BMC Software PATROL® Version 3.4.11 Security Target

Version 1.0

September 13, 2002

Prepared for:

BMC Software, Inc.
2101 City West Boulevard
Houston, TX 77042

Prepared by:

Computer Sciences Corporation
132 National Business Parkway
Annapolis Junction, MD 20701

Revisions to Document		
Date	Version	Changes Made
February 9, 2001	0.1	Original
April 11, 2001	0.1.1	Corrected grammatical and other minor issues provided in BPC_EDR_002
April 12, 2001	0.2	Made changes/corrections in accordance with BPC_EDR_003
April 25, 2001	0.3	Made changes/corrections in accordance with BPC_EDR_004
May 10, 2001	0.4	Made changes reference removing cryptography SFRs and addressed Validator/Evaluator comments.
June 12, 2001	0.5	Made changes for EDR 002, V0.2
November 29, 2001	0.6	Made changes for EDR 007
December 20, 2001	0.7	Made changes for EDR 011
January 16, 2002	0.8	Made changes to address remove AFL requirement.
February 11, 2002	0.9	
February 26, 2001	0.10a	
March 11, 2002	0.11	Response to EDR 018
May 2, 2002	0.12	Made changes to address practical OS Audit concerns and the role of the console user.
August 19, 2002	1.0d	Changed revision number to 1.0 draft
September 13, 2002	1.0	Final ST
September 17, 2002	1.0	Minor edits to remove blank “a” in section 2.1.14

Table of Contents

List of Tables

List of Figures

1 SECURITY TARGET INTRODUCTION

1 This Section presents security target (ST) identification information and an overview of the ST. A ST document provides the basis for the evaluation of an information technology (IT) product or system (e.g., Target of Evaluation). An ST principally defines:

- A security problem expressed as a set of assumptions about the security aspects of the environment; a list of threats which the product is intended to counter; and any known rules with which the product must comply (in Chapter 3, Security Environment).
- A set of security objectives and a set of security requirements to satisfy the objectives (in Chapters 4 and 5, Security Objectives and IT Security Requirements, respectively).
- The IT security functions provided by the Target of Evaluation (TOE) that meet the set of requirements (in Chapter 6, TOE Summary Specification).

2 The structure and contents of this ST comply with the requirements specified in the Common Criteria (CC), Part 1, Annex C, and Part 3, Chapter 5.

1.1 ST and TOE Identification

3 (ASE_INT.1-1) This section provides the information needed to identify and control this ST and its Target of Evaluation (TOE), the BMC Software, PATROL®, Version 3.4.11. This ST targets an Evaluation Assurance Level (EAL) 2 level of assurance. The TOE consists of BMC Software, PATROL® Version 3.4.11 and the Security Pack for PATROL® Version 3.4.11. There is a Windows and a Unix version for both PATROL® Version 3.4.11 and the Security Pack.

ST Title:	BMC Software, PATROL® Version 3.4.11, Security Target
ST Version:	Version 1.0
Publication Date:	September 13, 2002
TOE Identification:	PATROL® Version 3.4.11
CC Identification:	Common Criteria for Information Technology Security Evaluation, Version 2.1, August 1999
ST Author	Computer Sciences Corporation
ST Evaluation:	Computer Sciences Corporation
Key Words:	BMC Software, PATROL®, resource monitoring

1.2 References

4 The following documentation was used to prepare this ST:

[CC_PART1] Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, dated August 1999, version 2.1, CCIMB-99-031.

[CC_PART2] Common Criteria for Information Technology Security Evaluation – Part 2: Security functional requirements, dated August 1999, version 2.1, CCIMB-99-032.

[CC_PART3] Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance requirements, dated August 1999, version 2.1, CCIMB-99-033.

[CEM_PART1] Common Evaluation Methodology for Information Technology Security – Part 1: Introduction and General Model, dated 1 November 1997, version 0.6.

[CEM_PART2] Common Evaluation Methodology for Information Technology Security – Part 2: Evaluation Methodology, dated August 1999, version 1.0.

1.3 Conventions, Terminology, and Acronyms

5 (ASE_INT.1-4) This section identifies the formatting conventions used to convey additional information and terminology having specific meaning. It also defines the meanings of abbreviations and acronyms used throughout the remainder of the document. This section is provided to assist in the understandability of the ST by the target audience (i.e. evaluators and consumers).

1.3.1 Conventions

6 This section describes the conventions used to denote CC operations on security requirements and to distinguish text with special meaning. The notation, formatting, and conventions used in this ST are consistent with those used in the CC. Selected presentation choices are discussed here to aid the Security Target reader.

1.3.1.1 Operations

7 The CC allows several operations to be performed on functional requirements; *assignment, iteration, refinement*, and *selection* are defined in paragraph 2.1.4 of Part 2 of the CC.

8 The assignment operation is used to assign a specific value to an unspecified parameter, such as the length of a password. An assignment is indicated by showing the value in square brackets [assignment_value(s)].

9 The refinement operation is used to add detail to a requirement, and thus further restricts a requirement. Refinement of security requirements is denoted by **bold text.**

10 The selection operation is picking one or more items from a list in order to narrow the scope of a component element. Selections are denoted by *underlined italicized text.*

11 Plain *italicized text* is used for both official document titles and text meant to be emphasized more than plain text.

12 Iterated functional and assurance requirements are given unique identifiers by appending to the component name, short name, and functional element name from the CC an iteration number inside parenthesis, i.e., FMT_MTD.1.1 (1) and FMT_MTD.1.1 (2).

13 Also, explicitly stated requirements not based on the CC Part 2 security functional requirements will be identified by the extension EXP, i.e., **FCL_SSL_EXP.1.**

1.3.1.2 Naming Conventions

14 This section describes the naming conventions used for assumptions, threats, policies, and objectives given within this ST. When an assumption, threat, policy or objective applies to a subset of the TOE, a subscript is used to clarify the pertinent part of the TOE. A "C" subscript refers to the console and an "R" subscript refers to the remote portions of the TOE.

15 **Assumptions:** TOE security environment assumptions are given names beginning with "A." and are presented in alphabetical order. This prefix will be subscripted to reflect a given component for multi-component TOEs as required.

Examples:

16 A.ADMIN – Assumption allocated to TOE as an entity.

17 A_C.CONFIG – Assumption allocated to the Console component.

18 **Threats:** TOE security threats for the TOE and for the environment are given names beginning with "T." and "TE." Respectively, and are presented in alphabetical order. The TOE prefix will be subscripted to reflect threats to a given component for multi-component TOEs as required.

Examples:

19 T.ATTACK_DATA – Threat to/countered by the TOE as an entity.

20 T_R.ATTACK_DATA – Threat to/countered by the "remote" component of the TOE.

21 **Policies:** TOE security environment policies are given names beginning with "P." and are presented in alphabetical order. This prefix will be subscripted to reflect a given component for multi-component TOEs as required.

Examples:

22 P.ACCOUNT – Policy supported by the TOE as an entity.

23 P_C.ACCOUNT – Policy supported by the "Console" component of the TOE.

24 **Objectives:** Security objectives for the TOE and for the environment are given names beginning with "O." and "OE." respectively, and are presented in alphabetical order. These prefixes will be subscripted to reflect a given component for multi-component TOEs as required.

Examples:

25 O.ADMIN – Objective for the TOE as an entity.

26 OE.AUTHORIZATION – Objective for the environment.

27 O_R.ADMIN – Objective of the "remote" component of the TOE.

1.3.2 Terminology

28 In the Common Criteria, many terms are defined in Section 2.3 of Part 1. The following terms are a subset of those definitions. They are listed here to aid the user of the Security Target.

User	Any entity (human user or external IT entity) outside the TOE that interacts with the TOE.
Human user	Any person who interacts with the TOE.
Authorized User	A user that, in accordance with the TOE Security Policy (TSP) may perform an action.
External IT entity	Any IT product or system, untrusted or trusted, outside of the TOE that interacts with the TOE.
Role	A predefined set of rules establishing the allowed interactions between a user and the TOE.
Identity	A representation (e.g., a string) uniquely identifying an authorized user, which can be either the full or abbreviated name of that user or a pseudonym.
Authentication data	Information used to verify the claimed identity of a user.

29 In addition to the above general definitions, this Security Target provides the following specialized definitions:

PATROL® System Administrator	A role with which a human user is associated to administer both the functionality and security parameters of the TOE and the IT Environment. Such users are not subject to any access control requirements once identified to the TOE and are therefore trusted to not compromise the security policy enforced by the TOE.
User Role Administrator	A role with which a human user is associated to administer the user roles on the TOE. Such users are not subject to any access control requirements once identified to the TOE and are therefore trusted to not compromise the security policy enforced by the TOE.
Console User	A role with which a human user is associated that can start a PATROL® console. Such users have access to TOE directories and files, but do not have the authority to alter User Roles. Such users are trusted to not compromise the files of the TOE that allow the TOE to function.

30 When the general term **Administrator** is used, it refers to both the **PATROL® System Administrator** and the **User Role Administrator**. In the case of Patrol Classic, the user types

are PATROL System Administrator and User Role Administrator. Both these roles indicate a human user who is trusted to perform security critical operations within the TOE. No non-administrative users of the TOE have been identified.

