Microsoft® SQL Server® 2012 Database Engine
Common Criteria Evaluation (EAL4+)

Security Target

SQL Server 2012 Team

Author: Roger French
(Microsoft Corporation)

Version: 1.0

Date: 2013-01-11

Abstract

This document is the Security Target (ST) for the Common Criteria certification of the database engine of Microsoft® SQL Server® 2012.

Keywords

CC, ST, Common Criteria, SQL, Security Target, DBMS, Database Management System

This page intentionally left blank

Table of Contents

List of Tables

List of Figures

1 ST Introduction

This chapter presents Security Target (ST) and TOE identification information and a general overview of the ST. A ST contains the information technology (IT) security requirements of an identified Target of Evaluation (TOE) and specifies the functional and assurance security measures offered by that TOE to meet stated requirements. A ST principally defines:

a) A security problem expressed as a set of assumptions about the security aspects of the environment, a list of threats that the TOE is intended to counter, and any known rules with which the TOE must comply (chapter 3, Security Problem Definition).

b) A set of security objectives and a set of security requirements to address the security problem (chapters 4 and 6, Security Objectives and IT Security Requirements, respectively).

c) The IT security functionality provided by the TOE that meets the set of requirements (chapter 7, TOE Summary Specification).

1.1 ST and TOE Reference

This chapter provides information needed to identify and control this ST and its Target of Evaluation (TOE).

ST Title:	Microsoft® SQL Server® 2012 Database Engine Common Criteria Evaluation (EAL4+) Security Target
ST Version:	1.0
Date:	2013-01-11
Author:	Roger French, Microsoft Corporation
Certification-ID:	BSI-DSZ-CC-0811
TOE Identification:	Microsoft® SQL Server® 2012 Database Engine Enterprise Edition x64 (English) (and its related guidance documentation ([AGD] and [AGD_ADD]))
TOE Version:	11.0.3000.0 (This build refers to SQL Server 2012 including Service Pack 1.)
TOE Platform:	Windows Server 2008 R2 Enterprise Edition (English) or Windows Server 2012 Standard Edition (English)[1]
CC Identification:	Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 3 as of July 2009, English version ([CC]).
Evaluation Assurance Level:	EAL4 augmented by ALC_FLR.2
PP Conformance:	U.S. Government Protection Profile for Database Management Systems, Version 1.3, December, 24th 2010 ([PP])
Keywords:	CC, ST, Common Criteria, SQL, Security Target, DBMS, Database Management System

[1] Both including Core installation variant

1.2 TOE Overview

The TOE is the database engine of SQL Server 2012. SQL Server is a Database Management System (DBMS).

The TOE has been developed as the core of the DBMS to store data in a secure way.

The security functionality of the TOE comprises:

- Security Management
- Access Control
- Identification and Authentication
- Security Audit
- Session Handling

A summary of the TOE security functionality can be found in chapter 1.3.4. A more detailed description of the security functionality can be found in chapter 7, TOE Summary Specification.

Note that only the SQL Server 2012 database engine is addressed in this ST. Other related products of the SQL Server 2012 platform, such as Analysis Services, provide services that are useful but are not central to the enforcement of security policies. Hence, security evaluation is not directly applicable to those other products.

1.3 TOE Description

This chapter provides context for the TOE evaluation by identifying the product type and describing the evaluated configuration. The main purpose of this chapter is to bind the TOE in physical and logical terms. The chapter starts with a description of the product type before it introduces the physical scope, the architecture and last but not least the logical scope of the TOE.

1.3.1 Product Type

The product type of the TOE described in this ST is a database management system (DBMS) with the capability to limit TOE access to authorized users, enforce Discretionary Access Controls on objects under the control of the database management system based on user and/or role authorizations, and to provide user accountability via audit of users' actions.

A DBMS is a computerized repository that stores information and allows authorized users to retrieve and update that information. A DBMS may be a single-user system, in which only one user may access the DBMS at a given time, or a multi-user system, in which many users may access the DBMS simultaneously.

The TOE which is described in this ST is the database engine and therefore part of SQL Server 2012. It provides a relational database engine providing mechanisms for Access Control, Identification and Authentication and Security Audit.

The SQL Server platform additionally includes the following tools which are not part of the TOE:

- SQL Server Replication: Data replication for distributed or mobile data processing applications and integration with heterogeneous systems
- Analysis Services: Online analytical processing (OLAP) capabilities for the analysis of large and complex datasets.
- Reporting Services: A comprehensive solution for creating, managing, and delivering both traditional, paper-oriented reports and interactive, Web-based reports.

- Management tools: The SQL Server platform includes integrated management tools for database management and tuning as well as tight integration with tools such as Microsoft Operations Manager (MOM) and Microsoft Systems Management Server (SMS).
- Development tools: SQL Server offers integrated development tools for the database engine, data extraction, transformation, and loading (ETL), data mining, OLAP, and reporting that are tightly integrated with Microsoft Visual Studio to provide end-to-end application development capabilities.

The TOE itself only comprises the database engine of the SQL Server 2012 platform which provides the security functionality as required by this ST. Any additional tools of the SQL Server 2012 platform interact with the TOE as a standard SQL client. The scope and boundary of the TOE will be described in the next chapter.

1.3.2 Physical Scope and Boundary of the TOE

The TOE is the database engine of the SQL Server 2012 and its related guidance documentation. This engine is available in three different configurations (x86, x64, IA64). Only the x64 version is subject to this evaluation.

Further, SQL Server 2012 is available in different editions. Only the Enterprise Edition (EE) is subject to this evaluation.

The following figure shows the TOE (including its internal structure) and its immediate environment.

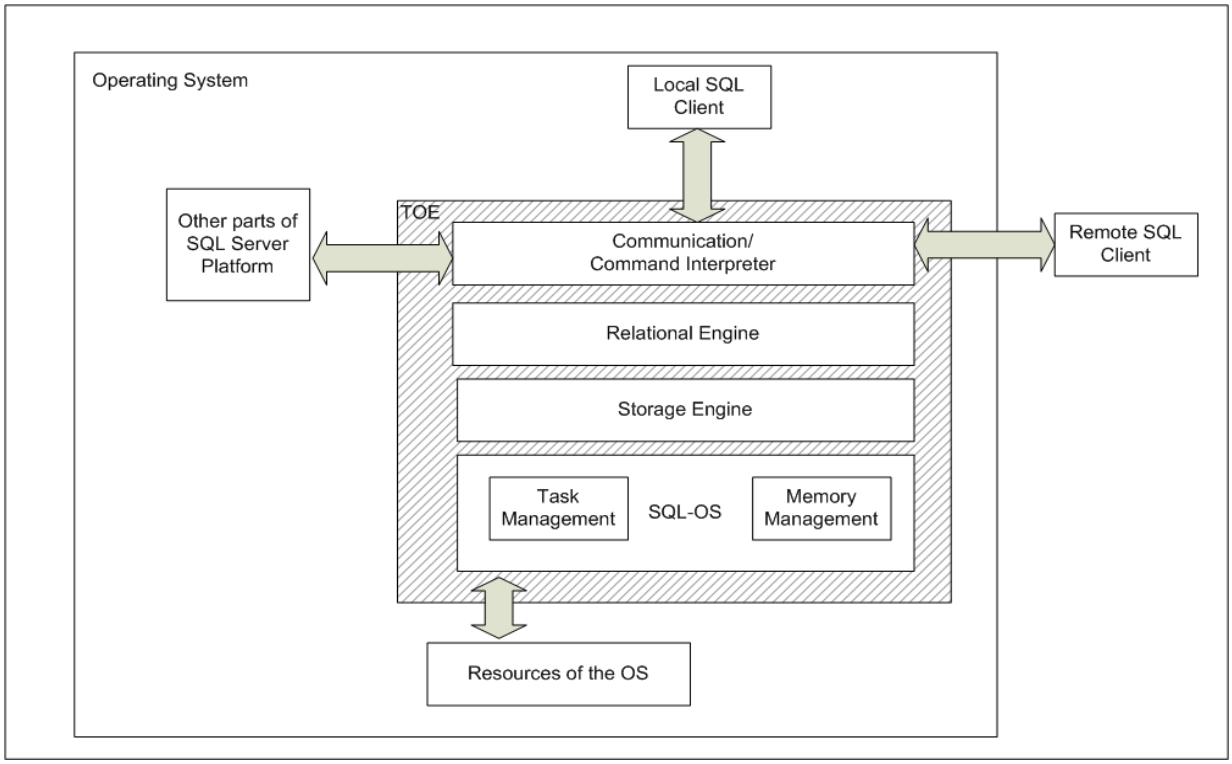

Figure 1 - TOE Structure

As seen in Figure 1 the TOE internally comprises the following units:

The **Communication** part is the interface for programs accessing the TOE. It is the interface between the TOE and clients performing requests.

All responses to user application requests return to the client through this part of the TOE.

The **Relational Engine** is the core of the database engine and is responsible for all security relevant decisions. The relational engine establishes a user context, syntactically checks every Transact SQL (T-SQL) statement, compiles every statement, checks permissions to determine if the statement can be executed by the user associated with the request, optimizes the query request, builds and caches a query plan, and executes the statement.

The **Storage Engine** is a resource provider. When the relational engine attempts to execute a T-SQL statement that accesses an object for the first time, it calls upon the storage engine to retrieve the object, put it into memory and return a pointer to the execution engine. To perform these tasks, the storage engine manages the physical resources for the TOE by using the Windows OS.

The **SQL-OS** is a resource provider for all situations where the TOE uses functionality of the operating system. SQL-OS provides an abstraction layer over common OS functions and was designed to reduce the number of context switches within the TOE. SQL-OS especially contains functionality for Task Management and for Memory Management.

For **Task Management** the TOE provides an OS-like environment for threads, including scheduling, and synchronization - all running in user mode, all (except for I/O) without calling the Windows Operating System.

The **Memory Management** is responsible for the TOE memory pool. The memory pool is used to supply the TOE with its memory while it is executing. Almost all data structures that use memory in the TOE are allocated in the memory pool. The memory pool also provides resources for transaction logging and data buffers.

The immediate **environment** of the TOE comprises:

The Windows Server Operating System hosts the TOE. As the TOE is software only it lives as a process in the Operating System (OS) and uses the resources of the OS. These resources comprise general functionality (e.g. the memory management and scheduling features of the OS) as well as specific functionality of the OS, which is important for the security functionality of the TOE (see chapter 7 for more details).

Other parts of the SQL Server 2012 Platform might be installed together with the TOE. The TOE is the central part of a complete DBMS platform, which realizes all security functionality as described in this ST. However other parts of the platform may be installed on the same machine if they are needed to support the operation or administration of the TOE. However these other parts will interact with the TOE in the same way, every other client would do.

