Teradata Database 13.0

Security Target

Version 1.4
November 2010

TRP Number: 541-0006458

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

This section identifies and provides and overview of the Security Target (ST). It also identifies the Target of Evaluation (TOE) and provides an evaluatable claim of Common Criteria (CC) conformance for the TOE.

1.1 SECURITY TARGET REFERENCE

The Security Target (ST) is identified as follows:

Teradata Database 13.0 Security Target
Version 1.4
November 2010

This ST describes the security assumptions, threats, objectives, requirements, and an associated rationale for the Teradata Database and its IT environment. The language used in this Security Target is consistent with the *Common Criteria for Information Technology Security Evaluation, Version 3.1 Revision 2*.

Chapter 1 of this ST provides an introduction, identifying information for the ST and the TOE, and a description of the TOE and guidance on its use. Chapter 2 describes the conformance claims made by the ST. Chapter 3 defines security problem addressed by the TOE in terms of assumptions and threats. Chapter 4 identifies the security objectives of the TOE and of the operational environment. Chapter 5 defines extended components. Chapter 6 describes the TOE security functional requirements and the security assurance requirements. Chapter 7 is the TOE Summary Specification, a description of the functions provided by the Teradata Database to satisfy the security functional and assurance requirements. Appendix A provides a listing of acronyms used throughout the document.

1.2 TOE REFERENCE

The Target of Evaluation (TOE) defined in this ST is identified as follows:

Teradata Database 13.0

The TOE is a product of Teradata Corporation and is referred to as the Teradata Database within this ST.

1.3 TOE OVERVIEW

The product type of the TOE described in this ST is a relational database management system (RDBMS). The TOE provides the capability to limit TOE access to authorized users, enforce

Discretionary Access Controls on objects under the control of the TOE based on user and/or access role authorizations, and to provide user accountability via audit of users' actions.

The Teradata Database is designed to access, store, and operate on data using Teradata Structured Query Language (Teradata SQL), which is compatible to ANSI SQL with extensions. The database was developed to allow users to view and manage large amounts of data as a collection of related tables. The Teradata Database includes security functionality for parallel database environments supporting multiple concurrent users. The security functionality includes:

- user management - including identification and authentication
- password management controls
- discretionary access control model to enforce access controls on database objects and resources (e.g., databases, users, tables, views, triggers, macros, stored procedures, external procedures, functions, types, GLOP objects, replication groups, authorization objects, access roles and profiles)
- extensive set of access rights
- access roles for management of access rights
- configurable auditing facility

The Teradata Database functions as a database server in a traditional client/server environment. Access requests are made via the Teradata Tools and Utilities that provide connectivity to the database and submit Teradata SQL statements to the database. For any access to the database through its defined external user interfaces, the database ensures that all security enforcement functions are invoked and succeed before any access request is allowed to proceed.

The Teradata Database operates as a parallel application executing as a set of cooperating processes on an underlying host operating system. The host operating system is not part of the TOE but rather part of the supporting operational environment. The operational environment provides several supporting security mechanisms to prevent compromise of the TOE security functions including:

- authentication and authorization of administrator access to database control utilities and other utilities used to manage system resources and I/O interfaces
- isolation of the TOE Security Functions (TSF) to prevent tampering with TSF components (e.g., the TOE processes managing the database)
- network perimeter controls to restrict network access to the database server to mitigate malicious attacks against the operating system upon which the TOE operates

The Teradata Database, as a software TOE, executes on non-TOE hardware and software systems. The major non-TOE hardware and software systems required for use of the TOE include:

- Symmetric multiprocessing (SMP) server with Intel Xeon EM64T processors (minimum 2.33 GHz.) and minimum 6GB of random access memory (RAM) running SUSE Linux Enterprise Server (SLES) 10 SP1 (64-bit)

- Massively parallel processing (MPP) server with Intel Xeon EM64T processors (minimum 2.33 GHz.) and minimum 6GB of random access memory (RAM) per node running SUSE Linux Enterprise Server (SLES) 10 SP1 (64-bit)

Note: This evaluation is limited to Teradata Database 13.0 running on SUSE Linux Enterprise Server (SLES) 10 SP1 (64-bit).

The Teradata Database is the only application executing on the server and underlying operating system. Other server applications, such as web server, e-mail server, domain server, directory server, etc. do not run on a Teradata Database server.

1.4 TOE DESCRIPTION

The Teradata Database is comprised of several software subsystems including the Parallel Database Extension (PDE), Gateway for LAN, Session Controller, Parser and Access Module Processors (AMP). A Session Controller and a Parser subsystem are always configured together in what is called a Parsing Engine (PE) virtual processor.

The PDE subsystem is a software interface layer that operates on top of the host operating system and provides an interface between the other database subsystems and the underlying operating system software. PDE includes a BYNET driver that manages the communication devices that interconnect the hardware nodes on which the server software is resident. It provides a standard interface for inter-process communications across nodes in a multi-node environment. PDE also includes a Console module (CNS) that manages the interface for input and output generated from a Database Window (DBW) on the Console.

The Gateway for LAN subsystem provides the client communications interface to Client applications connected via a network interface. It receives all messages sent from the client to the server. This includes messages containing Teradata SQL statements as well as messages for functions such as connecting and disconnecting sessions, determining the configuration of the server, receiving authentication data from the client, and responding to test messages that determine the health of the server over the LAN. For messages that contain Teradata SQL, the Gateway for LAN checks those messages to ensure that they conform to the specified protocol and forwards them to a Parser subsystem. The Gateway for LAN also receives response messages from the PE subsystems and returns them to the appropriate Client application. The Gateway for LAN also interacts with PDE for memory management and message handling services and for access to underlying operating system services.

A PE virtual processor always includes a Session Controller and a Parser subsystem. The Session Controller processes external requests to establish or terminate a logical connection between the application and the server. It also provides for the recovery of sessions following client or server failures. The Session Controller manages session activities, such as logon, password validation and logoff. The Parser decomposes SQL into relational data management processing steps. It processes external requests containing Teradata SQL by syntactically parsing the statements and generating a set of steps comprising an execution plan for the statements. Other Parser modules then access the generated steps and send them to one or more AMP subsystems for execution. Parser modules also monitor the execution of the steps, handle errors encountered during processing and return the execution results to the Gateway for return to the Client application.

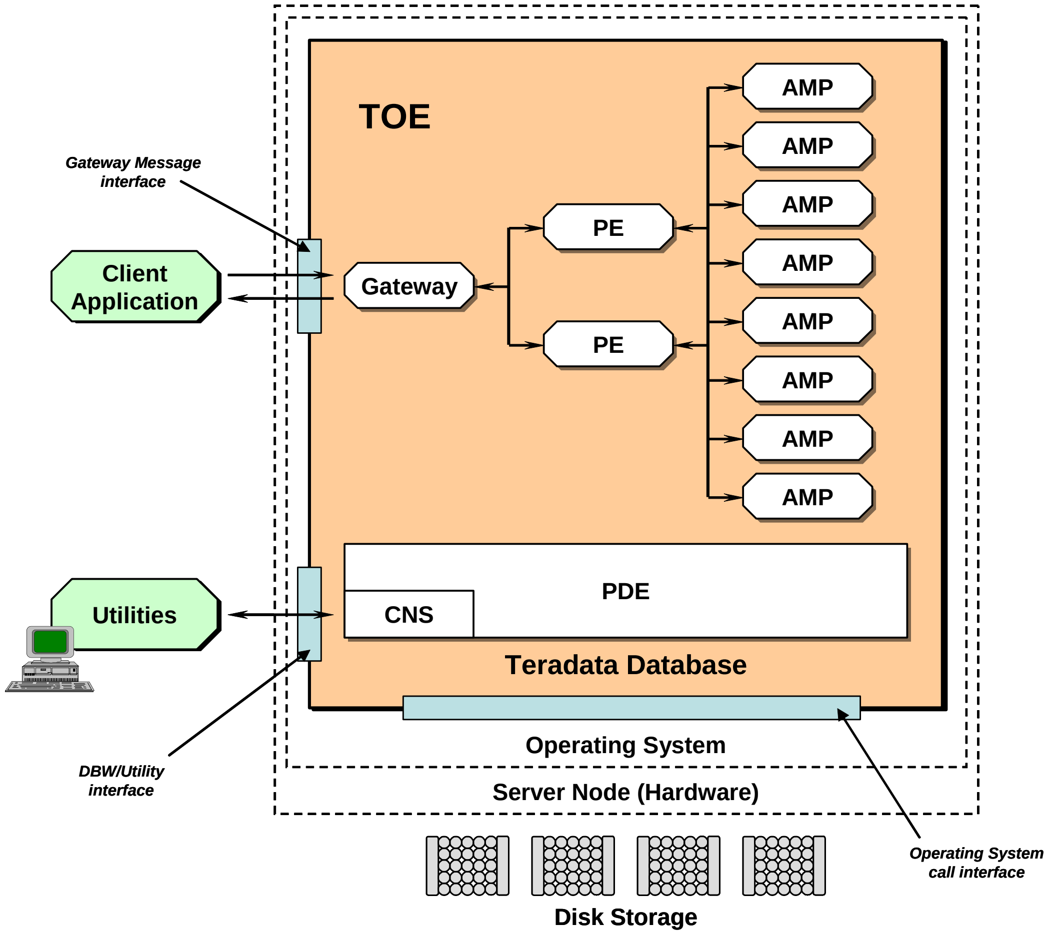
An AMP subsystem physically structures the TOE managed relational data and it processes the steps of an SQL execution plan to access that data. It also manages a set of relational tables containing the description of the user defined data objects. The AMP subsystem provides access to these Data Dictionary tables to Client applications through standard SQL and to other database subsystems as needed and is responsible for the integrity of the relational data structures. The AMP subsystem reads and writes the relational data structures from/to disk storage by making calls to the PDE subsystem which subsequently calls the underlying host operating system to perform the required physical read and write operations.