1.3.3 Acronyms

31 The following abbreviations are used in this Security Target:

API	Application Program Interface
CA	Certificate Authority
CC	Common Criteria for Information Technology Security Evaluation
CI	Configuration Items
CLR	Certificate Revocation Lists
DAC	Discretionary Access Control
EAL	Evaluation Assurance Level
EGID	Effective Group ID
EUID	Effective User ID
FIPS PUB	Federal Information Processing Standard Publication
IT	Information Technology
KM	Knowledge Module
OSP	Organizational Security Policy
PEM	PATROL® Event Manager or Privacy Enhanced Mail
PP	Protection Profile
PSL	PATROL® Script Language
SFP	Security Function Policy
SOF	Strength of Function
SSL	Secure Socket Layer
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSP	TOE Security Policy

1.4 Security Target Overview

32 (ASE_INT.1-2) This ST forms the basis for evaluation of the TOE, known as the BMC Software product PATROL®, Version 3.4.11 (NOTE: this is a limited distribution version), and includes the following PATROL® components: PATROL® Console, PATROL® Agents, the PATROL® Event Manager (PEM), and PATROL® Knowledge Modules (KMs). These basic components of the PATROL® suite of products provide a set of tools designed to assist in database, network, and system administration. In the context of PATROL® applications are any resource used by, or running on, a computer.

33 The PATROL® Console is the main interface with PATROL® Agents. It provides an object-oriented, graphical workspace where the status of vital resources in the distributed environment can be monitored.

34 A PATROL® Agent performs PATROL® activities using programmed knowledge stored in PATROL® Knowledge Modules (KMs). It runs autonomously on monitored computers.

35 The PATROL® Event Manager (PEM) displays events forwarded by PATROL® Agents in a manner that makes information about the enterprise more meaningful. It can be run as a stand-alone facility or from the PATROL® Console.

36 The PATROL® Knowledge Modules (KMs) are the programmed knowledge stored and used by PATROL® Agents to perform useful actions. PATROL® KMs are files that describe how to monitor and manage an application, how to identify objects, how to present them in an icon window, and what actions to take when monitored objects change state. These files contain commands written in PATROL® Script Language (PSL) and are loaded by PATROL® Agents.

37 The TOE with support from its IT environment provides the following security features:

a) Auditing,
b) User Data Protection,
c) Identification and Authentication,
d) Security Management, and
e) Protection of Security Functions,

38 A summary of the PATROL® security features can be found in Section 2, TOE Description. A detailed description of the PATROL® security features can be found in Section 6, TOE Summary Specification.

1.5 Common Criteria Conformance

39 (ASE_INT.1-3) This ST conforms to CC Part 2 extended, and is CC Part 3 conformant at the EAL 2 level of assurance.

2 TOE DESCRIPTION

40 This section provides a general description of the physical and logical scope and boundaries of the TOE.

2.1 Product Type

41 (ASE_DES.1-1) PATROL® is a systems application and event management tool. It provides an environment by which the status of every vital resource in the distributed environment being managed can be monitored. PATROL® is a suite of products consisting of:

- PATROL® Console,
- PATROL® Agents,
- PATROL® Event Manager (PEM), and
- PATROL® Knowledge Modules (KMs).

42 In the context of PATROL®, applications are any resource used by, or running on, a computer Figure 1 displays the basic PATROL® activities.

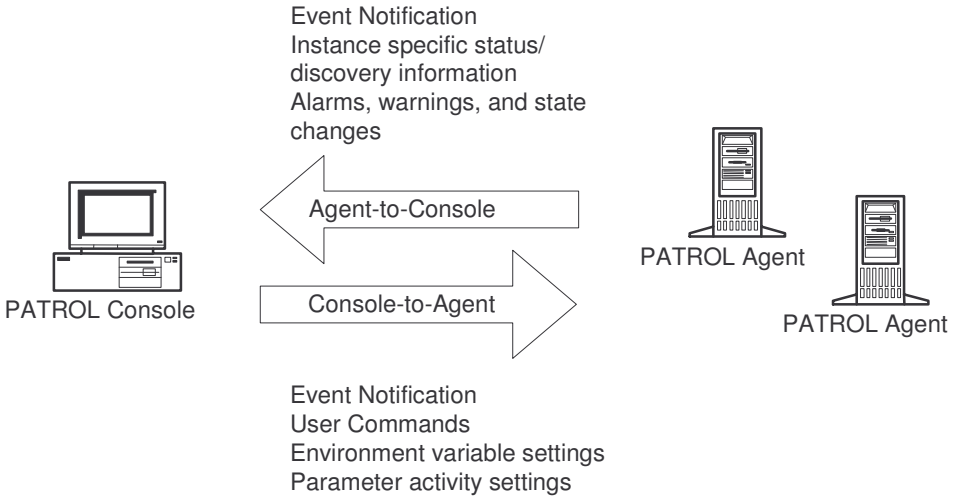

Figure 1: Basic PATROL® Activities

2.1.1 Scope and Boundaries of the Evaluated Configuration

43 This section provides a general description of the physical and logical scope and boundaries of the TOE.

2.1.1.1 Physical Scope and Boundary

44 (ASE_DES.1-2) The TOE configuration consists of two major executable components:

a) the PATROL® Console Workstation, and

b) one or more PATROL® Agents that execute on remote computer platform(s) functioning as either a workstation or server.

45 The Console workstation executes the PATROL® Console and PATROL® Agent. For UNIX™ workstations the PATROL® Event Manager is an additional sub-component that is instantiated on the PATROL® Console Workstation; under Windows 2000™, the PATROL® Event Manager is integral to the PATROL® Console applications. The remote server(s)/workstation(s) execute the PATROL® Agent(s).

46 PATROL Operator Console and the PATROL Developer Console are the graphical workspaces from which commands are issued to manage the distributed environment monitored by PATROL. The PATROL Console displays all of the monitored computers and applications. The PATROL Console can work in two console modes: Operator Console and Developer Console.

47 With the PATROL Operator Console the following tasks can be performed:

- define which applications PATROL should monitor
- monitor and manage computers and applications through the PATROL Agent and PATROL Knowledge Modules
- monitor the PATROL Agent's use of resources
- run predefined or user-defined commands and tasks against monitored machines
- run state change action commands on the PATROL Console machine when a state change occurs on a monitored computer
- log on to any managed computer (only for Unix and OpenVMS.)
- start and stop PATROL Agents remotely
- view parameter data
- retrieve historical data stored by the PATROL Agent

48 The PATROL Developer Console in the evaluated configuration is used only in the installation and initial start-up. The PATROL Developer Console is responsible for the following restricted Security Management activities:

- committing PATROL® KM changes to a PATROL® Agent; (Changes to the KM result in an unevaluated configuration.)
- issuing operating system commands at the PATROL® system output window; (Outside scope of evaluated configuration.)
- modifying the PATROL® Agent's parameter attributes; (Outside scope of evaluated configuration.)
- launching a PATROL® Console in developer mode. (Outside scope of evaluated configuration.)

49 PATROL Agent is the core piece of the PATROL architecture that monitors and manages host computers. The PATROL Agent performs the following tasks:

- Runs commands to collect system or application information; the information is collected according to applications and parameters defined in Knowledge Modules
- Stores information locally for retrieval by the PATROL consoles

- Loads specified Knowledge Modules (KMs) at start-up runs menu commands, and updates InfoBoxes in the PATROL Console
- Acts as a service provider for event management

50 The PATROL Event Manager (PEM Console) is the component by which the following tasks can be performed:

- View events
- Manage events and use events to control the managed environment
- Trigger events
- Generate event statistics
- Acknowledge events
- Close events

51 The PATROL Knowledge Module is a set of files from which a PATROL Agent receives information about all of the resources, such as databases and file systems running on a monitored computer. (Changes to the KM will result in an unevaluated configuration.) PATROL KMs provide information to the PATROL Agent about:

- The identity of objects
- Parameters
- Actions to take when an object changes a state
- How to monitor the application

52 Physically, each TOE platform consists of a processor architecture appropriate for the Operating System on which the TOE component runs. The TOE does not include any physical network components between the adapters of a connection between platforms. The ST assumes that any network connections, equipment (e.g., routers), and cables are appropriately protected in the TOE security environment.

53 The evaluated TOE configuration includes the hardware and software elements identified in Table 1.

Table 1: Evaluated TOE Configuration Components

Components	Items
Evaluated Software	BMC Software PATROL®, Version 3.4.11: • PATROL® Console for UNIX • PATROL® Event Manager (UNIX) • PATROL® Agent for (UNIX) • PATROL® Console for Microsoft Windows 2000 • PATROL® Agent for Microsoft Windows 2000 • PATROL® KM for UNIX V8.3 • PATROL® KM for NT V.3.5
Non-Evaluated Software (IT Environment)	Certificate Authority
Hardware (IT Environment)	**PATROL® Console/Agent for UNIX™:** • SUN SPARC-based platform running Solaris 2.7 **PATROL® for Microsoft Windows 2000 Server:** • Intel x86-based platform capable of running Microsoft Windows NT 4.0, SP 6a

54 Physically, each TOE component is composed of the functionally appropriate PATROL® software and the requisite networked computer platform.

2.1.1.2 Logical Scope and Boundary

55 (ASE_DES.1-3) The TOE logical boundary consists of the functionality inherent in the PATROL® Console, Agent, and Event Manager software. Additionally, the following KMs authored, and provided, by BMC Software are included:

- PATROL® KM for UNIX V8.3
- PATROL® KM for NT V3.5

56 The Unix and Windows KMs are actually a set of Scripts. The handling of BMC PSL scripts allow for flexibility to associate multiple scripts when loading the overall knowledge modules. While most of the internal functions are in compiled PSL scripts (.psl files), the KMs themselves are started by a KM list file (.kml) which contains the list of all sub-KMs to load. The sub-KMs (.km files) are actually KM scripts that load compiled PSL scripts (.psl files). These KM files are merely to make it simple to load compiled script modules.

