IBM i5/OS V5R3
Security Target

Version 1.0

07/08/05

Prepared for:
International Business Machines Corporation
Rochester, MN

Prepared By:
Science Applications International Corporation
Common Criteria Testing Laboratory
7125 Columbia Gateway Drive, Suite 300
Columbia, MD 21046

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is provided by International Business Machines Corporation. The TOE is the IBM i5/OS V5R3 proprietary operating system on an IBM iSeries hardware platform.

The Security Target contains the following additional sections:

- TOE Description (Section 2)
- Security Environment (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8).

1.1 Security Target, TOE and CC Identification

ST Title – IBM i5/OS V5R3 Security Target

ST Version – Version 1.0

ST Date – 07/08/05

TOE Identification – IBM i5/OS V5R3M0 running on IBM eServer models 520, 550, and 570 with software feature code 1930

CC Identification – Common Criteria for Information Technology Security Evaluation, Version 2.1, August 1999, ISO/IEC 15408.

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

- Controlled Access Protection Profile (CAPP), Version 1.d, 8 October 1999.
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional requirements, Version 2.1, August 1999, ISO/IEC 15408-2.
    - Part 2 Conformant
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance requirements, Version 2.1, August 1999, ISO/IEC 15408-3.
    - Part 3 Conformant
    - EAL 4 augmented with ALC_FLR.2

1.3 Conventions, Terminology, Acronyms

This section specifies the formatting information used in the Security Target.

1.3.1 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
    - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by a letter in parenthesis placed at the end of the component. For example FDP_ACC.1a and FDP_ACC.1b indicate that the ST includes two iterations of the FDP_ACC.1 requirement, a and b.
    - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]).
    - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).
    - Refinement: allows the addition of details. Refinements are indicated using bold, for additions and replacements, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …").
- If an operation was completed in a related Protection Profile or Interpretation, the operation will not be highlighted. Rather, the corresponding PP or Interpretation should be consulted to determine what operations might have already been performed. Note that Interpretations are identified where they are applied.

Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

1.3.2 Terminology and Acronyms

The following terminology and acronyms are used throughout the ST. Reference the U.S. Government Controlled Access Protection Profile (CAPP), Version 1.d for additional terms.

- Authorities – permissions
- IFS – Integrated File System
- IPL – Initial Program Load
- MI – Machine Interface
- SLIC – Software Licensed Internal Code
- Special authorities – privileges

2. TOE Description

The Target of Evaluation (TOE) is the IBM i5/OS V5R3 operating system, supporting hardware (specifically the IBM eServer), and those applications included with i5/OS necessary to manage, support, and configure the operating system.

2.1 TOE Overview

i5/OS is a complete operating system that operates on an IBM iSeries hardware platform. i5/OS is object-based and, in the evaluated configuration, implements approximately 50 object types. Data access and system management is controlled via access controls on the available objects, but only after the responsible user has been identified and authenticated by i5/OS and if the user has the required authorities. Additionally, i5/OS can audit security-relevant actions, including authentication attempts, access attempts, and security management functions.

Note that while the TOE is being identified as i5/OS running on an IBM eServer hardware platform, it should be understood that the evaluated configuration is actually i5/OS running on an IBM eServer hardware platform configured as described in the IBM i5/OS Installation Guide. Furthermore, the TOE is a subset of the appropriately configured product since the product includes a number of applications (i.e., the yellow area in Figure 1, below) that fall outside the scope of the TOE, and hence have not been evaluated, because they can have no effect on the security functions TSF.

2.2 TOE Architecture

Figure 1 depicts the basic i5/OS architecture. Like most other operating systems, i5/OS consists of layers ranging

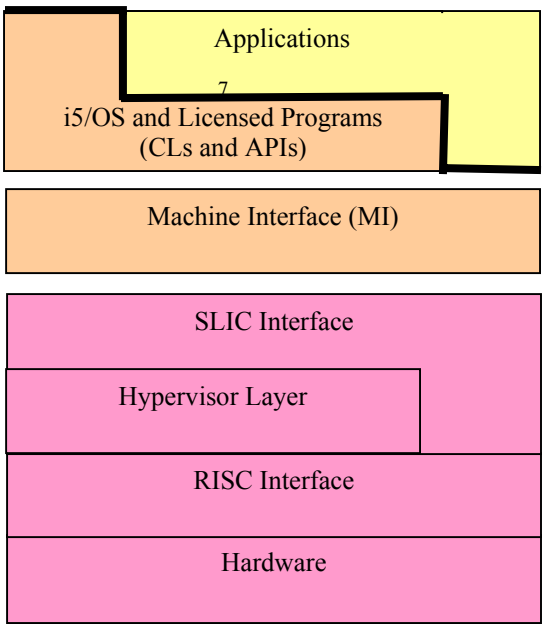

Figure 1 Basic i5/OS Architecture

from the most critical (hardware) to non-critical (user applications). The hardware is an IBM iSeries product and the lower layers of i5/OS are designed to abstract hardware details away from the higher layers of i5/OS. As a result, the SLIC and MI interfaces are essentially static regardless of the underlying hardware and it is these interfaces upon which i5/OS and user applications operate. Note that the TOE includes all of the elements depicted in Figure 1 with the exception of "Applications." Applications though controlled by the TOE, are assumed to be outside the scope of the TOE since they can have no effect on the operation of the TSF.

The iSeries running i5/OS interacts with an IT component. Specifically, a co-processor developed by IBM is used to orchestrate the i5/OS IPL and to perform diagnostic-type testing of i5/OS to ensure that it is operating properly. However, this co-processor does not play a role in the enforcement of the security policies offered by i5/OS and is considered part of the IT environment.

2.2.1 Physical Boundaries

The TOE is physically composed of 1) the IBM eServer models 520, 550, and 570 machine on which the i5/OS software operates; and 2) a console and keyboard, physically connected to the iSeries machine. The eSeries machine includes memory, disk drives, integrated network and disk controllers, tape drive, and CD-ROM drive. There are no additional external peripherials besides the console and keyboard. The IBM coprocessor referred to in Section 2.2 is physically attached to the eServer machine but is considered part of the IT Environment. Client workstations are connected to the iSeries machine, but are also considered part of the IT Environment.

Therefore, the physical boundaries of the TOE occur at 1) the eServer machine, 2) the console and keyboard, 3) the interface between the iSeries machine and the IBM coprocessor; 4) the interface between the iSeries machine and client workstations.

The following diagram is provided to help illustrate the separation between the TOE and its IT Environment. The TOE components are displayed in the dark grey shaded boxes, while the IT environment components are displayed in the light grey shaded boxes.

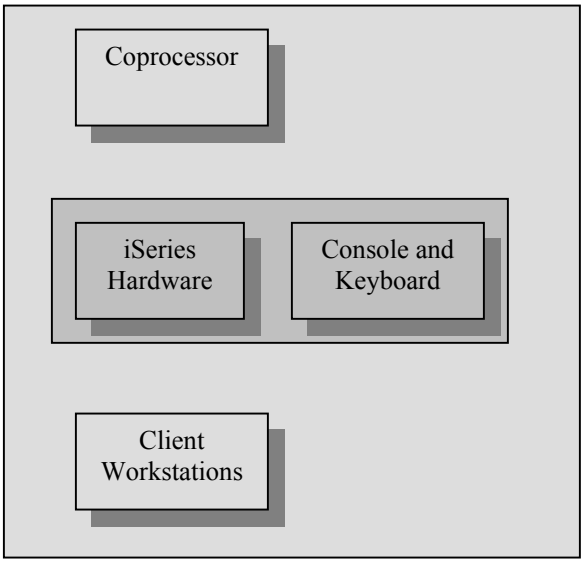

Figure 2 Physical TOE Boundaries

Physically, the iSeries hardware is connected to workstations that provide a primary interactive user interface, and a network, that offers network-oriented user services. Once connected to one of these interfaces, i5/OS software offers command line commands (CLs), application interfaces (APIs), and machine interface instructions (MIs) via available prompts, menus, and programs.

2.2.2 Logical Boundaries

The logical boundaries of i5/OS can be characterized as the set of security functions available at its physical interfaces. Each of these security functions is summarized below and discussed in greater detail later in this Security Target.

2.2.2.1 Security Audit

i5/OS has an audit mechanism that is invoked for access checks, authentication attempts, administrator functions, and at other times during its operation. When invoked, the date, time, responsible individual and other details describing the event are records to the audit trail.

i5/OS can be configured to halt when data can't be written to the audit log or to discard the data and continue processing. i5/OS can also be configured so that when audit log spaces (called journal receivers) fill, the system will automatically generate new ones so that audit data is not lost. The audit trail is composed of potentially many journal receivers. The audit entries associated with a filled journal receiver remain available for administrator review. Note that i5/OS allows an Administrator to configure an audit level parameter so that only selected types of auditable events will be collected and undesired audit events will not cause the audit trail to become full unnecessarily.

Tools are provided so that an administrator can effectively review the audit trail, including searching and sorting by user identities.

2.2.2.2 User Data Protection

i5/OS is object oriented and implements approximately 50 object types. Each of the objects has associated operations and access modes that can be configured so that individual users and groups of users can be restricted so that they can perform only selected operations on given objects.

2.2.2.3 Identification and Authentication

In the evaluated configuration, each user must provide a user name and password before they are allowed to exercise any i5/OS commands, regardless of the mechanism used to communicate with i5/OS. Once a user has been authenticated, i5/OS maintains the identity and other attributes with the resulting session to ensure proper access controls are enforced and individual accountability is maintained.

2.2.2.4 Security Management

i5/OS offers an extensive set of tools to manage and otherwise use its security services. i5/OS supports the notion of roles by assigning various special authorities to specific users. Access to essentially all of the i5/OS objects, including those used to store and manipulate the i5/OS security configuration, are protected using these authorities in conjunction with a discretionary access control policy.

2.2.2.5 Protection of the TOE Security Functions

Diagnostic tests exist to ensure that the hardware is functioning correctly. Some of the tests execute automatically during i5/OS initial program load (IPL) and additional tests can be exercised by an authorized administrator when necessary.

i5/OS protects itself using a combination of hardware support and strict control over the set of available applications. i5/OS includes a translator and compiler that are specifically designed to ensure that a given program will only access resources it is supposed to (e.g., the application will not be allowed to access memory from another user or system process).

i5/OS is object-based and provides a number of well-defined interfaces to access each object. Objects can only be accessed through the interfaces provided and those interfaces have been carefully designed to ensure that the appropriate access checks are made before they operate on any object.

2.3 TOE Documentation

In order to ensure that i5/OS can be operated securely, IBM has developed guidance for users and administrators to help them use the available security functions properly. The specific documents available for this purpose include:

- IBM iSeries Configure Your System For Common Criteria Security, Version 5 Release 3, SC41-5336-00
- iSeries Security Reference, Version 5, SC41-5302-07
- The iSeries Information Center
  - http://publib.boulder.ibm.com/pubs/html/as400/infocenter.html
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk631usersviewsecurity.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/apis/api.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbam6/rbam6clmain.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rzaiu/rzaiuicbackup.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/db2/rbafzmst02.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rzai2/rzai2kickoff.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rzahgictcp2.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk003planninguser.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapkaaacomface.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkprvaut.htm#prvaut
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk103physicalsec.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk004planningresource.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapkc17.htm

3. Security Environment

This section defines the security policies the TOE, in conjunction with its environment, is intended to fulfill as well as usage assumptions about the TOE's intended environment.