Other components exist in the Teradata Database environment and interface to the database, but are excluded from the definition of the TOE. These components include:

- The operating system on which the database executes.
- The database server node upon which the database software and underlying operating system operates. (The server node hardware, including processor and memory, are not developed by Teradata.)
- The disk storage subsystem and its associated SCSI or Fibre Channel interface. (The disk storage subsystem hardware is not developed by Teradata.)
- The Console's Database Window (DBW) utilities software.
- The Teradata Tools and Utilities (Client) applications including the Call Level Interface (CLI) software that processes messages sent to, and received from, the database.

The physical boundaries of the TOE are depicted in the Figure 1-1.

Figure 1-1 TOE Physical Boundaries



There are two external user interfaces to the Teradata Database. The Gateway Message interface receives service requests from Client applications and returns responses to the applications upon completion of a service request. The DBW/Utility interface provides for Console access to executable processes of the PDE subsystem. For both interfaces, remote client tools, utilities, and applications send messages to the interface and receive messages from the interface through the standard TCP/IP socket protocol.

The Gateway Message interface is the primary external user interface to the Teradata Database. The interface processes text messages which are generated by a client process. Messages are simply a string of character data consisting of a header and a body. The header of a message identifies the kind of message and its length along with other general information. The body consists of data structured for the kind of message defined in the header. The predominant kind of message is one in which the body contains a service request consisting of a SQL statement and associated data. The Gateway Message interface is used to process such service requests from both end users and authorized administrators.

The DBW/Utility interface is the external user interface to the Teradata Database PDE subsystem to provide for operational control of the server and for output of operational results of the server's execution. Utilities that use this interface are grouped into the following functional categories:

- Installation, configuration, migration, and upgrade
- System administration and maintenance
- Database administration and operation
- Diagnostics and troubleshooting

Utilities using the DBW/Utility interface do not provide any security functions and do not provide any interface to security functions described in this Security Target.

The Teradata Database makes calls to the underlying operating system to access operating system services and to access the associated disk storage subsystem. There is no direct access from the Teradata Database to the underlying hardware - only the operating system accesses the underling hardware.

Note that the TOE is defined as a software-only TOE. As such, the Server Node (Hardware) and Disk Storage is specifically outside the TOE boundary. (The disk storage resides in a separate disk array cabinet that is packaged completely separately from the Server Node hardware. In some very small environments where the Teradata Database may be running on a standalone server platform, the disk storage may be packaged as part of the server platform.)

The Teradata Database is designed with well-defined interfaces that ensure that all appropriate security checks are made before access is provided to protected database objects and resources. The Teradata Database operates as a set of cooperating processes which are managed by the underlying operating system. These processes operate as a parallel application such that no interference is allowed by processes associated with any non-TOE entities. Furthermore, the Teradata Database is designed such that its interfaces do not allow unauthorized users access to database resources.

Note that given the defined TOE physical boundaries, the TOE protection mechanisms could be bypassed through the underlying operational environment and it is assumed that the operational environment provides appropriate protection mechanisms. The hardware and the operating system upon which the TOE operates both contribute to the enforcement of domain separation between the processes and resources allocated to the TOE and processes and resources that may be allocated to other system functions.

The logical boundaries of the TOE are defined by the supported security functions. All five subsystems of the TOE contribute to meeting the security functional requirements.

TOE Access - The Teradata Database allows an authorized security administrator to restrict access to the database based on user identities, *hostid* associated with a network interface, and network (IP) address of the client system.

Identification and Authentication - The Teradata Database provides user identification and authentication through the use of user accounts and the enforcement of password policies. Users must provide a valid username and password before they can access any database objects or resources. Once identified and authenticated, all subsequent actions allowed within that user's session are based on the user's identity, access rights, and active access roles.

Administrator access to database control utilities and other utilities is controlled by a non-TOE component (i.e., the underlying operating system). As such, there is a dependency on the operational environment to provide identification and authentication mechanisms to restrict and control such administrator access.

User Data Protection - The Teradata Database enforces a Discretionary Access Control (DAC) policy for object access based on user identities, object ownership, and active access roles. All access to database objects subject to the DAC policy is controlled using access rights. The Teradata Database supports three types of access rights. Implicit rights (ownership rights) are implicitly granted to the immediate owner of a database or database object. Automatic rights are granted automatically by the system to the creator of a database, user, or object, and to a newly created user or database. Explicit rights are granted by any user having the `WITH GRANT OPTION` privilege for that right. The database ensures that the requestor has the appropriate access rights before access to a database object is allowed.

Upon initial installation of the Teradata Database, it has only one user. This user is called user `DBC` and will own all other databases and users in the system. User `DBC` also has access rights on all objects within the database. For the evaluated configuration, the administrator guidance also requires creating a separate authorized security administrator to perform security-related tasks. Creating an authorized security administrator user under user `DBC` provides protection of sensitive data and system objects owned by user `DBC`.

Security Audit - The Teradata Database automatically audits all successful and failed user logon attempts in the event log. An authorized security administrator may search and sort logon/logoff records using SQL statements to query a defined system view. Additionally, an authorized security administrator may control the monitoring of access rights checks performed by Teradata Database and may search and sort access log records using SQL statements to query a defined system view.

The time stamp used for recording the date and time on which an event is logged is obtained from a non-TOE component (i.e., the underlying operating system). As such, the TOE has a dependency upon the operational environment to provide a reliable time stamp for use by the security audit functions.

Security Management - The Teradata Database provides security management functions that enable an authorized security administrator to manage the secure operation of the database. These functions include management of users, user security attributes, access rights, access roles, and the audit facilities.

Resource Utilization - The Teradata Database enforces maximum quotas and limits on various resources to ensure that those resources are protected from monopolization by any individual database user. Specifically, an authorized security administrator can configure the database to enforce limits on permanent database space allocation, temporary database space usage, and spool database space usage.

2. CONFORMANCE CLAIMS

2.1 COMMON CRITERIA CONFORMANCE

This Security Target conforms to the following Common Criteria specifications:

Common Criteria for Information Technology Security Evaluation
Part 2: Security functional components
September 2007
Version 3.1 Revision 2

Common Criteria for Information Technology Security Evaluation
Part 3: Security assurance components
September 2007
Version 3.1 Revision 2

- EAL 4 augmented with ALC_FLR.3

The Security Target is Common Criteria Part 2 conformant in that all security functional requirements are based only upon functional components in Common Criteria Part 2.

The Security Target is Common Criteria Part 3 conformant in that all security assurance requirements are based only upon assurance components in Common Criteria Part 3.

2.2 PROTECTION PROFILE CLAIMS

This Security Target does not claim conformance to a Protection Profile.

2.3 PACKAGE CLAIMS

This Security Target is conformant with the EAL4 assurance package augmented with ALC_FLR.3.

3. SECURITY PROBLEM DEFINITION

The security problem addressed by this ST is defined by threats (T), organizational security policies (P), and assumptions (A) as described in the following sections.

3.1 THREATS

This section provides a description of threats to the assets against which specific protection within the TOE or its environment is required.

T.ACCOUNTABILITY	The authorized users of the TOE may not be held accountable for their actions within the TOE.
T.ADMIN_ERROR	An administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.
T. AUDIT_ COMPROMISE	A user or process may view audit records, cause audit records to be lost or modified, or prevent future audit records from being recorded, thus masking a user's action.
T.MASQUERADE	A user or process may masquerade as another entity in order to gain unauthorized access to data or TOE resources.
T.RESIDUAL_DATA	A user or process may gain unauthorized access to data through reallocation of TOE resources from one user or process to another.
T.RESOURCE	An authenticated database user might consume excessive database resources such that access to database resources by other database users is compromised.
T.NO_SECADMIN	The TOE may not be configured with an authorized security administrator, separate and distinct from other authorized database administrators, to provide for secure administration of the TOE.
T.TSF_COMPROMISE	A malicious user or process may cause configuration data to be inappropriately accessed (viewed, modified or deleted).
T.UNAUTHORIZED_ACCESS	A user may gain unauthorized access to user data for which they are not authorized according to the TOE security policy.

T.UNIDENTIFIED_ACTIONS	Failure of the authorized security administrator to identify and act upon unauthorized actions may occur.

3.2 ORGANIZATIONAL SECURITY POLICIES

This section provides a description of the organizational security policies, i.e., sets of rules, practices, and procedures, imposed by an organization to address its security needs.

P.ACCOUNTABILITY	The authorized users of the TOE shall be held accountable for their actions within the TOE.
P.SECADMIN	The TOE shall be configured with an authorized security administrator user for secure administration of the TOE. This user shall be separate and distinct from other authorized users.

3.3 ASSUMPTIONS

This section provides a description of assumptions that describe the security aspects of the operational environment in which the TOE will be used or is intended to be used.

A.DOMAIN_SEPARATION	The operational environment will provide a separate domain for the TOE's operation.
A.I_AND_A	It is assumed that the operational environment will provide identification and authentication mechanisms for use of utilities under the control of the operational environment.
A.NO_BYPASS	The operational environment will ensure the TSF cannot be bypassed in order to gain access to TOE data.
A.NO_EVIL	Administrators are non-hostile, appropriately trained, and follow all administrator guidance.
A.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the database server, other than those services necessary for the operation, administration and support of the database.

A.PHYSICAL	It is assumed that appropriate physical security is provided within the domain for the value of the IT assets protected by the TOE and the value of the stored, processed, and transmitted information.
A.RESTRICT_OS_ACCESS	It is assumed that logon access to the underlying operating system is restricted to authorized administrators only.
A.ROBUST_ENVIRONMENT	It is assumed that the operational environment is at least as robust as the TOE.
A.SECURE_COMMS	It is assumed that the operational environment will provide a secure (protected from disclosure, spoofing, and able to detect modification) line of communications between the remote user and the TOE.
A.TIME_STAMPS	It is assumed that the operational environment will provide the TOE with the necessary reliable timestamps.

4. SECURITY OBJECTIVES

The objectives listed in this section are intended to address all identified assumptions and counter all identified threats. The security objectives for the TOE are prefaced with an 'O' and those for the Environment are prefaced with an 'OE'.

