradata under change control xx/xx/xx
# to correct ARP Flux problem
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.all.arp_filter=0
net.ipv4.conf.all.arp_ignore=1
# end of change
```

In either case, validate that traffic REMAINS balanced by running "ethmon" for the duration of a job longer than 30 minutes.

6.6 Setting Explicit Host Routes

The static host routes go in the file **`/etc/sysconfig/network/routes`** and the usual # convention can be used for comments.

Some examples are shown below.

6.6.1 Example of A Poor Choice

This example shows a case where all the source TPA nodes are set to route through just one interface (eth4) while all the destination TPA nodes are set to route through the other interface (eth5). This poor configuration disregards the fact that the interfaces can send and receive simultaneously. This configuration does not allow the switch to provide us with extra buffering on the inbound traffic since it only arrives on one interface.

```
#routes to system 1

153.64.209.81 153.64.107.254 - eth4
153.64.209.82 153.64.107.254 - eth4
153.64.209.83 153.64.107.254 - eth4
153.64.209.84 153.64.107.254 - eth4
153.64.209.85 153.64.107.254 - eth4
153.64.209.86 153.64.107.254 - eth4
153.64.209.87 153.64.107.254 - eth4
153.64.209.88 153.64.107.254 - eth4
153.64.209.89 153.64.107.254 - eth4

#routes to system 2

153.64.209.91 153.64.107.254 - eth5
153.64.209.92 153.64.107.254 - eth5
153.64.209.93 153.64.107.254 - eth5
153.64.209.94 153.64.107.254 - eth5
153.64.209.95 153.64.107.254 - eth5
153.64.209.96 153.64.107.254 - eth5
```

```
153.64.209.97 153.64.107.254 – eth5
153.64.209.98 153.64.107.254 – eth5
153.64.209.99 153.64.107.254 – eth5
```

Please note that multiple Data Mover configurations are common and for load balancing they are generally set up with COP entries referencing different TPA nodes in each Data Mover server's **/etc/hosts** file. When setting up the static routes, it is recommended that ALL of the TPA nodes be included in the explicit host routes on the Data Mover servers. This will save editing and auditing time when the routes files are created and will avoid surprises if the COP assignments are changed later to change the way the load is balanced.

6.6.2 Example of a Good Choice

This example shows a case where both systems are (2+1) systems and 3 interfaces are available. Note that for clarity the Hot Spares are segregated to a later part of the file. The active nodes of the source system are distributed round-robin across all of the interfaces, and then the destination nodes are distributed similarly. Finally, the HS nodes are distributed. This results in the most even traffic distribution with 3 times the amount of inbound buffering in the switch. Note that the # comment lines are intended to be included in the file for clarity later.

```
#routes to system 1 (2+1 system) Active Nodes First

153.64.209.81 153.64.107.254 – eth4
153.64.209.82 153.64.107.254 – eth5
153.64.209.84 153.64.107.254 – eth6
153.64.209.85 153.64.107.254 – eth4
153.64.209.87 153.64.107.254 – eth5
153.64.209.88 153.64.107.254 – eth6

#routes to system 1 (2+1 system) Hot Spares

153.64.209.83 153.64.107.254 – eth4
153.64.209.86 153.64.107.254 – eth5
153.64.209.89 153.64.107.254 – eth6

#routes to system 2 (2+1) System Active Nodes First

153.64.209.91 153.64.107.254 – eth4
153.64.209.92 153.64.107.254 – eth5
153.64.209.94 153.64.107.254 – eth6
153.64.209.95 153.64.107.254 – eth4
153.64.209.97 153.64.107.254 – eth5
153.64.209.98 153.64.107.254 – eth6

#routes to system 2 (2+1) System Hot Spares
153.64.209.93 153.64.107.254 – eth4
153.64.209.96 153.64.107.254 – eth5
153.64.209.99 153.64.107.254 – eth6
```

6.7 Disable TCP Slow Start after Idle

Perform the following steps on the source and target Teradata nodes and the Data Mover servers:

1. Check the current setting

```
# sysctl net.ipv4.tcp_slow_start_after_idle
net.ipv4.tcp_slow_start_after_idle = 1
```

2. Change from 1 to 0

```
# sysctl -w net.ipv4.tcp_slow_start_after_idle=0
net.ipv4.tcp_slow_start_after_idle = 0
```

3. Add the following to `/etc/sysctl.conf` (to make reboot persistent)

```
# vi /etc/sysctl.conf
net.ipv4.tcp_slow_start_after_idle=0
```

7 Test and Validation

The packages “teradata-iperf” and “teradata-gsctools” are available on TSS and are “must-haves” for making before and after measurements and for implementing the changes recommended in this document. The package “teradata-ttcp” is also discussed below. It is still available on TSS, but most people will prefer to use “teradata-iperf” which was released in May 2014.

7.1 teradata-iperf

The “iperf” command is the Swiss Army Knife of network performance tools. Unlike its predecessor, “ttcp”, “iperf” can handle multiple simultaneous connections using a single TCP port on the receiving system. This simplifies execution and reduces the need to firewall changes at the customer site to run it. A “man page” is given at the end of this section. For more details on all its features, reference: https://sourceforge.net/projects/iperf2/.

It is recommended that you download it from TSS and install it. It doesn’t do anything unless you run it, so it is safe to pre-install.

Here are some thoughts on how to use “iperf” for network validation. It is recommended that you use PUT to install the “iperf” package in “non-vmf” install mode. You will probably need a change control authorization from the customer.

We use “iperf”, routinely, to measure the maximum sustained bandwidth which can be achieved between two nodes. Data Mover traffic is bursty, so we expect Data Mover to achieve lower bandwidths, but the iperf results can be very useful for establishing the health of the network under sustained load and measuring available bandwidth.

It is possible to run “iperf” in both directions simultaneously, but we normally run in one direction only. The direction tested would be the direction that the Data Mover traffic would be running in. For instance, run traffic FROM one (or more) TPA nodes on the source system TO the Data Mover server. Another test would be FROM the Data Mover server to one (or more) TPA nodes of the destination system.

“Iperf” can be started either as a client or server. A single server can receive data from one or more “iperf” clients. To run traffic from a TPA node to the Data Mover server, first start the “iperf” server on the Data Mover server. We specify to run as a server (-s) and we specify alternate TCP window (-w 1024k):

```
Receiving_node# iperf –s –w 1024k
```

This instance will run indefinitely until interrupted with a ctl-c or otherwise killed. When it starts, it will report what port it is listening on (5001 is default and will be used unless there is a firewall issue at the site). It also reports what TCP window size it negotiated with the kernel. If it reports 512k, then the system does not have the recommended TCP memory tuning sysctls from the latest teradata-linux-config applied. If it reports 512k, make a note of it, but continue with the tests. Normally it will round-up to 2.00 MByte, which is expected.

Example:

```
Server_aka_receiving_node # iperf -s -w 1024k
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 2.00 MByte (WARNING: requested 1.00 MByte)
------------------------------------------------------------
```

To send data to this instance, run a client instance of “iperf” on the node you want to send data from. It is necessary to specify the IP address of the destination node. It is not unusual for there to be 4 or more paths between the TPA nodes and the Data Mover servers, depending on the configuration:

- Primary Server management LAN

- Secondary Server management LAN
- Customer LAN for applications
- Customer LAN for data replication (optional)
- BYN1 (optional -- present for Teradata UDA systems)

It is possible to specify the IP address of any of these interfaces on the server node. You should pick the IP address that will be used during the Data Mover operation to the client node in question. Inspect the COP entries in the host file of the Data Mover server and use the IP address of the interface on the Data Mover server that corresponds to that network. If the Data Mover server has multiple IPs on that subnet, pick the correct address based on the routing on the Data Mover server.

The IP address is specified with "-c". In addition, the window size is specified (-w 1024k) and the number of TCP connections (-P 3).