3.1 Organizational Policies

P.ACCOUNTABILITY The users of the system shall be held accountable for their actions within the system.

P.AUTHORIZED_USERS Only those users who have been authorized to access the information within the system may access the system.

P.NEED_TO_KNOW The system must limit the access to, modification of, and destruction of the information in protected resources to those authorized users which have a 'need to know' for that information.

3.2 Assumptions

A.CONNECT	All connections to peripheral devices reside within the controlled access facilities. CAPP-conformant TOEs only address security concerns related to the manipulation of the TOE through its authorized access points. Internal communication paths to access points such as terminals are assumed to be adequately protected.
A.COOP	Authorized users possess the necessary authorization to access at least some of the information managed by the TOE and are expected to act in a cooperating manner in a benign environment.
A.LOCATE	The processing resources of the TOE will be located within controlled access facilities which will prevent unauthorized physical access.
A.MANAGE	There will be one or more competent individuals assigned to manage the TOE and the security of the information it contains.
A.NO_EVIL_ADM	The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the administrator documentation.
A.PEER	Any other systems with which the TOE communicates are assumed to be under the same management control and operate under the same security policy constraints. CAPP-conformant TOEs are applicable to networked or distributed environments only if the entire network operates under the same constraints and resides within a single management domain. There are no security requirements which address the need to trust external systems or the communications links to such systems.
A.PROTECT	The TOE hardware and software critical to security policy enforcement will be protected from unauthorized physical modification.

4. Security Objectives

This section defines the security objectives of the TSF and its supporting environment. Security objectives, categorized as being either for the TOE or Environment, reflect the stated intent to counter identified threats and/or comply with any organizational security policies identified. All of the identified organizational policies and assumptions are addressed under one of the categories below.

4.1 Security Objectives for the TOE

O.AUDITING The TSF must record the security relevant actions of users of the TOE. The TSF must present this information to authorized administrators.

O.AUTHORIZATION The TSF must ensure that only authorized users gain access to the TOE and its resources.

O.DISCRETIONARY_ACCESS The TSF must control access to resources based on identity of users. The TSF must allow authorized users to specify which resources may be accessed by which users.

O.ENFORCEMENT The TSF must be designed and implemented in a manner which ensures that the organizational policies are enforced in the target environment.

O.MANAGE The TSF must provide all the functions and facilities necessary to support the authorized administrators that are responsible for the management of TOE security.

O.RESIDUAL_INFORMATION The TSF must ensure that any information contained in a protected resource is not released when the resource is recycled.

4.2 Security Objectives for the Environment

O.CREDEN Those responsible for the TOE must ensure that all access credentials, such as passwords or other authentication information, are protected by the users in a manner which maintains IT security objectives.

O.INSTALL Those responsible for the TOE must ensure that the TOE is delivered, installed, managed, and operated in a manner which maintains IT security objectives.

O.PHYSICAL Those responsible for the TOE must ensure that those parts of the TOE critical to security policy are protected from physical attack which might compromise IT security objectives.

5. IT Security Requirements

5.1 TOE Security Functional Requirements

The following table describes the SFRs that are candidates to be satisfied by i5/OS.

Requirement Class	Requirement Component
FAU: Security Audit	FAU_GEN.1: Audit Data Generation
	FAU_GEN.2: User Identity Association
	FAU_SAR.1: Audit Review
	FAU_SAR.2: Restricted Audit Review
	FAU_SAR.3: Selectable Audit Review
	FAU_SEL.1: Selective Audit
	FAU_STG.1: Guarantees of Audit Data Availability
	FAU_STG.3: Action in Case of Possible Audit Data Loss
	FAU_STG.4: Prevention of Audit Data Loss
FDP: User Data Protection	FDP_ACC.1: Discretionary Access Control Policy
	FDP_ACF.1: Discretionary Access Control Functions
	FDP_RIP.2a: Object Residual Information Protection
	FDP_RIP.2b: Subject Residual Information Protection
FIA: Identification and Authentication	FIA_ATD.1: User Attribute Definition
	FIA_SOS.1: Strength of Authentication Data
	FIA_UAU.2: User authentication before any action
	FIA_UAU.7: Protected Authentication Feedback
	FIA_UID.2: User identification before any action
	FIA_USB.1: User-Subject Binding
FMT: Security Management	FMT_MSA.1: Management of Object Security Attributes
	FMT_MSA.3: Static Attribute Initialization
	FMT_MTD.1a: Management of the Audit Trail
	FMT_MTD.1b: Management of Audited Events
	FMT_MTD.1c: Management of User Attributes
	FMT_MTD.1d: Management of Authentication Data
	FMT_REV.1a: Revocation of User Attributes
	FMT_REV.1b: Revocation of Object Attributes
	FMT_SMR.1: Security Management Roles
FPT: Protection of the TOE Security Functions	FPT_AMT.1: Abstract Machine Testing
	FPT_RVM.1: Reference Mediation
	FPT_SEP.1: Domain Separation
	FPT_STM.1: Reliable Time Stamps

Table 1 Security Functional Components

5.1.1 Security Audit (FAU)

5.1.1.1 Audit Data Generation (FAU_GEN.1)

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the auditable events listed in column 'Event' of **the following table**. This includes all auditable events for the basic level of audit, except FIA_UID.**2**'s user identity during failures.

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information: a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; b) The additional information specified in the 'Details' column of **the following table**.

Component	Event	Details
FAU_GEN.1	Start-up and shutdown of the audit functions.	
FAU_GEN.2	None	
FAU_SAR.1	Reading of information from the audit records.	
FAU_SAR.2	Unsuccessful attempts to read information from the audit records.	
FAU_SAR.3	None	
FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	
FAU_STG.2	None	
FAU_STG.3	Actions taken due to exceeding of a threshold.	
FAU_STG.4	Actions taken due to the audit storage failure.	
FDP_ACC.1	None	
FDP_ACF.1	All requests to perform an operation on an object covered by the SFP.	The identity of the object.
FDP_RIP.2a	None	
FDP_RIP.2b	None	
FIA_ATD.1	None	
FIA_SOS.1	Rejection or acceptance by the TSF of any tested secret.	
FIA_UAU.2	All use of the authentication mechanism.	
FIA_UAU.7	None	
FIA_UID.2	All use of the user identification mechanism, including the identity provided during successful attempts.	The origin of the attempt (e.g. terminal identification.)
FIA_USB.1	Success and failure of binding user security attributes to a subject (e.g. success and failure to create a subject).	
FMT_MSA.1	All modifications of the values of security attributes.	
FMT_MSA.3	Modifications of the default setting of permissive or restrictive rules.	All modifications of the initial value of security attributes.
FMT_MTD.1a	All modifications to the values of TSF data.	
FMT_MTD.1b	All modifications to the values of TSF data.	The new value of the TSF data.
FMT_MTD.1c	All modifications to the values of TSF data.	The new value of the TSF data.
FMT_MTD.1d	All modifications to the values of TSF data.	
FMT_REV.1a	All attempts to revoke security attributes.	
FMT_REV.1b	All modifications to the values of TSF data.	
FMT_SMR.1	Modifications to the group of users that are part of a role.	
FMT_SMR.1	Every use of the rights of a role. (Additional / Detailed)	The role and the origin of the request.
FPT_AMT.1	Execution of the tests of the underlying machine and the results of the test.	
FPT_RVM.1	None	
FPT_SEP.1	None	
FPT_STM.1	Changes to the time.	

5.1.1.2 User Identity Association (FAU_GEN.2)

FAU_GEN.2.1 The TSF shall be able to associate each auditable event with the identity of the user that caused the event.

5.1.1.3 Audit Review (FAU_SAR.1)

FAU_SAR.1.1 The TSF shall provide authorized administrators with the capability to read all audit information from the audit records.

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

5.1.1.4 Restricted Audit Review (FAU_SAR.2)

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

5.1.1.5 Selectable Audit Review (FAU_SAR.3)

FAU_SAR.3.1 The TSF shall provide the ability to perform **[*searches* and *sorting*]** of audit data based on the following attributes: a) User identity; b) **[no additional attributes]**.

5.1.1.6 Selective Audit (FAU_SEL.1)

FAU_SEL.1.1 The TSF shall be able to include or exclude auditable events from the set of audited events based on the following attributes: a) User identity; b) **[object, and c) event type]**.

5.1.1.7 Guarantees of Audit Data Availability (FAU_STG.1)

FAU_STG.1.1 The TSF shall protect the stored audit records from unauthorized deletion.

FAU_STG.1.2 The TSF shall be able to **[*prevent*]** unauthorised modifications to the audit records in the audit trail. *(per International Interpretation #141)*

5.1.1.8 Action in Case of Possible Audit Data Loss (FAU_STG.3)

FAU_STG.3.1 The TSF shall generate an alarm to the authorized administrator if the audit trail exceeds **[the available mass storage capacity]**.

5.1.1.9 Prevention of Audit Data Loss (FAU_STG.4)

FAU_STG.4.1 The TSF shall be able to prevent auditable events, except those taken by the authorized administrator, and **[notify an authorized administrator]** if the audit trail is full.

5.1.2 User Data Protection (FDP)

5.1.2.1 Discretionary Access Control Policy (FDP_ACC.1)

FDP_ACC.1.1 The TSF shall enforce the Discretionary Access Control Policy on **[processes]** acting on the behalf of users, **[the following objects: block stream file, binding directory, class, command, controller description, distributed file directory, device description, FPFS directory, data area, data dictionary, data queue, edit description, exit registration, file, font mapping table, generic filter, job description, job queue, job schedule, journal, journal receiver, library, line description, locale space, menu definition, module, message file, message queue, output queue, page definition, program, query manager form, query manager query, query definition, subsystem description, SQL package, service program, bytestream file, symbolic link, table, time zone description, user index, user profile, user queue, and user space]** and all operations among subjects and objects covered by the DAC policy.

5.1.2.2 Discretionary Access Control Functions (FDP_ACF.1)

FDP_ACF.1.1 The TSF shall enforce the **[Discretionary Access Control Policy]** to objects based on the following:

a.) **The following access control attributes associated with a subject: authorities from associated user profiles, group profiles, and adoption from a program owner;** and

b.) **The following access control attributes associated with an object:** [**authorities from the associated owner profile and authorization list**]. *(per International Interpretation #103)*

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **[the process must have authorities sufficient to perform the requested operation based on: authorities held in the owner profile of the object when the associated user is the object owner, authorities held in the user profile, authorities granted to the associated user by the object authorization list, authorities held in an associated group profile, authorities granted to an associated group by the object authorization list, authorities granted to all users by the object authorization list, or authorities granted by adoption from that program owner's profile]**.

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based in the following additional rules: **[a process with all object special authority can always access the selected object]**.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the **[assignment of the *EXCLUDE authority will deny a process access to an object and its data]**.

5.1.2.3 Object Residual Information Protection (FDP_RIP.2a)

FDP_RIP.2a.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to all objects.

5.1.2.4 Subject Residual Information Protection (FDP_RIP.2b)

FDP_RIP.2b.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to all subjects.

5.1.3 Identification and Authentication (FIA)

5.1.3.1 User Attribute Definition (FIA_ATD.1)

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: a) User Identifier; b) Group Memberships; c) Authentication Data; d) Security-relevant Roles; and e) **[authorities]**.