Clients (comprising local clients and remote clients) are used to interact with the TOE during administration and operation. Services of the Operating System are used to route the communication of remote clients with the TOE.

The TOE relies on functionality of the Operating System and has the following hardware/software requirements:

Table 1 - Hardware and Software Requirements

Aspect	Requirement
CPU	AMD Opteron, AMD Athlon 64, Intel Xeon with Intel EM64T support, Intel Pentium IV with EM64T support at 1.4 GHz or faster [2]
RAM	1 GB
Hard Disk	Approx. 1500 MB of free space

[2] Please note that IA64 CPUs are not supported for the certified version of the database engine of SQL Server 2012.

Aspect	Requirement
Other	DVD ROM drive, display at Super VGA resolution, Microsoft mouse compatible pointing device, keyboard
OS	Windows Server 2008 R2 Enterprise Edition (English) or Windows Server 2012 Standard Edition (English)[3]
Software	.NET Framework 3.5.1SP1/4[4]

The TOE is downloadable as a DVD image via the Microsoft volume licensing service center (https://www.microsoft.com/licensing/servicecenter/default.aspx) and Service Pack 1 as a download that has to be installed separately.

The following guidance documents and supportive information belong to the TOE:

- SQL Server Books Online: This is the general guidance documentation for the complete SQL Server 2012 platform ([AGD]).
- SQL Server Guidance Addendum: This document contains the aspects of the guidance that are specific to the evaluated configuration of SQL Server 2012 SP1 ([AGD_ADD]).

The website https://www.microsoft.com/sqlserver/en/us/common-criteria.aspx (tab “SQL Server 2012 SP1”) contains additional information about the TOE and its evaluated configuration. Also the guidance addendum that describes the specific aspects of the certified version can be obtained via this website. The guidance addendum extends the general guidance of SQL Server 2012.

This website shall be visited before using the TOE.

1.3.3 Architecture of the TOE

The TOE which is described in this ST comprises one instance of the SQL Server 2012 database engine but has the possibility to serve several clients simultaneously.

1.3.4 Logical Scope and Boundary of the TOE

SQL Server 2012 is able to run multiple instances of the database engine on one machine. After installation one default instance exists. However the administrator is able to add more instances of SQL Server 2012 to the same machine.

The TOE comprises one instance of SQL Server 2012. Within this ST it is referenced either as "the TOE" or as "instance". The machine the instances are running on is referenced as "server" or "DBMS-server".

If more than one instance of SQL Server 2012 is installed on one machine these just represent multiple TOEs as there is no other interface between two instances of the TOE than the standard client interface.

In this way two or more instances of the TOE may only communicate through the standard client interface.

The TOE provides the following set of security functionality:

[3] Both including Core installation variant

[4] For Windows Server 2008 R2 only

- The **Access Control** function of the TOE controls the access of users to user and metadata stored in the TOE. It further controls that only authorized administrators are able to manage the TOE.
- The **Security Audit** function of the TOE produces log files about all security relevant events.
- The **Security Management** function allows authorized administrators to manage the behavior of the security functionality of the TOE.
- The **Identification and Authentication**[5] function of the TOE is able to identify and authenticate users.
- The **Session Handling** mechanism which limits the possibilities of users to establish sessions with the TOE and maintains a separate execution context for every operation. Also the Memory Management functionality belongs to the area of Session Handling and ensures that any previous information in memory is made unavailable before the memory is used either by overwriting the memory explicitly with a certain pattern or by overwriting the memory completely with new information.

The following functions are part of the environment:

- The **Audit Review** and **Audit Storage** functionality has to be provided by the environment and provide the authorized administrators with the capability to review the security relevant events of the TOE.
- The **Access Control Mechanisms** has to be provided by the environment for files stored in the environment.
- The environment provides **Identification and Authentication**5 for users for the cases where this is required by the TOE (The environment AND the TOE provide mechanisms for user authentication. See chapter 7.3 for more details).
- The environment has to provide **Time stamps** to be used by the TOE.
- The environment provides a **cryptographic** mechanism for **hashing** of passwords.
- The environment provides **residual information protection** for memory which is allocated to the TOE.

All these functions are provided by the underlying Operating System except Audit Review, . An additional tool (e.g. the SQL Server Profiler, which is part of the SQL Server Platform) has to be used for Audit Review.

Access to the complete functionality of the TOE is possible via a set of SQL-commands.

This set of commands is available via:

- Shared Memory
- Named Pipes
- TCP/IP

1.4 Conventions

For this Security Target the following conventions are used:

[5] Note that the TOE as well as the environment provides a mechanism for identification and authentication. Chapter 7 will describe this in more detail.

The CC allows several operations to be performed on functional requirements: *refinement*, *selection*, *assignment*, and *iteration* are defined in chapter C.4 of Part 1 of [CC]. Each of these operations is used in this ST.

A **refinement** operation (denoted by ~~**bold crossed out text**~~) is used to remove unnecessary details of a requirement, though it does not change the meaning of the requirement.

Moreover, a refinement operation (denoted by **bold text**) is used to add details to a requirement, and thus further restricts a requirement.

The **selection** operation is used to select one or more options provided by the CC in stating a requirement. Selections that have been made are denoted by italicized text.

The **assignment** operation is used to assign a specific value to an unspecified parameter, such as the length of a password. Assignments that have been made are denoted by showing the value in square brackets, [Assignment_value].

The **iteration** operation is used when a component is repeated with varying operations. Iteration is denoted by showing the iteration number in parenthesis following the component identifier, (iteration_number).

The CC paradigm also allows security target authors to create their own requirements. Such requirements are termed ‘extended requirements’ and are permitted if the CC does not offer suitable requirements to meet the authors’ needs. Extended requirements must be identified and are required to use the CC class/family/component model in articulating the requirements. In this ST, extended requirements will be indicated with the “EXP” following the component name.

2 Conformance Claims

2.1 CC Conformance Claim

This Security Target claims to be

- **CC Part 2 (Version 3.1, Revision 3, July 2009) extended** due to the use of the component FAU_STG_EXP.5 and
- **CC Part 3 (Version 3.1, Revision 3, July 2009) conformant** as only assurance components as defined in part III of [CC] have been used.

Further this Security Target claims to be conformant to the Security Assurance Requirements package EAL 4 augmented by ALC_FLR.2.

2.2 PP Conformance Claim

This Security Target claims to be conformant to:

- U.S. Government Protection Profile for Database Management, Version 1.3, December 24 2010 ([PP]).

Though [PP] allows a demonstrable conformance this Security Target claims strict conformance to [PP].

3 Security Problem Definition

This chapter describes

- the external entities interacting with the TOE,
- the assets that have to be protected by the TOE,
- assumptions about the environment of the TOE,
- threats against those assets, and
- organizational security policies that TOE shall comply with.

3.1 Assets

The following external entities interact with the TOE:

- Administrator:
  The administrator is authorized to perform the administrative operations and able to use the administrative functions.

- User:
  A person who wants to use the TOE.

- Attacker:
  An attacker is any individual who is attempting to subvert the operation of the TOE. The intention may be to gain unauthorized access to the assets protected by the TOE.

The TOE maintains two types of data which represent the assets: User Data and TSF Data.

The primary assets are the User Data which comprises the following:

- The user data stored in or as database objects;
- User-developed queries or procedures that the DBMS maintains for users.

The secondary assets comprise the TSF data that the TOE maintains and uses for its own operation. This kind of data is also called metadata. It specifically includes:

- The definitions of user databases and database objects
- Configuration parameters,
- User security attributes,
- Security Audit instructions and records.

3.2 Assumptions

The following table lists all the assumptions about the environment of the TOE. These assumptions have been directly taken from [PP] without any modification.

Table 2 - Assumptions

Assumption	Description
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 DBMS servers, other than those services necessary for the operation,

Assumption	Description
	administration and support of the DBMS.
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.

3.3 Threats

The following table identifies the threats to the TOE. These threats have been directly taken from [PP] without any modification.

Table 3 - Threats to the TOE

Threat	Description
T.ACCIDENTAL_ADMIN_ERROR	An administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.
T.MASQUERADE	A user or process may masquerade as another entity in order to gain unauthorized access to data or TOE resources
T.POOR_DESIGN	Unintentional errors in requirements specification or design of the TOE may occur, leading to flaws that may be exploited by a casually mischievous user or program.
T.POOR_IMPLEMENTATION	Unintentional errors in implementation of the TOE design may occur, leading to flaws that may be exploited by a casually mischievous user or program.
T.POOR_TEST	Lack of or insufficient tests to demonstrate that all TOE security functions operate correctly (including in a fielded TOE) may result in incorrect TOE behavior being discovered thereby causing potential security vulnerabilities.
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.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 administrator to identify and act upon unauthorized actions may occur.

3.4 Organizational Security Policies

An organizational security policy is a set of rules, practices, and procedures imposed by an organization to address its security needs. This chapter identifies the organizational security policies applicable to the TOE. These organizational security policies have been taken from [PP] without any changes.

Table 4 - Organizational Security Policies

Policy	Description
P.ACCOUNTABILITY	The authorized users of the TOE shall be held accountable for their actions within the TOE.
P.ROLES	The TOE shall provide an authorized administrator role for secure administration of the TOE. This role shall be separate and distinct from other authorized users.

4 Security Objectives

The purpose of the security objectives is to detail the planned response to a security problem or threat. This chapter describes the security objectives for the TOE and its operational environment.

4.1 Security Objectives for the TOE

This chapter identifies and describes the security objectives of the TOE. The objectives have been directly taken from [PP] with only the changes indicated by bold italic text.

Table 5 - Security Objectives for the TOE

Objective	Description
O.ACCESS_HISTORY	The TOE will store and retrieve information (to authorized users) related to previous attempts to establish a session.
O.ADMIN_GUIDANCE	The TOE will provide administrators with the necessary information for secure management.
O.ADMIN_ROLE	The TOE will provide authorized administrator roles to isolate administrative actions.
O.AUDIT_GENERATION	The TOE will provide the capability to detect and create records of security relevant events associated with users.
O.CONFIGURATION_IDENTIFICATION	The configuration of the TOE is fully identified in a manner that will allow implementation errors to be identified and corrected with the TOE being redistributed promptly.
O.DOCUMENTED_DESIGN	The design of the TOE is adequately and accurately documented.
O.INTERNAL_TOE_DOMAINS	The TSF will maintain internal domains for separation of data and queries belonging to concurrent users.
O.MANAGE	The TOE will provide all the functions and facilities necessary to support the authorized administrators in their 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.PARTIAL_FUNCTIONAL_TEST	The TOE will undergo some security functional testing that demonstrates that the TSF satisfies some of its security functional requirements.
O.PARTIAL_SELF_PROTECTION	The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure through its own interfaces.
O.RESIDUAL_INFORMATION	The TOE will ensure that any information contained in a protected resource within its Scope of Control is not

Objective	Description
	released when the resource is reallocated.
O.TOE_ACCESS	The TOE will provide mechanisms that control a user's logical access to the TOE.
O.VULNERABILITY_ANALYSIS	The TOE will undergo some vulnerability analysis to demonstrate that the design and implementation of the TOE does not contain any obvious flaws.
O.I&A	***The TOE will provide a mechanism for identification and authentication of users.***

4.2 Security Objectives for the operational Environment

The security objectives for the TOE Environment are defined in the following table. The objectives for the environment have been directly taken from [PP] with any changes.