```
Client_aka_Sending_node# iperf –c <IP_ADDRESS_OF_DATA MOVER> –w 1024k –P 3
```

When the client "iperf" starts, it will report the TCP Port number being used (5001) and the TCP window size negotiated with the kernel. As with the server side, we expect it to report 2.00 MByte. If it reports less (usually 512K), then the TCP memory tuning sysctls haven't been applied. Make a note of that but continue the test.

Example

```
# iperf -c 1.1.20.7 -w 1024k -P 3
------------------------------------------------------------
Client connecting to 1.1.20.7, TCP port 5001
TCP window size: 2.00 MByte (WARNING: requested 1.00 MByte)
------------------------------------------------------------
```

The client will now print out the thread IDs of the child tasks and what local port numbers and remote address were used. For this short test this information won't be needed, but on a long test (specified with `–t <a large length in seconds>`) these numbers can be used to look up values in the netstat output.

Example

```
[ 3] local 1.1.10.4 port 33604 connected with 1.1.20.7 port 5001
[ 5] local 1.1.10.4 port 33603 connected with 1.1.20.7 port 5001
[ 4] local 1.1.10.4 port 33602 connected with 1.1.20.7 port 5001
```

When the requested time period completes (the default is 10 seconds and can be overridden with `–t <seconds to test>`), the client will report performance for each thread and the sum for all the connections:

Example

```
[ ID] Interval    Transfer   Bandwidth
[ 3] 0.0-10.0 sec 3.84 GBytes 3.30 Gbits/sec
[ 4] 0.0-10.0 sec 3.84 GBytes 3.30 Gbits/sec
[ 5] 0.0-10.0 sec 3.85 GBytes 3.31 Gbits/sec
[SUM] 0.0-10.0 sec 11.5 GBytes 9.90 Gbits/sec
```

In this case, the 3 connections successfully saturated a 10 Gigabit link. In this example, the interfaces were configured with jumbo packets (MTU=9000 bytes). Had this been run with MTU=1500 the sum would have been approximately 9.45 Gbits/second due to the overhead of the IP & TCP headers in each smaller packet sent.

The server, at the end of the test, will also report its perception of the performance. We've found that the server's reporting isn't consistent, and we generally prefer the client's bandwidth report.

Example for the same job from the server:

```
[ ID] Interval    Transfer   Bandwidth
[ 4] 0.0-10.0 sec 3.84 GBytes 3.30 Gbits/sec
[ 6] 0.0-10.0 sec 3.84 GBytes 3.29 Gbits/sec
[ 5] 0.0-10.0 sec 3.85 GBytes 3.30 Gbits/sec
[SUM] 0.0-10.0 sec 11.5 GBytes 9.89 Gbits/sec
```

In some cases, the numbers aren't nearly as similar as shown above. The client's numbers are more believable.

It is meaningful to run two clients from two or more TPA nodes to the Data Mover server simultaneously. It is not recommended to start these using "pcl" but rather have separate "ssh" windows open to each TPA node used for the test. Use a longer time (-t 60, for instance) so that you can start all the tests at approximately the same time with respect to the length of the job.

These tests can be run with smaller window sizes (-w 32k) to accentuate the effects of large round-trip (aka ping) times. It is a good idea to measure the ping time between the nodes using "ping" and/or "traceroute" prior to running these "iperf" tests. Longer ping times will require a larger number of TCP connections (-P <number>).

While there are more recent versions of "iperf", we prefer iperf 2.0.5 which is the version on TSS. Iperf does not have a man page, but there is ample documentation. Online at: https://sourceforge.net/projects/iperf2/ and internally by running `iperf –help` which returns this information.

7.2 teradata-ttcp

This is an older package for TCP and UDP performance testing. It is referenced by a number of GSC procedures. However, teradata-ttcp is being phased out in favor of iperf. For performance tests requiring multiple connections, "iperf" is much easier to use since it can handle multiple connections on a single port. This package is available on TSS. Please consider learning to use "iperf" instead of "ttcp".

7.3 teradata-gsctools

This package, available on TSS, contains a growing set of tools provided by the Teradata GSC. We are interested in two sub-packages: **ethmon** and **ifedit**.

7.3.1 ethmon

This is a shell script that reports Ethernet interface and driver performance and diagnostic information periodically as well as certain global TCP counters such as retransmission rates. A typical invocation which displays all interfaces every 5 seconds would be:

```
ethmon –a 5
```

With typical output:

```
01:16:47 PM (%idle 99) eth0 Receive: 0 bit/s Transmit:  0 bit/s
           eth1 Receive: 0 bit/s Transmit:  0 bit/s
           eth2 Receive: 0 bit/s Transmit:  0 bit/s
           eth3 Receive: 9 kbit/s Transmit: 512 bit/s
           eth4 Receive: 0 bit/s Transmit:  0 bit/s
           eth5 Receive: 99 Mbit/s Transmit:  2 Mbit/s
           eth6 Receive: 0 bit/s Transmit:  0 bit/s
           eth7 Receive: 0 bit/s Transmit:  0 bit/s
           eth8 Receive: 20 Mbit/s Transmit: 690 Mbit/s

         segments retransmitted: +4895
```

The example above shows:

- Several idle interfaces
- Scaling of the displayed traffic rates in bits, kbits, and Mbits
- Virtual interfaces never shown. In the case above, eth5 and eth7 are bonded Active/Failover
- High retransmission rate. This is a global value. In this case it is likely that it is associated with the traffic on eth8 since the other interfaces are essentially idle.
- A very low CPU busy value (% idle 99)

Ethmon can be told to skip displaying interfaces having less than a certain amount of traffic. In the following example, the command does not display interfaces with less than 1 megabit/second of traffic:

```
ethmon –a –m 1 5
```

With typical output:

```
01:16:47 PM (%idle 99) eth5 Receive: 99 Mbit/s Transmit:  2 Mbit/s
            eth8 Receive: 20 Mbit/s Transmit: 690 Mbit/s

        segments retransmitted: +4895
```

The example below shows selective use of ethmon. In this case just the 3 interfaces used by Data Mover traffic are selected.