5.1.3.2 Strength of Authentication Data (FIA_SOS.1)

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets meet the following: a) For each attempt to use the authentication mechanism, the probability that a random attempt will succeed is less than one in 1,000,000; b) For multiple attempts to use the authentication mechanism during a one minute period, the probability that a random attempt during that minute will succeed is less than one in 100,000; and c) Any feedback given during an attempt to use the authentication mechanism will not reduce the probability below the above metrics.

5.1.3.3 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 the behalf of that user.

5.1.3.4 Protected Authentication Feedback (FIA_UAU.7)

FIA_UAU.7.1 The TSF shall provide only obscured feedback to the user while the authentication is in progress.

5.1.3.5 User Identification Before Any Action (FIA_UID.2)

FIA_UID.2.1 The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on the behalf of that user.

5.1.3.6 User-Subject Binding (FIA_USB.1)

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: a) The user identity which is associated with auditable events; b) The user identity or identities which are used to enforce the Discretionary Access Control Policy; c) The group membership or memberships used to enforce the Discretionary Access Control Policy; d) **[authorities]**.

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of a user: a) **[all initial subject security attributes are taken directly from the associated user's profile]**.

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of a user: a) **[subjects can invoke programs that allow the adoption authorities of the program owner if explicitly allowed by the program (object) attributes and authorized administrators can change any subject's security attributes]**.

5.1.4 Security Management (FMT)

5.1.4.1 Management of Object Security Attributes (FMT_MSA.1)

FMT_MSA.1.1 The TSF shall enforce the Discretionary Access Control Policy to restrict the ability to modify the access control attributes associated with a named object to **[**users authorized by the Discretionary Access Control Policy to modify object security attributes**]**.

5.1.4.2 Static Attribute Initialization (FMT_MSA.3)

FMT_MSA.3.1 The TSF shall enforce the Discretionary Access Control Policy to provide restrictive default values for security attributes that are used to enforce the Discretionary Access Control Policy.

FMT_MSA.3.2 The TSF shall allow the **[the object creator and an authorized administrator]** to specify alternative initial values to override the default values when an object or information is created.

5.1.4.3 Management of the Audit Trail (FMT_MTD.1a)

FMT_MTD.1a.1 The TSF shall restrict the ability to create, delete, and clear the audit trail to authorized administrators.

5.1.4.4 Management of Audited Events (FMT_MTD.1b)

FMT_MTD.1b.1 The TSF shall restrict the ability to modify or observe the set of audited events to authorized administrators.

5.1.4.5 Management of User Attributes (FMT_MTD.1c)

FMT_MTD.1c.1 The TSF shall restrict the ability to initialize and modify the user security attributes, other than authentication data, to authorized administrators.

5.1.4.6 Management of Authentication Data (FMT_MTD.1d)

FMT_MTD.1d.1 The TSF shall restrict the ability to initialize the authentication data to authorized administrators.

FMT_MTD.1d.2 The TSF shall restrict the ability to modify the authentication data to the following: a) authorized administrators; and b) users authorized to modify their own authentication data.

5.1.4.7 Revocation of User Attributes (FMT_REV.1a)

FMT_REV.1a.1 The TSF shall restrict the ability to revoke security attributes associated with the users within the TSC to authorized administrators.

FMT_REV.1a.2 The TSF shall enforce the rules: a) The immediate revocation of security-relevant authorizations; and b) **[no additional revocation rules concerning users]**.

5.1.4.8 Revocation of Object Attributes (FMT_REV.1b)

FMT_REV.1b.1 The TSF shall restrict the ability to revoke security attributes associated with objects within the TSC to users authorized to modify the security attributes by the Discretionary Access Control policy.

FMT_REV.1b.2 The TSF shall enforce the rules: a) The access rights associated with an object shall be enforced when an access check is made; and b) **[no additional revocation rules concerning objects]**.

5.1.4.9 Security Management Roles (FMT_SMR.1)

FMT_SMR.1.1 The TSF shall maintain the roles: a) authorized administrator; b) users authorized by the Discretionary Access Control Policy to modify object security attributes; c) users authorized to modify their own authentication data; and d) **[no other roles]**.

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

5.1.5 Protection of the TOE Security Functions (FPT)

5.1.5.1 Abstract Machine Testing (FPT_AMT.1)

FPT_AMT.1.1 The TSF shall run a suite of tests **[*during initial start-up or at the request of an authorized administrator*]** to demonstrate the correct operation of the security assumptions provided by the abstract machine that underlies the TSF.

5.1.5.2 Reference Mediation (FPT_RVM.1)

FPT_RVM.1.1 The TSF shall ensure that the TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.1.5.3 Domain Separation (FPT_SEP.1)

FPT_SEP.1.1 The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1.2 The TSF shall enforce separation between the security domains of subjects in the TSC.

5.1.5.4 Reliable Time Stamps (FPT_STM.1)

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.

5.2 TOE Security Assurance Requirements

The security assurance requirements for the TOE are the EAL 4 augmented with ALC_FLR.2 components as specified in Part 3 of the Common Criteria. No operations are applied to the assurance components.

Requirement Class	Requirement Component
ACM: Configuration management	ACM_AUT.1: Partial CM automation
	ACM_CAP.4: Generation support and acceptance procedures
	ACM_SCP.2: Problem tracking CM coverage
ADO: Delivery and operation	ADO_DEL.2: Detection of modification
	ADO_IGS.1: Installation, generation, and start-up procedures
ADV: Development	ADV_FSP.2: Fully defined external interfaces
	ADV_HLD.2: Security enforcing high-level design
	ADV_IMP.1: Subset of the implementation of the TSF
	ADV_LLD.1: Descriptive low-level design
	ADV_RCR.1: Informal correspondence demonstration

	ADV_SPM.1: Informal TOE security policy model
AGD: Guidance documents	AGD_ADM.1: Administrator guidance
	AGD_USR.1: User guidance
ALC: Life cycle support	ALC_DVS.1: Identification of security measures
	ALC_FLR.2: Flaw reporting procedures
	ALC_LCD.1: Developer defined life-cycle model
	ALC_TAT.1: Well-defined development tools
ATE: Tests	ATE_COV.2: Analysis of coverage
	ATE_DPT.1: Testing: high-level design
	ATE_FUN.1: Functional testing
	ATE_IND.2: Independent testing - sample
AVA: Vulnerability assessment	AVA_MSU.2: Validation of analysis
	AVA_SOF.1: Strength of TOE security function evaluation
	AVA_VLA.2: Independent vulnerability analysis

Table 2 EAL 4 augmented with ALC_FLR.2 Assurance Components

5.2.1 Configuration management (ACM)

5.2.1.1 Partial CM automation (ACM_AUT.1)

ACM_AUT.1.1d The developer shall use a CM system.
ACM_AUT.1.2d The developer shall provide a CM plan.
ACM_AUT.1.1c The CM system shall provide an automated means by which only authorised changes are made to the TOE implementation representation.
ACM_AUT.1.2c The CM system shall provide an automated means to support the generation of the TOE.
ACM_AUT.1.3c The CM plan shall describe the automated tools used in the CM system.
ACM_AUT.1.4c The CM plan shall describe how the automated tools are used in the CM system.
ACM_AUT.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.1.2 Generation support and acceptance procedures (ACM_CAP.4)

ACM_CAP.4.1d The developer shall provide a reference for the TOE.
ACM_CAP.4.2d The developer shall use a CM system.
ACM_CAP.4.3d The developer shall provide CM documentation.
ACM_CAP.4.1c The reference for the TOE shall be unique to each version of the TOE.
ACM_CAP.4.2c The TOE shall be labelled with its reference.
ACM_CAP.4.3c The CM documentation shall include a configuration list, a CM plan, and an acceptance plan.
ACM_CAP.4.4c The configuration list shall describe the configuration items that comprise the TOE.
ACM_CAP.4.5c The CM documentation shall describe the method used to uniquely identify the configuration items.
ACM_CAP.4.6c The CM system shall uniquely identify all configuration items.
ACM_CAP.4.7c The CM plan shall describe how the CM system is used.
ACM_CAP.4.8c The evidence shall demonstrate that the CM system is operating in accordance with the CM plan.
ACM_CAP.4.9c The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.
ACM_CAP.4.10c The CM system shall provide measures such that only authorised changes are made to the configuration items.
ACM_CAP.4.11c The CM system shall support the generation of the TOE.
ACM_CAP.4.12c The acceptance plan shall describe the procedures used to accept modified or newly created configuration items as part of the TOE.

ACM_CAP.4.13c The configuration list shall uniquely identify all configuration items that comprise the TOE. *(per International Interpretation #3)*

ACM_CAP.4.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.1.3 Problem tracking CM coverage (ACM_SCP.2)

ACM_SCP.2.1d The developer shall provide a list of configuration items for the TOE. *(per International Interpretation #4)*

ACM_SCP.2.1c The list of configuration items shall include the following: implementation representation; security flaws; and the evaluation evidence required by the assurance components in the ST. *(per International Interpretation #4)*

ACM_SCP.2.2c *(this element has been deleted per International Interpretation #4)*

ACM_SCP.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.2 Delivery and operation (ADO)

5.2.2.1 Detection of modification (ADO_DEL.2)

ADO_DEL.2.1d The developer shall document procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.2.2d The developer shall use the delivery procedures.

ADO_DEL.2.1c The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.

ADO_DEL.2.2c The delivery documentation shall describe how the various procedures and technical measures provide for the detection of modifications, or any discrepancy between the developer's master copy and the version received at the user site.

ADO_DEL.2.3c The delivery documentation shall describe how the various procedures allow detection of attempts to masquerade as the developer, even in cases in which the developer has sent nothing to the user's site.

ADO_DEL.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.2.2 Installation, generation, and start-up procedures (ADO_IGS.1)

ADO_IGS.1.1d The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

ADO_IGS.1.1c The installation, generation and start-up documentation shall describe all the steps necessary for secure installation, generation and start-up of the TOE. *(per International Interpretation #51 (rev 1))*

ADO_IGS.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1.2e The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

5.2.3 Development (ADV)

5.2.3.1 Fully defined external interfaces (ADV_FSP.2)

ADV_FSP.2.1d The developer shall provide a functional specification.

ADV_FSP.2.1c The functional specification shall describe the TSF and its external interfaces using an informal style.

ADV_FSP.2.2c The functional specification shall be internally consistent.

ADV_FSP.2.3c The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing complete details of all effects, exceptions and error messages.

ADV_FSP.2.4c The functional specification shall completely represent the TSF.

ADV_FSP.2.5c The functional specification shall include rationale that the TSF is completely represented.

ADV_FSP.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADV_FSP.2.2e The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements.

5.2.3.2 Security enforcing high-level design (ADV_HLD.2)

ADV_HLD.2.1d The developer shall provide the high-level design of the TSF.
ADV_HLD.2.1c The presentation of the high-level design shall be informal.
ADV_HLD.2.2c The high-level design shall be internally consistent.
ADV_HLD.2.3c The high-level design shall describe the structure of the TSF in terms of subsystems.
ADV_HLD.2.4c The high-level design shall describe the security functionality provided by each subsystem of the TSF.
ADV_HLD.2.5c The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.
ADV_HLD.2.6c The high-level design shall identify all interfaces to the subsystems of the TSF.
ADV_HLD.2.7c The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible.
ADV_HLD.2.8c The high-level design shall describe the purpose and method of use of all interfaces to the subsystems of the TSF, providing details of effects, exceptions and error messages, as appropriate.
ADV_HLD.2.9c The high-level design shall describe the separation of the TOE into TSPenforcing and other subsystems.
ADV_HLD.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADV_HLD.2.2e The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements.