57 Figure 2 illustrates the logical boundary of the TOE.

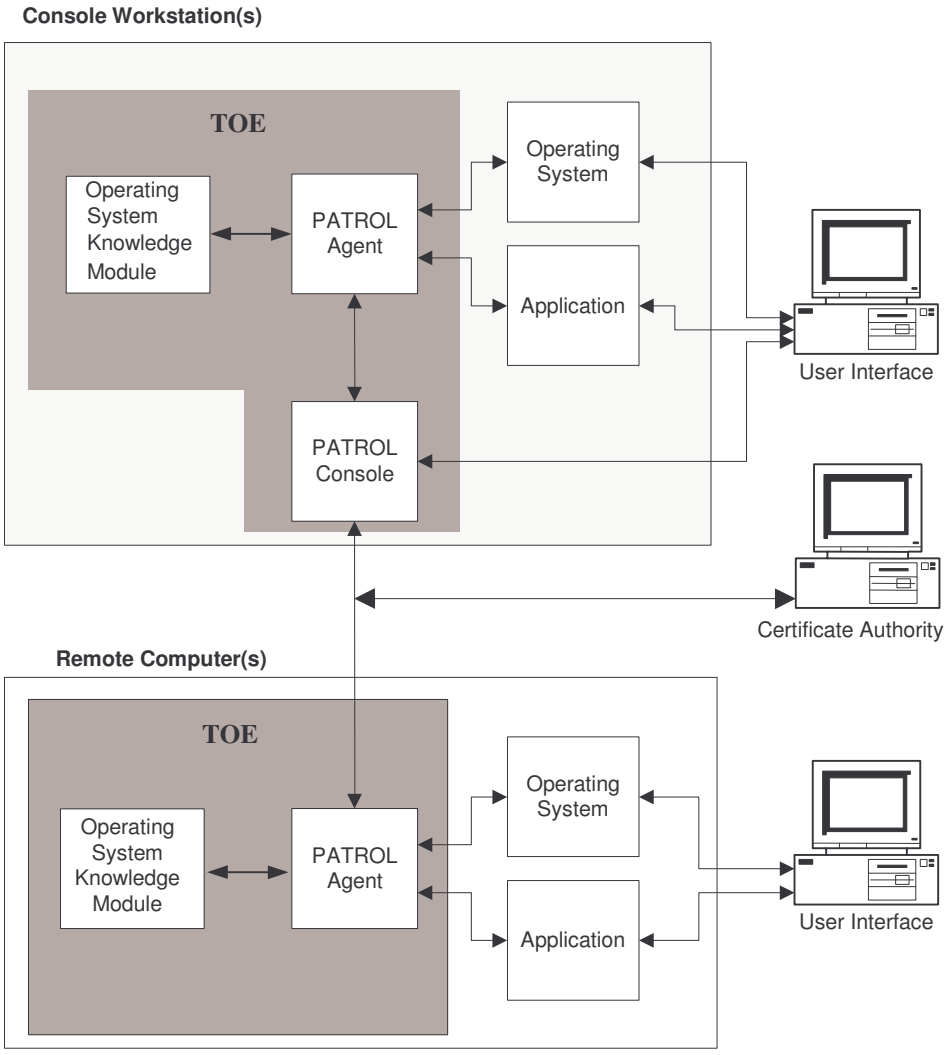

Figure 2: TOE Logical Boundary

2.1.1.3 TOE Security Functionality

58 The TOE provides the following security features:

a) Auditing,
b) User Data Protection,
c) Identification and Authentication, and
d) Security Management,

59 **Auditing –** PATROL® has the capability to generate audit logs. Audit information generated by the system is based on PATROL® Agent audit logs. Audit functionality provided by the IT Environment is outside the scope of this evaluation. The PATROL® Agent audit log feature permits the recording of various security-related aspects of PATROL® operation. PATROL® audit logs record information such as:

- commands that are executed as a result of Infobox or Menu commands,
- which console-connection runs commands (listed by console ID),

- connect/disconnect,
- commit operations,
- configuration operations,
- spawned commands.

60 **User Data Protection** – PATROL® provides discretionary access control restrictions; inter-TSF user data confidentiality; and data exchange integrity. PATROL® uses access control lists (ACLs) to restrict access to PATROL® Agents.

61 **Identification and Authentication** – PATROL® provides for identification and authentication of users on PATROL® Consoles and Agents through the use of a function called the DEFAULT ACCOUNT. The default account is used by the PATROL® Agent for executing monitoring commands, such as parameters and recovery actions.

62 **Security Management** – PATROL® includes a number of functions to manage security policy implementation. Policy management is controlled through a combination of ACLs, and security role definitions/assignments.

63 ACLs: Through an ACL the agent allows The PATROL System Administrator to define the following:

- Which users have access to the Agent,
- Which hosts have access to the Agent,
- Which type of PATROL® consoles and utilities have access to the agent, and
- any combination of the above three types of control.

64 HostName and UserName Attribute Conventions: In an ACL entry, any number of masking techniques can be used for the host name and user name attributes.

- UserName: The name of a local account that the connecting console may request to use.
- HostName: A machine (Console) that is authorized to connect to this agent. A hostname can be specified by using the fully qualified name, the short name, or a partial name (pattern) created with a wildcard specification in which the first character is a '*', with other characters following.

65 With respect to PATROL® User Roles:

a) The PATROL® System Administrator can edit the PATROL® user-roles file to protect the enterprise from unauthorized use of PATROL® Console operations.

b) The User Role Administrator can grant or remove the ability of specific users to perform specific console operations. For example, they may need to restrict certain PATROL® users from overriding agent parameter attributes while permitting certain other trusted operators to perform the same operation. The tasks controlled through user roles include:

- committing PATROL® KM changes to a PATROL® Agent;

- issuing operating system commands at the PATROL® system output window;
- modifying the PATROL® Agent's parameter attributes;
- launching a PATROL® Console in developer mode.

c) The Console User can perform console operations. This includes starting a PATROL® console and monitoring system activity.

66 The PATROL® user roles file allows the specification of the conditions under which the PATROL® Administrator permits or disables the console operations listed above. The criteria to consider when permitting or disabling operations are:

a) name of the logged in user;

b) name of the host machine on which the PATROL® Console is running;

c) name of the host machine to which the PATROL® Console is connected;

d) mode (developer or operator) in which the PATROL® Console is running.

2.1.1.4 TOE IT Environment Security Functionality

67 The TOE with support from the IT environment provides the following security features:

a) User Data Protection,
b) Identification and Authentication,
c) Protection of Security Functions

68 **User Data Protection** – PATROL® uses a Certificate Authority to provide additional access control protection and inter-TSF data confidentiality through the use of SSL protocols. The SSL only allows authorized users access to encrypted data.

69 PATROL® enforces the evidence of the origin of the transmission of data, and the verification this evidence as provided by a third party Certificate Authority. PATROL® Agent/Console communication supports TCP/IP standard protocols for communication across network connections on pre-determined well-known port numbers. PATROL® does not require the use of "proprietary" network protocols to function properly on an enterprise network.

70 **Identification and Authentication** –PATROL® uses SSL to automatically authenticate a peer by:

- Verifying all signatures in the certificate chain;
- Checking that the certificate chain terminates in a trusted root.

71 If both of these checks succeed, the peer is regarded as having passed the built-in authentication checks.

72 **Protection of Security Functions** – PATROL® uses a Certificate Authority to support protection of TOE security functions. The CA provides against unauthorized configuration data disclosure and modification by using a suite of standard protocols including Secure Socket Layer (SSL) and Data Encryption Standard (DES) encryption.

3 TOE SECURITY ENVIRONMENT

73 This section describes the security aspects of the intended environment for the evaluated TOE. This includes information about the physical, personnel, procedural, connectivity, and functional aspects of the environment.

74 The TOE is a distributed multi-component software product – the Console workstation and the remote/Agent platforms. Consequently, there are assumptions, threats, objectives, and organizational security policies for the TOE as a whole/system and for each of the components identified above.

3.1 Assumptions

75 The specific conditions listed in Table 2 are assumed to exist for the TOE as a whole system.

Table 2: Assumptions for the TOE – PATROL® "system."

Name	Description	Functional Aspect
A.ACCESS_CONTROL	The operating systems upon which the Console and Agent software runs will be configured to restrict modification to TOE executables and configuration files to only PATROL Authorized Administrators.	Functional
A.SAME_ADMIN	The operating systems upon which the PATROL Console and Agent software runs are under the same administrative management as the Console and Agent.	Functional
A.MANAGE	There will be one or more competent individuals assigned to manage the TOE. Those assigned to manage the TOE will have been appropriately trained.	Personnel
A.NOEVIL	Administrators are not careless, willfully negligent, nor hostile, and will follow and abide by all administrator guidance; however, they are capable of error.	Personnel
A.OPERATE_CORRECT	The computer platforms and operating systems upon which the Console and Agent software runs will operate correctly. This includes the hardware being able to provide reliable system time.	Functional
A.CAOPERATE_CORRECT	The Certificate Authority upon which the Agent security functionality depends will operate correctly.	Functional

Name	Description	Functional Aspect
A.PEER	Any other systems that will communicate with the TOE are under the same management control and will operate under the same security policy constraints.	Connectivity
A.PHYSICAL_PROTECT	The processing resources of the TOE will be located within facilities providing controlled access to prevent unauthorized physical access.	Physical

76 The specific conditions listed in Table 3 are assumed to exist for the TOE Console platform.

Table 3: Assumptions for the TOE – Console Platform

Name	Description	Functional Aspect
A_C.AUTHORIZED	Only authorized TOE Console software users and administrators will have accounts on those platforms on which the TOE Console software executes.	Personnel

77 The specific conditions listed in Table 4 are assumed to exist for the TOE Remote Agent Platforms.

Table 4: Assumptions for the TOE – Remote (Agent) Platform

Name	Description	Functional Aspect
A_R.BENIGN	Only authorized users will have physical access to the Agent platform(s) and are expected to operate in a cooperative manner in a benign environment.	Personnel

78 The specific conditions listed in Table 5 are assumed to exist for the TOE Remote Agent Platforms.

Table 5: Assumptions for the TOE IT Environment

Name	Description	Functional Aspect
A_E.Certificate_Authority	The IT Environment will provide a Certificate Authority using a suite of standard protocols including Secure Socket Layer (SSL) and Data Encryption Standard (DES) encryption compatible with the SPYRUS libraries used by the TOE.	Functional

3.2 Threats

79 Threats may be addressed either by the TOE or by its intended environment (for example, using personnel, physical, or administrative safeguards).