```
ethmon eth6 eth7 eth9 5


12:20:00 AM (%idle 86) eth6   Receive: 2268 Mbit/s  Transmit: 2310 Mbit/s
                rx_flow_control_xon: +8
                rx_flow_control_xoff: +8
                tx_flow_control_xon: +15
                tx_flow_control_xoff: +15

            eth7   Receive: 2283 Mbit/s  Transmit: 1387 Mbit/s
                rx_flow_control_xon: +456
                rx_flow_control_xoff: +456
                tx_flow_control_xon: +12
                tx_flow_control_xoff: +12

            eth9   Receive:  4 Mbit/s  Transmit: 1072 Mbit/s
                rx_flow_control_xon: +476
                rx_flow_control_xoff: +476

            segments retransmited: +13
            TCPFastRetrans: +12
```

The example above shows:

- A huge inbound imbalance that could be due to either a poor set of static routes or due to ARP Flux
- A smaller outbound imbalance. The imbalance in the example above was caused by congestion in the customer's network.
- Rx_flow_control_on & Rx_flow_control_off. These are counts of Ethernet flow control messages from the switch or fabric extender the port is connected to. These messages normally come in pairs. In the example above, the rx_flow_control causes the lower transmit performance on eth7 and eth9.

- Tx_flow_control_on & Tx_flow_control_off. These are counts of Ethernet flow control messages we presented to the switch on the given port. These are an indication of running out of RX_Descriptors. In this example, the fix has not been applied yet.

It is very useful to run “ethmon” on a couple of representative TPA nodes of the sending system AND on the Data Mover server(s) to get a benchmark BEFORE and AFTER applying the changes discussed earlier in this document. While running these instances of “ethmon”, run “iperf” test jobs and observe throughput and retransmissions. Also run real Data Mover table transfer jobs and observe the traffic. It is important for the Data Mover jobs to move enough data to be relevant. Transfer rates during the metadata portion of the transfer are usually not taxing to the network.

Run the same set of tests after applying the Network Best Practices and compare the results. Please consider that there are many variables related to system load and competition on the customer’s LAN from other traffic. Therefore, results may vary from run to run.

Also note that while we expect higher throughput and better traffic distribution after the changes are applied, we would not expect TCP retransmissions to go away as they are part of the normal process of establishing the available bandwidth on the network.

7.3.2 ifedit

This is a shell script that makes editing interface configuration files and restarting the interface much easier than doing each step manually. The program comes with 2 new “ixgbe profile” files which allow turning “TCP Segment Offload” On or Off on a case-by-case basis for 10Gig interface ports.

The `ifedit` command is part of the package teradata-gsctools version 02.01.01.01-1 and higher. It can be used for changing the variable assignments in the `ifcfg-eth<n> and ifcfg-bond<n>` files. More than one variable can be changed at a time. The interface can be restarted by adding `- -bounce`, circumventing the need to do a full network restart. In the case of bonded interfaces, the underlying “bonded” interfaces are restarted too, in the correct order.

8 Reference

8.1 Service Bulletins & Knowledge Articles

The articles listed below are in the Customer Services Knowledge Base. They can be looked up at: http://access.teradata.com

Enter the Article ID and select the **Knowledge Base** checkbox.

8.1.1 How to Configure 10-Gigabit NICs for Optimal Performance

Article ID: KAP3161CAE

Last Edited: 7/25/2014

Author: Robert Joe

This article reports that the global “TSO Off” and Rx/Tx Descriptor changes made by previous versions of teradata-gsctools are no longer done. It gives an example of how to use ifedit to get the same result on an interface-by-interface basis.

8.1.2 Teradata-gsctools ifedit – Edit Network Configuration

Article ID: KAP3161CA6

Last Edited: 7/3/2014

Author: Robert Joe

Contains the manual page for “ifedit”.

8.1.3 BAR Subnets Are Not Segregated if arp_filter Is Not Set (SLES)

Service Bulletin ID: SB 441

Article ID: KAP3C92AE

Last Edited: 2/3/2012

Author: Robert Joe

This was a first attempt to address the ARP Flux issue. The technique in SB 441/KAP3C92AE is incomplete and the method described earlier in this “Best Practices” document should be used instead.

8.1.4 Slow Network Throughput on 10-GiG NICS That Use Default Settings

Service Bulletin ID: SB573 (Deactivated)

Tech Alert ID: NTA 3407 (Deactivated)

Article ID: KAP314A85E

Last edited on 7/3/2014

References DR 168256 and DR 170660

This article introduced the use of improved NIC settings (RX/TX Descriptors, MTUL=9000, and TSO off). It shows how to use “ttcp” and “ethmon” to check the performance of the network.

This article is still interesting, but “ttcp” has been superseded by “iperf” and the correct way to apply the settings is with “ifedit” as described in this document and in KAP3161CAE.

8.1.5 The Complete Guide to Linux Routing

Article ID: KAP1ABEFA

Last Edited: 10/18/2011

Author: Robert Joe

This is an older article that provides a step-by-step guide for setup up routing for customer installations that have multiple customer networks. It discourages the use of multiple interfaces on a single network. With the application of the ARP Flux workaround discussed in this document, it is OK (if not desirable) on a Data Mover to have multiple interface ports on a single network with separate IP addresses.

8.1.6 Redundancy and Segregating Network Traffic across Multiple Network Interfaces

Article ID: KAP1ABEF6

Last Edited: 8/1/2012

Author: Robert Joe

Attachment: Source+IP+Address+Routing.doc

This technique is no longer required with the ARP Flux mitigation. If your system uses the technique described in this document, back the change out and apply the “ARP Flux“ parameters and build a source routed `/etc/sysconfig/network/routes` file.

8.1.7 Linux NIC Bonding Configuration Guide

Article ID: KAP19521A

Last Edited: 2/18/2014

Author: Robert Joe

Attachment: Linux NIC Bonding.doc

This article and attached MS Word document contain a worthwhile and detailed discussion of Bonding (aka Link Aggregation) on Linux systems. A reference is made to “Bare Metal Machines” in referenced document without defining “Bare Metal Machines” in this context means “not applicable to virtual machines”.

8.1.8 How to Configure VLAN Interfaces

Article ID KAP3CCDA2

Last edited 8/5/2014

Author: Robert Joe

Includes *How to Configure VLAN Interfaces* (KAP3CCDA2)

Modern Ethernet switches groups of ports on switches to be segregated into a “Virtual LAN” (aka VLAN). These VLANS can extend between switches via the links interconnected the switches. This makes it possible to have multiple networks on one set of switches and have the networks completely isolated from each other.

Internally, the switches manage the traffic by adding a “VLAN Tag” to each Ethernet message. Normally these VLAN tags are stripped off before the traffic is delivered to the destination Linux system. On rare occasion, there is a need to extend this tagging to the destination system so that two or more separate networks can be delivered to the Linux system over the same Ethernet connection. This is implemented by the switch by not stripping the VLAN tag on delivery and not adding a VLAN tag when receiving a packet.

While seldom used, Linux support the generation and stripping of VLAN tags using “Virtual Interfaces” which are layered on top of the physical interfaces. Article KAP19521A and the associated MS Word document explain how to set up VLANs.

8.1.9 Changing MTU on VLAN Interfaces Does Not Work (SLES)

Article ID: KAP3159B16

Last edited 4/24/2014

Author: Robert Joe

This article discusses the interaction of the MTU settings for VLAN interfaces and the underlying ETH<n> interfaces. Fortunately, the Linux bonding driver gets its MTU from the ifcfg-bond<n> file and the bonded interfaces inherit the MTU from `ifcfg-bond<n>` file.

8.2 Resources from the Internet

8.2.1 Large MTUs and Internet Performance

This study reveals that large MTUs offer throughputs much larger than a simplistic overhead analysis might suggest.

By David Murray, Terry Koziniec, Kevin Lee, and Michael Dixon

School of Information Technology

Murdoch University

http://www.kevin-lee.co.uk/work/research/murray_HSPR_2012.pdf

Abstract: Ethernet data rates have increased many orders of magnitudes since standardization in 1982. Despite these continual data rates increases, the 1500-byte Maximum Transmission Unit (MTU) of Ethernet remains unchanged. Experiments with varying latencies, loss rates and transaction lengths are performed to investigate the potential benefits of Jumboframes on the Internet. This study reveals that large MTUs offer throughputs much larger than a simplistic overhead analysis might suggest. The reasons for these higher throughputs are explored and discussed.

9 Security and Permissions