Table 6 - Security Objectives for the TOE Environment

Objective	Description
OE.NO_EVIL	Sites using the TOE shall ensure that authorized administrators are non-hostile, appropriately trained and follow all administrator guidance.
OE.NO_GENERAL_ PURPOSE	There will be no general-purpose computing capabilities (e.g., compilers or user applications) available on DBMS servers, other than those services necessary for the operation, administration and support of the DBMS.
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.

4.3 Security Objectives Rationale

4.3.1 Overview

The following table maps the security objectives to assumptions / threats / OSPs:

Table 7 - Summary of Security Objectives Rationale

Threats, Assumptions, OSP / Security Objectives	O.ACCESS_HISTORY	O.ADMIN_GUIDANCE	O.ADMIN_ROLE	O.AUDIT_GENERATION	O.CONFIGURATION_IDENTIFICATION	O.DOCUMENTED_DESIGN	O.INTERNAL_TOE_DOMAINS	O.MANAGE	O.MEDIATE	O.PARTIAL_FUNCTIONAL_TEST	O.PARTIAL_SELF_PROTECTION	O.RESIDUAL_INFORMATION	O.TOE_ACCESS	O.VULNERABILITY_ANALYSIS	O.I&A	OE.NO_EVIL	OE.NO_GENERAL_ PURPOSE	OE.PHYSICAL
T.ACCIDENTAL_ADMIN_ ERROR		X																
T.MASQUERADE													X		X			
T.POOR_DESIGN					X	X								X				
T.POOR_IMPLEMENTATION					X					X				X				
T.POOR_TEST						X				X				X				
T.RESIDUAL_DATA												X						
T.TSF_COMPROMISE							X	X			X	X						
T.UNAUTHORIZED_ACCESS	X								X						X			
T.UNIDENTIFIED_ACTIONS		X						X										
P.ACCOUNTABILITY				X									X		X			
P.ROLES			X															
A.NO_EVIL																X		
A.NO_GENERAL_PURPOSE																	X	
A.PHYSICAL																		X

Details are given in the following table. These details are directly taken from [PP] with only the changes indicated by bold italic text.

4.3.2 Rationale for TOE Security Objectives

Table 8 - Rationale for TOE Security Objectives

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
T.ACCIDENTAL_ADMIN_ERROR An administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.	O.ADMIN_GUIDANCE The TOE will provide administrators with the necessary information for secure management.	O.ADMIN_GUIDANCE helps to mitigate this threat by ensuring the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner. Having this guidance

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
		helps to reduce the mistakes that an administrator might make that could cause the TOE to be configured in insecurely.
T.MASQUERADE A user or process may masquerade as another entity in order to gain unauthorized access to data or TOE resources.	O.TOE_ACCESS The TOE will provide mechanisms that control a user's logical access to the TOE.	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 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.
	O.I&A ***The TOE will provide a mechanism for identification and authentication of users.***	***O.I&A mitigates this threat by providing the means to identify and authenticate the user. The correct identity of the user is the basis for any decision of the TOE about an attempt of a user to access data.***
T.POOR_DESIGN Unintentional errors in requirements specification or design of the TOE may occur, leading to flaws that may be exploited by a casually mischievous user or program.	O.CONFIGURATION_IDENTIFICATION The configuration of the TOE is fully identified in a manner that will allow implementation errors to be identified and corrected with the TOE being redistributed promptly.	O.CONFIGURATION_IDENTIFICATION plays a role in countering this threat by requiring the developer to provide control of the changes made to the TOE's design.

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
	O.DOCUMENTED_DESIGN The design of the TOE is adequately and accurately documented.	O.DOCUMENTED_DESIGN ensures that the design of the TOE is documented, permitting detailed review by evaluators.
	O.VULNERABILITY_ANALYSIS The TOE will undergo some vulnerability analysis to demonstrate the design and implementation of the TOE does not contain any obvious flaws.	O.VULNERABILITY_ANALYSIS ensures that the design of the TOE is analyzed for design flaws.
T.POOR_IMPLEMENTATION Unintentional errors in implementation of the TOE design may occur, leading to flaws that may be exploited by a casually mischievous user or program.	O.CONFIGURATION_IDENTIFICATION The configuration of the TOE is fully identified in a manner that will allow implementation errors to be identified and corrected with the TOE being redistributed promptly.	O.CONFIGURATION_IDENTIFICATION plays a role in countering this threat by requiring the developer to provide control of the changes made to the TOE's design, although the previous three objectives help minimize the introduction of errors into the implementation.
	O.PARTIAL_FUNCTIONAL_TEST The TOE will undergo some security functional testing that demonstrates the TSF satisfies some of its security functional requirements.	O.PARTIAL_FUNCTIONAL_TEST increases the likelihood that any errors that do exist in the implementation (with respect to the functional specification, high-level, and low-level design) will be discovered through testing.
	O.VULNERABILITY_ANALYSIS The TOE will undergo some vulnerability analysis to demonstrate the design and implementation of the TOE does not contain any obvious flaws.	O.VULNERABILITY_ANALYSIS helps reduce errors in the implementation that may not be discovered during functional testing. Ambiguous design documentation and the fact that exhaustive testing of the external interfaces is not required may leave bugs in the implementation undiscovered in functional testing.
T.POOR_TEST Lack of or insufficient tests to	O.DOCUMENTED_DESIGN	O.DOCUMENTED_DESIGN helps to ensure that the TOE's

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
demonstrate that all TOE security functions operate correctly (including in a fielded TOE) may result in incorrect TOE behavior being discovered thereby causing potential security vulnerabilities.	The design of the TOE is adequately and accurately documented.	documented design satisfies the security functional requirements. In order to ensure the TOE's design is correctly realized in its implementation, the appropriate level of functional testing of the TOE's security mechanisms must be performed during the evaluation of the TOE.
	O.PARTIAL_FUNCTIONAL_TEST The TOE will undergo some security functional testing that demonstrates the TSF satisfies some of its security functional requirements.	O.PARTIAL_FUNCTIONAL_TEST increases the likelihood that any errors that do exist in the implementation (with respect to the functional specification, high level, and low-level design) will be discovered through testing.
	O.VULNERABILITY_ANALYSIS The TOE will undergo some vulnerability analysis to demonstrate the design and implementation of the TOE does not contain any obvious flaws.	O.VULNERABILITY_ANALYSIS addresses this concern by requiring a vulnerability analysis be performed in conjunction with testing that goes beyond functional testing. This objective provides a measure of confidence that the TOE does not contain security flaws that may not be identified through functional testing. While these testing activities are a necessary activity for successful completion of an evaluation, this testing activity does not address the concern that the TOE continues to operate correctly and enforce its security policies once it has been fielded. Some level of testing must be available to end users to ensure the TOE's security mechanisms continue to operator correctly once the TOE is fielded.

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
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.	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.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.TSF_COMPROMISE A user or process may cause, through an unsophisticated attack, TSF data, or executable code to be inappropriately accessed (viewed, modified, or deleted).	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.RESIDUAL_INFORMATION is necessary to mitigate this threat, because even if the security mechanisms do not allow a user to 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.PARTIAL_SELF_PROTECTION The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure through its own interfaces.	O.PARTIAL_SELF_PROTECTION ensures the TOE is capable of protecting itself from attack.
	O.MANAGE The TOE will provide all the functions and facilities necessary to support the authorized administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use.	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.
	O.INTERNAL_TOE_DOMAINS The TSF will maintain internal domains for separation of data and	O.INTERNAL_TOE_DOMAINS ensures the TOE will establish separate domains for data belonging to users.

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
	queries belonging to concurrent users.	
T.UNAUTHORIZED_ACCESS A user may gain unauthorized access to user data for which they are not authorized according to the TOE security policy.	O.MEDIATE The TOE must protect user data in accordance with its security policy.	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. By implementing strong authentication to gain access to these services, an attacker's opportunity to conduct a man-in-the-middle and/or password guessing attack successfully is greatly reduced. 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 administrator. This feature ensures that no other user can modify the information flow policy to bypass the intended TOE security policy.
	O.ACCESS_HISTORY The TOE will store and retrieve information (to authorized users) related to previous attempts to establish a session.	O.ACCESS_HISTORY is important to mitigate this threat because it ensures the TOE will be able to store and retrieve the information that will advise the user of the last successful login attempt and performed actions without their knowledge.

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
	O.I&A ***The TOE will provide a mechanism for identification and authentication of users.***	***O.I&A*** ***contributes to countering this threat by providing the means to identify and authenticate the user. The correct identity of the user is the basis for any decision of the TOE about an attempt of a user to access data.***
T.UNIDENTIFIED_ACTIONS Failure of the authorized administrator to identify and act upon unauthorized actions may occur.	O.ADMIN_GUIDANCE The TOE will provide administrators with the necessary information for secure management.	The threat of an authorized administrator failing to know about malicious audit events produces the objectives of the authorized administrator having the facilities and knowing how to use them (O.ADMIN_GUIDANCE).
	O.MANAGE The TOE will provide all the functions and facilities necessary to support the authorized administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use.	The threat of an authorized administrator failing to know about malicious audit events produces the objectives of the authorized administrator having the capability to use the mechanisms (O.MANAGE) to review audit records.
P.ACCOUNTABILITY The authorized users of the TOE shall be held accountable for their actions within 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_GENERATION addresses this policy by providing the authorized 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 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, establishment of a trusted channel, etc.).

Threat/Policy	Objectives Addressing the Threat/Policy	Rationale
	O.TOE_ACCESS The TOE will provide mechanisms that control a user's logical access to the TOE.	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.
	O.I&A ***The TOE will provide a mechanism for identification and authentication of users.***	***O.I&A supports this policy by providing the means to identify and authenticate the user. The identity of the user is stored in the audit logs.***
P.ROLES The TOE shall provide an authorized administrator role for secure administration of the TOE. This role shall be separate and distinct from other authorized users.	O.ADMIN_ROLE The TOE will provide authorized administrator roles to isolate administrative actions.	The TOE has the objective of providing an authorized administrator role for secure administration. The TOE may provide other roles as well, but only the role of authorized administrator is required (O.ADMIN_ROLE).