4.1 SECURITY OBJECTIVES FOR THE TOE

O.AUDIT_GENERATION	The TOE will provide the capability to detect and create records of security relevant events associated with users.
O.AUDIT_REVIEW	The TOE will contain mechanisms to allow the authorized security administrator to view and sort the audit logs.
O.AUDIT_PROTECTION	The TOE will contain mechanisms to protect the audit log from unauthorized modifications or deletions.
O.I_AND_A	The TOE will contain identification and authentication mechanisms for users to login to the TOE.
O.MANAGE	The TOE will provide all the functions and facilities necessary to support the authorized security administrator in management of the security of the TOE, and restrict these functions and facilities from unauthorized use.
O.MEDIATE	The TOE must protect user data in accordance with its security policy.
O.RESIDUAL_INFORMATION	The TOE will ensure that any information contained in a protected resource within its Scope of Control is not released when the resource is reallocated.
O.RESOURCE	The TOE will provide for limiting the consumption of database resources by authorized users of the TOE.
O.SECADMIN	The TOE will provide for the creation of an authorized security administrator to isolate administrative actions.
O.TOE_ACCESS	The TOE will provide mechanisms that control a user's logical access to the TOE.

4.2 SECURITY OBJECTIVES FOR THE OPERATIONAL ENVIRONMENT

OE.DOMAIN_SEPARATION	The operational environment will provide an isolated domain for the execution of the TOE.
OE_I_AND_A	The operational environment will contain identification and authentication mechanisms for administrator access to database control utilities and other utilities.
OE.NO_BYPASS	The operational environment shall ensure the TOE security mechanisms cannot be bypassed in order to gain access to the TOE resources.
OE.NO_EVIL	Sites using the TOE shall ensure that authorized administrators are non-hostile, appropriately trained and follow all administrator guidance.
OE.CONFIG	The TOE and the underlying operating system will be installed, configured, managed and maintained in accordance with its guidance documentation and applicable security policies and procedures.
OE.NO_GENERAL_ PURPOSE	There will be no general-purpose computing capabilities (e.g., user applications) available on DBMS servers, other than those services necessary for the operation, administration and support of the database.
OE.PHYSICAL	Physical security will be provided within the domain for the value of the IT assets protected by the TOE and the value of the stored, processed, and transmitted information.
OE.RESTRICT_OS_ACCESS	The underlying operating system will be configured with only those user accounts required for access by authorized security administrators.
OE.ROBUST_ENVIRONMENT	The operational environment that supports the TOE for enforcement of its security objectives will be of at least the same level of robustness as the TOE.

OE.SECURE_COMMS	The operational environment will provide a secure line of communications between the remote user and the TOE.
OE.TIME_STAMPS	The operational environment will provide reliable time stamps.
OE.TRUST_IT	Each operational entity the TOE relies on for security functions will be installed, configured, managed and maintained in a manner appropriate to the IT entity, and consistent with the security policy of the TOE and the relationship between them.

4.3 SECURITY OBJECTIVES RATIONALE

This section shows that all secure usage assumptions, policies and threats are completely covered by the security objectives. In addition, each security objective is demonstrated to counter or address at least one assumption, policy or threat.

Table 4-1 Rationale for TOE Security Objectives

Threats/Policies	TOE Security Objectives	Rationale
T.ACCOUNTABILITY	O.AUDIT_GENERATION	O.AUDIT_GENERATION addresses this threat by providing the authorized security administrator with the capability of configuring the audit mechanism to record the actions of a specific user, or review the audit trail based on the identity of the user. Additionally, the authorized security administrator's ID is recorded when any security relevant change is made to the TOE.
	O.I_AND_A	O.I_AND_A ensures that the TOE will contain identification and authentication mechanisms for users to login to the TOE. This user identity is subsequently used by the TOE to identify the user in the audit logs.

Threats/Policies	TOE Security Objectives	Rationale
T.AUDIT_COMPROMISE	O.AUDIT_REVIEW	O.AUDIT_REVIEW ensures that the TOE will provide mechanisms to review the audit logs. These requirements will ensure the data is in a suitable manner for the authorized security administrator to interpret as well as giving the authorized security administrator a way to search and sort within the log to find appropriate data.
	O.AUDIT_PROTECTION	O.AUDIT_PROTECTION ensures the TOE will provide a secure mechanism for protecting the TOE audit log from unauthorized modifications and deletions.
	O.MANAGE	O.MANAGE ensures that the TOE will provide all the functions and facilities necessary to support the authorized security administrator in the management of the security of the audit logs, and restrict these functions and facilities from unauthorized use.
T.MASQUERADE	O.TOE_ACCESS	O.TOE_ACCESS mitigates this threat by controlling the logical access to the TOE and its resources. By constraining how and when authorized users can access the TOE, and by mandating the type and strength of the authentication mechanism this objective helps mitigate the possibility of a user attempting to login and masquerade as an authorized user. In addition, this objective provides the authorized security administrator the means to control the number of failed login attempts a user can generate before an account is locked out, further reducing the possibility of a user gaining unauthorized access to the TOE.
T.RESIDUAL_DATA	O.RESIDUAL_INFORMATION	O.RESIDUAL_INFORMATION counters this threat by ensuring that TSF data and user data is not persistent when resources are released by one user/process and allocated to another user/process.
T.RESOURCE	O.RESOURCE	O.RESOURCE ensures that the TOE provides an authorized security administrator with controls to limit the consumption of database resources by an authorized database user.
T.NO_SECADMIN	O.SECADMIN	The TOE has the objective of providing an authorized security administrator user for secure administration. This is a separate user from other administrative users that the TOE may provide.
T.TSF_COMPROMISE	O.RESIDUAL_INFORMATION	O.RESIDUAL_INFORMATION is necessary to mitigate this threat, because even if the security mechanisms do not allow a user to explicitly view TSF data, if TSF data were to reside inappropriately in a resource that was made available to a user, that user would be able to view the TSF data without authorization.
	O.MANAGE	O.MANAGE is necessary because an access control policy is specified to control access to TSF data. This objective is used to dictate who is able to view and modify TSF data, as well as the behavior of TSF functions.

Threats/Policies	TOE Security Objectives	Rationale
T.UNAUTHORIZED_ACCESS	O.MEDIATE	O.MEDIATE ensures that all accesses to user data are subject to mediation, unless said data has been specifically identifies as public data. The TOE requires successful authentication to the TOE prior to gaining access to any controlled-access content. Lastly, the TSF will ensure that all configured enforcement functions (authentication, access control rules, etc.) must be invoked prior to allowing a user to gain access to TOE or TOE mediated services. The TOE restricts the ability to modify the security attributes associated with access control rules, access to authenticated and unauthenticated services, etc. to the authorized security administrator.
T.UNIDENTIFIED_ACTIONS	O.MANAGE	The threat of an authorized security administrator failing to know about malicious audit events produces the O.MANAGE objectives of the authorized security administrator having the capability to specify rules to audit events and to access and review audit records.
P.ACCOUNTABILITY	O.AUDIT_GENERATION	O.AUDIT_GENERATION addresses this policy by providing the authorized security administrator with the capability of configuring the audit mechanism to record the actions of a specific user, or review the audit trail based on the identity of the user. Additionally, the authorized security administrator's ID is recorded when any security relevant change is made to the TOE (e.g., access rule modification, start-stop of the audit mechanism, etc.).
	O.TOE_ACCESS	O.TOE_ACCESS supports this policy by requiring the TOE to identify and authenticate all authorized users prior to allowing any TOE access or any TOE mediated access on behalf of those users.
P.SECADMIN	O.SECADMIN	O. SECADMIN addresses this policy by ensuring that the TOE has been configured with an authorized security administrator user for secure administration. The TOE may provide other administrative users or roles as well, but only the authorized security administrator is required.

Table 4-2 Rationale for Operational Environmental Objectives

Threats/Policies/ Assumptions	Operational Environmental Objectives	Rationale
T.ACCOUNTABILITY	OE.TIME_STAMPS	OE.TIME_STAMPS plays a role in addressing this threat by requiring the IT Environment to provide a reliable time stamp. The audit mechanism is required to include the current date and time in each audit record. All audit records that include the user ID, will also include the date and time that the event occurred.

Threats/Policies/ Assumptions	Operational Environmental Objectives	Rationale
T.ADMIN_ERROR	OE.CONFIG	Authorized administrators are trusted to properly configure the TOE so it enforces its security policies.
A.DOMAIN_SEPARATION	OE.DOMAIN_SEPARATION	OE.DOMAIN_SEPARATION ensures the operational environment will provide an isolated domain for the TOE's execution.
A.I_AND_A	OE.I_AND_A	OE.I_AND_A ensures the operational environment will provide mechanisms for administrators to be authenticated before any database control utilities and other utilities used to manage system resources and I/O interfaces may be used.
A.NO_BYPASS	OE.NO_BYPASS	OE.NO_BYPASS ensures the TOE cannot be bypassed in order to gain unauthorized access of TOE resources.
A.NO_EVIL	OE.NO_EVIL	All authorized administrators are trustworthy individuals, having background investigations commensurate with the level of data being protected, have undergone appropriate admin training, and follow all admin guidance.
A.NO_GENERAL_PURPOSE	OE.NO_GENERAL_PURPOSE	The DBMS server must not include any general-purpose computing or storage capabilities. This will protect the TSF data from malicious processes.
A.PHYSICAL	OE.PHYSICAL	The TOE, the TSF data, and protected user data is assumed to be protected from physical attack (e.g., theft, modification, destruction, or eavesdropping). Physical attack could include unauthorized intruders into the TOE environment, but it does not include physical destructive actions that might be taken by an individual that is authorized to access the TOE environment.
A.RESTRICT_OS_ACCESS	OE.RESTRICT_OS_ACCESS	The underlying operating system running on the DBMS server must include only those user accounts required by authorized administrators. Restricting access to the operating system protects against tampering by malicious users.
A.ROBUST_ENVIRONMENT	OE.ROBUST_ENVIRONMENT OE.TRUST_IT	The TOE shall only be installed in an operational environment that is at least as robust as the TOE. The IT entities in the environment are correctly installed, configured, managed and maintained.