5.2.3.3 Subset of the implementation of the TSF (ADV_IMP.1)

ADV_IMP.1.1d The developer shall provide the implementation representation for a selected subset of the TSF.
ADV_IMP.1.1c The implementation representation shall unambiguously define the TSF to a level of detail such that the TSF can be generated without further design decisions.
ADV_IMP.1.2c The implementation representation shall be internally consistent.
ADV_IMP.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADV_IMP.1.2e The evaluator shall determine that the least abstract TSF representation provided is an accurate and complete instantiation of the TOE security functional requirements.

5.2.3.4 Descriptive low-level design (ADV_LLD.1)

ADV_LLD.1.1d The developer shall provide the low-level design of the TSF.
ADV_LLD.1.1c The presentation of the low-level design shall be informal.
ADV_LLD.1.2c The low-level design shall be internally consistent.
ADV_LLD.1.3c The low-level design shall describe the TSF in terms of modules.
ADV_LLD.1.4c The low-level design shall describe the purpose of each module.
ADV_LLD.1.5c The low-level design shall define the interrelationships between the modules in terms of provided security functionality and dependencies on other modules.
ADV_LLD.1.6c The low-level design shall describe how each TSP-enforcing function is provided.
ADV_LLD.1.7c The low-level design shall identify all interfaces to the modules of the TSF.
ADV_LLD.1.8c The low-level design shall identify which of the interfaces to the modules of the TSF are externally visible.
ADV_LLD.1.9c The low-level design shall describe the purpose and method of use of all interfaces to the modules of the TSF, providing details of effects, exceptions and error messages, as appropriate.
ADV_LLD.1.10c The low-level design shall describe the separation of the TOE into TSP-enforcing and other modules.
ADV_LLD.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_LLD.1.2e The evaluator shall determine that the low-level design is an accurate and complete instantiation of the TOE security functional requirements.

5.2.3.5 Informal correspondence demonstration (ADV_RCR.1)

ADV_RCR.1.1d The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.

ADV_RCR.1.1c For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.

ADV_RCR.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.3.6 Informal TOE security policy model (ADV_SPM.1)

ADV_SPM.1.1d The developer shall provide a TSP model.

ADV_SPM.1.2d The developer shall demonstrate correspondence between the functional specification and the TSP model.

ADV_SPM.1.1c The TSP model shall be informal.

ADV_SPM.1.2c The TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_SPM.1.3c The TSP model shall include a rationale that demonstrates that it is consistent and complete with respect to all policies of the TSP that can be modeled.

ADV_SPM.1.4c The demonstration of correspondence between the TSP model and the functional specification shall show that all of the security functions in the functional specification are consistent and complete with respect to the TSP model.

ADV_SPM.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.4 Guidance documents (AGD)

5.2.4.1 Administrator guidance (AGD_ADM.1)

AGD_ADM.1.1d The developer shall provide administrator guidance addressed to system administrative personnel.

AGD_ADM.1.1c The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE.

AGD_ADM.1.2c The administrator guidance shall describe how to administer the TOE in a secure manner.

AGD_ADM.1.3c The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.

AGD_ADM.1.4c The administrator guidance shall describe all assumptions regarding user behaviour that are relevant to secure operation of the TOE.

AGD_ADM.1.5c The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.

AGD_ADM.1.6c The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_ADM.1.7c The administrator guidance shall be consistent with all other documentation supplied for evaluation.

AGD_ADM.1.8c The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.

AGD_ADM.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.4.2 User guidance (AGD_USR.1)

AGD_USR.1.1d The developer shall provide user guidance.

AGD_USR.1.1c The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE.

AGD_USR.1.2c The user guidance shall describe the use of user-accessible security functions provided by the TOE.
AGD_USR.1.3c The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.
AGD_USR.1.4c The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.
AGD_USR.1.5c The user guidance shall be consistent with all other documentation supplied for evaluation.
AGD_USR.1.6c The user guidance shall describe all security requirements for the IT environment that are relevant to the user.
AGD_USR.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.5 Life cycle support (ALC)

5.2.5.1 Identification of security measures (ALC_DVS.1)

ALC_DVS.1.1d The developer shall produce development security documentation.
ALC_DVS.1.1c The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.
ALC_DVS.1.2c The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.
ALC_DVS.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ALC_DVS.1.2e The evaluator shall confirm that the security measures are being applied.

5.2.5.2 Flaw reporting procedures (ALC_FLR.2)

ALC_FLR.2.1d The developer shall document the flaw remediation procedures.
ALC_FLR.2.2d The developer shall establish a procedure for accepting and acting upon user reports of security flaws and requests for corrections to those flaws.
ALC_FLR.2.1c The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.
ALC_FLR.2.2c The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.
ALC_FLR.2.3c The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.
ALC_FLR.2.4c The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users.
ALC_FLR.2.5c The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.
ALC_FLR.2.6c The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.
ALC_FLR.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.5.3 Developer defined life-cycle model (ALC_LCD.1)

ALC_LCD.1.1d The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.
ALC_LCD.1.2d The developer shall provide life-cycle definition documentation.
ALC_LCD.1.1c The life-cycle definition documentation shall describe the model used to develop and maintain the TOE.
ALC_LCD.1.2c The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.

ALC_LCD.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.5.4 Well-defined development tools (ALC_TAT.1)

ALC_TAT.1.1d The developer shall identify the development tools being used for the TOE.
ALC_TAT.1.2d The developer shall document the selected implementation-dependent options of the development tools.
ALC_TAT.1.1c All development tools used for implementation shall be well-defined.
ALC_TAT.1.2c The documentation of the development tools shall unambiguously define the meaning of all statements used in the implementation.
ALC_TAT.1.3c The documentation of the development tools shall unambiguously define the meaning of all implementation-dependent options.
ALC_TAT.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.6 Tests (ATE)

5.2.6.1 Analysis of coverage (ATE_COV.2)

ATE_COV.2.1d The developer shall provide an analysis of the test coverage.
ATE_COV.2.1c The analysis of the test coverage shall demonstrate the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.
ATE_COV.2.2c The analysis of the test coverage shall demonstrate that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.
ATE_COV.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.6.2 Testing: high-level design (ATE_DPT.1)

ATE_DPT.1.1d The developer shall provide the analysis of the depth of testing.
ATE_DPT.1.1c The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TSF operates in accordance with its high-level design.
ATE_DPT.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.6.3 Functional testing (ATE_FUN.1)

ATE_FUN.1.1d The developer shall test the TSF and document the results.
ATE_FUN.1.2d The developer shall provide test documentation.
ATE_FUN.1.1c The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.
ATE_FUN.1.2c The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.
ATE_FUN.1.3c The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.
ATE_FUN.1.4c The expected test results shall show the anticipated outputs from a successful execution of the tests.
ATE_FUN.1.5c The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.
ATE_FUN.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.6.4 Independent testing - sample (ATE_IND.2)

ATE_IND.2.1d The developer shall provide the TOE for testing.

ATE_IND.2.1c The TOE shall be suitable for testing.
ATE_IND.2.2c The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.
ATE_IND.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ATE_IND.2.2e The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.
ATE_IND.2.3e The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

5.2.7 Vulnerability assessment (AVA)

5.2.7.1 Validation of analysis (AVA_MSU.2)

AVA_MSU.2.1d The developer shall provide guidance documentation.
AVA_MSU.2.2d The developer shall document an analysis of the guidance documentation.
AVA_MSU.2.1c The guidance documentation shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.
AVA_MSU.2.2c The guidance documentation shall be complete, clear, consistent and reasonable.
AVA_MSU.2.3c The guidance documentation shall list all assumptions about the intended environment.
AVA_MSU.2.4c The guidance documentation shall list all requirements for external security measures (including external procedural, physical and personnel controls).
AVA_MSU.2.5c The analysis documentation shall demonstrate that the guidance documentation is complete.
AVA_MSU.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_MSU.2.2e The evaluator shall repeat all configuration and installation procedures, and other procedures selectively, to confirm that the TOE can be configured and used securely using only the supplied guidance documentation.
AVA_MSU.2.3e The evaluator shall determine that the use of the guidance documentation allows all insecure states to be detected.
AVA_MSU.2.4e The evaluator shall confirm that the analysis documentation shows that guidance is provided for secure operation in all modes of operation of the TOE.

5.2.7.2 Strength of TOE security function evaluation (AVA_SOF.1)

AVA_SOF.1.1d The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.
AVA_SOF.1.1c For each mechanism with a strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST.
AVA_SOF.1.2c For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric defined in the PP/ST.
AVA_SOF.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_SOF.1.2e The evaluator shall confirm that the strength claims are correct.

5.2.7.3 Independent vulnerability analysis (AVA_VLA.2)

AVA_VLA.2.1d The developer shall perform a vulnerability analysis. *(per International Interpretation #51 (rev 1))*
AVA_VLA.2.2d The developer shall provide vulnerability analysis documentation. *(per International Interpretation #51 (rev 1))*
AVA_VLA.2.1c The vulnerability analysis documentation shall describe the analysis of the TOE deliverables performed to search for ways in which a user can violate the TSP. *(per International Interpretation #51 (rev 1))*

AVA_VLA.2.2c The vulnerability analysis documentation shall describe the disposition of identified vulnerabilities. *(per International Interpretation #51 (rev 1))*

AVA_VLA.2.3c The vulnerability analysis documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE. *(per International Interpretation #51 (rev 1))*

AVA_VLA.2.4c The vulnerability analysis documentation shall justify that the TOE, with the identified vulnerabilities, is resistant to obvious penetration attacks. *(per International Interpretation #51 (rev 1))*

AVA_VLA.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.2.2e The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure the identified vulnerabilities have been addressed.

AVA_VLA.2.3e The evaluator shall perform an independent vulnerability analysis.

AVA_VLA.2.4e The evaluator shall perform independent penetration testing, based on the independent vulnerability analysis, to determine the exploitability of additional identified vulnerabilities in the intended environment.

AVA_VLA.2.5e The evaluator shall determine that the TOE is resistant to penetration attacks performed by an attacker possessing a low attack potential.

6. TOE Summary Specification

This chapter describes the security functions and associated assurance measures.

6.1 TOE Security Functions

6.1.1 Security Audit

i5/OS includes the capability to audit all security relevant events. Audit records are stored in journal receivers associated with the security audit journal. The journal receivers and security journal are all objects that are protected from unauthorized access or destruction using the discretionary access control mechanism.

Only an authorized administrator holding the *AUDIT special authority may create the security audit journal and a journal receiver. (By default, only the Security Officer (*SECOFR) user class may hold the *AUDIT special authority.) They do not exist when a system is initially "scratch" installed. The journal and its receiver are owned by the user that created them. i5/OS provides a command to create the initial security journal receiver and the security audit journal. The use of this command is recommended for two reasons. First, the command creates the journal and the journal receiver with the public authority of *EXCLUDE. This prevents non-administrators from accessing the audit data. Second, the command creates the journal with the attribute "system managed." A system managed journal will create a new journal receiver, attach the new receiver, and detach the old one any time the previous receiver is full. This prevents the potential loss of audit data. When the system generates a new receiver, it will have the same authorities and owner as the previous receiver.