4.3.3 Rationale for environmental Security Objectives

The following table contains the rationale for the IT Environmental Objectives. This rationale has directly been taken from [PP] without any changes.

Table 9 - Rationale for IT Environmental Objectives

Assumption	Environmental Objective Addressing the Assumption	Rationale
A.NO_EVIL Administrators are non-hostile, appropriately trained, and follow all administrator guidance.	OE.NO_EVIL Sites using the TOE shall ensure that authorized administrators are non-hostile, are appropriately trained and follow all administrator guidance.	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 There are no general-purpose computing capabilities (e.g., compilers or user applications) available on DBMS servers, other than those services	OE.NO_GENERAL_PURPOSE There will be no general-purpose computing capabilities (e.g., compilers or user applications) available on DBMS servers, other than those services necessary for	The DBMS server must not include any general-purpose commuting capabilities. This will protect the TSF data from malicious processes.

Assumption	Environmental Objective Addressing the Assumption	Rationale
necessary for the operation, administration and support of the DBMS.	the operation, administration and support of the DBMS.	
A.PHYSICAL Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the IT environment.	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.	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.

5 Extended Component Definition

The following extended components are defined in [PP] and corresponding dependencies are identified in chapter 6.3.2.

Table 10 - Extended PP components

FAU_GEN.1-NIAP-0410	Audit data generation
FAU_GEN_(EXT).2	User and/or group identity association
FAU_SEL.1-NIAP-0407	Selective audit
FDP_ACF.1-NIAP-0407	Security attribute based access control
FMT_MSA_(EXT).3	Static attribute initialization
FPT_TRC_(EXT).1	Internal TSF consistency
FTA_TAH_(EXT).1	TOE access history

5.1 Definition for FAU_STG_EXP.5

This chapter defines the extended functional component FAU_STG_EXP.5 (Administrable prevention of audit data loss) of the existing functional class FAU (Security audit). All other extended requirements that are used in this Security Target have been directly taken from [PP].

This component was defined because part II of [CC] does not contain any SFR which allows specifying a *set* of allowed actions which can be taken in the case where the audit is full. Moreover, none of the components in part II of [CC] allows *only authorized administrators* to specify what should happen if the audit log is full.

The family FAU_STG is extended by the new component FAU_STG_EXP.5 as shown in the following figure:

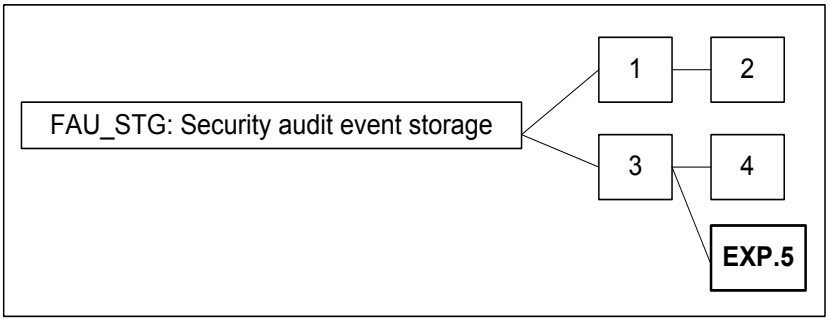

Figure 2 - FAU_STG Component Levelling

FAU_STG_EXP.5 Extended prevention of audit loss, specifies administrable actions in case the audit trail is full.

Management for FAU_STG_EXP.5:

The following actions could be considered for management functions in FMT:

a) Maintenance (deletion, modification, addition) of actions to be taken in case of audit storage failure.

Audit for FAU_STG_EXP.5:

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Basic: Actions taken due to potential audit storage failure.

FAU_STG_EXP.5	**Administrable prevention of audit data loss**
	Hierarchical to: FAU_STG.3 Action in case of possible audit data loss
	Dependencies: FAU_STG.1 Protected audit trail storage
FAU_STG_EXP.5.1	The TSF shall **take the following actions**: **[selection: *"ignore auditable events", "prevent auditable events, except those taken by the authorised user with special rights", "overwrite the oldest stored audit records", "stop the TOE",* [assignment: *other actions*]] as defined by [assignment: *authorised role*]** if the audit trail **is full.**

6 IT Security Requirements

This chapter defines the IT security requirements that shall be satisfied by the TOE or its environment:

Common Criteria divides TOE security requirements into two categories:

- Security functional requirements (SFRs) (such as, identification and authentication, security management, and user data protection) that the TOE and the supporting evidence need to satisfy to meet the security objectives of the TOE.
- Security assurance requirements (SARs) that provide grounds for confidence that the TOE and its supporting IT environment meet its security objectives (e.g., configuration management, testing, and vulnerability assessment).

These requirements are discussed separately within the following subchapters.

6.1 TOE Security Functional Requirements

The TOE satisfies the SFRs delineated in the following table. The rest of this chapter contains a description of each component and any related dependencies.

Table 11 - TOE Security Functional Requirements

Class FAU: Security Audit	
FAU_GEN.1-NIAP-0410	Audit data generation
FAU_GEN_(EXT).2	User and/or group identity association
FAU_SEL.1-NIAP-0407	Selective audit
FAU_STG_EXP.5	Administrable Prevention of audit data loss
Class FDP: User Data Protection	
FDP_ACC.1	Subset access control
FDP_ACF.1-NIAP-0407	Security attribute based access control
FDP_RIP.1	Subset residual information protection
Class FIA: Identification and Authentication	
FIA_ATD.1	User attribute definition
FIA_UAU.2	User authentication before any action
FIA_UAU.5	Multiple authentication mechanisms
FIA_UID.2	User identification before any action
Class FMT: Security Management	
FMT_MOF.1	Management of security functions behaviour
FMT_MSA.1	Management of security attributes
FMT_MSA_(EXT).3	Static attribute initialization
FMT_MTD.1	Management of TSF data
FMT_REV.1(1)	Revocation (user attributes)
FMT_REV.1(2)	Revocation (subject, object attributes)

FMT_SMF.1	Specification of management functions
FMT_SMR.1	Security roles
Class FPT: Protection of the TSF	
FPT_TRC_(EXT).1	Internal TSF consistency
Class FTA: TOE Access	
FTA_MCS.1	Basic limitation on multiple concurrent sessions
FTA_TAH_(EXT).1	TOE access history
FTA_TSE.1	TOE session establishment

6.1.1 Class FAU: Security Audit

Audit data generation (FAU_GEN.1-NIAP-0410)

FAU_GEN.1.1-NIAP-0410 **Refinement**: 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 *minimum* level of audit listed **in Table 12;**
c) [Start-up and shutdown of the DBMS;
d) Use of special permissions (e.g., those often used by authorized administrators to circumvent access control policies); and
e) [*events as specified in Table 13*]].

FAU_GEN.1.2-NIAP-0410 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 (if applicable), 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, [no additional information].

Table 12 - Auditable Events

Security Functional Requirement	Auditable Event(s)
FAU_GEN.1-NIAP-0410	None
FAU_GEN_(EXT).2	None
FAU_SEL.1-NIAP-0407	All modifications to the audit configuration that occur while the audit collection functions are operating.
FDP_ACC.1	None
FDP_ACF.1-NIAP-0407	Successful requests to perform an operation on an object covered by the SFP.
FIA_ATD.1	None
FMT_MOF.1	None
FMT_MSA.1	None
FMT_MSA_(EXT).3	None

Security Functional Requirement	Auditable Event(s)
FMT_MTD.1	None
FMT_REV.1(1)	Unsuccessful revocation of security attributes.
FMT_REV.1(2)	Unsuccessful revocation of security attributes.
FMT_SMF.1	Use of the management functions.
FMT_SMR.1	Modifications to the group of users that are part of a role.
FIA_UAU.2	Every use of the authentication mechanism.
FIA_UAU.5	The final decision on authentication.
FIA_UID.2	Every use of the authentication mechanism.

Table 13 - Auditable Events for additional SFRs

Security Functional Requirement	Auditable Event(s)	Additional Audit Record Content
FAU_STG_EXP.5	Every modifications to the setting	-
FIA_UAU.2	Every use of the authentication mechanism.	-
FIA_UAU.5	The final decision on authentication;	-
FIA_UID.2	Every use of the identification mechanism.	-

User identity association (FAU_GEN_(EXT).2)

FAU_GEN_(EXT).2.1 For audit events resulting from actions of identified users and/or identified groups, the TSF shall be able to associate each auditable event with the identity of the user and/or group that caused the event.

Selective audit (FAU_SEL.1-NIAP-0407)

FAU_SEL.1.1-NIAP-0407 **Refinement**: The TSF shall allow **only the administrator** to include or exclude auditable events from the set of audited events based on the following attributes:

a) user identity and/or group identity,
b) event type,
c) object identity,
d) [*none*];
e) [success of auditable security events;
f) failure of auditable security events; and
g) [*no additional criteria*].]

Administrable Prevention of audit data loss (FAU_STG_EXP.5)

FAU_STG_EXP.5.1 The TSF shall take one of the following actions:

- *Overwrite the oldest stored audit records*
- *Stop the TOE*

as defined by [the administrator] if the audit trail is full.

6.1.2 Class FDP: User Data Protection

Subset access control (FDP_ACC.1)

FDP_ACC.1.1 The TSF shall enforce the [Discretionary Access Control policy] on [all subjects, all DBMS-controlled objects and all operations among them].

Security attribute based access control (FDP_ACF.1-NIAP-0407)

FDP_ACF.1.1-NIAP-0407 The TSF shall enforce the [Discretionary Access Control policy] to objects based on the following:

- [the authorized user identity and/or group membership associated with a subject,
- access operations implemented for DBMS-controlled objects, and
- object identity].

FDP_ACF.1.2-NIAP-0407 **Refinement:** The TSF shall enforce the following rules to determine if an operation among controlled subjects and **DBMS-**controlled objects is allowed:

- **The Discretionary Access Control policy mechanism shall, either by explicit authorized user action or by default, provide that database management system controlled objects are protected from unauthorized access according to the following ordered rules:**[
  a) *If the requested mode of access is denied to that authorized user deny access*
  b) *If the requested mode of access is denied to [any] group of which the authorized user is a member, deny access*
  c) *If the requested mode of access is permitted to that authorized user, permit access.*
  d) *If the requested mode of access is permitted to any group of which the authorized user is a member, grant access*
  e) *Else deny access]*

FDP_ACF.1.3-NIAP-0407 **Refinement**: The TSF shall explicitly authorize access of subjects to **DBMS-**controlled objects based on the following additional rules: [

- Authorized administrators, the owner of an object and owners of parent objects have access
- in case of Ownership-Chaining access is always granted].

FDP_ACF.1.4-NIAP-0407 The TSF shall explicitly deny access of subjects to objects based on the following rules: [*no additional explicit denial rules*].