This section describes best practices for using Data Mover user and permission features, including granting users access to Data Mover, job permissions, user roles, and general security management.

Viewpoint is a major dependency for Data Mover user and permission features and is used as the basis for user authentication for Data Mover. This includes all Data Mover interfaces, not just the Data Mover portlet. When Data Mover is configured with Viewpoint, Data Mover users must supply a valid Viewpoint user when accessing the portlet, running command-line interface commands, or calling Data Mover REST API's. The Viewpoint user is also used when assigning Data Mover permissions. Without Viewpoint, the Data Mover security features are limited. For more information on security best practices when Viewpoint is not available, see Configure Without Viewpoint.

9.1 Choose Permission Enforcement Level to Match Usage

Data Mover provides two permission enforcement levels:

- Daemon-level
- Job-level

Job-level enforcement is enabled by default and provides the ability to share access to the same Data Mover instance between many groups and users. This however may be overkill if you expect to have only a few users accessing Data Mover, in which case Daemon-level enforcement may be more appropriate. So first consider which model best fits your use case before proceeding.

9.1.1 When to Choose Job Level Enforcement

Job-level enforcement provides tight user control and is enabled by default. Job-level enforcement is an extension of daemon-level enforcement; therefore, users are required to have both the daemon-level and job-level permissions. For example, to execute a job, the user must have both the daemon-level execute privilege and the job-level execute privilege for the specific job to be executed. This makes permission management very powerful, but still requires additional work to maintain. Job-level enforcement is recommended in the following cases:

- Mixture of power users and weak users. Not all users are completely trusted.
- Need to isolate specific work. Users should have access to some jobs, but not all jobs.
- Need for job ownership. User needs to create own jobs and then determine if others can access them.

9.1.2 When to Choose Daemon-Level Enforcement

Daemon-level enforcement provides broad permission controls. Data Mover users are granted general read, write, and execute privileges. With daemon-level enforcement, a user who has daemon-level read privilege can view all Data Mover jobs that exist, including jobs the user did not create. This makes permission management very simple but does not allow for tight control of what users can do. Daemon-level enforcement is recommended in the following cases:

- Only a few power users are using Data Mover, and all users are trusted with broad control.
- No need to isolate specific work. All users can have some level of access to all Data Mover work.
- No need for specific job ownership. All jobs are in one general pool that all users can access at some level.

9.2 Use Viewpoint Roles for Easy Group Permission Management

Viewpoint roles provide a convenient method of managing Data Mover permissions, particularly when dealing with groups of users. Data Mover user permissions can be granted to Viewpoint roles in the same manner as they are granted to individual Viewpoint users. This makes managing groups of users easier by allowing you to define roles

with certain Data Mover permissions and grant that role to the users, instead of having to grant those permissions to each user separately.

Using Viewpoint roles is recommended when you have a group or groups of users who require the same set of permissions.

Granting a group of users the same role does not mean those users will have access to all of the same jobs. Individual users can be granted multiple roles as well as permissions not included in a role. Roles provide a method of centrally maintaining a set of permissions; therefore, roles can be used even if certain users within the group need different permissions than the other users.

9.3 Grant Permissions when Creating New Jobs

When creating a new Data Mover job, it is important to consider what other users or roles should be granted permissions on that job. By default, only the job creator (referred to as the job's owner) is granted job-level permissions for a new job. If you want other users or roles to have access to this job, you should grant those users and roles permissions for the job during the job creation process, including cases where daemon-level enforcement is currently used but a switch to job-level enforcement is possible in the future. Job-level permissions can be assigned after creating a job and permissions for multiple jobs can be granted using the portlet.

9.4 Limit User Resource Usage

With multiple users creating and executing Data Mover jobs, it is important to manage what resources each user can use to avoid resource shortages. To limit user resources. use job settings permissions as follows:

- Restrict which utilities each user or role can use. Force users to use slower, but less resource intensive, utilities like JDBC or Teradata PT API Stream. Prevent certain users from using utilities that require a load slot such as Teradata PT API Load and Update.
- Limit the number of data streams that can be specified for each job if using legacy copy methods (ARC or TPT). When using ARC or TPT, each data stream is a set of new processes running on the Data Mover agent. Prevent a single user from using too many data streams and slowing down other work.

For more information, see "Job Settings Permissions" in the Teradata Data Mover User Guide section Permissions.

9.5 View All Data Mover Work

This section describes options for viewing all Data Mover work with daemon-level enforcement and job-level enforcement, including the advantage of using the command-line interface.

9.5.1 With Daemon-Level Enforcement

Viewing all Data Mover jobs and work currently being executed is straight forward when using daemon-level permission enforcement. The user needs daemon-level read permission to see all Data Mover jobs from either the portlet or the command-line interface.

9.5.2 With Job-Level Enforcement

Viewing all Data Mover work in the portlet, requires daemon-level read permission and job-level read permission for all Data Mover jobs, which may be difficult. The easier approach is to use the command-line interface or Data Mover REST API with the super user (dmcl_admin). Run the list_jobs command using the super user; a list of all jobs with execution status will be displayed.

9.6 Usage Models

This section describes potential usage models for the users and permissions feature. You are not limited to these models and other potential models exist; they provide some examples of what can be done with the users and permissions feature.

9.6.1 Limited Power Users

You can allow only a few trusted power users to access Data Mover by doing the following:

- Use daemon-level enforcement. This is a simple method of limiting users that eliminates the complexity of job-level enforcement.
- Define the power user role with daemon-level read, write, and execute permissions.

9.6.2 Multiple Creators, Single Executor

You can allow multiple users to access Data Mover and define work to be done but prevent them from executing the work. In this model, work is executed by only a limited set of trusted power users. A possible use case for this model is when multiple people or groups make data movement requests to the database administrator (DBA). Instead of the DBA creating all of the jobs, outside users can set up a job to do the requested work. The DBA can then review and configure the job as needed, schedule the work via outside scheduling processes, and execute the job. Outside users can view and monitor the progress and results of their requested work. This model allows multiple users to interact with Data Mover while maintaining an efficient and controlled workflow. To implement this usage model, do the following:

- Use job-level enforcement to control what outside users can do. Daemon-level enforcement would not be used in this case because you want to prevent users from editing another users' work.
- Define requester role with daemon-level read and write permissions only. Users will automatically inherit job-level permissions on the jobs they create themselves.
- Define executor role with daemon-level read, write, and execute permissions for the DBA.
- When creating a job, the requester grants all job-level permissions (read, write, and execute) to the executor role so the DBA can access the job.

9.6.3 Configure Without Viewpoint

When Data Mover cannot be paired with Viewpoint, the Data Mover user and permission features are limited. In this case, the command-line interface or the Data Mover REST API are the only method of interacting with Data Mover. Individual users or roles cannot be defined, and job permissions are not available. The only user security model available is to force all Data Mover users to provide the super username and password. Best practices for this case are as follows:

- Enable security management to force command-line interface and REST users to supply the super username and password
- Modify the default super user password to a site-unique password. Rotate the password periodically.

10 Testing Connectivity

Testing should be performed to verify the following connectivity:

- Verify all required ports that Data Mover requires are open. Telnet can be used to test if a port is open. See DM ICU for details on which ports.
- DM Portlet to source/target and DM Daemon. When adding a new DM Daemon to the DM portlet, the DM admin portlet will verify that it can connect to the DM Daemon. When creating a new job and providing source and target systems, use the test button to verify that the DM portlet can successfully use your credentials to login to that system.
  Note: The test button only verifies that the DM portlet can access the source and target system. It does not verify that the DM Daemon and DM Agent can access the source and target system. The system alias name used when defining the job needs to properly resolve to COP entrees on the Viewpoint server and the DM servers.
- DM Daemon to source and target systems. Create a job to verify that the DM Daemon can access both the source and target systems using the provided credentials.
  Note: If using a user pool, the DM Daemon will only use one credential from the pool when creating a job so this process will not verify all users in the pool.
- DM Agent to source and target systems. Run a test job to verify that the DM Agent can connect the source and target systems.
  Note: If using more than one agent, check the job output to see which agent was used and verify that all agents can execute tasks successfully. You can either run tests until all agents get used or only enable one agent at a time and run jobs for each.
  Note: For DSA tasks, the Agent does not directly connect to the source or target system.
- DM Daemon/DM Agent to DSC and DSC to DM Daemon. Run a DM DSA job to verify that the DM Daemon and the DM Agent can connect to the DSC to execute DSA jobs. After completion of job, get job status output level 4 and verify that the object status section shows up in the output. This is indication that the DSC is able to communicate back to DM. If multiple DM Agents, make sure to repeat tests to prove that each DM Agent can run a DSA task successfully.

Testing Credentials and Permissions

If using a user pool, verify that all users in the pool are able to connect to the source/target systems and have proper permissions to perform the required work. Refer to DM User Guide for required permissions. Creating and running DM jobs will only use one user from the pool so all users should be verified outside of Data Mover.

Section 3: Workload Management

11 Running Jobs

11.1 Job Sizing

Data Mover performs best when you break up your workload into several jobs, rather than putting all of the data you need to copy into a single job. This is where job sizing comes in. The technique is to group objects by size and then break down those groups into one or more jobs depending on the amount of data in each group. The goal is to turn your workload into a set of well-balanced jobs that can then be optimized and executed in batches. The advantages of job sizing are the following:

- **Puts your eggs in separate baskets**. There have been cases where users have attempt to copy thousands of objects in a single job and have gotten stuck when a single failure brings down the job. Data Mover does have a restart job feature that causes completed objects to be skipped but the all in one approach tends to result in longer execution. Having separate jobs means a single failure only brings down one job while the others continue to completion.
- **Allows better job optimization**. A lot of Data Mover's job settings are set at the job level rather than the object level. Having one job with lots of objects of various sizes makes it difficult to choose settings that work best for all of those objects. If we instead group objects by size, we can choose settings that match best with the size of object.

Use the following guide lines when sizing jobs:

Object	Guideline	Notes
General	Create jobs with similar sized objects	In many cases, the copy method used by Data Mover will utilize the same resource settings for all objects in a job. If a job contains objects varying widely in size, this can result in resource settings that are either overkill or insufficient for the size of individual objects being copied. Best to group objects by size as much as possible.
Small Objects (less than 100 GB)	Group large sets of small objects (100+) into single job	There is an overhead for copying any object (gathering metadata, connecting sessions, creating schema, etc). For smaller objects, that overhead is as significant if not more so than the time needed to copy the data. As such it is often more efficient to group lots of small objects together to minimize the overhead (i.e. connect sessions once, copy many objects)
	Copy whole table, don't worry about copying just delta.	If the ratio of delta to table size is close enough, it is more time efficient to copy the entire table than to use a partial copy to copy just the delta.
Medium/Large Objects (100 to 500 GB)	Limit to a few dozen per job	With medium to large sized objects, the amount of data becomes more important than the overhead of copy an object. Limiting the number of objects of this size to a few dozen allows us to break up the workload into batches and take better advantage of Data Mover's parallelism. For example, a single job using DSA will result in a single DSA job processing those objects. If we instead had two or more jobs, that results in two or more DSA jobs which can run in parallel.
	Use partial copy for copying delta	Since the delta is likely much smaller than the total amount of data in the table, it is more efficient to use a partial copy job to copy the delta than to try to copy the whole table.

Object	Guideline	Notes
Extra Large Objects (greater than 500 GB)	Limit to 10 or less per job. Consider 1 per job if very large.	It is best to copy very large tables separately from other workloads. A very large table may be best copied with a higher number of resources (sessions, streams), that would be overkill for smaller objects. The smaller objects will also finish much quicker when separated into their own jobs. Also, best to limit the number of very large objects per job. Two very large objects might be copied faster by having each in its own job so that work on both can occur in parallel. Depending on network bandwidth and other workloads that need to run, may limit the number of very large objects being copied at once.
	Use partial copy for copying delta	Since the delta is significantly smaller than the total amount of data in the table, it is far more efficient to use a partial copy job to copy the delta than to try to copy the whole table.
Full Databases	Consider one job per database if databases have large number of objects (1000's)	When dealing with a large number of objects, it very convenient to copy the parent database, rather than copying all objects individually. The catch here is that a single database may contain 1000's of objects. Users have run into issues trying to use a single job to copy half a dozen databases than in turn contained tens of thousands of objects. Data Mover gathers metadata for all of the objects within the databases being copied. When dealing with tens of thousands of objects, even if all are small, this can bog down Data Mover, slowing down not just the job copying the databases but also other workloads. In such cases, best to break down the work into separate jobs to keep each job copying only a few 1000 objects.
	Copy large/very large objects separate from database	To make sure we are using settings that are most optimal for the individual objects being copied, best to copy larger tables separate from the database.

11.2 Job Naming

Give jobs meaningful names to make it easier to understand the job's purpose. Best practice is to follow a specific methodology while naming jobs like source/target system names or department/business unit names. Consider using abbreviated names if entire name could be too long, like suffix F for FULL and P for PARTIAL table copy. Example naming pattern: `<DepartName>_<BusinessUnitName>_<DatabaseName>_<UUID>_<F or P>`.

11.3 Source/Target Alias

Don't use IP/TDPID for source/target tdpid when creating jobs as this can lead to poor performance and is inflexible. Using an IP address will result in traffic going to one particular TPA node. If the address of the node is reconfigured, then the job must also be edited. Instead, use alias names which are mapped in DNS, host file, or cop entries. The alias name can be used to route traffic to multiple TPA nodes and the IP addresses that that the alias resolves to can be modified without needing to change the job. Refer to network best practices section for more details.

E.g.: Refer below server as 'proddr' in job instead of ip/tdpid.