Each i5/OS component that implements security-relevant functions is responsible to collect the necessary data and to call the i5/OS auditing routines. The auditing routines will only send audit data to the journal receiver if auditing is active and the security action is pre-selected.

The audit routines also monitor the size of the journal receiver in storage. If the journal receiver reaches an authorized-administrator defined threshold and the journal is system managed, the audit routine generates a new journal receiver, attaches it to the security audit journal, and detaches the old receiver. If the journal is not system managed, warning messages are sent to the authorized administrator. This allows the authorized administrator to react and possibly avoid a journal-receiver full condition. Warnings are generated when the available mass storage space is exhausted, regardless of the method selected to manage the journal.

There are numerous security relevant events that are auditable by i5/OS, including but not limited to:

- Start-up and shutdown of the audit functions
- Successful and unsuccessful access to the audit trail
- Modification to the audit configuration
- Actions taken due to exceeding an audit threshold
- Actions taken due to audit storage failure
- Access control decisions
- Use of an identification or authentication mechanism, including successful and unsuccessful authentication
- Successful and unsuccessful subject creation
- Modification or revocation of access control attributes
- Modification to default and initial object access settings
- Creating, deleting or clearing the audit trail
- Modifying or observing audit data
- Initializing, modifying, or revoking user security attributes

- Modification to the set of defined users
- Use of the authorized administrator role
- Abstract machine tests
- Changes to time

Each record event includes at least the following information:

- Date and time
- Event type (which differentiate between successes and failures)
- User name
- Object name (when an object is involved)
- Origination of the attempt (for sign-on attempts)
- The new values (for changes to audit settings and user security attributes)
- Role identification and request origination (for use of rights associated with a role)

Authorized administrators may configure auditing to audit or not audit specific events based on the following parameters:

- User identity
- Object
- Event type

In order to control the collected audit data, i5/OS provides systems-wide, user profile, and object based auditing levels. These settings allow an authorized administrator to enable and disable auditing, as well as to select actions to take when an audit record can't be deposited in to the security audit log for any reason. In particular, an authorized administrator can configure the system to shut down when the system is unable to deposit the audit record into the journal receiver.

Auditing of security relevant actions is accomplished by having the software that produces such actions call the appropriate routines to append an entry into the journal receiver. These routines determine whether auditing is currently active by testing the QAUDCTL (audit control) system value values *AUDLVL (audit level) and *OBJAUD (object audit). QAUDCTL turns auditing on or off at the audit level or at the object level. It functions as a master switch to turn auditing on or off.

When an auditable event occurs, all three audit controls; system-wide, user profile, and object based audit levels are checked. The event will be logged if any of the three controls indicate the event should be logged.

When an authorized administrator is ready to review audit data, they must issue a command to extract the journal entries from the security journal receiver. The command used to extract the records can do some data filtering by date and time range, job ID, or entry type. In addition, the data can be placed in objects that make advanced searching possible. Sorting can be based on numerous attributes, including user identities.

The Security Audit function is designed to satisfy the following security functional requirements:

- FAU_GEN.1
- FAU_GEN.2
- FAU_SAR.1
- FAU_SAR.2
- FAU_SAR.3
- FAU_SEL.1

- FAU_STG.1
- FAU_STG.3
- FAU_STG.4

6.1.2 User Data Protection

i5/OS controls access to the objects listed in the FDP_ACC.1.1 requirement using authorities and authorization lists. In addition, the system also requires a user to have special authorities (privileges) to access certain external objects by commands or APIs and some MI instructions (special privileged instructions).

i5/OS provides each object that is part of the TOE with an interface that defines what actions users may perform, and how i5/OS should treat the encapsulated data for each of these objects.

Authorities can be granted to users, groups, and to all users (known as *PUBLIC) and there are four types of authorities:

- Owner authority – Authority of the owning user profile. Each object has an associated owning user profile and by default the owning user profile is the user profile of the creating process and all authorities are granted to the owner.
- Primary group authority – Authority of the primary group. An object may optionally have a primary group. IF it does, the primary group authority is stored with the object and not with the group profile.
- Private authority – Authority that is explicitly granted to a user or group profile. Private authorities to objects are stored in the user and group profiles and if both a user and group profile have private authorities for the same object, the user profile takes priority.
- Public authority – Authorities that apply to users that don't have explicit authority to an object. Public authorities are used only in the absence of owner, primary group, and private authorities.

There are also two categories of authorities: object authorities and data authorities. Object authorities pertain to operations that are performed on the object as a whole. Data authorities pertain to operations that can be performed on the contents of the object.

The following object authorities are defined in i5/OS:

- Object Operation (*OBJOPR) – allows a user to look at the description of an object (type, text, etc.) and use the object as determined by the specified data authorities. To open a file, for example, the user must have this authority.
- Object Management (*OBJMGT) – allows the user the ability to specify authorities for the object, move and rename the object, and add members to database files.
- Object Existence (*OBJEXIST) – allows the user control of the object's ownership and the ability to delete the object.
- Object Alter (*OBJALTER) – allows the user to add, clear, initialize, and reorganize members of a database file.
- Object Reference (*OBJREF) – allows the user to specify a database file as the parent in a referential constraint.
- Authorization List Management (*AUTLMGT) – allows users to add, remove, and change users and their authorities on an authorization list. *ALLOBJ special authority or ownership of the authorization list is needed to give *AUTLMGT authority for the authorization list to another user. This authority pertains only to authorization lists.

The following data authorities are defined in i5/OS:

- Read (*READ) – allows a user to display the contents of the object (e.g. displaying records of a file) and add spool files to an output queue.

- Update (*UPDATE) – allows a user to change entries (e.g., records) in the object.
- Add (*ADD) – allows a user to add entries to the objects.
- Delete (*DLT) – allows users to remove entries (e.g., records and messages) from the object.
- Execute (*EXECUTE) – allows users to run a program, service program, or SQL package, or search for an object in a library or directory.
- Exclude (*EXCLUDE) – prevents user access to the object. (Note that in group authority and program adoption access checks, if an entity holds both *EXCLUDE authority and some other authority, the other authority takes precedence over the *EXCLUDE authority.) Exclude authority is different from having no authority. Based on the algorithm for granting authority, having no authority implies that the public authority defined for the object is used. Note that *EXCLUDE is both a data and an object authority.

Additionally, authorities can be used individually or in predefined system-supplied groups. The following groups of authorities are defined for all objects:

- *ALL – *OBJOPR, *OBJMGT, *OBJEXIST, *OBJALTER, *OBJREF, *READ, *ADD, *UPD, *DLT, and *EXECUTE
- *CHANGE – *OBJOPR, *READ, *ADD, *UPD, *DLT, and *EXECUTE
- *USE – *OBJOPR, *READ, and *EXECUTE

The following groups of authorities are defined for use with the integrated file system:

- *RWX – *OBJOPR, *READ, *ADD, *UPD, *DLT, and *EXECUTE
- *RW – *OBJOPR, *READ, *ADD, *UPD, and *DLT
- *RX – *OBJOPR, *READ, and *EXECUTE
- *R – *OBJOPR and *READ
- *WX – *OBJOPR, *ADD, *UPD, *DLT, and *EXECUTE
- *W – *OBJOPR, *ADD, *UPD, and *DLT
- *X – *OBJOPR and *EXECUTE

Authorization lists are objects of type *AUTL. They are used to assign specific authorities for different users and groups to a set of objects. All objects except profiles and authorization lists can be secured by an authorization list. Furthermore, an object can have only a single authorization list while a single authorization list can be used to secure multiple objects.

The owner of an object and users with all object (*ALLOBJ) special authority can grant or revoke any authority to the target object. Other users with *OBJMGT or *AUTLMGT authority can:

- Revoke any authorities that they have
- Grant any authority that they have, other than *OBJMGT or *AUTLMGT.

The owner of an object, a user with all object (*ALLOBJ) special authority, or a user with *ALL authority to an object can secure or unsecure an object with an authorization list.

When a user attempts to access an object in i5/OS, an access check is made to determine whether the requested operation is allowed. i5/OS goes through the authority sources in a priority order until it finds sufficient or insufficient authority to the target object. i5/OS supports an explicit authority called *EXCLUDE. When this authority is encountered, the user is denied access to the target object. The following authority sources are examined in the order listed below to determine access:

- User Profile. This is sometimes called private authority checking. This is the notion of explicit users having explicit authority or *ALLOB special authority.
    - Authority is granted if the user has *ALLOBJ special authority.

  - When the user profile has an explicit private authority entry to the object, authority is granted or denied based on the amount of authority found in the entry. If a private entry is found, and it has enough authority to grant access to the object, no further checks are performed (access is granted regardless of authorization list entries, if any). If a private authority entry is found, but it does not have enough authority to grant access to the object, the only other possible source for authority to the object will be from adopted authority.
  - When the object is secured by an authorization list (*AUTL) and the user profile has an explicit private authority entry to the *AUTL, authority is granted or denied based on the amount of authority found in the entry. If a private authority entry is found for the authorization list, but it does not have enough authority to grant access to the object, the only other possible source for authority to the object will be from adopted authority.
  - If no private authority entries are found, i5/OS will check the user's group authority.

- Group profiles. This is the notion that the user has one or more group profiles associated with their profile and the authority to the objects comes from an individual group or the additive authorities of 2 to 16 groups. The following is done for each group profile until authority is granted or there are no more groups left to check.
  - Authority is granted if the group profile has *ALLOBJ Special authority.
  - When a private authority entry is found for the group profile, the authority is added to any authority found for previous group profiles. If the total authority is enough, the user is granted access to the object. If the authority is not enough, then the next group in the list is checked
  - When the object is secured by an authorization list (*AUTL) and the group profile has an explicit private authority entry to the *AUTL, the authority is added to any authority found for previous group profiles. If the total authority is enough, the user is granted access to the object. If the authority is not enough, then the next group in the list is checked
  - After all the groups have been checked, some group authority is found, but the authority is not enough to grant access to the object, the only other possible source for authority to the object will be from adopted authority.
  - After all the groups have been checked and no group authority is found, i5/OS will check the object's public authority.
- Public Authorities. This is the authority granted to users that aren't explicitly authorized to the object.
  - If the public authority is sufficient, access is immediately granted
  - If the public authority is not sufficient to gain access to the object, the only other possible source for authority to the object will be from adopted authority.
- Program adoption. i5/OS allows programs to run with the owner's authority.
  - i5/OS allows a program or service program to run with the authority of its owner. When an adopted authority check is done, the "user profile" authority test is done to determine if the program's owner has the required authority.
  - Adopted authority checks do not include the program owner's groups.
  - If the user does not obtain authority to the object using adopted authority, i5/OS will continue to walk the program stack to determine if there are any other programs that are running with adopted authority. If a program is found, that owner's authority is checked and added to any previous found adopted authority.
  - The adopted authority checks will continue until enough authority is found or there are no more programs in the call stack that adopt authority.

  NOTE: In the group authority and program adoption cases authority checking is additive. Where authorities are added:

"*EXCLUDE authority" + "some authority" = "some authority".

When an object is created, it is assigned an owning profile (commonly called the object owner). The default owner is the profile that was in control when the object was created. In addition, the owner is given a *ALL private authority entry to the object.