Subset residual information protection (FDP_RIP.1)

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 [*objects that are related to or may be exposed through user sessions*].

6.1.3 Class FIA: Identification and authentication

User attribute definition (FIA_ATD.1)

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users:

- [Database user identifier and/or group memberships;
- Security-relevant database roles; and
- [login-type (SQL-Server login or Windows Account Name)
- For SQL-Server login: Hashed password]].

User authentication before any action (FIA_UAU.2)

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Multiple authentication mechanisms (FIA_UAU.5)

FIA_UAU.5.1 The TSF shall provide [

- SQL Server Authentication and
- Access to Windows Authentication]

to support user authentication.

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the [following rules:

- If the login is associated with a Windows user or a Windows group Windows Authentication is used,
- If the login is a SQL Server login the SQL Server authentication is used.

].

User identification before any action (FIA_UID.2)

FIA_UID.2.1 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.4 Class FMT: Security Management

Management of security functions behaviour (FMT_MOF.1)

FMT_MOF.1.1 The TSF shall restrict the ability to *disable and enable* the functions [relating to the specification of events to be audited] to [authorized administrators].

Management of security attributes (FMT_MSA.1)

FMT_MSA.1.1 **Refinement:** The TSF shall enforce the [Discretionary Access Control policy] to restrict the ability to [*manage*] **all** the security attributes to [authorized administrators].

Static attribute initialization (FMT_MSA_(EXT).3)

FMT_MSA_ (EXT).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.

Note: FMT_MSA_(EXT).3 as defined in [PP] is identical to FMT_MSA.3 in [CC, Part 2].

Management of TSF data (FMT_MTD.1)

FMT_MTD.1.1 The TSF shall restrict the ability to [*include or exclude*] the [auditable events] to [authorized administrators].

Revocation (FMT_REV.1(1))

FMT_REV.1.1(1) The TSF shall restrict the ability to revoke security attributes associated with the *users* within the TSC to [the authorized administrators].

FMT_REV.1.2(1) The TSF shall enforce the rules [Changes to logins are applied at the latest as soon as a new session for the login is established]

Note: FMT_REV.1.1(1) as defined in [PP] differs from [CC, Part 2] in that it does not assign the list of security attributes and makes editorial changes. The list of security attributes associated with users is defined in FIA_ATD.1.1.

Revocation (FMT_REV.1(2))

FMT_REV.1.1(2) The TSF shall restrict the ability to revoke security attributes associated with the *objects* within the TSC to [the authorized administrators and database users as allowed by the Discretionary Access Control policy].

FMT_REV.1.2(2) The TSF shall enforce the rules [The changes have to be applied immediately].

Note: FMT_REV.1.1(2) as defined in [PP] differs from [CC, Part 2] in that it does not assign the list of security attributes and makes editorial changes. The list of security attributes associated with objects is implicitly defined in FDP_ACF.1-NIAP-0407.

Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [

- Add and delete logins
- Add and delete users
- Change role membership for DB scoped roles and Server scoped roles
- Create and destroy database scoped groups
- Create, Start and Stop Audit
- Include and Exclude Auditable events
- Define the mode of authentication
- Manage Attributes for Session Establishment
- Define the action to take in case the audit file is full]

Security roles (FMT_SMR.1)

FMT_SMR.1.1 **Refinement**: The TSF shall maintain the roles:

- [sysadmin]; **and**
- [roles as defined in the following tables
- Roles to be defined by authorized administrators].

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

Table 14 - Default Server Roles

Role	Description
sysadmin	Members of the sysadmin fixed server role can perform any activity in the server. By default, all members of the Windows BUILTIN\Administrators group, the local administrator's group, are members of the sysadmin fixed server role.
Serveradmin	Members of the serveradmin fixed server role can change server-wide configuration options and shut down the server.
Securityadmin	Members of the securityadmin fixed server role manage logins and their properties. They can GRANT, DENY, and REVOKE server-level permissions. They can also GRANT, DENY, and REVOKE database-level permissions. Additionally, they can reset passwords for SQL Server logins.
Processadmin	Members of the processadmin fixed server role can end processes that are running in an instance of SQL Server.
Setupadmin	Members of the setupadmin fixed server role can add and remove linked servers.
Bulkadmin	Members of the bulkadmin fixed server role can run the BULK INSERT statement.
Diskadmin	The diskadmin fixed server role is used for managing disk files.
Dbcreator	Members of the dbcreator fixed server role can create, alter, drop, and restore any database.

Table 15 - Default Database Roles

Role	Granted Permission(s)
db_owner	Members of the db_owner fixed database role can perform all configuration and maintenance activities on the database, and can also drop the database.
db_securityadmin	Members of the db_securityadmin fixed database role can modify role membership and manage permissions. Adding principals to this role could enable unintended privilege escalation.
db_accessadmin	Members of the db_accessadmin fixed database role can add or remove access to the database for Windows logins, Windows groups, and SQL Server logins.
db_backupoperator	Members of the db_backupoperator fixed database role can back up the database.
db_ddladmin	Members of the db_ddladmin fixed database role can run any Data Definition Language (DDL) command in a database.
db_datawriter	Members of the db_datawriter fixed database role can add, delete, or change data in all user tables.
db_datareader	Members of the db_datareader fixed database role can read all data from all user tables.
db_denydatawriter	Members of the db_denydatawriter fixed database role cannot add, modify,

Role	Granted Permission(s)
	or delete any data in the user tables within a database.
db_denydatareader	Members of the db_denydatareader fixed database role cannot read any data in the user tables within a database.

6.1.5 Class FPT: Protection of the TOE Security Functions

Internal TSF consistency (FPT_TRC_(EXT).1)

FPT_TRC_(EXT).1.1 The TSF shall ensure that TSF data is consistent between parts of the TOE by providing a mechanism to bring inconsistent TSF data into a consistent state in a timely manner.

6.1.6 Class FTA: TOE Access

Basic limitation on multiple concurrent sessions (FTA_MCS.1)

FTA_MCS.1.1 The TSF shall restrict the maximum number of concurrent sessions that belong to the same user.

FTA_MCS.1.2 The TSF shall enforce, by default, a limit of [*5*] sessions per user.

TOE access history (FTA_TAH_(EXT).1)

FTA_TAH_(EXT).1.1 Upon successful session establishment, the TSF shall store and retrieve the date and time of the last successful session establishment to the user.

FTA_TAH_(EXT).1.2 Upon successful session establishment, the TSF shall store and retrieve the *date and time* of the last unsuccessful attempt to session establishment and the number of unsuccessful attempts since the last successful session establishment.

TOE session establishment (FTA_TSE.1)

FTA_TSE.1.1 **Refinement**: The TSF shall be able to deny session establishment based on [attributes that can be set explicitly by authorized administrators, including user identity and/or group identity, time of day, day of the week], **and [no additional attributes]**.

6.2 TOE Security Assurance Requirements

The assurance requirements for the TOE comprise all assurance requirements for EAL 4 as defined in [CC] augmented by ALC_FLR.2.

6.3 Security Requirements rationale

6.3.1 Security Functional Requirements rationale

The following table contains the rationale for the TOE Security Requirements. This rationale has been directly taken from [PP] with only the changes indicated by bold italic text.

Note that [PP] maps some objectives to SARs only. To be conformant to [PP] the ST writer adopts this approach.

Table 16 - Rationale for TOE Security Requirements

Objective	Requirements Addressing the Objective	Rationale
O.ACCESS_HISTORY The TOE will store and retrieve information (to authorized users) related to previous attempts to establish a session.	FTA_TAH_(EXT).1	The TOE must be able to store and retrieve information about previous unauthorized login attempts and the number times the login was attempted every time the user logs into their account. The TOE must also store the last successful authorized login. This information will include the date, time, method, and location of the attempts. When appropriately displayed, this will allow the user to detect if another user is attempting to access their account. These records should not be deleted until after the user has been notified of their access history. (FTA_TAH_(EXT).1)
O.ADMIN_GUIDANCE The TOE will provide administrators with the necessary information for secure management.	ALC_DEL.1	ALC_DEL.1 ensures that the administrator is provided documentation that instructs them how to ensure the delivery of the TOE, in whole or in parts, has not been tampered with or corrupted during delivery. This requirement ensures the administrator has the ability to begin their TOE installation with a clean (e.g., malicious code has not been inserted once it has left the developer's control) version of the TOE, which is necessary for secure management of the TOE.
	AGD_PRE.1	AGD_PRE.1 ensures the administrator has the information necessary to install the TOE in the evaluated configuration. Often times a vendor's product contains software that is not part of the TOE and has not been evaluated. The Preparative User Guidance (AGD_PRE) documentation ensures that once the administrator has followed the installation and configuration guidance the result is a TOE in a secure configuration.

Objective	Requirements Addressing the Objective	Rationale
	AGD_OPE.1	AGD_OPE.1 mandates the developer provide the administrator with guidance on how to operate the TOE in a secure manner. This includes describing the interfaces the administrator uses in managing the TOE, security parameters that are configurable by the administrator, how to configure the TOE's rule set and the implications of any dependencies of individual rules. The documentation also provides a description of how to setup and review the auditing features of the TOE. The guidance must show the administrator how to use the functionality available, review the results of any tests and/or alerts, and act accordingly.
	AGD_OPE.1	AGD_OPE.1 is also intended for non-administrative users, but it could be used to provide guidance on security that is common to both administrators and non-administrators (e.g., password management guidelines).
	AGD_OPE.1 AGD_PRE.1	AGD_OPE.1 and AGD_PRE.1 analysis during evaluation will ensure that the guidance documentation is complete and consistent, and notes all requirements for external security measures.
O.ADMIN_ROLE The TOE will provide authorized administrator roles to isolate administrative actions.	FMT_SMR.1	The TOE will establish, at least, an authorized administrator role. The authorized administrator will be given privileges to perform certain tasks that other users will not be able to perform. These privileges include, but are not limited to, access to audit information and security functions (FMT_SMR.1).
O.AUDIT_ GENERATION The TOE will provide the capability to detect and create records of security relevant events	FAU_GEN.1-NIAP-0410	FAU_GEN.1-NIAP-0410 defines the set of events that the TOE must be capable of recording. This requirement ensures that the administrator has the ability to audit any security relevant events that takes place in the TOE. This requirement also defines the