Threats/Policies/ Assumptions	Operational Environmental Objectives	Rationale
A.SECURE_COMMS	OE.SECURE_COMMS	OE.SECURE_COMMS states that the environment must provide a secure line of communication for transfer of TSF data. This is necessary because access to the TOE may be distributed geographically with users and authorized administrators in different locations. The objective OE.SECURE_COMMS does not necessarily mandate that the communications between the remote user or administrator and the TOE be encrypted.
A.TIME_STAMPS	OE.TIME_STAMPS	OE.TIME_STAMPS states that the environment will maintain reliable timestamps and those will be used by the TOE to stamp each audit record with a date and time.

5. EXTENDED COMPONENTS DEFINITION

There are no extended components defined in this Security Target.

6. SECURITY REQUIREMENTS

This section identifies the security functional requirements for the TOE and its' operational environment. In addition, this section also presents the security assurance requirements for the TOE. The operations performed on the security functional and assurance requirements contained in this section adhere to the following conventions:

- Iteration: Allows a component to be used more than once with varying operations. In the ST, a number in parenthesis appended to a component indicates iteration.
- Assignment: Allows the specification of an identified parameter. Assignments are indicated using italicized text and are surrounded by brackets (e.g., [*assignment*]).
- Selection: Allows the specification of one or more elements from a list. Selections are indicated using bold italicized text and are surrounded by brackets (e.g., [***selection***]).
- Refinement: Allows the addition of details. Refinements are indicated using bold text for additions to the requirements (e.g., **refinement**).

All of the security functional requirements are from Common Criteria Part 2 and there are no extended requirements defined in this ST.

6.1 SECURITY FUNCTIONAL REQUIREMENTS

The following table provides a summary of the security functional requirements implemented by the TOE.

Table 6-1 TOE Security Functional Requirements

Security Functional Class	Security Functional Requirement
Security Audit (FAU)	FAU_GEN.1 Audit data generation
	FAU_GEN.2 User identity association
	FAU_SAR.1 Audit review
	FAU_SAR.2 Restricted audit review
	FAU_SAR.3 Selectable audit review
	FAU_SEL.1 Selective audit
	FAU_STG.1 Protected audit trail storage
User Data Protection (FDP)	FDP_ACC.1 Subset access control
	FDP_ACF.1 Security attribute based access control
	FDP_RIP.1 Subset residual information protection
Identification and Authentication (FIA)	FIA_AFL.1 Authentication failure handling
	FIA_ATD.1 User attribute definition
	FIA_SOS.1 Verification of secrets

Security Functional Class	Security Functional Requirement
	FIA_UAU.1 Timing of authentication
	FIA_UID.1 Timing of identification
	FIA_USB.1 User-subject binding
Security Management (FMT)	FMT_MOF.1 Management of security functions behavior
	FMT_MSA.1 Management of security attributes
	FMT_MSA.3 Static attribute initialization
	FMT_MTD.1 Management of TSF data
	FMT_REV.1 Revocation
	FMT_SMF.1 Specification of management functions
	FMT_SMR.1 Security roles
Resource Utilisation (FRU)	FRU_RSA.1 Maximum quotas
TOE Access (FTA)	FTA_TSE.1 TOE session establishment

The following subsections present the security functional requirements for the TOE.

6.1.1 Class FAU: Security Audit

6.1.1.1 FAU_GEN.1 Audit data generation

Hierarchical to: No other components.

Dependencies: FPT_STM.1 Reliable time stamps

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;
b) All auditable events for the [***not specified***] level of audit; and
c) [*Events specified in Table 6-2*].

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*other audit relevant information, as provided under “Additional Data” in Table 6-2*]

Table 6-2 Auditable Events

Component	Event	Additional Data
FAU_SAR.1	Reading of information from the database audit records	None

Component	Event	Additional Data
FAU_SEL.1	All modifications to the database audit configuration that occur while the database audit collection functions are operating	Modified configuration element
FDP_ACF.1	All requests to perform an operation on a database object covered by the SFP	The assigned user access right
FIA_UAU.1	All use of the user authentication mechanism	None
FIA_UID.1	All use of the user identification mechanism, including the user identity provided	None
FIA_USB.1	Success or failure of binding user security attributes to a database subject (e.g., success and failure to create a database subject)	None
FMT_MOF.1	Modifications in the behavior of the functions of the TSF	Change of threshold for unsuccessful authentication attempts or actions to be taken in the event of an authentication failure
FMT_MSA.1	All modifications of the values of database security attributes	Modification, deletion or addition of database security attributes
FMT_MTD.1	All modifications to the values of TSF data	None
FMT_REV.1	All attempts to revoke database security attributes	None
FMT_SMR.1	Modifications to the group of users that are part of a role	None
FTA_TSE.1	All attempts at establishment of a user session	None

6.1.1.2 FAU_GEN.2 User identity association

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation
FIA_UID.1 Timing of identification

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

6.1.1.3 FAU_SAR.1 Audit review

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation

FAU_SAR.1.1 The TSF shall provide [*the authorized security administrator*] with the capability to read [*all audit information*] from the audit records.

FAU_SAR.1.2 The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

6.1.1.4 FAU_SAR.2 Restricted audit review

Hierarchical to: No other components.

Dependencies: FAU_SAR.1 Audit review

FAU_SAR.2.1 The TSF shall prohibit all users read access to the audit records, except those users that have been granted explicit read-access.

6.1.1.5 FAU_SAR.3 Selectable audit review

Hierarchical to: No other components.

Dependencies: FAU_SAR.1 Audit review

FAU_SAR.3.1 The TSF shall provide the ability to apply [*searches, sorting*] of audit data based on [*all attributes contained within the audit records*].

6.1.1.6 FAU_SEL.1 Selective audit

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation
FMT_MTD.1 Management of TSF data

FAU_SEL.1.1 The TSF shall be able to select the set of audited events from the set of all auditable events based on the following attributes:

a) [***object identity, user identity, event type***]

b) [*success or failure of access, frequency of access*].

6.1.1.7 FAU_STG.1 Protected audit trail storage

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorized deletion **for actions within the TOE Scope of Control**.

FAU_STG.1.2 The TSF shall be able to [***prevent***] unauthorized modifications to the stored audit records in the audit trail **when attempts to modify audit records occur within the TOE Scope of Control**.

6.1.2 Class FDP: User Data Protection

6.1.2.1 FDP_ACC.1 Subset access control

Hierarchical to: No other components.

Dependencies: FDP_ACF.1 Security attribute based access control

FDP_ACC.1.1 The TSF shall enforce the [*Discretionary Access Control policy*] on [*all subjects (users), all DBMS-controlled objects (DATABASE, USER, TABLE, INDEX, VIEW, TRIGGER, MACRO, PROCEDURE, EXTERNAL PROCEDURE, FUNCTION, TYPE, GLOP, REPLICATION GROUP, AUTHORIZATION, ROLE, PROFILE) and all operations among them*].

6.1.2.2 FDP_ACF.1 Security attribute based access control

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialization

FDP_ACF.1.1 The TSF shall enforce the [*Discretionary Access Control policy*] to objects based on the following: [*database subject attributes: user identity, active access roles; database object attributes: object owner and access rights granted on the object*].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [

a) *If the authorized user associated with the subject is the owner of the object, then the requested access is allowed; or*

b) *If the authorized user associated with the subject has the object access right for the requested access to the object, then the requested access is allowed; or*

c) *If the authorized user associated with the subject is the member of an active access role or nested access role which has the object access right for the requested access to the object, then the requested access is allowed; or*

d) *If `PUBLIC` has the object access right for the requested access to the object, then the requested access is allowed; or*

e) *Otherwise, the access is denied*].

Note: `PUBLIC` is a special internal user provided by the Teradata Database. Access rights granted to `PUBLIC` are applicable to every valid user of the system and all future users of the system.

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [*no additional rules*].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the [*no additional rules*].

6.1.2.3 FDP_RIP.1 Subset residual information protection

Hierarchical to: No other components.

Dependencies: No dependencies

FDP_RIP.1.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the [***allocation of the resource to***] the following objects: [*database tables, database indexes*].

6.1.3 Class FIA: Identification and Authentication

6.1.3.1 FIA_AFL.1 Authentication failure handling

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication

FIA_AFL.1.1 The TSF shall detect when [***an authorized security administrator configurable positive integer within*** [*3 - 127*]] unsuccessful authentication attempts occur related to [*the last successful authentication for the indicated user identity*].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been [***met***], the TSF shall [*disable the account until unlocked by the authorized security administrator or until a configurable number of minutes have elapsed*].

6.1.3.2 FIA_ATD.1 User attribute definition

Hierarchical to: No other components.

Dependencies: No dependencies

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: [*user identity, authentication data, access roles as defined by an authorized security administrator, profile*].

Note: Each database user is associated with a user object that is identified by a username. This username represents the user identity.

Note: A profile is a database object that defines a set of system parameters that can be assigned to a group of database users. A profile, if assigned to a database user, may contain user-specific password control attributes. Other system parameters that may be specified by a profile are account string, default database, spool and temporary space allocations, and cost profile.

6.1.3.3 FIA_SOS.1 Verification of secrets

Hierarchical to: No other components.

Dependencies: No dependencies

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets meet [*the following requirements:*

- *Passwords will be restricted to a minimum and maximum number of characters in length,*
- *Passwords will contain a combination of upper and lower case characters,*
- *Passwords will contain at least one numeric character,*
- *Passwords will contain at least one special character,*
- *Passwords will not contain the user's username,*
- *Passwords will not contain a restricted (dictionary) word,*
- *Passwords will be valid for a maximum number of days before expiration,*
- *Previously used passwords may not be re-used for a minimum number of days*].

6.1.3.4 FIA_UAU.1 Timing of authentication

Hierarchical to: No other components.

Dependencies: FIA_UID.1 Timing of identification

FIA_UAU.1.1 The TSF shall allow [*establishment of a virtual circuit for the purpose of transferring authentication information, receipt of error messages upon authentication failure*] on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

6.1.3.5 FIA_UID.1 Timing of identification

Hierarchical to: No other components.