When objects are created, they are assigned a *PUBLIC authority. This authority can come from:

- The CRTAUT (Create Authority) parameter that is available on some interfaces.
- The default "create authority" value associated with the target library. This value often points to a system-wide setting for the default value.
- For IFS based objects, the authority can be inherited from the parent.

All storage objects (memory, disks, workstations, optical drives, magnetic tapes and printers) used in i5/OS are cleared when they are allocated. Input/output processor and other device buffers are controlled by keeping track of how much data is present and disallowing read attempts beyond the current data.

The User Data Protection function is designed to satisfy the following security functional requirements:

- FDP_ACC.1
- FDP_ACF.1
- FDP_RIP.2a
- FDP_RIP.2b

6.1.3 Identification and Authentication

i5/OS defines users and groups using profiles. Each profile is an object with field-level access controls in order to ensure that only authorized administrators can change profile information that is considered security relevant. Furthermore, user and group profiles can be created and deleted by authorized administrators. Note that a group profile can be deleted only if it has no members and is not the primary group of any object.

Each profile contains numerous fields of information, most of which are not security relevant. The important fields are summarized as follows:

- Profile name
- Password
- Expired password indicator
- Profile status (i.e., enabled or disabled)
- Failed sign-on attempt count
- Date password expires
- User class (Security Officer, Security Administrator, System Programmer, System Operator, or End User)
- Primary group profile
- Supplemental group profiles (up to 15)
- OBJAUD (object auditing value)
- AUDLVL (indicates security-relevant events to audit)
- UID (User Identification Number)
- GID (Group Identification Number)

- Pointers to the objects the profile owns
- Pointers to the objects for which the profile has a private authority entry.
- Pointers to the objects for which the profile is the primary group
- For each owned object, pointers to profiles that have private authorities to the object and the authorities
- Special authority and privileged instruction masks can allow execution of privileged instructions

Of these fields, a user who is not an authorized administrator can only change their own password, albeit indirectly using the services of i5/OS provided for that purpose. With regard to the FIA_ATD.1.1 requirement, the UID attribute corresponds to the user identifier, the primary group profile and supplemental group profiles correspond to the group memberships, and the password corresponds to the authentication data. The security-relevant roles are implemented as the user class attribute, pointers to the objects the profile owns, pointers to the objects for which the profile has a private authority entry, and pointers to profiles for each owned object that have private authorities to the object and the authorities.

i5/OS requires all users to identify and authenticate themselves before they are allowed to access system resources. Users are identified by a user profile and authenticated by a password. Each user has a unique user profile that must also contain a unique UID. A password of 6 to 128 characters in length is used to authenticate users. There are eleven system values that control passwords. These system values require users to change passwords regularly and help prevent users from assigning trivial, easily guessed passwords. The administrator and user guidance document provide recommendations for password construction that includes:

. a minimum of six characters be used

. prevention of the use of repeated characters

. each character in a password be different from the character in the same position of the previous password

. a password must contain at least one numeric character

. intervals between password re-use be a minimum (5)

. frequency of password expiration (60 days)

A user can obtain access to the i5/OS by signing on. The user must provide an identity and the associated password when logging in; the password will not be displayed on the screen when it is typed. Additionally, authorized administrators may configure the TOE so that, when a user unsuccessfully attempts to log in, the error message displayed does not indicate whether the username, the password, or both were incorrect. If successfully authenticated, a process will be instantiated with the user's profile in order to implement the current and subsequent user requests. User authentication occurs when the following functions are used:

- AUTOSTART
- SIGNON command
- STRxxxJOB or SBMxxxJOB commands.
- FTP Sign on
- TELNET Sign on
- RUNRMTCMD

In order to be successfully authenticated, the user identity must correspond with an existing user profile and the provided password must match the password stored in the profile. Additionally, the user profile must be enabled and have the required access to resources associated with the connection attempt (e.g., access to the workstation device). Note that if the user's password has expired, it must be changed before the sign-on can be completed. Finally, if the user is signing on interactively, via workstation or TELNET, the user is provided information regarding the date and time of their last sign-on as well as the number of unsuccessful sign-on attempts since then along with the number of days before their password will expire.

In order to be successfully authenticated using remote commands, the user on the client system initiates the session by using using the CL command RUNRMTCMD that causes the client to establish a TCP/IP connection to the REXEC server. Once the connection is established, the client sends 4 null-terminated strings to the server. The second null-terminated string is a user name and the third null-terminated string is a password, which is not

obscured. The REXEC server uses standard security interfaces and validates the supplied user name and supplied password.

Note that RUNRMTCMD does not obscure the password until the user presses the enter key. This is consistent with the application note in the CAPP that states "Obscured feedback implies the TSF does not produce a visible display of any authentication data entered by a user, such as through a keyboard (e.g., echo the password on the terminal). It is acceptable that some indication of progress be returned instead, such as a period returned for each character sent. Some forms of input, such as card input based batch jobs, may contain human-readable user passwords."

If a user fails a sign-on attempt, a count of unsuccessful attempts is incremented and the event is audited if the administrator is auditing these events. Furthermore, an authorized administrator can define actions to take when a specific number of unsuccessful sign-on attempts are reached. These actions include disabling the workstation and/or the user's profile.

As indicated above, once a user is successfully authenticated, a process is instantiated with their user profile, which includes any group profile(s) - a special form of user profile - that the user may hold, to operate on their behalf. The security attributes contained within applicable user and group profiles are associated with the process. Any commands issued by the user subsequently execute in the context of the user's profile with the following two exceptions. A trusted subject, such as an authorized administrator, can change the user profile associated with a process thread and thereby change the security attributes. Any process can potentially augment its security attributes by calling a program that adopts authority. Such programs can be created by a user and assigned attributes such that when another user executes that program the associated process can acquire the authorities of the program's owner.

Adopted authority is added to any other authority found for the user. Only the authorities of the owner are adopted. If the owner has a group profile, the group's authorities are not considered. Adopted authority is a program attribute that is specified when the program is created. If program adoption is specified, then the authorities associated with the program owner's user profile are checked to determine whether authority is sufficient to access the object. The system may use the adopted authority from the original program the user called or from earlier programs in the program stack. If the adopted authority check locates sufficient authority, then access is granted. If the result is insufficient, then access is denied.

Note that when a user profile is created using either CRTUSRPRF or CHGUSRPRF, a user class may optionally be assigned. User classes can be used as a convenient way to assign special authorities that are associated with a given class. However, user classes are not used for DAC, and despite also controlling what menu options are displayed for a given user class on i5/OS menu interfaces, user classes do not limit the use of commands.

The Identification and Authentication function is designed to satisfy the following security functional requirements:

- FIA_ATD.1
- FIA_SOS.1
- FIA_UAU.2
- FIA_UAU.7
- FIA_UID.2
- FIA_USB.1

6.1.4 Security Management

i5/OS allows users to be assigned to roles based on their user class; the user class controls the options that are available to the user. The user class can be assigned one of the following values:

- Security Officer (*SECOFR), who performs all security functions including creating security administrators
- Security Administrator (*SECADM), who performs all security functions including creating security administrators

- System Programmer (*PGMR), who performs system programming functions
- System Operator (*SYSOPR), who also performs system maintenance and operation functions and can back up the system and save and restore objects
- End User (*USER), who performs application functions

In addition to the user classes, special authorities determine the user's role. Special authorities are used to specify the type of actions a user can perform on system resources. A user can be given zero or more special authorities. Special authorities are specified in the special authority (SPCAUT) field of the user profile. If the value of SPCAUT is user class (*USRCLS), then special authorities are granted to the user based on the user class. If *USRCLS is specified, specific special authorities cannot be specified for this user.

Special authorities also determine what special privileged MI instructions a user can issue. When the security administrator updates the special authorities field in a user profile, the corresponding special privileged instruction bits in the user profile are also updated.

The algorithm for checking a user process' special authority is similar to the DAC algorithm. The algorithm consists of process user profile, process group user profiles and check program adoption. Special authority checking is additive. The algorithm searches the user profile, the group profiles and the program adoption until sufficient authority is found or the algorithm reaches the last step.

The available special authorities include:

- *ALLOBJ: All Object - allows the user to have *ALL access to each object whether or not private authorities exist for the user. Even if the user has *EXCLUDE authority to an object, *ALLOBJ special authority still allows the user to access the object.
- *SECADM: Security Administration - allows the user to create, change, and delete user profiles. The security administrator can grant only those special authorities (excluding *SECADM) that the administrator has. A security administrator must have *SECADM and *ALLOBJ special authorities to grant the *SECADM special authority.
- *JOBCTL: Job Control - allows the user to change, display, hold, release, cancel, and clear jobs that are executing on the system or are on a job queue that has the operational control attribute set to YES (i.e., OPRCTL(*YES)). *JOBCTL also allows the user to display or copy files on output queues specified as (OPRCTL(*YES)). *JOBCTL special authority allows the user to IPL the system and control the subsystems.
- *SPLCTL: Spool Control - allows the user to delete, display, hold, and release spool files that are owned by other users in libraries that the user has *EXECUTE authority to.
- *SAVSYS: Save System - allows the user to save and restore all objects on the system.
- *AUDIT: Audit - allows the user to specify or change the audit system values to determine system wide audit criteria. *AUDIT also allows the user to set auditing levels for individual objects and users.
- *SERVICE: Service - allows the user to perform service operations.
- *IOSYSCFG: System Configuration - allows the user to change system configuration information.

For the purposes of this Security Target, authorized administrators are considered any user who has any special authority. Additionally, any user whose user class is set to a value other than *USER (i.e., *SECOFR, *SECADM, *PGMR, or *SYSOPR) is considered an authorized administrator.

i5/OS controls the ability to perform security management functions using the discretionary access control mechanism. As indicated in previous sections, the ability to manage the audit function, including event selection and audit data review are restricted to authorized administrators by ensuring that only authorized administrators have the necessary authority. Similarly, only authorized administrators have the authorities necessary to manage security-relevant aspects of user and group profiles. Only authorized administrators are able to initialize authentication data.

The ability to manage discretionary access control attributes of an object is not restricted to authorized administrators. Discretionary access can be controlled by an authorized administrator, the object's owner, or any user with object management or authorization list management authority to the object, as described in section 6.1.2.Some changes to authentication data (user and group profiles) only take effect when initial job attributes are collected and some changes take effect immediately. For example, if an administrator adds or subtracts group profiles from a user, any active jobs for the user will not be affected. The jobs will continue to run with the groups the user had when the user signed on. However, an authorized administrator can force a user that is currently signed-on off the system so that any changes can be effective immediately. Changes to access control attributes become effective the next time an access check is made. The *SECADM special authority is required to create, change or delete user profiles.

When objects are created, the user can either specify the public authority (permission) to the object or use the default system value. This can be explicitly specified at creation or set to the default system value (QCRTAUT). The default authority for objects created on the system are *ALL authority for the owner and the default system value (QRCTAUT) for *PUBLIC (all users). Public authority is used in the absence of owner, primary group or private authority.

The TOE uses a more restrictive authority for the default system value (QCRTAUT); its value is set to *EXCLUDE which prevents user access to the object. Exclude authority is different than having no authority. Having no authority implies that the public authority defined for the object is used.