Objective	Requirements Addressing the Objective	Rationale
associated with users.		information that must be contained in the audit record for each auditable event. This requirement also places a requirement on the level of detail that is recorded on any additional security functional requirements an ST author adds to this [PP].
	FAU_GEN_(EXT).2	FAU_GEN_(EXT).2 ensures that the audit records associate a user and/or group identity with the auditable event. In the case of authorized users, the association is accomplished with the user ID. In the case of authorized groups, the association is accomplished with the group ID.
	FAU_SEL.1-NIAP-0407	FAU_SEL.1-NIAP-0407 allows the administrator to configure which auditable events will be recorded in the audit trail. This provides the 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.
	FAU_STG_EXP.5	***FAU_STG_EXP.5 allows the administrator to define what should happen in the case where the audit file is full. This provides the administrator with the possibility to decide about possible audit data loss or stopping of services based on the information stored in the database.***
O.CONFIGURATION_IDENTIFICATION The configuration of the TOE is fully identified in a manner that will allow implementation errors to be identified and corrected with the TOE being redistributed promptly.	ALC_CMS.***4***	ALC_CMS.***4*** addresses this objective by requiring that there be a unique reference for the TOE, and that the TOE is labeled with that reference. It also requires that there be a CM system in place, and that the configuration items that comprise the TOE are uniquely identified. This provides a clear identification of the composition of the TOE.
	ALC_FLR.2	ALC_FLR.2 addresses this objective by requiring that there be a mechanism

Objective	Requirements Addressing the Objective	Rationale
		in place for identifying flaws subsequent to fielding, and for distributing those flaws to entities operating the system.
O.DOCUMENTED_DESIGN The design of the TOE is adequately and accurately documented.	ADV_FSP.***4***	ADV_FSP.***4*** requires that the interfaces to the TOE be documented and specified.
	ADV_TDS.***3***	ADV_TDS.***3*** requires the high-level design of the TOE be documented and specified and that said design be shown to correspond to the interfaces.
	ADV_TDS.***3***	ADV_TDS.***3*** requires that there be a correspondence between adjacent layers of the design decomposition.
O.MANAGE The TOE will provide all the functions and facilities necessary to support the authorized administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use.	FMT_MOF.1	FMT_MOF.1 requires that the ability to use particular TOE capabilities be restricted to the administrator.
	FMT_MSA.1	FMT_MSA.1 requires that the ability to perform operations on security attributes be restricted to particular roles.
	FMT_MSA_(EXT).3	FMT_MSA_(EXT).3 requires that default values used for security attributes are restrictive.
	FMT_MTD.1	FMT_MTD.1 requires that the ability to manipulate TOE content is restricted to administrators.
	FMT_REV.1(1) FMT_REV.1(2)	FMT_REV.1 restricts the ability to revoke attributes to the administrator.
	FMT_SMF.1	FMT_SMF.1 identifies the management functions that are available to the authorized administrator.
	FMT_SMR.1	FMT_SMR.1 defines the specific security roles to be supported.
	FIA_ATD.1	***FIA_ATD.1 defines the user level security attributes.***
O.MEDIATE The TOE must protect user data in accordance with its security policy.	FDP_ACC.1	The FDP requirements were chosen to define the policies, the subjects, objects, and operations for how and when mediation takes place in the TOE.

Objective	Requirements Addressing the Objective	Rationale
		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-NIAP-0407	FDP_ACF.1-NIAP-0407 defines the security attribute used to provide access control to objects based on the TOE's access control policy.
	FPT_TRC_(EXT).1	Replicated TSF data that specifies attributes for access control must be consistent across distributed components of the TOE. The requirement is to maintain consistency of replicated TSF data.
O.INTERNAL_TOE_DOMAINS The TSF will maintain internal domains for separation of data and queries belonging to concurrent users.	ADV_ARC.1	ADV_ARC.1 provides the security architecture description of the security domains maintained by the TSF that are consistent with the SFRs. Since self-protection is a property of the TSF that is achieved through the design of the TOE and TSF, and enforced by the correct implementation of that design, self-protection will be achieved by that design and implementation.
O.PARTIAL_FUNCTIONAL_TEST The TOE will undergo some security functional testing that demonstrates the TSF satisfies some of its security functional requirements.	ATE_COV.***2***	ATE_COV.***2*** requires that there be a correspondence between the tests in the test documentation and the TSF as described in the functional specification.
	ATE_FUN.1	ATE_FUN.1 requires that the developer provide test documentation for the TOE, including test plans, test procedure descriptions, expected test results, and actual test results. These need to identify the functions tested, the tests performed, and test scenarios. There require that the developer run those tests, and show that the expected results were achieved.
	ATE_IND.2	ATE_IND.2 requires that the

Objective	Requirements Addressing the Objective	Rationale
		evaluators test a subset of the TSF to confirm correct operation, on an equivalent set of resources to those used by the developer for testing. These sets should include a subset of the developer run tests.
O.PARTIAL_SELF_PROTECTION The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure through its own interfaces.	ADV_ARC.1	ADV_ARC.1 provides the security architecture description of the security domains maintained by the TSF that are consistent with the SFRs. Since self-protection is a property of the TSF that is achieved through the design of the TOE and TSF, and enforced by the correct implementation of that design, self-protection will be achieved by that design and implementation.
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.	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.TOE_ACCESS The TOE will provide mechanisms that control a user's logical access to the TOE.	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/or group memberships and enforce what type of access the user has to the TOE.
	FTA_MCS.1	FTA_MCS.1 ensures that users may only have a maximum of a specified number of active sessions open at any given time.
	FTA_TSE.1	FTA_TSE.1 allows the TOE to restrict access to the TOE based on certain criteria.
O.VULNERABILITY_ANALYSIS The TOE will undergo some vulnerability analysis to demonstrate	AVA_VAN.***3***	The AVA_VAN.***3*** component provides the necessary level of confidence that vulnerabilities do not exist in the TOE that could cause the security policies to be violated. AVA_VAN.***3*** requires

Objective	Requirements Addressing the Objective	Rationale
the design and implementation of the TOE does not contain any obvious flaws.		the evaluator to perform a search for potential vulnerabilities in all the TOE deliverables. For those vulnerabilities that are not eliminated by the developer, a rationale must be provided that describes why these vulnerabilities cannot be exploited by a threat agent with an ***enhanced*** basic attack potential, which is in keeping with the desired assurance level of this TOE. This component provides the confidence that security flaws do not exist in the TOE that could be exploited by a threat agent of ***enhanced*** basic attack potential to violate the TOE's security policies.
O.I&A ***The TOE will provide a mechanism for identification and authentication of users.***	***FIA_UAU.2***	***FIA_UAU.2 realizes the authentication part of O.I&A as it requires that each user has to get successfully authenticated before allowing any other TSF-mediated action on behalf of that user.***
	FIA_UID.2	***FIA_UID.2 realizes the identification part of O.I&A as it requires that each user has to get successfully identified before allowing any other TSF-mediated action on behalf of that user.***
	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/or group memberships and enforce what type of access the user has to the TOE.***
	FIA_UAU.5	***FIA_UAU.5 specifies that the TOE uses two methods to ensure that every user has to be successfully authenticated.*** ***On the one hand the TOE is able to reuse the authentication results from the environment and on the other hand the TOE provides a password based authentication***

Objective	Requirements Addressing the Objective	Rationale
		mechanism.

The following table includes the rationale for the IT Environment Requirements. This rationale has been directly taken from [PP] with only the changes indicated by bold italic text.

Table 17 - Rationale for Environment Requirements

Environmental Objective	Requirements Addressing the Objective	Rationale
OE.NO_EVIL Sites using the TOE shall ensure that authorized administrators are non-hostile, are appropriately trained and follow all administrator guidance.	N/A	This objective does not contain any IT security requirements because it is a non-IT related objective. Thus, the CC does not mandate it map to any requirements.
OE.NO_GENERAL_ PURPOSE There will be no general-purpose computing capabilities (e.g., compilers or user applications) available on DBMS servers, other than those services necessary for the operation, administration and support of the DBMS.	N/A	This objective does not contain any IT security requirements because it is a non-IT related objective. Thus, the CC does not mandate it map to any requirements.
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.	N/A	This objective does not contain any IT security requirements because it is a non-IT related objective. Thus, the CC does not mandate it map to any requirements.

6.3.2 Rationale for satisfying all Dependencies

The following table contains the rationale for satisfying all dependencies of the Security Requirements. This rationale has been taken from [PP] with only the changes indicated by bold italic text[6].

[6] Both [PP, 5.1] and [PP, 6.5] list the dependencies of Security Functional Requirements. The rationale in Table 18 is based on the dependencies given in [PP, 6.5].

Table 18 - Rationale for satisfying all dependencies

Requirement	Dependency	Satisfied
FAU_GEN.1-NIAP-0410	FPT_STM.1	This requirement must be satisfied by the IT environment because the DBMS is a software only TOE.
FAU_GEN_(EXT).2	FAU_GEN.1-NIAP-0410 FIA_UID.1	***Satisfied*** ***The dependency to FIA_UID.1 is fulfilled by the TOE (FIA_UID.2 is hierarchical to FIA_UID.1).***
FAU_SEL.1-NIAP-0407	FAU_GEN.1-NIAP-0410 FMT_MTD.1	Satisfied Satisfied
FAU_STG_EXP.5	***FAU_STG.1***	***The dependency to FAU_STG.1 is satisfied by the environment. The TOE as a DBMS has to rely on the Operating System to protect the files.***
FDP_ACC.1	FDP_ACF.1-NIAP-0407	Satisfied
FDP_ACF.1-NIAP-0407	FDP_ACC.1 FMT_MSA.3	Satisfied The dependency on FMT_MSA.3 is satisfied by FMT_MSA_(EXT).3.
FDP_RIP.1	None	N/A
FIA_ATD.1	None	N/A
FIA_UAU.2	***FIA_UID.1***	***The dependency to FIA_UID.1 is fulfilled by the TOE (FIA_UID.2 is hierarchical to FIA_UID.1).***
FIA_UAU.5	***None***	***N/A***
FIA_UID.2	***None***	***N/A***
FMT_MOF.1	FMT_SMF.1 FMT_SMR.1	Satisfied Satisfied
FMT_MSA.1	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 1 FMT_SMR.1	Dependency satisfied by FDP_ACC.1. Satisfied Satisfied
FMT_MSA_(EXT).3	FMT_MSA.1 FMT_SMR.1	Satisfied Satisfied
FMT_MTD.1	FMT_SMF.1	Satisfied

Requirement	Dependency	Satisfied
	FMT_SMR.1	Satisfied
FMT_REV.1(1)	FMT_SMR.1	Satisfied
FMT_REV.1(2)	FMT_SMR.1	Satisfied
FMT_SMF.1	None	N/A
FMT_SMR.1	FIA_UID.1	***Satisfied*** ***The dependency to FIA_UID.1 is fulfilled by the TOE (FIA_UID.2 is hierarchical to FIA_UID.1).***
FPT_TRC_(EXT).1	FPT_ITT.1	***This dependency does not need to be fulfilled because the TOE does not comprise physically separated parts.***
FTA_MCS.1	FIA_UID.1	***The dependency to FIA_UID.1 is either fulfilled by the TOE (FIA_UID.2 is hierarchical to FIA_UID.1) or by the environment.***
FTA_TAH_(EXT).1	None	N/A
FTA_TSE.1	None	N/A

6.3.3 Rationale for extended requirements

Table 19 presents the rationale for the inclusion of the extended functional and assurance requirements. The extended requirements that are included as NIAP interpretations do not require a rationale for their inclusion per CCEVS management. The rationale has been directly taken from [PP] with no other than the indicated changes.