```
COP entries:
10.1.1.4 sdx1325cop1 proddrcop1
10.1.1.8 sdx1325cop2 proddrcop2
10.1.1.5 sdx1325cop3 proddrcop3
```

11.4 Credentials

Use Data Mover's user pool feature when providing credentials for a job rather than providing the username and password in the job definition. When you need to update passwords or add/remove credentials, you can do this in a single step and all jobs using the updated user pool will pick up the change the next time they are executed.

11.5 Copy Method

In general, it is best not to force the use of a particular copy method when creating a job. This allows for Data Mover to choose the best copy method available. This also makes it easier to adopt new copy methods in the future. For example, if your jobs initially forced the use of legacy ARC, all of those jobs would need to be modified when switching to use DSA. If instead no copy method was forced, the switch to DSA could occur without having to modify each job.

QueryGrid is an exception. Data Mover is unable to automatically choose QueryGrid as the copy method. To use QueryGrid, you must provide the name of the foreign server when defining the job.

See Section I for more details on choosing between copy methods.

11.6 Streams/Sessions

Data Streams and source/target session settings work differently depending on the copy method you are using. See the following recommendations based on the copy method.

<table>
<tr><th>Copy Method</th><th>Guideline</th><th>Notes</th></tr>
<tr><td>DSA</td><td>Specify data streams as following:<table><tr><th>Tables/Database Size</th><th>Data stream Value</th></tr><tr><td>250 GB</td><td>1</td></tr><tr><td>250 GB - 500 GB</td><td>2</td></tr><tr><td>500 GB - 1 TB</td><td>3</td></tr><tr><td>1 TB or above</td><td>8</td></tr></table></td><td>If you do not provide data stream value, DSA uses as many data streams as possible which consumes excess resources for a single job. For best performance set a data stream value which limits the use of data streams.</td></tr>
<tr><td></td><td>Source/target sessions ignored</td><td>Setting not used in DSA jobs</td></tr>
<tr><td>Query Grid</td><td>Data streams and source/target sessions ignored</td><td>Settings not used in Query Grid jobs</td></tr>
<tr><td>ARC</td><td>Increase sessions first if dynamic settings not producing desired performance</td><td>Each ARC data stream will connect the number of source/target sessions provided. TASM rules will override this setting.</td></tr>
<tr><td></td><td>Increase data streams if additional performance needed</td><td>Data Mover will dynamically choose number of data streams if you do not provide a value. However, you may need to increase this setting to achieve better performance. TASM rules do not apply to number of data streams but they will limit the number of sessions each data stream connects.</td></tr>
<tr><td>TPT</td><td>Increase sessions first if dynamic settings not producing desired performance</td><td>The number of sessions each TPT data stream connects is determined by taking the source/target session value and dividing by the number of data streams. TASM rules will override the total number of sessions connected.</td></tr>
</table>

Copy Method	Guideline	Notes
	Increase sessions when increasing data streams	If you increase data streams for a DM TPT job, make sure to also increase the number of sessions. If you increase data streams but not sessions, the same total number of sessions will be connected and each data stream will have fewer sessions to use. TASM rules will override session setting but not data stream setting.

11.7 Compression

When Copying Data

Data Mover does not provide a mechanism to compress data that is being copied. However, in certain cases, data that is already compressed on the source system will be copied in its compressed state. DSA preserves BLC, MVC, and algorithmic compression. The data will be compressed as it goes across the network though it may need to be uncompressed by the target system if the data needs to be re-distributed.

Legacy ARC maintains MVC and algorithmic compression but not BLC. All other copy methods require the data to be uncompressed on the source system before copying the data to the target system.

Compressing Data on Target

If BLC is used on the source system and you are copying data using TPT, you need to add 'BLOCKCOMPRESSION=YES' to the jobs query band settings to ensure that the data is also compressed on the target system. This does not cause the data to be copied in a compressed state, but it will tell the target system to compress the data when it is received. See Teradata DBS guide for more details.

You can also control data temperature along with block level compression in query band 'TVSTEMPERATURE=COLD;', See Teradata DBS user guide for more details about the 'TVSTemperature' data temperatures setting.

11.8 Job Validation

Data Mover provides a simple row count validation feature to verify that source and target have the same number of rows after a job completes. You can choose to compare the total table row count or just the count of rows that match the WHERE clause provided in partial table copies. False failures can occur if rows are inserted or deleted in the source table during the DM copy or in the time between DM completing the copy and performing the row count. Teradata Ecosystem Manager offers more robust table comparison features that allow you to define your own custom comparison or to allow the row count to be successful if within a certain percentage.

12 Optimizing Workloads

After getting Data Mover jobs created and running, the next question is how to improve performance. You may need current jobs to run faster to meet your goal timelines or you may be looking to increase the workload. The first major aspect of performance is network configuration which we discussed in the networking best practice section. Here we discuss the next step which is optimizing the execution of your workload.

12.1 Efficient Job Execution

Executing each job as efficiently as possible is a key to both improving the performance for that particular job as well as freeing up resources to allow more jobs to run.

12.1.1 Reduce Number of Commands Used to Execute Each Job

As you begin to automate your Data Mover workload and have jobs executing more often, the number of commands that you use as part of that automated execution becomes a limiting factor for performance. For example, let's say you use the following pattern to execute a job:

1. Check to see if job exists using list_jobs or call to /datamover/jobs
2. If job exists, delete job
3. Create job to copy latest delta
4. Start job with sync option so that command line or REST API block until job finishes
5. Once job finishes, get full job status

If you are only running a few jobs at a time, this pattern may work fine to meet your needs. However, consider that as you increase the number of jobs executing in parallel, all of the commands above are multiplied by the number of jobs you have running. This will cause increased contention in Data Mover which will not only prevent you from increasing the number of jobs you have executing but also decrease the performance of the jobs you already have running.

Best to use as few commands as possible when executing each job. The following pattern would be better than our initial example:

1. Start job with sync option. If need to update the job on each execution, pass in updated parameters when calling start.
2. Once job finishes, only gather full job status if conditionally needed (i.e. job failed)

12.1.2 Use Freeze Job Steps

By default, every time you execute a Data Mover job, Data Mover rebuilds the job plan. Data Mover does this to make sure that the job plan matches the current environment. However, this is not necessary in most cases. Best to enable the freeze job steps feature. When this is enabled, Data Mover will re-use the existing job plan when the job is executed rather than going through the work of creating a new job plan. This can be a significant time saver, reducing the overhead for each job execution. If the environment does change, your job may fail at that point due to the job plan being out of date, but you can then use the update job steps command to re-fresh the job plan.