There are two categories of authorities: object authorities and data authorities. Object authority defines the operations that can be performed on the object as a whole. Data authority defines the operations that can be performed on the contents of an object. *EXCLUDE is both a data and an object authority.

The Security Management function is designed to satisfy the following security functional requirements:

- FMT_MSA.1
- FMT_MSA.3
- FMT_MTD.1a
- FMT_MTD.1b
- FMT_MTD.1c
- FMT_MTD.1d
- FMT_REV.1a
- FMT_REV.1b
- FMT_SMR.1

6.1.5 Protection of the TOE Security Functions

In support of the security audit function, the underlying hardware includes a real-time clock that is synchronized by i5/OS during each IPL so that i5/OS can offer reliable time information to itself and its applications.

6.1.5.1 TSF Isolation and Protection

i5/OS maintains a domain for its own execution, and separates this domain from the user domain, by a combination of the state and domain attributes implemented in software. i5/OS runs in system or inherit state. All user code runs in user state. Most i5/OS objects are created in system domain storage. Therefore, code running in user state cannot access system domain objects directly; instead, they must use the defined i5/OS interfaces. Manipulation of the state and domain attributes requires use of blocked i5/OS instructions. Code written on the evaluated configuration cannot use the blocked i5/OS instructions because the translator in the evaluated configuration does not translate blocked i5/OS instructions. Code written on the evaluated configuration cannot issue hardware instructions directly since the availability of compilers and translators is carefully controlled. The administrator guidance provides procedures for

the system administrator to guard against object code being restored (or otherwise introduced) to the system without retranslation, which will ensure the integrity of the domains.

i5/OS blocks some instructions, and these are analogous to machine instructions that can only be executed while the machine is in supervisor mode. MI instructions can be blocked at translate time or at runtime. Instructions that are blocked at translate time are those that the translator will not translate.

i5/OS administrator documentation warns that introducing a translator that is called by the MI instructions Create Program (CRTPG) and Create Module (CRTMOD) other than the "**Error! Reference source not found.**" removes the system from the evaluated configuration. Regardless, such a translator cannot create an encapsulated program object because MI programs can write data only into spaces and spaces cannot have the program MI object type.

i5/OS restore function prevents programs and other objects from being restored that are not allowed on the evaluated configuration. These programs may have been altered by use of the Dedicated Service Tools (DST).

i5/OS creates objects using a hardware storage protection attribute. During execution of each RISC instruction, the hardware determines whether the page frame is hardware storage protected. In this way, user state programs can have hardware storage protection read-only access to objects such as the entry point table.

i5/OS uses hardware tag bits set to identify a valid pointer data object. Obtaining a System Pointer (SYP) gives a process thread addressability to an MI object. However, that pointer will only be valid as input to MI instructions that will operate on an object of the type addressed by the pointer. Any attempt to modify any type of tagged pointer, except with an MI instruction designed to modify a pointer, causes the tag bit to be cleared. The storage will no longer be viewed as a pointer by any MI instruction. This, along with translator control of addressability to space pointer machine objects, prevents non-pointer data from being used as pointer data.

6.1.5.2 Resource Isolation and Protection

i5/OS provides an object-based interface. This object-based interface enforces discretionary access control and auditability for objects. The objects are accessed through system pointers; these system pointers are protected against improper modification by hardware tags. Hardware storage protection bits also protect objects in main storage from improper access or modification by user state processes.

i5/OS creates protected objects in system domain, and DAC and audit requirements are enforced on these objects. Protected objects created in system domain are not directly accessible by user state programs, but only indirectly accessible using system provided APIs.

i5/OS passes parameters between user state and system state programs after validating both the input and output parameters to ensure that an invalid parameter value cannot cause the TSF to perform some function which the user state program is not allowed to perform, or that an invalid parameter does not contain a value that would unintentionally modify a system domain object.

i5/OS internal control blocks, or internal objects in i5/OS terminology, may be in user domain or in system domain and are protected by the hardware storage protection mechanism. These internal control blocks include the process static, heap, and dynamic storage working areas.

i5/OS handles messages as follows:

- Any user state program can send a message of any type to any other user state program.
- Any system state program can send a message of any type to any user or system state program.
- A user state program can send a non-exception message to any system state program.
- A user state program can send an exception type message (status, notify, or escape) to a system state program if either the system state program is a request processor, the system state program called a user state program, or the system state program called program QPXXCALL in library QSYS, and QPXXCALL called a user state program. The user state program sending the exception message does not have to be the user state program called by the system state program.

6.1.5.3 Abstract Machine Tests

i5/OS provides start-up and on-demand self-tests to verify the correct operation of the underlying hardware. Test results are sent to the control panel, appear in a system console message, appear in the primary partition's console for secondary partitions, or appear in either the service action log or the product activity log. Problems are reported using system reference codes (SRC) that:

- Identifies a system status
- Describes a detected hardware, Licensed Internal Code (LIC), or software failure
- Describes the unit that is reporting the failure and its location

An SRC encodes information used to evaluate or identify a system-detected hardware or software error, failure, or status. The failure information may include the failing condition or part (or unit) that can be exchanged or replaced and its location.

The Protection of the TOE Security Functions function is designed to satisfy the following security functional requirements:

- FPT_AMT.1
- FPT_RVM.1
- FPT_SEP.1
- FPT_STM.1

6.2 TOE Security Assurance Measures

6.2.1 Configuration management

The configuration management measures applied by IBM ensure that configuration items are uniquely identified, and that documented procedures are used to control and track changes that are made to the TOE. IBM ensures changes to the implementation representation are controlled with the support of automated tools and that TOE associated configuration item modifications are properly controlled. IBM performs configuration management on the TOE implementation representation, design documentation, tests and test documentation, user and administrator guidance, delivery and operation documentation, life-cycle documentation, vulnerability analysis documentation, configuration management documentation, and security flaws.

These activities are documented in:

- IBM iSeries OS/400 Configuration Management Plan, Revision 2, May 11, 2005

The Configuration management assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- ACM_AUT.1
- ACM_CAP.4
- ACM_SCP.2

6.2.2 Delivery and operation

IBM provides delivery documentation and procedures to identify the TOE, allow detection of unauthorized modifications of the TOE and installation and generation instructions at start-up. IBM's delivery procedures describe all applicable procedures to be used to detect modification to the TOE. IBM also provides documentation that describes the steps necessary to install i5/OS in accordance with the evaluated configuration.

These activities are documented in:

- IBM iSeries OS/400 V5R3 Common Criteria System Delivery Procedures, Revision 1, January 15, 2004

- IBM iSeries Configure Your System For Common Criteria Security, Version 5 Release 3, SC41-5336-00

The Delivery and operation assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- ADO_DEL.2
- ADO_IGS.1

6.2.3 Development

IBM has numerous documents describing all facets of the design of the TOE. In particular, they have a functional specification that describes the accessible TOE interfaces; a high-level design that decomposes the TOE architecture into subsystems and describes each subsystem and their interfaces; a low-level design that further decomposes the TOE architecture into modules and describes each module and their interfaces; and, correspondence documentation that explains how each of the design abstractions correspond from the TOE summary specification in the Security Target to the actual implementation of the TOE. Furthermore, IBM has a security model that describes each of the security policies implemented by i5/OS. Of course, the implementation of the TOE itself is also available as necessary.

These activities are documented in:

- IBM Corporation OS/400 OS400 Audit Methodology, Revision 1.1, February 6, 2005
- IBM CorporationOS/400 Design Documentation, Revision 0.41, February 24, 2005
- iSeries Operating System/400 Commands (pdf files), Version 5 Release 3
- V5R3 MI Instructions Documentation (html pages)
- IBM Corporation OS/400 Interfaces, Revision 0.1, March 17, 2005
- Proprietary PowerPC AS documentation
- IBM OS/400 V5R3 Security Policy Model, Version 0.3, 04/29/05
- All Unblocked MIs Mapped SFRs via exceptions spreadsheet – version 0.3
- CMD and API Mapping, 16 Nov 2004
- MI Mapping Notes - Version 0.1, 23 Nov04
- OS400 API List spreadsheet – 16 November, v2
- Implementation subset

The Development assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- ADV_FSP.2
- ADV_HLD.2
- ADV_IMP.1
- ADV_LLD.1
- ADV_RCR.1
- ADV_SPM.1

6.2.4 Guidance documents

IBM provides administrator and user guidance on how to utilize the TOE security functions and warnings to administrators and users about actions that can compromise the security of the TOE.

These activities are documented in:

- IBM iSeries Configure Your System For Common Criteria Security, Version 5 Release 3, SC41-5336-00
- iSeries Security Reference, Version 5, SC41-5302-07
- The iSeries Information Center
  - http://publib.boulder.ibm.com/pubs/html/as400/infocenter.html
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk631usersviewsecurity.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/apis/api.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbam6/rbam6clmain.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rzaiu/rzaiuicbackup.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/db2/rbafzmst02.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rzai2/rzai2kickoff.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rzahgictcp2.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk003planninguser.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapkaaacomface.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkprvaut.htm#prvaut
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk103physicalsec.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapk004planningresource.htm
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/info/rbapk/rbapkrbapkc17.htm

The Guidance documents assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- AGD_ADM.1
- AGD_USR.1

6.2.5 Life cycle support

IBM ensures the adequacy of the procedures used during the development and maintenance of the TOE through the use of a comprehensive life-cycle management plan. IBM includes security controls on the development environment that are adequate to provide the confidentiality and integrity of the TOE design and implementation that is necessary to ensure the secure operation of the TOE. IBM achieves this through the use of a documented model of the TOE life cycle and well-defined development tools that yield consistent and predictable results. As part of its life-cycle management process, IBM documents and uses flaw remediation procedures.

These activities are documented in:

- IBM iSeries OS/400 System Life Cycle Document Revision 2, April 16, 2004
- IBM Proprietary Compiler Documentation
- IBM Proprietary Change Control Process Documentation

The Life cycle support assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- ALC_DVS.1

- ALC_FLR.2
- ALC_LCD.1
- ALC_TAT.1

6.2.6 Tests

IBM has a test plan that describes how each of the necessary security functions is tested, along with the expected test results. IBM has documented each test as well as an analysis of test coverage and depth demonstrating that the security aspects of the design evident from the functional specification and high-level design are appropriately tested. Actual test results are created on a regular basis to demonstrate that the tests have been applied and that the TOE operates as designed.

These activities are documented in:

- IBM iSeries OS/400 Common Criteria Test Plan, Revision 1.1, December 22, 2004
- TestcaseInfo.xls (test coverage)
- OSMInstructionTests.xls (test coverage)
- Test Cases as referenced by TestcaseInfo.xls
- Test Results as referenced by test cases

The Tests assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- ATE_COV.2
- ATE_DPT.1
- ATE_FUN.1
- ATE_IND.2

6.2.7 Vulnerability assessment

The TOE administrator and user guidance documents describe the operation of i5/OS and how to maintain a secure state. These guides also describe all necessary operating assumptions and security requirements outside the scope of control of the TOE. They have been developed to serve as complete, clear, consistent, and reasonable administrator and user references. Furthermore, IBM has conducted a misuse analysis demonstrating that the provided guidance is complete.

IBM has conducted a strength of function analysis wherein all permutational or probabilistic security mechanisms have been identified and analyzed resulting in a demonstration that all of the relevant mechanisms fulfill the minimum strength of function claim, SOF-Medium. The mechanism associated with the SOF claim is the TOE password mechanism. The TOE password mechanism is associated with the Identification and Authentication security function.