Table 19 - Rationale for Explicit Requirements

Explicit Requirement	Identifier	Rationale
FAU_GEN_(EXT).2	User and/or group identity association	This requirement was needed to replace FAU_GEN.2.1-NIAP-0410 because this PP does not require the TOE to implement a user identity. It does require the TOE to implement a user identity and/or a group identity to satisfy the DAC policy. Therefore, this extended requirement was created to allow the audit function to use the user identity or the group identity or both.
FPT_TRC_(EXT).1	Internal TSF consistency	FPT_TRC_(EXT).1 has been created to require timely consistency of replicated TSF data. Although there is a Common Criteria Requirement that attempts to address this functionality, it falls short of the needs of the environment in this protection profile.

Explicit Requirement	Identifier	Rationale
		Specifically, FPT_TRC.1.1 states "The TSF shall ensure that TSF data is consistent when replicated between parts of the TOE." In the widely distributed environment of this PP's TOE, this is an infeasible requirement. For TOEs with a very large number of components, 100 percent TSF data consistency is not achievable and is not expected at any specific instant in time. Another concern lies in FPT_TRC.1.2 that states that when replicated parts of the TSF are "disconnected", the TSF shall ensure consistency of the TSF replicated data upon "reconnection". Upon first inspection, this seems reasonable, however, when applying this requirement it becomes clear that it dictates specific mechanisms to determine when a component is "disconnected" from the rest of the TSF and when it is "reconnected". This is problematic in this PP's environment in that it is not the intent of the authors to dictate that distributed TSF components keep track of connected/disconnected components. In general, to meet the needs of this PP, it is acceptable to only require a mechanism that provides TSF data consistency in a timely manner after it is determined that it is inconsistent.
FTA_TAH_(EXT).1	TOE Access History	This PP does not require the TOE to contain a client. Therefore, the PP cannot require the client to display a message. This requirement has been modified to require the TOE to store and retrieve the access history instead of displaying it.
FMT_MSA_(EXT).3	Static attribute initialization	The CC does not allow the PP author to specify restrictive values that are not modifiable. This extended requirement eliminates the element FMT_MSA.3.2 from the component FMT_MSA.3 and makes the component more secure by requiring the security attributes of the objects on creation to be restrictive and not allowing any user to be able of override the restrictive default values.

Explicit Requirement	Identifier	Rationale
FAU_STG_EXP.5	***Administrable Prevention of audit data loss***	***It has been necessary to develop this explicit Security Functional Requirement because part II of [CC] does not contain any SFR which allows specifying a set of allowed actions which can be taken in the case where the audit is full.*** ***For the TOE described in this ST it was necessary to provide authorized administrators with the possibility to specify what should happen if the audit log is full. However there should only be one action to be taken in this case.*** ***However this SFR has been developed based on the definition of FAU_STG.4 and has the same family behaviour except that it is not hierarchical to any other SFR. .***

6.3.4 Rationale for Assurance Requirements

The assurance level of [PP] requires an assurance level of at least EAL2 for this Security Target. However, the developer believes that a database management system should be resistant against attacks that are performed by attackers with an attack potential of enhanced basic. For this reason the assurance component AVA_VAN.3 has been used. Because of the dependencies of the component AVA_VAN.3, and to gain a higher level of assurance in the correct implementation of the security functionality, the author decided to specify an Evaluation Assurance Level 4 augmented by ALC_FLR.2.

The additional use of ALC_FLR.2 is necessary in order to stay compliant to [PP].

7 TOE Summary Specification

This chapter presents an overview of the security functionality implemented by the TOE.

7.1 Security Management (SF.SM)

This security functionality of the TOE allows modifying the TSF data of the TOE and therewith managing the behavior of the TSF.

This comprises the following management functions (FMT_SMF.1):

- Add and delete logins on an instance level,
- Add and delete users on a database level,
- Change role membership for DB scoped roles and Server scoped roles,
- Create and destroy database roles,
- Create, Start and Stop Security Audit,
- Include and exclude Auditable events,
- Define the mode of authentication for every login,
- Manage attributes for Session Establishment,
- Define the action to take in case the audit file is full.

All these management functions are available via T-SQL statements directly or realized by Stored Procedures within the TOE which can be called using T-SQL.

The TOE maintains a set of roles on the server level and on the database level as listed in Table 14 and Table 15. The TOE maintains a security ID for each login on a server level and each database user. This security ID is used to associate each user with his assigned roles (FIA_ATD.1, FMT_SMR.1).

Changes to logins that are preformed via the management functions are applied at the latest as soon as a new session for the login is established (FMT_REV.1(1)).

7.2 Access Control (SF.AC)

The TOE provides a Discretionary Access Control (DAC) mechanism to control the access of users to objects based on the identity of the user requesting access, the membership of this user to roles, the requested operation and the ID of the requested object.

The TOE maintains two kinds of user representations:

1. On an instance level an end user is represented by a login. On this level the TOE controls the access of logins to objects pertaining to the instance (e.g. to view a database).
2. On a database level an end user is represented by a database user. On this level the TOE controls the access of database users to objects of the database (e.g. to read or create a table).

Further the TOE is able to manage a user account completely within a database. In this case the user account in the database is associated with a login that is also contained in this database. The authentication then happens against this database.

Members of the database roles “db_owner” or “db_accessadmin” are able to add users to a database. The TOE maintains an internal security identifier (SID) for every user and role. Each database user can be associated with at most one instance “login”.

Every object controlled by the TOE has an ID, an owner and a name.

Objects in the TOE form a hierarchy and belong to one of three different levels: server, database and schema.

The TOE maintains an Access Control List (ACL) for each object within its scope. These ACLs are stored in a system table which exists in every database for database related ACLs and in a system table in the 'master' database for instance level ACLs.

Each entry of an ACL contains a user SID and defines whether a permission is an "Allow" or a "Deny" permission for that SID.

When a new object is created, the creating user is assigned as the owner of the object and has complete control over the object. The ACL for a newly created object is always empty by default (FMT_MSA_(EXT).3).

After creation, grant, deny or revoke permissions on objects can be assigned to users. Changes to the security relevant attributes of objects are immediately applied (FMT_REV.1(2)).

When a user attempts to perform an action to an object under the control of the TOE, the TOE decides whether the action is to be permitted based on the following rules:

1. If the requested mode of access is denied to that authorized user, the TOE will deny access
2. If the requested mode of access is denied to any role of which the authorized user is a member, the TOE will deny access
3. If the requested mode of access is permitted to that authorized user, the TOE will permit access
4. If the requested mode of access is permitted to any role of which the authorized user is a member, the TOE will permit access
5. Else: The TOE will deny access

The TOE permission check for an action on an object includes the permissions of its parent objects. The permissions for the object itself and all its parent objects are accumulated together before the aforementioned rules are evaluated. Note: Some actions require more than one permission.

This means that if a user or a role has been granted a permission to an object this permission is also valid for all child objects. E.g. if a user has been granted a permission to a schema, he automatically has the same permission on all tables within that schema, if the permission has not explicitly been denied. Similarly, if a user has been denied a permission on a schema, he will be denied the same permission to all tables within that schema, regardless of explicit grant permissions.

The rules as described before are always applied when a user requests access to a certain object using a certain operation. There are only two situations where these access control rules are overridden:

1. The system administrator, the owner of an object and owners of parent objects always have access, so for these users the TOE will always allow access to the object
2. In the case of "Ownership Chaining" which is described in chapter 8.1 in more detail the access is allowed

(FDP_ACC.1 and FDP_ACF.1-NIAP-0407)

As the access to management functions of the TOE is controlled by the same functionality as the access to user data this security functionality additionally ensures that the management functions are only available for authorized administrators (FMT_MOF.1, FMT_MSA.1, FMT_MTD.1, FMT_REV.1(1)).

7.3 Identification and Authentication (SF.I&A)

This security functionality requires each user to be successfully authenticated before allowing any other actions on behalf of that user. This is done on an instance level and means that the user has to be associated with a login of the TOE.

The TOE knows two types of logins: Windows accounts and SQL Server logins. The administrator has to specify the type of login for every login he is creating.

The possibility for the TOE to perform its own authentication is necessary because not all users connecting to the TOE are connecting from a Windows environment.

Microsoft Windows account names

These logins are associated with a user account of the Windows Operating System in the environment.

For these logins the TOE requires that the Windows environment passes on the Windows SID(s) of that user to authenticate the user before any other action on behalf of that user is allowed.[7]

For these logins the Windows security identifier (SID) from the Windows account or group is used for identification of that login within the TOE. Any permission is associated with that SID (FIA_UAU.2, FIA_UID.2, FIA_ATD.1, FIA_UAU.5).

SQL Server login names

SQL Server logins are not associated with a user of Windows but are maintained by the TOE itself. Logins exist on a server level (and users in databases can be associated with a login) and on a database level itself (for contained databases). For every SQL Server login the TOE maintains a login name and a password. The password is not stored in plain text, but hashed using the SHA-1 hash function provided by the Operating System in the environment.

Each SQL Server login name is stored in a system table. SQL Server generates a SID that is used as a security identifier and stores it in this table.

This SID is internally used as a security identifier for the login.
If a user is connecting to the TOE using a SQL Server login he has to provide the username and password. The TOE hashes the password using the hash function provided by the Operating System in the environment, and compares the hash to the value stored for that user. If the values are identical the TOE has successfully authenticated the user (FIA_UAU.2, FIA_UID.2, FIA_ATD.1, FIA_UAU.5).

7.4 Security Audit (SF.AU)

The TOE produces audit logs for all security relevant actions. These audit logs are stored into files in the environment of the TOE.

The Security Audit of the TOE especially comprises the following events:

- Startup and Shutdown of the TOE,
- Start and Shutdown of Security Audit Function,
- Every login attempt including the processes for authentication and session establishment,

[7] Windows authentication of users may be based on a username and password or alternative mechanisms. After successful authentication of a user Windows associates a list of SID(s) with every user which represent the user and every group the user is a member of.

- Every successful request to perform on operation on an object covered by the access control function,
- Modifications to the role membership of users,
- The use of SF.SM,
- Every rejected attempt to establish a session.

The TOE maintains a set of events which can be additionally audited and provides the administrator with the capability to start a Security Audit process to capture these events.

For each event in the Security Audit logs the following information is stored:

1. Date and Time of the event,
2. Identity of the user causing the event (if available),
3. Type of the event,
4. ID of the object,
5. Outcome (success or failure) of the event.