3.2.1 Threats Addressed by the TOE

80 Table 6 identifies threats to the protected resources that are addressed by the TOE in a system context.

Table 6: Threats Addressed by the TOE – PATROL® "system."

Name	Description
T.REPLAY	A hostile/unauthorized user would use replay to obfuscate unauthorized activity.
T.TRAFFIC_SPOOF	A hostile/unauthorized user would attempt to spoof Agent communications in order to hide or perform unauthorized activity, or provide false data.
T.TROJAN	A hostile/unauthorized user will attempt to use the PATROL® Scripting Language "files create" as a mechanism to get file access.
T.UNAUTH_ACCESS_DATA	A hostile/unauthorized user would attempt to read TOE data/configuration files in order to: • Ascertain TOE, or managed application, secrets. • Modify TOE behavior.
T.UNDETECTED_ACTIONS	Authorized and unauthorized users would use the fact that identification and recording of their actions was not taking place in order to circumvent the TSP.

81 Table 7 identifies those threats that are addressed by the TOE from the Console platform perspective.

Table 7: Threats addressed by the TOE – Console Platform

Name	Description
T_C.UNAUTH_DEPLOY	A hostile/unauthorized user would attempt to deploy an unauthorized KM(s) on a remote platform to change/modify/attack the "system"/system management.
T_C.UNAUTH_CHANGES	A hostile/unauthorized user would attempt to make unauthorized changes to the Agent and KM configuration to change/modify/attack the "system"/system management.
T_C.UNAUTH_COMMANDS	A hostile/unauthorized user would attempt to execute unauthorized system commands on the target system to change/modify/attack the "system"/system management.

82 Table 8 presents the threats to the protected resources that are addressed from the PATROL® remote (Agent) context.

Table 8: Threats Addressed by the TOE – Remote (Agent) Platforms

Name	Description
T_R.ELEVATE_ACCESS	A hostile/unauthorized user may attempt to bypass the security of the TOE through attempting to use the PATROL® Agent to elevate access to remote machines.
T_R.APPLICATION_SECRETS	A hostile/unauthorized user will attempt to access Agent configuration/data files in order to obtain secrets (e.g., passwords) to monitored applications in order to gain unauthorized access to those applications.

Name	Description
T_R.KM_TAMPER	A hostile/unauthorized user will attempt to modify Agent and/or KM behavior by making unauthorized changes to KM script files to modify TOE behavior, or gain unauthorized access.

3.2.2 Threats Addressed by the Operating Environment

83 Table 9 identifies threats to the assets against which specific protection within the IT Environment is required.

Table 9: Threats Addressed by Operating Environment

Name	Description
TE.UNAUTH_ACCESS	Hostile/unauthorized users can read from, or write to, PATROL® configuration and/or data files in order to modify system behavior without being detected.
TE.UNAUTH_ACCESS_NETWORK	A hostile/unauthorized user would attempt to read packets sent between TOE components in order to: • Ascertain the status of network resources for which they were not authorized, • Ascertain TOE, or managed application, secrets.
TE.UNAUTH_DATA_MOD_NETWORK	A hostile/unauthorized user would attempt to alter data packets between a PATROL® Agent and the PATROL® Console in order to hide unauthorized activity.
TE.UNAUTH_USAGE	Hostile/unauthorized users can instantiate, or terminate TOE software processes to circumvent system management.
TE.UNDETECTED_ACTIONS	Authorized and unauthorized users will not have their actions recorded and thereby circumvent the TSP.

3.3 Organizational Security Policies

84 The Organizational Security Policies (OSPs) given in Table 10 are identified for the TOE.

Table 10: Organizational Security Policies

Name	Description
P.ACCOUNTABLE	Users of the system must be held accountable for their actions.
P.AUTHORIZATION	The system must have the ability to limit the extent of each user's authorization.
P.INFO_ACCESS	Information shall only be accessible by authorized individuals and processes with a "need to know."

Name	Description
P.INTEGRITY	The system must have the ability to protect system data in transmission between distributed parts of the protected system.
P.MANAGE	The TOE shall be managed and maintained so that its security functions are implemented and preserved throughout its operational lifetime.
P.TRACE	The system will have the ability to review the actions of, and interactions between, components of the system.

4 SECURITY OBJECTIVES

85 The purpose of a security objective is to detail the planned response to a security problem or threat. Threats can be directed against the TOE or the security environment or both, therefore, the CC identifies two categories of security objectives:

- Security objectives for the TOE, and
- Security objectives for the IT Environment.

4.1 SECURITY OBJECTIVES FOR THE TOE

86 Table 11 identifies the security objectives to address security concerns that are directly addressed by the TOE.

Table 11: Security Objectives for the TOE

Name	Description
O.ADMIN	The TOE must provide functions to enable system administrators and administrators to effectively manage and maintain the TOE and its security functions, ensuring that only they can access administrative functionality.
O.AUDIT	The TOE must provide an audit capability to report security relevant events so that the responsible subjects can be held accountable for their actions.
O.CONNECT	The TOE must only allow connectivity between Consoles and Agents as determined by the PATROL® System Administrator.
O.CONFIDENTIALITY	The TOE must provide confidentiality by protecting the content of the information transferred between components of the TOE.
O.ENTITY_IDENTIFICATION	The TOE must identify entities to verify that permission for connection/access to TOE components, or data is authorized.
O.INTEGRITY	The TOE must apply integrity protection to all information it releases between components. Upon receipt of protected data, the TOE must verify that the received data accurately represents that the data was protected.
O.SEPARATE_ROLES	The TOE must accommodate separate roles for Authorized Administrators to limit their access to the TOE security mechanisms.

4.2 SECURITY OBJECTIVES FOR THE ENVIRONMENT

87 Table 12 identifies security objectives to address security concerns that are directly addressed by the IT Environment.

Table 12: Security Objectives for the IT Environment

Name	Description
OE.CERTIFICATE_SUPPORT	The TOE environment must provide reliable Certificate Authority functions including correct operation and functionality.
OE.DISCRETIONARY_ACCESS	The TOE environment must provide discretionary access control (DAC) to protect TOE resources and limit TOE application instantiation.
OE.INSTALL	Those responsible for the TOE must ensure that the TOE is delivered, installed, managed, and operated in a manner that maintains IT security objectives.
OE.PHYSICAL_PROTECTION	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
OE.PLATFORM_SUPPORT	The TOE environment must provide reliable platform functions including: correct hardware operation and functionality including providing system time; correct platform software operation and functionality.

5 IT SECURITY REQUIREMENTS

88 IT security requirements include:

- TOE security requirements, and (optionally)
- TOE's IT Environment security requirements upon which satisfaction of the TOE's security objectives depend.

89 These requirements are discussed separately below.

5.1 TOE Security Requirements

90 The CC divides security requirements into two categories:

- *Security functional requirements (SFRs)*: that is, requirements for security functions such as information flow control, audit, and identification.
- *Security assurance requirements (SARs)*: provide grounds for confidence that the TOE meets its security objectives (for example, configuration management, testing, and vulnerability assessment).

91 This section presents the security functional and assurance requirements for the TOE and its supporting IT Environment.

5.1.1 TOE Security Functional Requirements

92 Table 13 identifies the Security Functional Requirements (SFRs) for the TOE.

Table 13: TOE Security Functional Requirements

Functional Component ID	Functional Component Name	Dependencies
Security Audit		
FAU_GEN.1	Audit data generation	FPT_STM.1
User Data Protection		
FDP_ACC.2	Complete Access Control	FDP_ACF.1
FDP_ACF.1	Security attribute based access control	FDP_ACC.1 FMT_MSA.3
Identification and Authentication		
FIA_ATD.1	User Attribute Definition	None
FIA_UAU.3	Unforgeable Authentication	None
FIA_UID.2	User Identification before any action	None
FIA_USB.1	User-subject binding	FIA_ATD.1
Security Management		
FMT_MOF.1 (1)	Management of security functions behavior	FMT_SMR.1
FMT_MOF.1 (2)	Management of security functions behavior	FMT_SMR.1
FMT_MSA.1	Management of security attributes	[FDP_ACC.1 or FDP_IFC.1], FMT_SMR.1

Functional Component ID	Functional Component Name	Dependencies
FMT_MSA.3	Static Attribute Initialization	FMT_MSA.1 FMT_SMR.1
FMT_MTD.1 (1-2)	Management of TSF data	FMT_SMR.1
FMT_SMR.1	Security roles	FIA_UID.1
Explicitly-Stated		
FCL_SSL_EXP.1	Secure Socket Layer	FAU_GEN.1

93 Requirements Note: This ST consists of two access control Security Function Policies (SFP). The first is called the AccessControl SFP and is satisfied by the TOE. The subjects under control of the AccessControl SFP are the Console(s) and Agent(s). The objects controlled are the connections/communication between the subjects. The second SFP is the Discretionary Access Control (DAC) and is satisfied by the hardware platform in the IT Environment. The subjects, objects, and controlled operations are named in FDP_ACC.1.