Dependencies: No dependencies

FIA_UID.1.1 The TSF shall allow [*establishment of a virtual circuit for the purpose of transferring identification information, receipt of error messages upon identification failure*] on behalf of the user to be performed before the user is identified.

FIA_UID.1.2 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.3.6 FIA_USB.1 User-subject binding

Hierarchical to: No other components.

Dependencies: FIA_ATD.1 User attribute definition

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: [*user identity and active access roles*].

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [*subject security attributes are derived from TSF data maintained for each defined user after a successful login with the defined user identity*].

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [*a user can set the active access role to any or all access roles assigned to them by an authorized security administrator*].

6.1.4 Class FMT: Security Management

6.1.4.1 FMT_MOF.1 Management of security functions behavior

Hierarchical to: No other components.

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MOF.1.1 TSF shall restrict the ability to [***modify the behavior of***] the functions [*relating to the specification of events to be audited*] to [*the authorized security administrator*].

6.1.4.2 FMT_MSA.1 Management of security attributes

Hierarchical to: No other components.

Dependencies: [FDP_ACC.1 Subset access control or
FDP_IFC.1 Subset information flow control]
FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MSA.1.1(1) The TSF shall enforce the [*Discretionary Access Control policy*] to restrict the ability to [*grant or revoke*] the security attributes [*database object access rights*] to [*authorized users*].

FMT_MSA.1.1(2) The TSF shall enforce the [*Discretionary Access Control policy*] to restrict the ability to [*create, drop, grant, or revoke*]] the security attributes [*access roles as defined by an authorized security administrator*] to [*the authorized security administrator*].

6.1.4.3 FMT_MSA.3 Static attribute initialization

Hierarchical to: No other components.

Dependencies: FMT_MSA.1 Management of security attributes
FMT_SMR.1 Security roles

FMT_MSA.3.1 The TSF shall enforce the [*Discretionary Access Control policy*] to provide [***restrictive***] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the [*no identified roles*] to specify alternative initial values to override the default values when an object or information is created.

6.1.4.4 FMT_MTD.1 Management of TSF data

Hierarchical to: No other components.

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MTD.1.1(1) The TSF shall restrict the ability to [[*create, drop*]] the [*user identities*] to [*the authorized security administrator*].

FMT_MTD.1.1(2) The TSF shall restrict the ability to [[*create, drop, grant, revoke*]] the [*access roles as defined by an authorized security administrator*] to [*the authorized security administrator*].

FMT_MTD.1.1(3) The TSF shall restrict the ability to [***change_default***] the [*authentication data*] to [*the authorized security administrator*].

FMT_MTD.1.1(4) The TSF shall restrict the ability to [***modify***] the [*authentication data*] to [*the authorized security administrator and users authorized to modify their own authentication data*].

FMT_MTD.1.1(5) The TSF shall restrict the ability to [[*grant, revoke*]] the [*database object access rights*] to [*the authorized user creating or immediately owning the database object and an authorized user explicitly granted the access rights WITH GRANT OPTION*].

FMT_MTD.1.1(6) The TSF shall restrict the ability to [***change_default, modify,***] the [*audit rules*] to [*the authorized security administrator*].

FMT_MTD.1.1(7) The TSF shall restrict the ability to [***delete, query***] the [*audit records*] to [*the authorized security administrator*].

FMT_MTD.1.1(8) The TSF shall restrict the ability to [***change_default, modify, delete***] the [*maximum quotas*] to [*the authorized security administrator and the authorized database administrator*].

Note: The Discretionary Access Control policy, as described in Section 7.3, controls how database object access rights are enforced. The FMT_MTD.1.1(5) iteration addresses the management of the DAC policy by specifying the functional requirement for management of the grants and revokes of the database object access rights which are enforced by the DAC policy.

6.1.4.5 FMT_REV.1 Revocation

Hierarchical to: No other components.

Dependencies: FMT_SMR.1 Security roles

FMT_REV.1.1 The TSF shall restrict the ability to revoke [*database object access rights*] associated with the [***users, objects***] under the control of the TSF to [*authorized users (only for the database objects they own or database* *objects for which they have been granted database object access rights allowing them to revoke security attributes)*].

FMT_REV.1.2 The TSF shall enforce the rules [*revocation of database object access rights shall affect all subsequent attempts to access the database object*].

6.1.4.6 FMT_SMF.1 Specification of management functions

Hierarchical to: No other components.

Dependencies: No dependencies

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [*beginning and ending the audit function, selection of the audited events, review of audit data, management of database users and authentication data, management of database object access rights, management of access roles, and management of maximum quotas*].

6.1.4.7 FMT_SMR.1 Security roles

Hierarchical to: No other components.

Dependencies: FIA_UID.1 Timing of identification

FMT_SMR.1.1 The TSF shall maintain the roles [

a) DBC,
b) authorized database administrators,
c) authorized security administrator,
d) authorized user,
e) access roles as defined by an authorized security administrator, and
f) users authorized to modify their own authentication data].

Note: There is a difference in terminology between CC Part 2 and the Teradata Database regarding the use of the word "role" in FMT_SMR.1.1. The first usage (e.g., a), b), c), and d) above), which is part of the CC Part 2 requirement, generally refers to specific database roles as types of users that are created within the TSF and authorized to perform security management functions . The second usage (e.g., e) above) specifically refers to the use of Teradata Database ROLE objects that can be created and granted to database users. In this context, the role defines a collection of access rights. Such roles are commonly used to manage user access security for groups of users. The term "access role" will be used throughout this Security Target to refer to this type of role object.

Note: The authorized user role applies to any database user that has been properly authenticated and authorized. This role is included because, per the Discretionary Access Control policy, any authorized user can grant and revoke access rights on any database objects created or owned by that user.

FMT_SMR.1.2 The TSF shall be able to associate users with roles.

6.1.5 Class FRU: Resource Utilization

6.1.5.1 FRU_RSA.1 Maximum quotas

Hierarchical to: No other components.

Dependencies: No dependencies

FRU_RSA.1.1 The TSF shall enforce maximum quotas of the following resources: [*permanent database space allocation, temporary database space allocation, and spool database space usage for a specified job*] that [***individual user***] can use [***simultaneously***].

6.1.6 Class FTA: TOE Access

6.1.6.1 FTA_TSE.1 TOE session establishment

Hierarchical to: No other components.

Dependencies: No dependencies

FTA_TSE.1.1 The TSF shall be able to deny session establishment based on [*user identity, hostid, client system network address*].

6.2 SECURITY ASSURANCE REQUIREMENTS

This section identifies the security assurance requirements that are met by the TOE. These assurance requirements conform to the CC Part 3 requirements for EAL4 augmented with ALC_FLR.3 and are identified in the following table.

Table 6-3 TOE Security Assurance Requirements

Security Assurance Class	Security Assurance Component
ASE: Security Target evaluation	ASE_INT.1 ST introduction
	ASE_CCL.1 Conformance claims

Security Assurance Class	Security Assurance Component
	ASE_SPD.1 Security problem definition
	ASE_OBJ.2 Security objectives
	ASE_ECD.1 Extended components definition
	ASE_REQ.2 Derived security requirements
	ASE_TSS.1 TOE summary specification
ADV: Development	ADV_ARC.1 Security architecture description
	ADV_FSP.4 Complete functional specification
	ADV_IMP.1 Implementation representation of the TSF
	ADV_TDS.3 Basic modular design
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures
ALC: Life cycle support	ALC_CMC.4 Production support, acceptance procedures and automation
	ALC_CMS.4 Problem tracking CM coverage
	ALC_DEL.1 Delivery procedures
	ALC_DVS.1 Identification of security measures
	ALC_FLR.3 Systematic flaw remediation
	ALC_LCD.1 Developer defined life-cycle model
	ALC_TAT.1 Well-defined development tools
ATE: Tests	ATE_COV.2 Analysis of coverage
	ATE_DPT.2 Testing: security enforcing modules
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing – sample
AVA: Vulnerability assessment	AVA_VAN.3 Focused vulnerability analysis

6.3 SECURITY REQUIREMENTS RATIONALE

Table 6-4 demonstrates the mapping of Security Requirements to TOE Security Objectives. Rationale for each mapping is included in the table.

Table 6-4 Rationale for TOE Security Requirements

Security Objective	Security Requirements	Rationale
O.AUDIT_GENERATION	FAU_GEN.1	FAU_GEN.1 defines the set of events that the TOE must be capable of recording. This requirement ensures that the authorized security administrator has the ability to audit any security relevant events that takes place in the TOE. This requirement also defines the information that must be contained in the audit record for each auditable event.
	FAU_GEN.2	FAU_GEN.2 ensures that the audit records associate a user identity with the auditable event. In the case of authorized users, the association is accomplished with the userid.
	FAU_SEL.1	FAU_SEL.1 allows the authorized security administrator to configure which auditable events will be recorded in the audit trail. This provides the authorized security administrator with the flexibility in recording only those events that are deemed necessary by site policy, thus reducing the amount of resources consumed by the audit mechanism.
	FIA_USB.1	FIA_USB.1 requires that all subjects that act on behalf of users must have a binding that associates the subjects with a user. This is necessary to be able to associate audit records with user identities.
O.AUDIT_REVIEW	FAU_SAR.1	FAU_SAR.1 requires that only the authorized security administrator has the capability to read the audit records which must be presented in a manner suitable for the authorized security administrator to interpret them.
	FAU_SAR.2	FAU_SAR.2 prohibits all other users read access of the audit records.
	FAU_SAR.3	FAU_SAR.3 requires the TOE to provide a mechanism for the authorized security administrator to search and sort through the audit records.
O.AUDIT_PROTECTION	FAU_STG.1	FAU_STG.1 requires that only the authorized security administrator may delete the audit records ensuring that no malicious users may compromise the data stored within the audit records.
	FMT_MTD.1	FMT_MTD.1.1(7) allows only the authorized security administrator to query the audit logs and delete the audit log records.
	FMT_SMF.1	FMT_SMF.1 lists the mechanisms available to the authorized security administrator for managing the audit records.