Note that if you pass in new parameters when starting jobs, that the job plan will always be rebuilt, even if freeze job steps is enabled. If you want to get the benefit of freeze job steps, try to see if you can avoid passing in new parameters on every execution. This may not be possible for partial copy jobs that need to be updated every time to grab the latest delta.

12.1.3 Use Credential Pools

Your credentials for the source and target systems will need to be updated on a regular basis. If you supplied the credentials directly when creating your jobs, you will need to update every job when your credentials change. If using the freeze job steps feature, this will cause a full rebuild when you provide the new credentials. To avoid this and to make credential management easier, make use of Data Mover's credential pools. With credential pools, individual jobs do not need to be updated when credentials change. You instead update the credentials in the pool and your jobs will pick up those changes the next time they are executed.

12.2 Batch Management

Even if you size your jobs appropriately and execute them efficiently, you will still run into limitations in terms of how many jobs can be executed at a given time. If you have a large workload, it is best to execute your jobs in small batches and to space out those batches to keep Data Mover working but to avoid overload. The exact size of the batches and how often to execute will depend on the size of the jobs. Smaller jobs may finish quick and take up less resources so you might be able to execute 20 to 30 at a time. Bigger jobs will take longer to finish and use up more resources so you will need to use smaller batches and give jobs more time to finish before starting the next batch. If you can monitor Data Mover's usage and dynamically execute jobs based on available capacity, that is ideal. Another strategy is to break your work into batches and then play with the size of batches and time in between batches until you reach the ideal throughput.

12.2.1 Throttle Your Job Submission Rate

By default, Data Mover has a job limit of 20 running jobs. This means that if you started 30 jobs, 20 of those jobs would move into a RUNNING state, while the remaining 10 would be marked as QUEUED. The 10 QUEUED jobs will be promoted to RUNNING whenever one of the current RUNNING job finishes. Technically, if you have 100 jobs that you need to run, you could start all 100 at the same time and let Data Mover work through the backlog. However, this ends up resulting in poor performance and even failures if the number of jobs is too high. The first 20 jobs are put into a RUNNING state. The next 20 are put into a QUEUED state which causes reads and writes the DM repository as well as takes up memory as the jobs are staged. The remaining 60 jobs are put on a waiting list. For the 80 QUEUED and waiting list jobs, the command line or REST API continuously ping the DM Daemon with alive messages if jobs are started using the SYNC option as recommended. If run with command line, each start command also results in a JVM instance that runs until the job finishes. This ends up being a lot of wasted resource usage for the 80 jobs that aren't even running and that has been shown to have a negative performance impact on the 20 jobs that are actually running.

Best practice is to avoid exceeding the max running job limit. Having a few QUEUED jobs is ok but keep it to a low number.

12.2.2 Be Careful with Raising Max RUNNING Job Limit

If we want to avoid QUEUED jobs, then raising the max RUNNING job limit might solve the issue of handling larger workloads. Unfortunately, increasing the max RUNNING job limit results in the same issues seen with having large number of QUEUED jobs. When a job is in RUNNING state, it still has to wait for spot to open up on the available DM Agents to execute its tasks. This means that jobs may be marked as RUNNING but are currently not doing anything as all the DM Agents are busy. Having a high number of RUNNING jobs also means we will have a lot of work querying metadata from source and target systems as well as reading and writing to the DM repository which will slow down the execution time of individual jobs.

Best to start with the default 20 max RUNNING job limit and only increase that limit once you have determined that there is sufficient capacity to do so.

12.2.3 Use List Tasks to Tune Resources

As mentioned, as job can be in a RUNNING state and not be doing anything. This is because the job's tasks may be QUEUED waiting for resources on the DM Agent to execute. In the DM portlet, you can view all of the tasks for all jobs by clicking on the "Daemon Status" button. Likewise, you can run the list tasks command or call the `/datamover/tasks` API. This will show you why tasks are being QUEUED. All tasks require an open slot on a DM Agent. Certain types of tasks like TPT tasks have their own separate limit like load slots that will cause those tasks to be queued even if a DM Agent has an available slot. By viewing this information, you can get an idea of what resources are being maxed out.

12.2.4 Increase DM Agent Max Task Limit

Modern copy methods like DSA and Query Grid use very little resources on the DM Agent. If you are using these copy methods and see that job tasks are getting queued waiting for a DM Agent slot, try increasing the DM Agent's max task limit to allow more of these tasks to execute.

Legacy copy methods ARC and TPT use CPU, memory, and network on the DM Agent to copy data so there is a limitation on how many of these types of tasks can run. Still you can try increasing the limit in small steps to see if that improves your throughput.

12.3 Job History Management

Experience in the field has shown that the amount of data in the DM repository can cause poor job performance. In order to prevent Data Mover repository tables from overflowing with job history data, purge should be enabled and configured properly. It is best if user purge based on both the percentage of repository space used and by age of the data. Purge properties should be tuned base on the customer utilization of Data Mover and business requirements. It is highly recommended that customer:

- Enable the daily execution of the feature by setting `repository.purge.enabled` to “true”
- Not keep records older than 60 days maximum by setting `repository.purge.history.unit` to “days” and `repository.purge.history.unitcount` to a value bellow 60
- Ensure that the repository stays at least 50% empty by setting `repository.purge.percent` to “50”

Since purge by the age of the records feature is more optimized than purge by free repository space percentage, one should set the former option such that the latter execution performs no work. Data Mover also checks the size of repository tables on daily bases and alerts the user if any table's row-count exceeds a million rows. If the TVI alert is triggered, then customer must reevaluate the purge settings. Purge should be configured to be executed during the time of the day when Data Mover jobs are not running by setting `repository.purge.hour` and `repository.purge.minute` properties and unexpected behavior could happen if this recommendation is not followed.

13 Using Data Mover for Disaster Recovery

Data Mover is a part of DR solutions at many customer sites. Typically, Teradata Professional Services is involved in helping the customer implement a solution. Over the years, Teradata PS have developed an automation framework that sits on top of Data Mover. This framework provides a method of recording the objects that have changed and generating Data Mover jobs to copy those changes over from the production system to the DR system. This framework is not a part of the Data Mover product. Users can create their own frameworks if desired or work with the PS framework to avoid needing to start from scratch. In this section, we'll discuss the techniques used to use Data Mover as part of a DR solution.

13.1 Seeding DR System

Must Copy Over Databases Definitions, Users, Permissions, Roles First

Data Mover cannot copy users, permissions, and roles. Data Mover can copy databases but has a restriction that the database is already defined on the target system. When setting up a DR system, an outside process must be used to first copy these objects over to the DR system before Data Mover can be used.

Can Copy Databases and Tables