5.1.1.1 Class FAU: Security Audit

94 FAU_GEN.1: Audit data generation

Hierarchical to: No other components.

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

a) Start-up and shutdown of the audit functions;

b) All auditable events for the *not specified* level of audit; and

c) [the events in Table 14].

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, the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [information specified in column three of Table 14.]

Table 14: Auditable Events

Functional Component	Auditable Event	Additional Audit Record Contents
PATROL Agent	Commands executed as a result of Infobox or Menu commands	The entry in the log file records the console-ID of the peer and the local account name used for the connection.
PATROL Agent	Connections/Disconnections	The entry in the log file records the console ID of the

Functional Component	Auditable Event	Additional Audit Record Contents
		peer, the console type, and the local account name used for this connection.
PATROL Agent	Commit Operations	The entry in the log file records the name of the file, the console ID of the connection performing the commit, and the local account that is used for the connection.
PATROL Agent	Configuration Operations	The entry in the log file records the events that change variables, kill the agent, and send a license file and PSL pconfig() operations.
PATROL Agent	Spawned Commands	The entry in the log file records explicitly created external processes. Note: The agent does not create a log entry for implicitly created commands. This means that the PATROL agent will not log the commands that are created by a process that it creates. Example: Using PSL popen() to create a process, and then sending a command down the channel for this process to execute. The agent logs the creation of the popen() process.

Dependencies: FPT_STM.1 Reliable time stamps

5.1.1.2 Class FDP: User Data Protection

95 FDP_ACC.2 Complete access control

Hierarchical to: FDP_ACC.1

FDP_ACC.2.1 The TSF shall enforce the [AccessControl SFP] on [communication requests between the Console and Agents] and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2 The TSF shall ensure that all operations between any subject in the TSC and any object within the TSC are covered by an access control SFP.

Dependencies: FDP_ACF.1 Security attribute based access control

96 FDP_ACF.1 Security attribute based access control

Hierarchical to: No other components.

FDP_ACF.1.1 The TSF shall enforce the [AccessControl SFP] to objects based on [verification of the Console's authorization to connect as reported in the Agent's access control list].

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

a) [The requesting Console must be on the Agent's access control list;

b) The PATROL® Agent is running under a valid user account on the monitored host that is properly defined on the requesting console;

c) The Default account is configured properly on the PATROL® Agent such that it matches the locally defined account; and

d) The requesting console has the proper roles defined for the operation attempted (Operator or Developer Console)].

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rule: [none].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the [none].

Dependencies: FDP_ACC.1 Subset access control

FMT_MSA.3 Static attribute initialization

5.1.1.3 Class FIA: Identification and Authentication

97 FIA_ATD.1 User Attribute definition

Hierarchical to: No other components.

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

a) [defined role,

b) user-ids,

c) hostname].

Dependencies: No dependencies

98 FIA_UAU.3 Unforgeable authentication

Hierarchical to: No other components

FIA_UAU.3.1 The TSF shall *prevent* use of authentication data that has been forged by any **Agent or Console** of the TSF.

FIA_UAU.3.2 The TSF shall *prevent* use of authentication data that has been copied by any **Agent or Console** of the TSF.

Dependencies: No dependencies

99 FIA_UID.2 User Identification before any action

Hierarchical to: FIA_UID.1 Timing of identification

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

Dependencies: No dependencies

100 FIA_USB.1 User-subject binding

Hierarchical to: No other components.

FIA_USB.1.1 The TSF shall associate the appropriate user security attributes with subjects acting on behalf of that user.

Dependencies: FIA_ATD.1 User attribute definition

5.1.1.4 Class FMT: Security Management

101 FMT_MOF.1 Management of security functions behavior

Hierarchical to: No other components.

FMT_MOF.1.1 (1) The TSF shall restrict the ability to *disable, enable, and/or modify the behavior of* the functions: [

a) management of audit record generation;

b) modification of PATROL® Knowledge Modules]

c) management of the certificate revocation list (CRL)]

to [PATROL® System Administrator].

Dependencies: FMT_SMR.1 Security Roles

FMT_MOF.1.1 (2) The TSF shall restrict the ability to *enable and/or modify the behavior of* the functions[

a) committing KM changes to a PATROL® Agent;

b) issuing operating system commands at the PATROL® system output window;

c) modifying the PATROL® Agent's parameter attributes;

d) launching a PATROL® Console in developer mode,

to [**users as specified by the User Role Administrator**]].

Dependencies: FMT_SMR.1 Security Roles

102 FMT_MSA.1 Management of security attributes

Hierarchical to: No other components.

FMT_MSA.1.1 The TSF shall enforce the [AccessControl SFP] to restrict the ability to *delete, modify,**and** [add]* the security attributes **in a rule** [listed in section FDP_ACF.1.2] to [the PATROL® System Administrator].

Dependencies: [FDP_ACC.1 Subset access control
or
FDP_IFC.1 Subset information flow control]
FMT_SMR.1 Security Roles

103 FMT_MSA.3 Static Attribute Initialization

Hierarchical to: No other components.

FMT_MSA.3.1 The TSF shall enforce the [AccessControl and DAC SFPs] to provide *permissive* default values for security attributes that are used to enforce the SFPs.

FMT_MSA.3.2 The TSF shall allow the [PATROL® System Administrator] to specify alternative initial values to override the default values when an object or information is created.

Dependencies: FMT_MSA.1 Management of security attributes

FMT_SMR.1 Security roles

104 FMT_MTD.1 (1) Management of TSF data

Hierarchical to: No other components.

FMT_MTD.1.1 (1) The TSF shall restrict the ability to [*manage*] the [PATROL audit trail] to [the PATROL® System Administrator].

Dependencies: FMT_SMR.1 Security roles

105 FMT_MTD.1 (2) Management of TSF data

Hierarchical to: No other components.

FMT_MTD.1.1 (2) The TSF shall restrict the ability to *modify, delete, and clear* the [user identity used in FIA_UID.2] to [the PATROL® System Administrator, User Role Administrator].

Dependencies: FMT_SMR.1 Security roles

106 FMT_SMR.1 Security roles

Hierarchical to: No other components.

FMT_SMR.1.1 The TSF shall maintain the roles [PATROL® System Administrator, and User Role Administrator, and Console User].

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

Dependencies: FIA_UID.1 Timing of identification

5.1.1.5 Class FCL_EXP: Explicitly Stated Protocol Requirement

107 FCL_SSL_EXP.1 Secure Socket Layer Protocol

Hierarchical to: No other components.

FCL_SSL_EXP.1.1 The TSF will provide Secure Socket Layer (SSL) standard protocol based on the TOE's SPYRUS libraries.

Dependencies: FAU_GEN.1 Audit data generation

5.1.2 IT Environment Functional Requirements

108 Table 15: IT Environment Security Functional Requirements identifies the Security Functional Requirements (SFRs) for the IT Environment.

Table 15: IT Environment Security Functional Requirements

Functional Component ID	Functional Component Name	Dependencies
User Data Protection		
FDP_ACC.1	Subset access control	FDP_ACF.1
FDP_ACF.1	Security attribute based access control	FDP_ACC.1
Identification and Authentication		
FIA_ATD.1	User Attribute Definition	None
FIA_UAU.2	User Authentication before any action	FIA_UID.1
FIA_UID.2	User Identification before any action	None
FIA_USB.1	User-subject binding	FIA_ATD.1
Protection of TOE Security Functions		
FPT_STM.1	Reliable time stamps	None

5.1.2.1 Class FDP: User Data Protection

109 FDP_ACC.1 Subset Access Control

Hierarchical to: No other components.

FDP_ACC.1.1 The **host platform** shall enforce the [Discretionary Access Control (DAC) SFP] on

a) [the subjects listed in Table 16 acting on the behalf of users,

b) the named objects in Table 16; and

c) all operations among subjects and objects covered by the DAC SFP].

Table 16: DAC SFP Subjects, Objects, Operations

Subject	Object	Name Object	Operations between Subject/Named Object
NT: processes acting on behalf of a specific user or acting on behalf of the system	File System	Patrol® Directory(ies) Files	Read/Write/Exec/Delete/Change Permissions/Take Ownership
UNIX: processes acting on behalf of a specific user or acting on behalf of the system	File System	Patrol directory – regular File – regular, system, audit, PATROL® database	Read/Write/Exec/Delete/Change Permissions/Take Ownership

Dependencies: FDP_ACF.1 Security attribute based access control

110 FDP_ACF.1 Security Attribute Based Access Control (1)

Hierarchical to: No other components

FDP_ACF.1.1 (1) The **host platform** shall enforce the [DAC SFP] to objects based on **the following**:

a) [The user identity and group membership(s) associated with a subject; and

b) The following access control attributes associated with an object:

- The permission bits;
- Group ownership.

c) The ability to associate allowed or denied operations with one or more user identities;

d) The ability to associate allowed or denied operations with one or more group identities; and

e) Defaults for allowed or denied operations.]

FDP_ACF.1.2 (1) The **host platform** shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

a) [The object's owner is the process's EUID and the owner read/write/exec bit is set.

b) The object's group is one of the process's EGID's and the group read/write/exec bit is set.]

FDP_ACF.1.3 (1) The **host platform** shall explicitly authorize access of subjects to objects based on the following additional rules: [None.]

FDP_ACF.1.4 (1) The **host platform** shall explicitly deny access of subjects to objects based on the **following**: [None.]