Security Objective	Security Requirements	Rationale
O.I_AND_A	FIA_AFL.1	FIA_AFL.1 requires the TOE to provide a mechanism for the authorized security administrator to restrict the number of unsuccessful authentication attempts allowed before a user account is locked.
	FIA_ATD.1	FIA_ATD.1 requires the TOE to maintain user identities and passwords belonging to individual users.
	FIA_SOS.1	FIA_SOS.1 requires the TOE to enforce rules requiring the construction of strong passwords and to prevent brute-force password attacks.
	FIA_UAU.1 FIA_UID.1	FIA_UAU.1 and FIA_UID.1 require the TOE to successfully identify and authenticate a user before establishing a session on behalf of the user.
	FIA_USB.1	FIA_USB.1 requires that all subjects that act on behalf of users must have a binding that associates the subjects with a user uniquely.
O.MANAGE	FMT_MOF.1	FMT_MOF.1 requires that the ability to use particular TOE capabilities be restricted to the authorized security administrator.
	FMT_MSA.1	FMT_MSA.1 requires that the ability to perform operations on security attributes be restricted to particular roles. FMT_MSA.1.1(1) provides for restrictions on granting and revoking of access rights to authorized users. FMT_MSA.1.1(2) provides for the restriction on management of access roles to the authorized security administrator.
	FMT_MSA.3	FMT_MSA.3 requires that default values used for security attributes are restrictive.
	FMT_MTD.1	FMT_MTD.1 requires that the TOE restrict the ability to manipulate TSF data to the authorized security administrator. FMT_MTD.1.1(1) provides for restrictions on management of users. FMT_MTD.1.1(2) provides for restrictions on management of access roles. FMT_MTD.1.1(3) and FMT_MTD1.1(4) provide for restrictions on management of authentication data. FMT_MTD1.1(5) provides for restrictions on management of database object access rights. FMT_MTD.1.1(6) and FMT_MTD.1.1(7) provide for restrictions on management of audit log rules and audit data. FMT_MTD.1.1(8) provides for restrictions on management of maximum quota.
O.MANAGE (continued)	FMT_REV.1	FMT_REV.1 restricts the ability to revoke attributes to authorized users.
	FMT_SMF.1	FMT_SMF.1 identifies the management functions that are available to an authorized security administrator.
	FMT_SMR.1	FMT_SMR.1 defines the specific database and access roles to be supported.

Security Objective	Security Requirements	Rationale
O.MEDIATE		The FDP requirements were chosen to define the policies, the subjects, objects, and operations for how and when mediation takes place in the TOE.
	FDP_ACC.1	FDP_ACC.1 defines the Access Control policy that will be enforced on a list of subjects acting on the behalf of users attempting to gain access to a list of named objects. All the operation between subject and object covered are defined by the TOE's policy.
	FDP_ACF.1	FDP_ACF.1 defines the security attribute used to provide access control to objects based on the TOE's access control policy.
O.RESIDUAL_INFORMATION	FDP_RIP.1	FDP_RIP.1 is used to ensure the contents of resources are not available to subjects other than those explicitly granted access to the data.
O.RESOURCE	FRU_RSA.1	FRU_RSA.1 requires that the TOE provide controls to limit the consumption of database resources by authorized database users.
O.SECADMIN	FMT_SMR.1	FMT_SMR.1 requires that the TOE will establish, at least, an authorized security administrator user. The authorized security administrator will be given rights to perform certain tasks that other users will not be able to perform. These rights include, but are not limited to, access to audit information and security functions.
O.TOE_ACCESS	FIA_ATD.1	FIA_ATD.1 defines the attributes of users, including a user ID that is used by the TOE to determine a user's identity and enforce what type of access the user has to the TOE.
	FIA_USB.1	FIA_USB.1 ensures that all subjects that act on behalf of users will have a binding that associates the subjects with a user uniquely.
	FTA_TSE.1	FTA_TSE.1 allows the TOE to restrict access to the TOE based on certain criteria.

The target assurance level is EAL4 augmented with ALC_FLR.3. EAL4 is chosen for the evaluation of the TOE because the TOE is designed for use in commercial and government environments where EAL4 assurance is required to demonstrate that the Teradata Database security functionality has been methodically designed, tested, and reviewed and where users require a moderate to high level of independently assured security.

ALC_FLR.3 has been included to provide for the further evaluation of the TOE's flaw remediation procedures. These procedures are required to provide continuing assurance that potential security flaws detected in the TOE can be reported, properly corrected, and fixes distributed in a timely manner as necessary to protect sensitive or business critical information assets maintained in the RDBMS.

7. TOE SUMMARY SPECIFICATION

This chapter describes the high-level specification of each TOE Security Function (TSF) that contributes to satisfaction of the SFRs presented in Chapter 6.

Each of the following subsections describes a security function of the Teradata Database. For each security function, details are provided that substantiate how the Teradata Database meets the security function, and ensures that no function can be subverted or bypassed without being traced. At the end of each subsection, the SFRs that are satisfied by the TSF are listed, and when it provides added clarity, short additional details specific to the TSF are provided.

7.1 TOE ACCESS

Users are identified using a username. Also, a network interface through which client systems connect to the Teradata Database must have its own unique identifier known as a *hostid*. The database grants logon permission to all users from all *hostid*s by default. However, the authorized security administrator can control which users have access to the database by granting or revoking logons for specific usernames on specific *hostid*s. Therefore, it is possible to deny session establishment based upon user identity and *hostid*.

The Teradata Database also supports a feature to restrict logons by network (IP) address. The authorized security administrator may configure a set of IP filters which can be used to allow or deny a user logon based upon the network address of the client system from which the user is initiating the logon.

The TOE Access function satisfies the following security functional requirements:

- FTA_TSE.1 TOE session establishment – The TOE satisfies this requirement by allowing the establishment of a session to be denied based upon the user's identity, *hostid*, and client system network address.

7.2 IDENTIFICATION AND AUTHENTICATION

When a user attempts to logon to the Teradata Database, a virtual circuit is established to facilitate the transfer of identification and authentication information between the client and the database and to facilitate the transfer of error messages - such as notification of an expired password or authentication failure - to the client. These are the only TSF-mediated actions performed on behalf of a user prior to the user being identified and authenticated.

The Teradata Database Data Dictionary is composed of tables and defined system views that reside in the system user DBC. The tables are reserved for use by the system and contain information, called metadata, about the data in the system. The defined system views are actually virtual tables that are used to provide access to the information in one or more underlying Data Dictionary tables. A unique set of security attributes for each user (including a username and password) is maintained within Data Dictionary tables. Optionally, these security attributes can include a default access role and profile.

The Teradata Database provides several configurable controls related to password authentication. The system default password controls are maintained in Data Dictionary tables. Optionally, the authorized security administrator may assign a profile to a user that specifies a different set of password controls. If a user has a profile assigned, then any password controls specified by the profile override the corresponding system default controls. The following configurable controls combine to satisfy the requirements regarding passwords:

- The minimum and maximum number of characters required for a valid password
- The requirement that a valid password contain a combination of upper and lower case characters, at least one numeric character, and at least one special character
- The requirement that a valid password may not contain the user's username
- The requirement that a valid password may not contain a restricted (dictionary) word
- The maximum number of days to elapse before a password expires
- The minimum number of days to elapse before a previously used password may be re-used

Note: Appendix B of the Teradata Database Security Administration reference manual provides guidelines that must be followed "to operate the system at a level of security equivalent to the Common Criteria evaluated configuration." This guidance requires that the default password control policy be configured as follows:

- minimum number of characters required in a valid password string - PasswordMinChar = 8 characters
- maximum number of characters allowed in a valid password string - PasswordMaxChar = 30 characters
- digits required in password - PasswordDigits = 'r'
- alpha and special characters required in password - PasswordSpecChar = 'r'
- username not allowed in password - PasswordSpecChar = 'r'
- restricted (dictionary) words not allowed in password - PasswordRestrictWords='Y'
- number of erroneous sequential logon attempts a user is allowed before the user is locked to further logon attempts – MaxLogonAttempts = 3 attempts
- user lock time duration after the user has exceeded the maximum number of logon attempts - LockedUserExpire = 5 minutes
- time span during which a password is valid - ExpirePassword = 90 days
- time span during which a password may not be reused - PasswordReuse = 270 days

Note: Appendix I of the Teradata Database Security Administration reference manual provides the default list of restricted (dictionary) words that cannot be included in a valid password.

To logon to the Teradata Database, a user provides a username and password. The database verifies that the username and password are valid by comparing them to the corresponding security attributes stored in the Data Dictionary. If either the username or password is not valid, then the logon will fail and no TSF-mediated actions will be performed on behalf of the user.

If the user exceeds the maximum number of failed logon attempts, then the database will prevent further logon attempts by the user until the applicable password lockout time has elapsed. The guidance provided to the authorized security administrator for the evaluated configuration requires configuring the password lockout time control such that the user remains locked for a period of 5 minutes.

Upon successful identification and authentication of a user, the Teradata Database establishes a binding between the user identity and the established database session. When a session is established, the session's active access role is set to the default access role indicated by the user's security attributes if a default access role has been granted by the authorized security administrator. If no access role has been specified as the default access role, then no access role is made active for the session. During a session, a user may set the active access role to any other access role, or `ALL` access roles, that have been granted to the user by the authorized security administrator. Also during a session, a user may change his or her password in accordance with the password control policy.

In addition to database authentication of user passwords, the Teradata Database also supports external authentication of users via LDAP, Kerberos, and SPNEGO mechanisms as discussed in the Teradata Database Security Administration reference manual. However, the use of any of these mechanisms requires additional configuration of either the TDGSS and/or Kerberos subsystems. The guidance provided in Appendix B of the Teradata Database Security Administration reference manual for configuring the system to the Common Criteria evaluated configuration does not include the additional configuration of either the TDGSS or Kerberos subsystems rendering the use of external authentication effectively disabled in the evaluated configuration. Consequently, this Security Target makes no claims regarding the support of external authentication.

Administrator access to the Console's Database Window (DBW) utilities is controlled by the underlying operating system. As such, there is a dependency on the operational environment to provide identification and authentication mechanisms to control access to these utilities under the control of the operating system.