IBM performs regular vulnerability analyses of the entire TOE (including documentation) to identify weaknesses that can be exploited in the TOE.

These activities are documented in:

- IBM OS/400 V5R3 Vulnerability Analysis, version 0.2, 4/29/05
- IBM OS/400 V5R3 Misuse Analysis, version 0.1, 10/18/04

The Vulnerability assessment assurance measure satisfies the following EAL 4 augmented with ALC_FLR.2 assurance requirements:

- AVA_MSU.2
- AVA_SOF.1

- AVA_VLA.2

7. Protection Profile Claims

As documented in this Security Target, IBM i5/OS V5R3 complies with the Controlled Access Protection Profile (CAPP), Version 1.d, 8 October 1999.

The Security Environment, Objectives, and Requirements have been reproduced from the CAPP with the following modifications: obvious typographical errors have been corrected, all current International Interpretations have been applied, iterated requirements have been relabeled in accordance with the conventions described in this Security Target, all operations left uncompleted in the CAPP have been completed appropriately in this security target, and as specifically indicated below. All such operations are identified in the applicable requirement elements.

The FDP_ACF.1.1 security functional requirement has been refined in order to describe the capabilities of the TOE in greater detail and to comply with Interpretation #103, which stipulates that this requirement must state which attributes are associated with subjects and which are associated with objects. Specifically, the attributes listed are a superset of those attributes required in the CAPP. The user profiles referred to in the version of the requirement in the ST contain both the user identity (in the form of the UID attribute) required by the CAPP and other attributes contained in the user profile. The group profiles referred to in the ST version of the requirement are associated with both the group membership required by the CAPP and other security attributes contained in the group profile.

FIA_UID.1 and FIA_UAU.1 have been upgraded to FIA_UID.2 and FIA_UAU.2 since user identification and authentication is required prior to accessing all TSF mediated functions.

Note that the rationale for the security environment, objectives, and requirements has been included primarily by reference to the CAPP.

8. Rationale

This section provides the rationale for completeness and consistency of the Security Target. The rationale addresses the following areas:

- Security Objectives;
- Security Functional Requirements;
- Security Assurance Requirements;
- TOE Summary Specification;
- Security Functional Requirement Dependencies; and
- Internal Consistency.

8.1 Security Objectives Rationale

This section shows that all secure usage assumptions, organizational security policies, and threats are completely covered by security objectives. In addition, each objective counters or addresses at least one assumption, organizational security policy, or threat.

8.1.1 Security Objectives Rationale for the TOE and Environment

This section provides evidence demonstrating the coverage of organizational policies and usage assumptions by the security objectives.

	P.ACCOUNTABILITY	P.AUTHORIZED_USERS	P.NEED_TO_KNOW	A.CONNECT	A.COOP	A.LOCATE	A.MANAGE	A.NO_EVIL_ADM	A.PEER	A.PROTECT
O.AUDITING	X									
O.AUTHORIZATION		X								
O.DISCRETIONARY_ACCESS			X							
O.ENFORCEMENT	X	X	X							
O.MANAGE	X	X	X							
O.RESIDUAL_INFORMATION			X							
O.CREDEN					X					
O.INSTALL							X	X	X	
O.PHYSICAL				X		X				X

Table 3 Environment to Objective Correspondence

The security environment definition and all of the security objectives have been drawn from the Controlled Access Protection Profile (CAPP). See the CAPP for any applicable rationale.

8.2 Security Requirements Rationale

This section provides evidence supporting the internal consistency and completeness of the components (requirements) in the Security Target. Note that **Table 4** indicates the requirements that effectively satisfy the individual objectives. .

8.2.1 Security Functional Requirements Rationale

All Security Functional Requirements (SFR) identified in this Security Target are fully addressed in this section and each SFR is mapped to the objective for which it is intended to satisfy.

	O.AUDITING	O.AUTHORIZATION	O.DISCRETIONARY_ACCESS	O.ENFORCEMENT	O.MANAGE	O.RESIDUAL_INFORMATION
FAU_GEN.1	X					
FAU_GEN.2	X					
FAU_SAR.1	X				X	
FAU_SAR.2	X					
FAU_SAR.3	X				X	
FAU_SEL.1	X				X	
FAU_STG.1	X					
FAU_STG.3	X				X	
FAU_STG.4	X				X	
FDP_ACC.1			X			
FDP_ACF.1			X			
FDP_RIP.2a						X
FDP_RIP.2b						X
FIA_ATD.1		X	X			
FIA_SOS.1		X				
FIA_UAU.2		X				
FIA_UAU.7		X				
FIA_UID.2		X				
FIA_USB.1	X		X			
FMT_MSA.1			X			
FMT_MSA.3			X			
FMT_MTD.1a	X				X	
FMT_MTD.1b	X				X	
FMT_MTD.1c					X	
FMT_MTD.1d		X			X	
FMT_REV.1a					X	
FMT_REV.1b			X			

FMT_SMR.1					X	
FPT_AMT.1				X		
FPT_RVM.1				X		
FPT_SEP.1				X		
FPT_STM.1	X					

Table 4 Objective to Requirement Correspondence

All of the security requirements and security objectives have been drawn from the Controlled Access Protection Profile (CAPP), except for FIA_UID.2 and FIA_UAU.2 (which are upgraded from FIA_UID.1 and FIA_UAU.1 from the CAPP). See the CAPP for any applicable rationale. The rationale in the CAPP is applicable with the substitution of FIA_UID.2 for FIA_UID.1 and FIA_UAU.2 for FIA_UAU.1 since the hierarchically greater requirements essentially represent the most restrictive case of their hierarchically lesser counterparts.

8.3 Security Assurance Requirements Rationale

All of the security assurance requirements have been drawn from the Controlled Access Protection Profile (CAPP). See the CAPP for any applicable rationale.

Note that the CAPP rationale addresses the EAL 3 assurance level, while the i5/OS V5R3 TOE supports EAL 4 augmented with ALC_FLR.2. Since EAL 4 provides a higher degree of assurance than EAL 3, the CAPP rationale remains valid. EAL 4 augmented with ALC_FLR.2 was chosen for this TOE because i5/OS V5R3 provides this level of assurance and is suitable for environments requiring assurance greater than the minimum level of assurance required for a CAPP-compliant TOE.

8.4 Requirement Dependency Rationale

All of the security requirements have been drawn from the Controlled Access Protection Profile (CAPP), except FIA_UID.2, FIA_UAU.2, and ALC_FLR.2. See the CAPP for any applicable dependency rationale. Note that any requirements depending on FIA_UID.1, including FIA_UAU.2, are answered by the hierarchically greater FIA_UID.2 requirement and ALC_FLR.2 has no dependencies.

Subsequent to the publication and verification of the CAPP, International Interpretation #65 was finalized. This interpretation introduces a new family of Security Management requirements, Specification of Management Functions (FMT_SMF). While this should not normally affect dependency rationale, that interpretation introduces dependencies from FMT_MSA.1 and FMT_MTD.1, both contained in this Security Target. Hence, it seems as though some FMT_SMF security requirements should be added to this Security Target to fulfill those dependencies. However, while the CAPP is clearly intended to ensure that certain security management functions are controlled if they are made available, it is not evident from the CAPP which, if any, of those security management functions must be present in the first place. This Security Target identifies all applicable security management functions in the TOE and explains how they are appropriately controlled and it is effectively unnecessary to introduce a security functional requirement to demand that certain security management functions must be present.

8.5 Explicitly Stated Requirements Rationale

All of the security requirements have been drawn from the Controlled Access Protection Profile. No security requirements have been explicitly defined in the context of this Security Target.

8.6 Strength of Function Rationale

The TOE minimum strength of function is SOF-medium. The evaluated TOE is intended to operate in commercial and DoD environments processing unclassified information. This security function is in turn consistent with the security objectives described in section 4.

This security target includes a probabilistic or permutational function. The list of relevant security functions and security functional requirements includes:

- Identification and Authentication
  - FIA_UAU.2 - User authentication before any action
  - FIA_SOS.1 – Verification of secrets

The system places the following restrictions on the passwords selected by the user:

- The minimum password length is recommended to be at least six characters long;

The password space is calculated as follows:

Patterns of human usage are important considerations that can influence the approach to searching a password space, and thus affect SOF. Assuming the worst case scenario and the user chooses a number comprising only six characters, the number of password permutations is:

52 alpha characters (upper and lower)
10 digits
+ 17 special characters (!, @, #, $, %, ^, &, *, (,), +, =, <, >, :, ;, ‘ ‘)
79 possible values

79^6 = (79*79*79*79*79*79) = 243,087,455,521

The amount of time it takes to manually type a password given that authentication can only occur based upon manual input is 7 seconds. An attacker can at best attempt (60/7= 8.6 password entries every minute, or 514 password entries every hour.

On average, an attacker would have to enter (243,087,455,521/ 2 = 121,543,727,760) passwords, over (121,543,727,760/ 514) 236,466,396.42 hours, before entering the correct password. The average successful attack would, as a result, occur in slightly less than:

(236,466,396.42 / 24 / 365 =) 26,993.88 years

Another measure to increase the provided protection of this authentication mechanism is the recommendation that the account lockout setting be set to 3 consecutive attempts, and that the account only be unlocked with administrative assistance. This guarantees that there would be no possibility of attempting more than 3 passwords before the account would be locked

In accordance with annex B.3 in the CEM, the elapse time of attack is not practical and thus results in a High strength of function rating, which exceeds SOF-medium.

8.7 TOE Summary Specification Rationale

Each subsection in Section 6, the TOE Summary Specification, describes a security function of the TOE. Each description is followed with rationale that indicates which requirements are satisfied by aspects of the corresponding security function. The set of security functions work together to satisfy all of the security functions and assurance requirements. Furthermore, all of the security functions are necessary in order for the TSF to provide the required security functionality.

This Section in conjunction with Section 6, the TOE Summary Specification, provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions work together to provide all of the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF. **Table 5 Security Functions vs. Requirements Mapping** demonstrates the relationship between security requirements and security functions.

	Security Audit	User Data Protection	Identification and Authentication	Security Management	Protection of the TOE Security Functions
FAU_GEN.1	X				
FAU_GEN.2	X				
FAU_SAR.1	X				
FAU_SAR.2	X				
FAU_SAR.3	X				
FAU_SEL.1	X				
FAU_STG.1	X				
FAU_STG.3	X				
FAU_STG.4	X				
FDP_ACC.1		X			
FDP_ACF.1		X			
FDP_RIP.2a		X			
FDP_RIP.2b		X			
FIA_ATD.1			X		
FIA_SOS.1			X		
FIA_UAU.2			X		
FIA_UAU.7			X		
FIA_UID.2			X		
FIA_USB.1			X		
FMT_MSA.1				X	
FMT_MSA.3				X	
FMT_MTD.1a				X	
FMT_MTD.1b				X	
FMT_MTD.1c				X	
FMT_MTD.1d				X	
FMT_REV.1a				X	
FMT_REV.1b				X	
FMT_SMR.1				X	
FPT_AMT.1					X
FPT_RVM.1					X
FPT_SEP.1					X
FPT_STM.1					X

Table 5 Security Functions vs. Requirements Mapping

8.8 PP Claims Rationale

See Section 7, Protection Profile Claims.