Furthermore each audit file contains an introduction with the list of events which are audited in the file. (FAU_GEN.1-NIAP-0410 and FAU_GEN_(EXT).2).

The administrator has the possibility to specify, what should happen in case an audit file is full. The following two scenarios are supported in the evaluated version:

1. Rollover

   The administrator specifies a maximum size per audit file and a maximum number of files for the Security Audit. If one audit file is full, the TOE starts the next file until the maximum number of files has been reached. When the maximum number of files has been reached and the last audit file is full, the TOE will start overwriting the oldest audit file.

2. Shutdown

   The administrator specifies one audit file with a maximum size and the option to shut down the TOE on any audit error. When the maximum size of the audit file has been reached the TOE will stop operation (FAU_STG_EXP.5).

The TOE provides the possibility to create a filter for the audit function. Using this filter mechanism the administrator is able to exclude auditable events from being audited based on the following attributes:

- User identity,
- Event Type,
- Object identity,
- Success or failure of auditable security events.

However, to modify the behavior of the Security Audit function by including additional or excluding events from being audited the administrator has to stop the Security Audit process, modify the Security Audit function and start the Security Audit process again (FAU_SEL.1-NIAP-0407).

7.5 Session Handling (SF.SE)

After a user attempting to establish a session has been successfully authenticated by SF.I&A this security functionality decides whether this user is actually allowed to establish a session to the TOE.

The TOE uses two sets of additional criteria to decide whether a user is allowed to establish a session. First the TOE enforces a limit of the number of concurrent sessions a user is allowed to have at one time. This limit is set to 5 by default but can be modified by authorized administrators as described in

SF.SM. If a user reached the limit of concurrent sessions the TOE will deny establishing another session for that user (FTA_MCS.1).

As a second criterion the admin is able to specify a set of rules to explicitly deny session establishment based on:

- User's identity,
- Time of the day, and
- Day of the week.

The TOE only establishes a session for a user if no explicit deny rule for that user has been specified (FTA_TSE.1).

For every attempt to establish a session (whether successful or not) the TOE stores the date and time of the event and the number of unsuccessful attempts since the last successful attempt (FTA_TAH_(EXT).1).

After the TOE established a session to a user the user context is held in a context with limited permission. SF.SE maintains a separate context for the execution of each operation by a user. As soon as a user performs an operation on an object the TOE starts at least one thread to perform this operation.

When the TOE reuses memory which could contain previous information content and which is related to the context of a user's session (i.e. to which a user could gain access), this previous information will not be available for any user. To ensure this, the TOE either directly overwrites any memory that will be used for users sessions completely with new information or with a certain pattern. Before the previous information has been overwritten the resource is not available for any usage. For memory which is allocated using the Operating System the TOE uses a function of the OS, which ensures that only empty memory is provided to the TOE. Whenever data is written to or loaded from disc this is done page wise where a page has the size of 8 KB (FDP_RIP.1).

8 Appendix

8.1 Concept of Ownership Chains

Database Objects within the TOE are not always only passive objects. Some objects refer to other objects. This is especially true for Stored Procedures and Views. When multiple database objects access each other sequentially, the sequence is known as a chain. Although such chains do not independently exist, when the TOE traverses the links in a chain, the TOE evaluates access permissions on the constituent objects differently than it would if it were accessing the objects separately. These differences have important implications for managing security.

Ownership chaining enables managing access to multiple objects, such as multiple tables, by setting permissions on one object, such as a view. Ownership chaining also offers a slight performance advantage in scenarios that allow for skipping permission checks.

8.1.1 How Permissions Are Checked in a Chain

When an object is accessed through a chain, the TOE first compares the owner of the object to the owner of the calling object. This is the previous link in the chain. If both objects have the same owner, permissions on the referenced object are not evaluated. In the context of the Discretionary Access Control Mechanism this is not a circumvention of access control as the owner of an object always has complete control over his objects. So if one user is the owner of both objects, the calling object and the called object, the owner also would have direct access to both objects.

8.1.2 Example of Ownership Chaining

In the following illustration, the July2003 view is owned by Mary. She has granted to Alex permissions on the view. He has no other permissions on database objects in this instance. What happens when Alex selects the view?

Alex executes SELECT * on the July2003 view. The TOE checks permissions on the view and confirms that Alex has permission to select on it.

The July 2003 view requires information from the SalesXZ view. The TOE checks the ownership of the SalesXZ view. Because this view has the same owner (Mary) as the view that calls it, permissions on SalesXZ are not checked. The required information is returned.

The SalesXZ view requires information from the InvoicesXZ view. The TOE checks the ownership of the InvoicesXZ view. Because this view has the same owner as the previous object, permissions on InvoicesXZ are not checked. The required information is returned. To this point, all items in the sequence have had one owner (Mary). This is known as an unbroken ownership chain.

The InvoicesXZ view requires information from the AcctAgeXZ view. The TOE checks the ownership of the AcctAgeXZ view. Because the owner of this view is different from the owner of the previous object (Sam, not Mary), full information about permissions on this view is retrieved. If the AcctAgeXZ view has permissions that allow access by Alex, information will be returned.

The AcctAgeXZ view requires information from the ExpenseXZ table. The TOE checks the ownership of the ExpenseXZ table. Because the owner of this table is different from the owner of the previous object (Joe, not Sam), full information about permissions on this table is retrieved. If the ExpenseXZ table has permissions that allow access by Alex, information is returned.

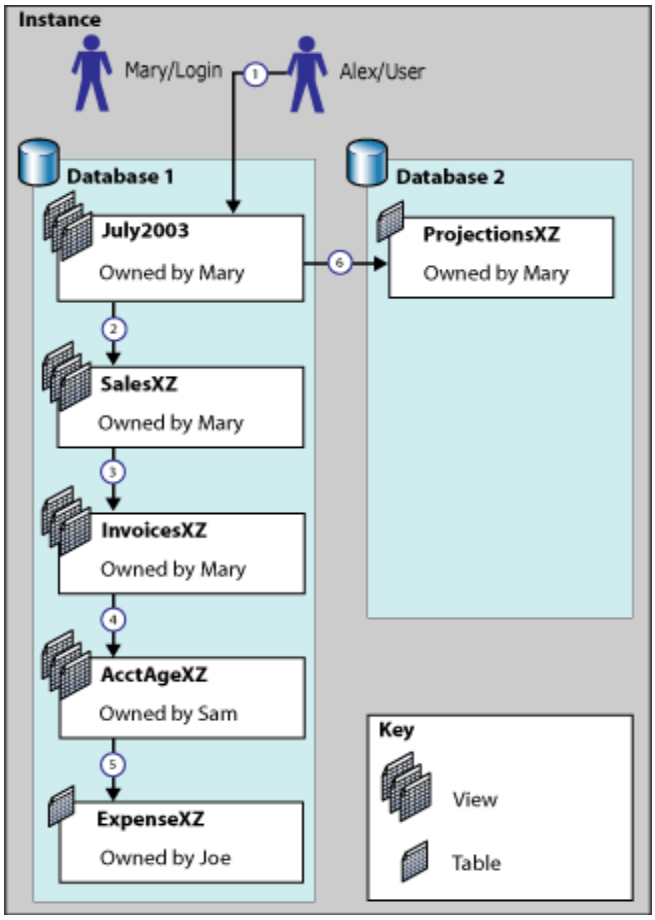

Figure 3 - Concept of Ownership Chaining

When the July2003 view tries to retrieve information from the ProjectionsXZ table, the TOE first checks to see whether cross-database chaining is enabled between Database 1 and Database 2. If cross-database chaining is enabled, the TOE will check the ownership of the ProjectionsXZ table. Because this table has the same owner as the calling view (Mary), permissions on this table are not checked. The requested information is returned.

8.2 References

The following documentation was used to prepare this ST:

[AGD]	Microsoft SQL Server 2012 Books Online
[AGD_ADD]	Microsoft SQL Server 2012 Database Engine Common Criteria Evaluation – Guidance Addendum (delivered via https://www.microsoft.com/sqlserver/en/us/common-criteria.aspx (tab “SQL Server 2012 SP1”)
[CC]	Common Criteria for Information Technology Security Evaluation
	Part 1: Introduction and general model, dated July 2009, version 3.1 R3
	Part 2: Security functional requirements, dated July 2009, version 3.1, R3
	Part 3: Security assurance requirements, dated July 2009, version 3.1, R3
[PP]	U.S. Government Protection Profile for Database Management Systems, Version 1.3, December 24, 2010

8.3 Glossary and Abbreviations

8.3.1 Glossary

The following terms are used in this Security Target:

Term	Definition
Attacker	The term attacker refers to any individual (or technical entity) that is attempting to subvert the security functionality of the TOE. In this Security Target it is assumed that the attacker has an attack potential of “enhanced basic”.
Authorized Administrators	This term refers to a group of users which comprise the “sysadmin” (sa) and any user who is allowed to perform a management operation because the permission has been granted to him within the DAC either by assigning him to a role with administrator permissions or by granting him the possibility to perform an administrative operation explicitly.
DAC	Discretionary Access Control is a mechanism to limit the access of users to objects based on the ID of the user, the ID of the object and a set of access control rules.
DBMS	A DBMS is a computerized repository that stores information and allows authorized users to retrieve and update that information.
Named Pipe	Method for inter process communication.
Object	An object within the TOE contains data and can be accessed by subjects. However in the TOE an object is not necessarily only a passive entity as some objects refer to other objects.
OC	Ownership Chaining.
SQL	The Structured Query Language is a language which can be used to create, modify and retrieve data from a DBMS.
SQL Server	SQL Server is a product of Microsoft to which the TOE belongs.
TDS	Tabular Data Stream is a data format which is used for communication with the TOE.
T-SQL	Extension of the SQL language in order to support control flow, variables, user authentication and various other functions.
User	The term user refers to technical entities (e.g. applications, other instances of the TOE) or human users (using a SQL-client) that are using the services of the TOE.

8.3.2 Abbreviations

The following abbreviations are used in this Security Target:

Abbreviation	Definition
ACL	Access Control List
CC	Common Criteria
DAC	Discretionary Access Control

Abbreviation	Definition
DBMS	Database Management System
EAL	Evaluation Assurance Level
ETL	Extract, Transform, Load
IT	Information Technology
MOM	Microsoft Operations Manager
MS	Microsoft
NIAP	National Information Assurance Partnership
NSA	National Security Agency
OLAP	Online analytical processing
OS	Operating System
OSP	Organizational Security Policy
PP	Protection Profile
SAR	Security Assurance Requirement
SF	Security Functionality
SFR	Security Functional Requirement
SID	Security ID
SMS	System Management Server
SQL	Structured Query Language
ST	Security Target
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functionality
T-SQL	Transact SQL