The Identification and Authentication security function satisfies the following security functional requirements:

- FIA_AFL.1 Authentication failure handling – The TOE satisfies this requirement by allowing an authorized security administrator to define the maximum number of failed login attempts allowed before the user account is locked.
- FIA_ATD.1 User attribute definition – The TOE satisfies this requirement by maintaining an association between users, passwords, access roles, and profiles.
- FIA_SOS.1 Verification of secrets – The TOE satisfies this requirement by allowing an authorized security administrator to define rules required for construction of a valid password.
- FIA_UAU.1 Timing of authentication – The TOE satisfies this requirement by ensuring that, after establishment of a virtual circuit, a user is properly authenticated before allowing any other TSF-mediated actions on behalf of that user.

- FIA_UID.1 Timing of identification – The TOE satisfies this requirement by ensuring that, after establishment of a virtual circuit, a user is properly identified before allowing any other TSF-mediated actions on behalf of that user.
- FIA_USB.1 User-subject binding – The TOE satisfies this requirement by associating the user's security attributes with the user's session.

7.3 USER DATA PROTECTION

The Teradata Database implements a discretionary access control (DAC) policy defined as a means of restricting access to database objects based on the identity of database users (subjects) and database object access rights held by the database users (subjects). The controls are discretionary in the sense that a database user (subject) with appropriate access rights is authorized to grant those access rights to other database users (subjects) and access roles.

The parser module is directly responsible for enforcing the DAC policy. It is responsible for both generating rows in the system access rights table that give a user (subject) the right to access an object and for checking of those access rights on subsequent execution of SQL statements.

All user (subject) access to database objects subject to the DAC policy is controlled using access rights. Types of database objects include `DATABASE`, `USER`, `TABLE`, `INDEX`, `VIEW`, `TRIGGER`, `MACRO`, `PROCEDURE`, `EXTERNAL PROCEDURE`, `FUNCTION`, `TYPE`, `GLOP`, `REPLICATION GROUP`, `AUTHORIZATION`, `ROLE`, and `PROFILE`. [Note that the Teradata Database supports the concept of a `USER` object - distinct and separate from the concept of a user as a subject. A `DATABASE` object and a `USER` object are *almost* identical in the Teradata Database. The difference is that a `USER` object contains all of the attributes of a `DATABASE` object (space allocations, journals, etc.) plus a set of attributes specifically used during session establishment (collation sequence, time zone, default character set, etc.).] Access rights are maintained through the Data Dictionary for which access is similarly controlled.

Upon initial installation of the Teradata Database, it has only one user. This user is called user `DBC` and will own all other databases and users in the system. User `DBC` also has access rights on all objects within the database. For the evaluated configuration, the administrator guidance requires creating a separate authorized security administrator under user `DBC` and granting that user access rights for creating and managing users, access roles, profiles, configuration of password management controls, and controlling the security audit facility. Creating an authorized security administrator user under user `DBC` provides protection of sensitive data and system objects owned by user `DBC`.

The Teradata Database supports three types of access rights. Implicit rights (ownership rights) are implicitly granted to the immediate owner of a database or database object. Automatic rights include all rights on a database, user, or object and are granted automatically by the system to the creator of a database, user, or object, and to a newly created user or database. The immediate owner of an object is not necessarily the same as the creator of the object. Explicit rights are granted by any user having the `WITH GRANT OPTION` privilege for that right. The database ensures that the requestor has the appropriate access rights before access to a database object is allowed. Explicit rights can be granted to one or more specific databases or users or, alternatively, granted to "`ALL`" which results in the right being granted to every existing database or user owned by that database or user as well as any database or user owned by that database or user that may be created in the future.

The Teradata Database allows for ownership of database and user objects to be transferred from the immediate owner to another owner. Upon the transfer of ownership, implicit rights (ownership rights) on the database or user being given are removed from the previous owner and given to the new owner. The transfer of ownership does not revoke any explicit rights on the given database or user. No explicit rights on the given database or user are granted to the new ownership hierarchy as a result of the transfer of ownership, nor does the database or user being given receive any explicit rights.

Access roles define access rights on database objects for groups of users. An authorized security administrator can assign one or more access roles to a user (including a default access role). A user who is a member of an access role can access all the objects for which the access role has access rights. Users can switch from the default access role to any other access role for which they are a member. Active access role(s) refers to the access role(s) active for the user's session - either the default access role or access role(s) switched to by the user.

The DAC policy for object access is based on user identities, access rights, and active access roles. The following enforceable rules combine to control access to database objects (e.g., databases, users, views, macros, stored procedures, and functions) and the operations that can be performed on these objects:

- Every object created in the database is uniquely identified and the TOE correctly resolves all references to an object.
- The TOE enforces Discretionary Access Control (DAC) on objects based on the following user (subject) attributes: (a) identity of the user associated with the session, (b) access rights associated with the user, and (c) access rights associated with any access roles active for the user's session.
- The TOE enforces Discretionary Access Control (DAC) on objects based on the following object attributes: (a) the identity of the owner of the object, (b) the object rights granted on the object, and (c) any security policies in force for the object.
- An access right is effective in a user session only if: (a) the access right was granted directly and has not been revoked, (b) the access right was granted indirectly (e.g., implicitly, automatically) and has not been revoked, or (c) the access right was granted to the user via membership in an access role and has not been revoked from the access role, and the access role is active in the current session.

The TOE enforces the following rules to determine if access by a subject to a database object is allowed:

- If the user associated with the database session is the owner of the database object, then access to the database object is allowed.
- If the required access right on the database object (or the containing database) has been explicitly or implicitly granted to the user associated with the database session, then access to the database object is allowed.
- If an access role is active within the session and the required access right on the database object (or the containing database) has been explicitly granted to the access role associated with the database session or to an access role nested within the access role associated with the database session, then access to the database object is allowed. (A nested access role is an access role that has been granted to another access role. Access roles can only be nested one level deep.)
- If the required access right on the database object (or the containing database) has been explicitly granted to `PUBLIC`, then access to the database object is allowed.

Table 7-1 provides a mapping of typical access rights that may be implicitly, automatically, or explicitly granted to a user (subject) that govern access to the different types of database objects.

Table 7-1 Database Object and Access Rights Mapping

	CREATE	DROP	ALTER	SELECT	INSERT	UPDATE	DELETE	INDEX	SHOW	STATISTICS	EXECUTE	CREATE OWNER PROCEDUE	UDTMETHOD	UDTTYPE	UDTUSAGE	REPLCONTROL	DUMP	RESTORE	CHECKPOINT
DATABASE	X	X		X	X	X	X	X	X	X		X	X	X	X		X	X	
USER	X	X		X	X	X	X					X					X	X	
TABLE	X	X		X	X	X	X	X	X	X							X	X	X
INDEX								X	X	X							X	X	
VIEW	X	X		X	X	X	X												
TRIGGER	X	X																	
MACRO	X	X									X								
PROCEDURE	X	X	X								X	X							
EXTERNAL PROCEDURE	X	X	X								X								
FUNCTION	X	X	X								X								
TYPE															X				

GLOP	X	X																	
REPLICATION GROUP															X				
AUTHORIZATION	X	X																	
ROLE	X	X																	
PROFILE	X	X																	

The TOE protects against inappropriate reuse of any resource associated with an allocated database table or index object by ensuring that any previous information content associated with that resource is unavailable by ensuring that rows written to a table or index will overlay all of the storage allocated to the row and by insuring that only rows or the columns extracted from rows are returned as part of a result set. Also, the TOE allows no access to any table or index object once the object has been deleted or dropped.

The Teradata Database operates as a set of cooperating processes which are managed by the underlying operating system. These processes operate such that no interference is allowed by processes associated with any non-TOE entities. Furthermore, the Teradata Database is designed such that its interfaces do not allow unauthorized users access to database resources.

Note that given the defined TOE boundaries, the TOE protection mechanisms could be bypassed through the underlying operational environment. As such, it is assumed that the operational environment provides an isolated domain for the execution of the TOE and ensures that the TOE security mechanisms cannot be bypassed in order to gain access to the TOE resources.

The User Data Protection security function satisfies the following security functional requirements:

- FDP_ACC.1 Subset access control – The TOE satisfies this requirement by associating access rights with all operations that can be performed on database objects and requiring that a user (subject) have the appropriate right in order to perform the corresponding operation.
- FDP_ACF.1 Security attribute based access control – The TOE satisfies this requirement by enforcing a discretionary access control policy based upon user identity and access rights associated with the user, active access roles, and database objects.
- FDP_RIP.1 Subset residual information protection – The TOE satisfies this requirement by ensuring that no previous information content associated with a database table or index object is available when the object is re-used.

7.4 SECURITY AUDIT

The parser module is directly responsible for access logging. (The term “access logging” is used as opposed to “audit” since auditing is done based upon accesses to database objects.)

The following steps are required to enable and manage the access logging facility:

- A database initialization program (DIP) script must be run to create the special access log rule macro. (DIP is accessed through the Database Window interface. The specific script used to create the access log rule macro is executed from the `/opt/teradata/tdat/tdbms/[release number]/etc/dipacc.bteq` file where `[release number]` is the version of the Teradata Database that is installed.)

- The database must be reset to initialize the logging software.

By default, only user `DBC` has the rights to control the access logging facility. The rights to control the access logging facility are determined by the right to `EXECUTE` the access log rule macro. User `DBC` may grant this right to an authorized security administrator.

The authorized security administrator can perform the following actions to control the monitoring of access rights checks performed by Teradata Database:

- Change the default rules or modify the rules for beginning logging of auditable events, including specifying the rules that determine which accesses are logged
- Change the default rules or modify the rules for ending logging auditable events
- Query the rules that are used to determine which accesses are logged
- Query access log table records containing auditable events
- Delete aged records from the access log table

System event log records are used to indicate startup and shutdown of the auditing facility.

Rules can be specified to audit events based upon one, all, or any combination of the following items:

- Type of access (event)
- Frequency of access
- Name of the requesting user
- Objects referenced
- Success or failure of the access

The Teradata Database will record the following information into each access log record that is generated: Date and time on which the event was logged; username of the user for whom the log entry was made; the name of the database object for which the log entry was made; the type of access for which the check that generated the log entry was performed; indication as to whether the access request succeeded or failed.