5.1.2.2 Class FIA: Identification and Authentication

111 FIA_ATD.1 User Attribute definition

Hierarchical to: No other components.

FIA_ATD.1.1 The **host platform** shall maintain the following list of security attributes belonging to individual users [user-id; group membership(s); real name].

Dependencies: No dependencies

112 FIA_UAU.2 User authentication before any action

Hierarchical to: FIA_UAU.1 Timing of authentication

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

Dependencies: FIA_UID.1 Timing of identification

113 FIA_UID.2 User Identification before any action

Hierarchical to: FIA_UID.1 Timing of identification

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

Dependencies: No dependencies

114 FIA_USB.1 User-subject binding

Hierarchical to: No other components.

FIA_USB.1.1 The **host platform** shall associate the appropriate user security attributes with subjects acting on behalf of that user.

Dependencies: FIA_ATD.1 User attribute definition

5.1.2.3 Class FPT: Protection of the TOE Security Functions

115 FPT_STM.1 Reliable time stamps

Hierarchical to: No other components.

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

Dependencies: No dependencies.

5.1.3 SFRs With SOF Declarations

116 The overall Strength of Function (SOF) claim for the TOE is SOF-basic.

5.1.4 TOE Security Assurance Requirements

117 The security assurance requirements (SARs) for the TOE evaluation are all the SARs, without tailoring through iteration, assignment, selection, or refinement, as identified for the EAL 2 level of assurance from CC Part 3 Security Assurance Requirements. These SARs are identified in Table 17.

Table 17: EAL 2 Assurance Requirements

Assurance Component ID	Assurance Component Name	Dependencies
ACM_CAP.2	Configuration items	None
ADO_DEL.1	Delivery procedures	None
ADO_IGS.1	Installation, generation, and start-up procedures	AGD_ADM.1
ADV_FSP.1	Informal functional specification	ADV_RCR.1
ADV_HLD.1	Descriptive high-level design	ADV_FSP.1, ADV_RCR.1
ADV_RCR.1	Informal correspondence demonstration	None
AGD_ADM.1	Administrator guidance	ADV_FSP.1
AGD_USR.1	User guidance	ADV_FSP.1
ATE_COV.1	Evidence of coverage	ADV_FSP.1, ATE_FUN.1
ATE_FUN.1	Functional testing	None
ATE_IND.2	Independent testing-sample	ADV_FSP.1, AGD_ADM.1, AGD_USR.1, ATE_FUN.1

Assurance Component ID	Assurance Component Name	Dependencies
AVA_SOF.1	Strength of TOE security function evaluation	ADV_FSP.1, ADV_HLD.1
AVA_VLA.1	Developer vulnerability analysis	ADV_FSP.1, ATE_HLD.1 AGD_ADM.1, AGD_USR.1

6 TOE SUMMARY SPECIFICATION

118 This section presents an overview of the security functions implemented by the TOE and the Assurance Measures applied to ensure their correct implementation.

6.1 TOE Security Functions

119 This section presents the security functions performed by the TOE to satisfy the SFRs identified in Section 5.1.1.

6.1.1 Security Audit

120 (FAU_GEN.1.1) The Audit Log feature of the PATROL Agent records security-related aspects of PATROL. The Log records information such as:

- Commands that are executed as a result of Infobox or Menu commands,
- Which console-connect runs commands (listed by console ID),
- Connect/disconnect,
- Commit operations,
- Configuration operations,
- Most spawned commands,

121 The auditing feature is controlled by the configuration variable /AgentSetup/auditLog. The standard PATROL installation process does not create this variable. The PATROL System Administrator must create and set this variable to enable audit logging.

Table 18: Audit Log Entries

Type of Audit Event	Description of Audit Record
Commands Executed	Each command (i.e. script) that is executed as a result of a Menu Command or an InfoBox Command. The entry in the log file records the console-ID of the peer and the local account name used for the connection.
Connect/Disconnect	Details each connection/disconnection. The entry in the log file records the console ID of the peer, the console type, and the local account name used for this connection.
Commit Operations	Each file that is transferred during a commit. The entry in the log file records the name of the file, the console ID of the connection performing the commit, and the local account that is used for the connection.
Configuration Operations	Each explicit pconfig, wpconfig, or xpconfig action that affects the state of the PATROL® Agent. The entry in the log file records the events

Type of Audit Event	Description of Audit Record
	that change variables, kill the agent, and send a license file and PSL pconfig() operations.
Spawned Commands	Each explicit entry in the log file records explicitly created external processes. Note: The agent does not create a log entry for implicitly created commands. This means that the PATROL agent will not log the commands that are created by a process that it creates. Example: Using PSL popen() to create a process, and then sending a command down the channel for this process to execute. The agent logs the creation of the popen() process.

122 Audit logging can be configured with the following KEY=VALUE pairs defined in the */AgentSetup/auditLog* variable:

Table 19: Audit Logging Key Values

Key Value	Description
Active	Determines whether the audit logging feature is turned on or not. The recognized values include: TRUE, FALSE
Delimiter	Determines the delimiter that separates the fields in the log file. The default character is "|".
FileAging	Determines the interval at which new log files are created. Options are Daily, Entries = N, Size = N. N equals a predetermined number as entered by the PATROL® System Administrator.
File Count	Determines how many old log files are retained. The default is 5.
Filename	Determines the pathname and file naming convention for the audit log file.

123 The log file stores data in the following format:

124 `Time|Host|EntryType|User|Entry-specific-data`

125 (FAU_GEN.1.2) Each field is separated by the delimiter character (the default is a pipe, |) specified in a configuration variable.

Table 20: Audit Log File Format

Field	Description
Time	the date and local time in **yyyymmdd:hh:mm:ss** format
Host	the name of the machine on which the agent is running
EntryType	the type of action being recorded • audit

Field	Description
	• execute • connect • disconnect • commit • config • command
User	the name of the local account used to perform the action
Entry- specific-data	offers details on what type of information each entry type provides

126 The Entry Type is determined by the type of action being recorded. The left column lists the action; the right describes the entry.

Table 21. Entry Type Actions

EntryType	Description of Entry Type
audit	Indicates file opened/closed
command	the console ID running the command; if the command originates from the system-output window, it displays the actual command
commit	the console ID and the name of the file being transferred
config	two types of entries • The first indicates where the connection originated. It contains the console ID and the high-level action taking place such as reboot agent. • The second gives a specific action such as store or delete, and lists the variable affected.
connect	the console ID and the connection type
disconnect	the console ID
execute	the command name and its arguments

127 **TOE Functional Requirements Satisfied:** FAU_GEN.1;

128

6.1.2 User Data Protection

129 (FCL_SSL_EXP.1) PATROL enforces the generation of evidence of origin for transmitted TOE component data transfer with the support of a Certificate Authority. The use of SSL permits PATROL to relate the identity of the transmitting PATROL component originating the information to the packet content information. The Certificate Authority provides immediate verification of the evidence to the PATROL component receiving the transmittal.

130 BMC has 4 levels of security for which the product can be configured. For this evaluation the highest security level (Level 4) is claimed. At Level 4, all communicating components must authenticate with each other, and key databases must validate all connection requests. All communications are secure between all Consoles and Agents.

131 BMC is utilizing SSL Version 3.0 and supports PKCS-7 certificate chains and a single X.509 certificate. SSL 3.0 also provides additional generality including support for certificate chains and new ciphers. The Transport-Layer Security (TLS) protocol is an IETF-standardized version of SSL 3.0. The specification is given in RFC2246 and it contains only a few differences with the specification of SSL 3.0. Specifically, the protocol version is number is 3.1, and there are several new alert codes.

132 BMC's PATROL® Security Level 4 provides the following:

- SSL provides private communications and authentication;
- SSL for mutual authentication of agent (server) and console (client);
- Attended agent restart;
- Agent (server) provides certificate so the console can authenticate the Agent;
- Console is required to authenticate to the Agent.

133 A data server (Agent) must provide a certificate to all clients (Consoles) wishing to establish communication. This certificate is kept in an encrypted key database. In order to open this database, the user must provide a password. This password does not reside in plain text, but is generated from an encrypted password string and a key material string. These items are provided in a configuration file (or as registry variables in the case of Windows NT).

134 As a matter of policy the password and key material are not provided, requiring attended operation to provide a key database and the password with which to open it when the agent starts. This configuration of password and key material is independent of the security levels.

135 The SSL connection begins with the client establishing a TCP/IP connection with a server. It sends the server a message identifying itself. The server responds with a "server certificate" and other supplemental information with which the client can verify that the server certificate is genuine. At this point the client can elect to accept the server's certificate.

136 The server must have a key database in order to operate. This database contains at a minimum the server's public-private key pair, the server's certificate, and the certificate of the entity that signed the server's certificate.

137 The exchange of information which results in an SSL connection of the types described above is performed during an SSL "handshake". If the server so desires, it can upon completion of the handshake, send an additional message stating it requests a certificate from the client. Such a message starts a protocol called a "rehandshake". The client presents the server with its certificate. Like the client, the server can elect to verify the client's certificate back to a root authority, or it can accept it without rigorous verification one time only or always. Client side authentication results in the key used to encrypt messages from client to server differing from that used to encrypt messages from server to client. This results in additional privacy. Additionally, if the server verifies the client's certificate back to a root authority, the server can now be certain of the identity of the client. Needless to say, this type of connection is the most difficult to configure since additional information must be present in the key databases of both sides if such a connection is to succeed.

138 (FDP_ACC.2) The PATROL® Agent provides complete access control on communication requests between the Console and Agents and all operations among subjects and objects covered by the access control policy.