Once you have copied over the necessary users, permissions, roles, and database definition using an outside process, Data Mover can be utilized to copy over databases and tables along with their data. Remember to reference the job sizing and workload techniques discussed in earlier sections to properly break up and space out your work. Note: To utilize the Database Copy feature, the secondary Database system should have a higher or equal version as the source system. This is highly recommended as DSA utility cannot be used otherwise.

13.2 Keeping DR Synchronized

Copying Whole Table vs. Delta

The rule of thumb is that delta backups are useful when tables are large in size and capturing the new data from source system is simple. Example of this is log tables with their timestamp column used as primary index.

Utilizing partial copies could be beneficial also if:

- Old captured data is not changed so much.
- Faster force utility could be used for partial copy than full table copy.
- Customer does not need the old data. Example: analysis has been done on the data already and actions has been taken accordingly.
- Old data from the table is being utilized by other tools and shorter downtimes are essential to the business.
- Data on the target table is corrected/optimized and customer does not want it to be replaced.
- Staging tables needs to be used for full copy and not enough free space available on source or target systems.

Full copies are best to be used when:

- Tables are empty or have few rows.
- Tables have few old rows and countless new rows.
- Great number of rows have been updated/changed within the tables.
- Tables/Views definition have changed
- It is hard or impossible to identify new changes

13.2.1 How to Copy Delta

Tracking primary database changes is not within the scope of Data Mover capabilities. There are few proven methods that can be utilized to identify updated/inserted rows within the primary table.

Batch ID Method

This option requires the user to add an integer column to the table. Every time delta backup is performed, the tool inserting data into the table will increment the batch ID number by one and use it for the newly updated/inserted rows. During the next partial copy, the WHERE clause will select the rows with the highest batch number.

Timestamp Column Method

A timestamp column is going to be added to the table possibly with the default value set to CURRENT_TIMESTAMP. For changes done to the table the value of this column must be updated with the current time. The last backup time needs to be recorded and will be used within the WHERE clause of the next delta backup.

Using External Table for Batch ID or Timestamp Column Method

To avoid recreating a job every time delta backup is needed, an external table should be created containing last Batch ID or last Backup timestamp. The WHERE clause defined in the delta job should SELECT its time range or Batch ID from this table. After each backup the value within the table should be updated.

Soft Delete Method

In order to take into account the delete rows, a new Boolean column needs to get added to the customers tables. The default value for the column should be false, and it must be set to true if the row needs to be deleted. Batch ID column or timestamp column must be updated as well. During the next scheduled backup, after performing delta copy, marked rows should be deleted from the source and target tables.

14 Migrating Jobs from ARC to DSA

14.1 Overview

Teradata ARC (legacy BAR) is being deprecated in Teradata SQL-E 17.00. As a replacement, Data Mover offers support for DSA in addition to Query Grid, TPT, and JDBC. Data Mover support for DSA was introduced in Data Mover 16.20.00.00 and can be utilized to copy data between Teradata SQL-E versions as old as TD 14.10.

Note: Similar to ARC, DSA does not support copying data from newer TD SQL-E versions to older versions. For those cases, consider using Data Mover with Query Grid, TPT, or JDBC.

Migrating from using Data Mover with ARC to using Data Mover with DSA requires configuration changes as well as changes the path by which data travels as it is being copied. When planning such a migration, take these considerations into account:

14.1.1 DM with DSA Copies Data Directly from Source to Target

When Data Mover uses DSA to copy data, the data will travel directly from the source Teradata SQL-E nodes to the target Teradata SQL-E nodes. This means that the source and target Teradata SQL-E nodes need to be able to directly connect to one another and the network between these systems needs to be able to handle the data of the objects being copied.

This is a big change from using Data Mover with ARC. With ARC, the data travelled from the source Teradata SQL-E system to the Data Mover server and then to the target Teradata SQL-E system. As such, many sites optimized the network connections to and from the Data Mover server(s) to handle the large amount of data being copied. DM with TPT and JDBC will still use this network path but DM with DSA will only use this path for querying metadata and coordinating with the BAR ClientHandler and DSMain components that reside on the Teradata SQL-E nodes.

14.1.2 DSA Needs to be Configured before it can be used in Data Mover Jobs

With ARC, all of the required client software comes included on the Data Mover server and can be utilized immediately assuming the Data Mover server has connectivity to the source and target Teradata SQL-E systems. With DSA, Data Mover includes embedded DSA software but configuration is required before Data Mover can utilize DSA as a copy method.

When utilized by Data Mover, DSA requires BAR ClientHandler software to be installed on at least the target Teradata SQL-E nodes. It is recommended to install the ClientHandler on both source and target Teradata SQL-E nodes. The source and target systems need to be registered with the embedded DSC. Network netmask or fabric needs to be configured to define the network path that will utilized when copying data from source nodes to target nodes. These steps are defined in detail in the Teradata Data Mover Installation, Configuration, and Upgrade Guide.

14.1.3 Data Mover Jobs May Need to Be Modified to Use DSA

After configuring Data Mover with DSA, there may be additional changes required to allow existing jobs to use DSA:

- The "disable.default.dsa" DM Daemon configuration parameter needs to be set to false. If you upgraded to a version of Data Mover that supports DSA and you did not immediately configure and use DSA, this parameter may have been set to "true" to prevent Data Mover from trying to use DSA before it was configured. When set to "false", Data Mover defaults to using DSA whenever both the source and target Teradata SQL-E are versions 16.00 or later and all the conditions for using DSA have been met (ex: copying between same versions of Teradata SQL-E or copying older to newer).
- The force utility job setting needs to be unspecified or set to "DSA". If your jobs were previously forcing the use of ARC using the force utility job setting, you will now need to modify your jobs to either not specify a copy method so that Data Mover will choose DSA or to force the use of DSA. Recommend setting the force utility job to "unspecified" if possible, to avoid needing to modify jobs again.

- Alter the data stream job setting. By default, ARC uses 1 data stream, but DSA uses as many data streams as possible which generally does not give the best performance. Best practice is to change the data stream value based on the following recommendations:

Tables/Database Size	Data stream Value
250 GB	1
250 GB - 500 GB	2
500 GB - 1 TB	3
1 TB or above	8

- (Optional) Remove source and target session job settings. These settings are not used by DSA and are ignored if specified in the job definition. You can therefore remove these settings to avoid confusion.

Note: A migration tool is included in Data Mover 17.00 and later to assist in making these changes. See the *Teradata Data Mover Install, Configuration, and Upgrade for Customers* (B035-4102) for details.

14.1.4 Consult Your Solution Architect Before Migrating

Finally, before migrating your jobs from using ARC to using DSA, consult your Teradata Solution Architect to verify that all considerations have been accounted.