The time stamp used for recording the date and time on which an event is logged is obtained from the underlying operating system. As such, the security audit functions have a dependency upon the operational environment to provide a reliable time stamp.

All audit logs are maintained in protected tables within the Data Dictionary. Access to the audit log tables and defined system views is restricted to the authorized security administrator. The authorized security administrator may search and sort access log records using SQL statements to query a defined system view. As with any SQL query, a result set can be sorted by any of the view's columns, which include date, time, type of access, frequency of access, username, object name, or success/failure of the access.

The authorized security administrator may purge aged records from the access log using a system view defined within the Data Dictionary. The view contains logic to ensure that only records older than 30 days from the current date can be deleted.

The Teradata Database automatically audits all successful and failed user logon attempts in the event log. The authorized security administrator may search and sort logon/logoff records using SQL statements to query a defined system view.

Since the audit logs are maintained in protected tables within the Data Dictionary, the size of the logs is limited only by available permanent space for the user `DBC`. If permanent space for user `DBC` is exhausted such that no new access log records can be generated, then the database will reject any requests that would generate an access log record.

The Security Audit security function satisfies the following security functional requirements:

- FAU_GEN.1 Audit data generation – The TOE satisfies this requirement by generating the necessary audit records associated with each auditable event and by including the date and time, event type, user identity, object identity, and indication of success or failure in each record.
- FAU_GEN.2 User identity association – The TOE satisfies this requirement by including the associated user identity in each audit record.
- FAU_SAR.1 Audit review – The TOE satisfies this requirement by allowing the authorized security administrator to access audit records using SQL commands.
- FAU_SAR.2 Restricted audit review - The TOE satisfies this requirement by restricting access to audit records to only the authorized security administrator.
- FAU_SAR.3 Selectable audit review – The TOE satisfies this requirement by allowing the authorized security administrator to search and sort audit records based upon information contained in the records.
- FAU_SEL.1 Selective audit – The TOE satisfies this requirement by allowing the authorized security administrator to establish rules to determine whether events will be included or excluded from the audit log.
- FAU_STG.1 Protected audit trail storage – The TOE satisfies this requirement by protecting the audit records such that no alterations can be made to the audit records, only the authorized security administrator may have access to the audit records, and only the authorized security administrator may purge aged audit records.

7.5 SECURITY MANAGEMENT

The Security Management functions enable the authorized security administrator to manage the secure operation of the Teradata Database. During the initial database configuration, the initial user in the system (DBC) creates a new authorized security administrator user (e.g., SECADMIN) with appropriate access rights in order to manage security in a structured manner.

The DBC user has all security rights on the database. The Teradata Database security features allow for the DBC user to grant rights associated with security management solely to an authorized security administrator. The authorized security administrator typically performs the following duties:

- Establishes and modifies logon rules
- Creates drops access roles
- Grants, and if necessary, revokes use of access roles
- Defines the users, objects, and event types, if any, to be audited
- Monitors audit logs to detect security incidents and initiate corrective action

Access to database objects is controlled through the use of a user identity and associated access roles, as explained in the access control security function. Only the authorized security administrator may create, drop, grant, and revoke TSF data associated with users and access roles. However, other users may be authorized to grant or revoke access rights on database objects based upon ownership rights or other rights which have been explicitly granted to the user.

A user who creates a database or another user becomes the owner and is implicitly granted ownership rights on that space. As the owner of the new space, the user is also automatically granted access rights to anything created in that space. If the new space is a user, the owning user is considered the parent and the newly created user is considered its child. In turn, a child becomes the parent of any new users it creates. A parent may grant itself rights on any objects owned by any of its child users. The immediate owner of an object is the containing user or database. However, the parents in the hierarchy above the containing user or database are also indirect owners of the object. An owner can grant or revoke any right applied to an owned object.

A new user or database is automatically granted all rights on itself, with the exception of the GRANT (WITH GRANT OPTION) and CREATE DATABASE/USER, CREATE PROCEDURE, and EXECUTE PROCEDURE rights. Thus, a newly created user can create tables, views, and macros within its own user space. The automatic right to create objects can be explicitly revoked from a new user by an owner or by the creator of the user. A user can be explicitly granted the right to create databases and other users in its own user space. This right can only be granted by a user who has the GRANT right (WITH GRANT OPTION) and the right to create such users and databases. In the case of stored procedure-related rights, a newly created user gets only the DROP PROCEDURE access right. The rights to create or execute stored procedures are not automatic. These can be explicitly granted to any user by user DBC or by a user having the rights WITH GRANT OPTION. If a user has been granted the CREATE DATABASE right and subsequently creates a new database, the Teradata Database automatically grants to that user a series of creator rights on the created space. Similarly, a series of creator rights are automatically granted to the user that creates any database object.

A user may revoke rights on an object from another user, access role, or 'ALL USERS' only if the user is an owner of the object to which the rights refer, or has the GRANT right (WITH GRANT OPTION), plus all of the rights that are to be revoked on the object. If the object is a view, stored procedure, or macro, the owner of the view or macro must also have the GRANT right (WITH GRANT OPTION), plus all other applicable rights, on the objects referenced by the view, stored procedure, or macro. Implicit ownership rights cannot be revoked. Revoked access rights take effect immediately.

In the evaluated configuration, only the authorized security administrator will be granted the ROLE rights necessary to create and drop access roles. As the creator of an access role, the authorized security administrator is automatically granted WITH ADMIN OPTION on the access role and, as such, is authorized to grant and revoke the access role to or from a user or to or from another access role.

A grantor does not need to have any rights, including WITH ADMIN OPTION, on the grantee to grant or revoke a right to or from it, whether the grantee is an access role or a user.

Rules to configure access logging are managed using the BEGIN LOGGING and END LOGGING SQL statements. By default, only user DBC has the rights to control the access logging facility. The rights to control the access logging facility are determined by the right to EXECUTE the access log rule macro. User DBC may grant this right to the authorized security administrator.

The Teradata Database enforces maximum quotas and limits on various resources to ensure that those resources are protected from monopolization by any individual database user. Permanent, temporary, and spool space limits are managed by the authorized security administrator through the use of CREATE/MODIFY USER/DATABASE statements.

The Security Management security function satisfies the following security functional requirements:

- FMT_MOF.1 Management of security functions behavior – The TOE satisfies this requirement by restricting the ability to modify the behavior of the audit function to the authorized security administrator.
- FMT_MSA.1 Management of security attributes – The TOE satisfies this requirement by restricting to only authorized users the ability to grant or revoke access rights associated with operations on database objects and to only the authorized security administrator for creating, dropping, granting, or revoking use of an access role.
- FMT_MSA.3 Static attribute initialization – The TOE satisfies this requirement by ensuring that default access rights are granted to database objects and owners whenever an object is created.
- FMT_MTD.1 Management of TSF data – The TOE satisfies this requirement by ensuring that only the authorized security administrator may create or modify TSF data associated with users, access roles, authentication data, the audit function, and maximum quotas.

- FMT_REV.1 Revocation – The TOE satisfies this requirement by ensuring that only authorized users may revoke rights associated with operations on database objects.
- FMT_SMF.1 Specification of management functions – The TOE satisfies this requirement by providing an interface through which the authorized security administrator may manage users, authentication data, access roles, and the audit function, and set maximum quotas.
- FMT_SMR.1 Security roles – The TOE satisfies this requirement by allowing the authorized security administrator to create and maintain access roles and to associate users with access roles.

7.6 RESOURCE UTILIZATION

The Teradata Database enforces maximum quotas and limits on various resources to ensure that those resources are protected from monopolization by any individual database user.

Permanent space is used to store tables, indexes, stored procedures, triggers, functions, and permanent journals. (A permanent journal is not a type of database object. Rather, a permanent journal is associated with a table object and is used to maintain a sequential history of all changes made to the rows of the table.) Permanent space limits are enforced at the database or user (not table) level. The authorized security administrator defines the maximum limit for a user with the `PERM` parameter of a `CREATE/MODIFY USER` statement and an authorized database administrator defines the maximum limit for a database with the `PERM` parameter of a `CREATE/MODIFY DATABASE` statement.

Temporary space is used to hold rows of materialized global temporary tables. It is allocated at the database or user level, but not the table level. The authorized security administrator may define the maximum limit with the `TEMPORARY` parameter of a `CREATE/MODIFY USER` statement or through a user profile assigned to a user. If a temporary space limit is not explicitly specified for a user or database, then the temporary space limit is inherited from the specification for the immediate owner of the user or database.

Spool space is used to hold the response rows of every query run by a user during a session, to hold intermediate result sets produced during execution of a query, and to hold volatile tables produced during execution of a query. The Teradata Database allocates spool space dynamically only from space that is not being used for permanent or temporary data. The authorized security administrator may define the maximum limit with the `SPOOL` parameter of a `CREATE/MODIFY USER/DATABASE` statement or through a user profile assigned to a user. If a spool limit is not explicitly specified for a user or database, then the spool limit is inherited from the specification for the immediate owner of the user or database.

The Resource Utilization function satisfies the following security functional requirements:

- FRU_RSA.1 Maximum Quotas – The TOE satisfies this requirement by enforcing maximum space limits to prevent excessive monopolization of resources by any individual database user.

APPENDIX A - ACRONYMS

AMP	Access Module Processor
AWS	Administration Workstation
CLI	Call Level Interface
CNS	Console Subsystem
CPU	Central Processing Unit
DAC	Discretionary Access Control
DBMS	Database Management System
DBW	Database Window
DIP	Database Initialization Program
LAN	Local Area Network
MPP	Massive Parallel Processing
OS	Operating System
PDE	Parallel Database Extensions
PE	Parsing Engine
RDBMS	Relational Database Management System
SMP	Symmetric Multi Processing
SLES	SUSE Linux Enterprise Server
SQL	Structured Query Language
TDP	Teradata Director Program
vdisk	Virtual Disk
vproc	Virtual Processor