139 (FDP_ACF.1) This access control policy is based on the PATROL Agent's credentials that are incorporated within its assigned X.509 certificate, authentication of the TOE component's cryptographically bound signature, and verification of the PATROL Console's authorization to connect as defined in the Agent's Access Control Lists (ACL): These are security controls within the Agent that limit access to the PATROL® Agent. The ACL details are stored in the agent database and can be configured using the standard agent configuration tools such as wpconfig (on NT™), xpconfig (on UNIX™) or pconfig from the command line on either platform. Some aspects include:

- Which types of consoles connect to it;
- Which types of PATROL® Event Manager (PEM) messages the Agent sends to the PATROL® Event Manager (PEM) Console;
- Where Unix applications display their information;
- How the Agent behaves if no consoles connect to it;
- Which users can connect to it.

140 (FCL_SSL_EXP.1) Basic data exchange confidentiality is provided by PATROL® with support from the Certificate Authority. The Certificate Authority uses data encryption mechanisms to avoid the disclosure of sensitive information to a malicious listener on the network, the transfer of such information is encrypted. The encryption is performed at the transport level and is completely unrelated to application level security. Additionally, the transport level protocol can be used to verify the identity of the data server (the Agent) to the client (the Console) and vice versa.

141 (FCL_SSL_EXP.1) Data exchange integrity is provided by PATROL with the support of the Certificate Authority. PATROL® protects user data through the encryption of stored variables in the PATROL® Agent configuration database. This database remains open while the Agent is running. PATROL® provides a utility that permits extraction of parameter information from the database based on its class, instance, and time period. During the SSL handshake it is possible for the client and the server to negotiate which types of encryption, authentication, and message digest algorithms to use during the life of the session. A configuration file defines which encryption algorithms are used by the SSL; supported encryption standards are: SSLv2, SSLv3, or SSLv23.

142 (FCL_SSL_EXP.1) PATROL® through the use of the Certificate Authority utilizes "trusted channels" of communication via the use of SSL protocols and digital certificates obtained from the authorized Certificate Signing Authority. This digital certificate contains the authority's public key that the PATROL System Administrator will need when requesting the certificate. It is also used to authenticate certificates validated by this Certificate Authority.

143 PATROL® supports the use of digital certificates in the X.509 PEM (Privacy-Enhanced Mail) format only. The key database administrator utility uses the X.509 PEM format, an ASCII string format, to import certificates. A translator program must be used for certificate formats other than the X.509 PEM format.

144 Any process that either presents a certificate or verifies one will require a configuration file. The configuration file contains stanzas that contain the information needed by a process operating in a server context or a client context. Each stanza contains at a minimum the path to the key database to be used to verify incoming certificates. Another entry specifies the name or tag of the certificate the process will present to the other party if it is requested to do so. Other entries specify control information with regard to timeout conditions and what level of verification is to be enforced. The protocols to support may be specified with special entries. Additional stanzas specify the path to the log file to be maintained by the security module if one is desired.

145 Any process that either presents a certificate or verifies one will require a key database. This database is built and maintained by a database administrator process. Servers must have a key database and have, at a minimum, a public - private key pair, a digitally signed certificate identifying itself and associated with the key pair, and the certificate of the root authority which signed the server's certificate.

146 **Functional Requirements Satisfied:** FDP_ACC.2; FDP_ACF.1; FCL_SSL_EXP.1

147 (FDP_ACC.1) The IT Environment also provides subset access control through the implementation of discretionary access control (DAC).

148 (FDP_ACF.1 (1)) The DAC access control policy is based on host platform operating system permission bits and subject group membership.

149 **IT Environment Functional Requirements:** FDP_ACC.1; FDP_ACF.1;

6.1.3 Identification and Authentication

150 (FIA_ATD.1) PATROL relies on the host platform to maintain the user-id and group membership security attributes for individual users. PATROL maintains the defined role, user-id, and hostname for individual users.

151 (FCL_SSL_EXP.1) SSL automatically authenticates the peer by:

- Verifying all signatures in the certificate chain.
- Checking that the certificate chain terminates in a trusted root

If both of those checks succeed, the peer is regarded as having passed the built-in authentication checks.

152 (FIA_UAU.3) The use of SSL provides PATROL support in preventing the use of forged or copied authentication data by any Agent or Console.

153 (FIA_UID.2) and (FIA_USB.1) PATROL uses the DEFAULT ACCOUNT for identification before any allowing any TSF-mediated action on behalf of the user: The default account is used by the PATROL® Agent for executing commands at each server. It is specified by the default Account variable in the agent configuration file. The Agent cannot run application discovery and parameters properly without a valid (local) user name.

154 (FIA_UAU.2) and (FIA_UID.2) The host platform requires each user to be identified and authenticated before allowing any TSF-mediated actions on behalf of the user.

155 **Functional Requirements Satisfied:** FIA_ATD.1; FIA_UAU.3, FIA_UID.2; FIA_USB.1,FCL_SSL_EXP.1

156 **IT Environment Functional Requirements:** FIA_UAU.2; FIA_UID.2;

6.1.4 Security Management

157 PATROL® supports the definition of roles, as well as, providing a number of functions to manage the various security policies and features provided by the TOE.

158 (FMT_SMR.1) With respect to PATROL® User Roles:

a) The PATROL® System Administrator can edit the PATROL® user roles file to protect the enterprise from unauthorized use of PATROL® Console operations.

b) The User Role Administrator can grant or remove the ability of specific users to perform specific console operations. For example, they may need to restrict certain PATROL® users from overriding agent parameter attributes while permitting certain other trusted operators to perform the same operation. The tasks controlled through user roles include:

- committing PATROL® KM changes to a PATROL® Agent;
- issuing operating system commands at the PATROL® system output window;
- modifying the PATROL® Agent's parameter attributes;
- launching a PATROL® Console in developer mode.

c) The Console User can perform console operations. This includes starting a PATROL® console and monitoring system activity.

159 The PATROL® user roles file allows the specification of the conditions under which the PATROL® Administrator permits or disables the console operations listed above.

160 (FMT_MOF.1 (1) & (2)) PATROL maintains access control through ACLs. It is through the ACL that the PATROL System Administrator defines which users are authorized to connect to an agent; in which modes the user can connect; and from which hosts the user can connect.

161 ACCESS CONTROL LISTS (ACL): The ACL details are stored in the agent database and can be configured using the standard agent configuration tools such as wpconfig (on NT™), xpconfig (on UNIX™) or pconfig from the command line on either platform.

Through an ACL, the Agent allows the following to be defined:

- Who has access to the Agent;
- What hosts have access to the Agent;
- What type of PATROL® Consoles and utilities have access to the Agent;
- And, any combination of these three types of control.

162 **UserName and HostName Attribute Conventions**: In an ACL entry, the PATROL System Administrator can use a number of masking techniques for the host name and user name attributes.

163 **UserName:** The name of a local account that the connecting console may request to use. Valid values include:

- *—any username (assuming the account exists)
- *username*—an actual name of an account

164 **HostName**: A machine (Console) that is authorized to connect to this Agent. The PATROL System Administrator can specify a hostname by using the fully qualified name, the short name, or a partial name (pattern) created with a wildcard specification in which the first character is a '*', with other characters following.

- *—any host name (assuming the host exists)
- ***hostname***—an IP Address or actual name of the host indicating that this entry is for that host only
- ***partial_hostnam**e—a wildcard specification, in which the first character is an asterisk followed by other characters

If the HostName value is not provided for an ACL entry, it defaults to '*'.

165 (FMT_MSA.1) (FMT_MSA.3), & (FMT_MTD.1 (1), (2)). The following table defines the format and type of data provided in an ACL:

Table 22. ACL Format and Type of Data

Format and Type of Data	For each access control list (ACL), the format is a comma-separated list of entries. Each entry has the following format: `UserName/HostName/Mode` **UserName**-the name of a local account that the connecting console may request to use. It defaults to *. **HostName**-a machine (console) that is authorized to connect to this agent. It defaults to *. **Mode**-a list of application and application modes that are authorized to access the agent. **C**-Configure (pconfig, wpconfig, xpconfig) **D**-Developer (console) **O**-Operator (console) **P**-PEM (event manager console) If the Mode value is missing from an individual ACL entry, it defaults to O.
Default Value	*/*/CDOP
Minimum and Maximum	Not applicable
Dependencies	none
Recommendation	Not applicable

166 **Connection Modes and Accounts:** Table 23 below describes how the various Consoles and utilities connect to the Agent and what type of account each uses. The accounts to connect to the Agent include:

- connection account—account used to connect the PATROL® Console to the Agent.
- default account—account used by the PATROL® Agent for executing monitoring commands, such as parameters and recovery actions.
- system log-on account—account used to log on to the operating system and used to access the PATROL® Console.

Table 23: Client/Agent Connections

Client	Account Used
developer console	connection account
operator console	connection account
pconfig, xpconfig, wpconfig	When started from the command line, these utilities use the system log-on account. When started from within a developer console, these utilities use the system log-on account.
pconfig()	When this function is run by a parameter, recovery action, or application discovery, it uses the default account. When this function is run by a Menu command or an Infobox command, it uses the connection account.
PATROL Event Manager (PEM)	system log-on account. User-coded client that uses PATROL® (API) Application Program Interface.

167 CERTIFICATE REVOCATION LISTS (CRL): The CRL is stored in SSL Key Database. To prevent the use of invalid certificates, CAs main lists of invalid certificates called certificate revocation lists (CRLs). Because SSL accepts certificates as identification, it must be able to verify that a presented certificate has not been revoked. Therefore, the Patrol system administrator must obtain a new CRL from the CA and install it in the SSL key database using the sslcmd function following an CRL installation schedule determined by organization physical security policy.

168 **Functional Requirements Satisfied:** FMT_MOF.1.1 (1); FMT_MOF.1.1 (2); FMT_MSA.1; FMT_MSA.3; FMT_MTD.1 (1); FMT_MTD.1 (2); FMT_SMR.1

6.1.5 Protection of TOE Security Functions

169 (FCL_SSL_EXP.1) PATROL® with support from the Certificate Authority provides replay detection and basic internal TSF data transfer protection. The Certificate Authority protects against unauthorized configuration data disclosure and modification by using a suite of standard protocols including Secure Socket Layer (SSL) and Data Encryption Standard (DES) encryption.

170 The SSL protocol is designed to provide privacy between two communicating applications. The protocol is designed to authenticate the server and the client (at Level 4). SSL requires a reliable transport protocol (e.g., TCP) for data transmission and reception.

171 **Functional Requirements Satisfied:** FCL_SSL_EXP.1

172 (FPT_STM) PATROL relies on the host platform to provide reliable time stamps in the audit security functionality.

173 **IT Environment Requirements:** FPT_STM.1

6.2 Assurance Measures

174 PATROL satisfies the CC EAL 2 assurance requirements. BMC Software has assurance measures for PATROL to satisfy the stated SARs. This section identifies the Configuration Management, System Delivery Procedures, System Development Procedures, Guidance Documents, Life Cycle Support, Testing, and Vulnerability Analysis measures applied by BMC Software to satisfy the CC EAL 2 assurance requirements.

6.2.1 Configuration Management

175 The configuration management measures applied by BMC Software include providing a reference for the TOE, using a CM system, and providing CM documentation.

176 The CM system uniquely identifies all configuration items (CIs) and provides the measures that are used to maintain and ensure that only authorized changes are made to the configuration items. The CM documentation shows that the CM system, at a minimum, tracks the following: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, evidence that demonstrates that the CM system is operating in accordance with the CM plan, and CM documentation. The CM documentation also describes how configuration items are tracked by the CM system.

177 The configuration management measures are documented within the following BMC Software documents:

- BMC Software PATROL® Version 3.4.11, Software Configuration Management Document
- BMC Software PATROL® Version 3.4.11, Configuration Management: CI List

178 Assurance Requirements Satisfied: ACM_CAP.2

6.2.2 Delivery and Operation

179 BMC Software provides delivery and operation documentation that describes what components are delivered with PATROL®, guidance for initially installing it, and warnings about the importance of properly unpacking, installing, and configuring the TOE. The installation and start-up document provides a set of procedures for initially installing and configuring the TOE into the evaluated configuration. These delivery and operation measures are documented within the following BMC Software documents:

- BMC Software PATROL® Classic, PATROL® Enterprise Manager and PATROL® Perform/Predict Product Packaging and Delivery Procedures
- BMC Software PATROL® Version 3.4.11 Security Target under NIAP Common Criteria EAL2 Installation Instructions

180 Assurance Requirements Satisfied: ADO_DEL.1 and ADO_IGS.1

6.2.3 Development

181 The development documents provided by BMC Software satisfy the CC functional specification and high-level design development requirements, as well as provide a correspondence between that information and this ST. These architecture measures are documented within the following BMC Software documents:

- BMC Software PATROL® Version 3.4.11, Security Functional Specification (FSP)
- BMC Software PATROL® Version 3.4.11, High-Level Design
- BMC Software PATROL® Version 3.4.11, Informal Correspondence Documentation

182 **Assurance Requirements Satisfied**: ADV_FSP.1, ADV_HLD.1, and ADV_RCR.1.

6.2.4 Guidance

183 The Guidance assurance measures provided by BMC Software include system administrative and user guidance documents.

184 The system administrative guidance contains the following administrative functions and interfaces:

- Warnings about functions and privileges that should be controlled in a secure processing environment,
- All assumptions regarding user behavior that are relevant to secure operation of the TOE,
- All security parameters under the control of the administrator,
- Indicates secure values as appropriate,
- Descriptions of 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, which is consistent with all other documentation supplied for evaluation,
- Describes all security requirements for the IT Environment that are relevant to the administrator.

185 The user guidance is consistent with other evaluation documents and contains the following:

- All security requirements for the IT Environment that are relevant to the user functions and interfaces available to the non-administrative user of the TOE,
- The use of user-accessible security functions provided by the TOE,
- Warnings about user-accessible functions and privileges that should be controlled in a secure processing environment,
- All user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behavior found in the statement of TOE, security environment;

186 These guidance measures are documented within the following BMC Software documents:

- BMC Software PATROL® Version 3.4.11, Security Target Admin and User Guide
- PATROL Security Technical Bulletin mmddyy

187 Assurance Requirements Satisfied: AGD_ADM.1 and AGD_USR.1.

6.2.5 Test

188 The test assurance provided by BMC Software includes documentation that provides an analysis of the test coverage, an analysis of the depth of testing, and TSF test documentation.

189 The analysis of the test coverage demonstrates correspondence between the tests identified in the test documentation and the TSF as described in the functional specification, and demonstrates that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.

190 The analysis of the depth of testing demonstrates that the tests identified in the test documentation are sufficient to demonstrate that the TSF operates in accordance with its high-level design and confirms that the information provided meets all requirements for content and presentation of evidence.

191 The TSF test documentation consists of test plans, test procedure descriptions, expected test results and actual test results. The test plans identify the security functions to be tested and describe the goal of the tests to be performed. The test procedure descriptions identify the tests to be performed and describe the scenarios for testing each security function. These scenarios include any ordering dependencies on the results of other tests.

192 The expected test results show the anticipated outputs from a successful execution of the test. The test results from the developer execution of the tests demonstrate that each tested security function behaved as expected.

193 The developer will provide the TOE suitable for and an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

194 These tests measures are documented in the following BMC Software documents:

- BMC Software PATROL® Version 3.4.11, Analysis of Coverage
- BMC Software PATROL® Version 3.4.11, Security Target Test Coverage Document

195 **Assurance Requirements Satisfied**: ATE_COV.1, ATE_FUN.1, and ATE_IND.2.

6.2.6 Vulnerability Assessment

196 The vulnerability assessment assurance measures provided by BMC Software include guidance documentation; a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim; and documentation of an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP, and disposition of obvious vulnerabilities.

197 The guidance documents identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation, are complete, clear, consistent and reasonable, list all assumptions about the

intended environment, and list all requirements for external security measures (including external procedural, physical and personnel controls).

198 The strength of TOE security function analysis show for each mechanism identified in the ST as having a strength of TOE security function claim that it meets or exceeds the minimum strength level and metric defined in the ST.

199 The vulnerability analysis shows that the developer performed a search analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP and the disposition of obvious vulnerabilities.

200 These measure are documented within the following BMC Software documents:

- BMC Software PATROL® Version 3.4.11, Strength of Function Analysis
- BMC Software PATROL® Version 3.4.11, Independent Vulnerability Analysis

201 Assurance Requirements Satisfied: AVA_SOF.1, and AVA_VLA.1.

7 PP CLAIMS

202 The BMC Software PATROL® Version 3.4.11 Security Target was not written to comply with any Protection Profile.

8 RATIONALE

203 This section shows that all threats and organizational security policies are completely covered by security objectives.

8.1 TOE Security Objectives Rationale

204 This section shows that all threats and organizational security policies are completely covered by security objectives.

8.1.1 Rationale for Security Objectives

205 Table 24 demonstrates that each TOE security objective counters, or addresses, at least one organizational security policy, or threat.

Table 24: Security Objectives Rationale Mapping

IT Security Objectives	Threats and Organizational Policies
O.ADMIN	T.TROJAN T_C.UNAUTH_DEPLOY T_C.UNAUTH_CHANGES T_C.UNAUTH_COMMANDS T_R.ELEVATE_ACCESS T_R.KM_TAMPER P.AUTHORIZATION P.INFO_ACCESS P.MANAGE
O.AUDIT	T.UNDETECTED_ACTIONS P.ACCOUNTABLE P.TRACE
O.CONNECT	P.INFO_ACCESS
O.ENTITY_IDENTIFICATION	P.ACCOUNTABLE P.AUTHORIZATION T.REPLAY T.TRAFFIC_SPOOF
O.CONFIDENTIALITY	P.INFO_ACCESS
O.INTEGRITY	P.INFO_ACCESS P.INTEGRITY.
O.SEPARATE_ROLES	T.UNAUTH_ACCESS_DATA T_C.UNAUTH_COMMANDS P.AUTHORIZATION

206 The following objectives are sufficient to address the named threats and to help implement the named organizational policies as described in Section 3 of the ST.

207 O.ADMIN – This security objective, by requiring the TOE provide the functions necessary to adequately administer the system, helps implement the following OSPs:

- P.AUTHORIZATION because an administrator can limit a user's authorization

- P.INFO_ACCESS because an administrator can limit a user's access to information, and
- P.MANAGE because the administrator can manage and maintain the system security functions.

208 The O.ADMIN security objective also helps to counter the threats:

- T.TROJAN because an administrator can limit a user's authorization;
- T_C.UNAUTH_DEPLOY because an administrator can limit a user's authorization;
- T_C.UNAUTH_CHANGES because an administrator can limit a user's authorization;
- T_C.UNAUTH_COMMANDS because an administrator can limit a user's authorization;
- T_R.ELEVATE_ACCESS because an administrator can limit a user's access to information; and
- T_R.KM_TAMPER because an administrator can limit a user's access to information.

209 O.AUDIT – This security objective, by requiring the TOE provide audit capability to report security relevant events to provide user accountability helps implement the OSPs:

- P.ACCOUNTABLE and P.TRACE because the objective requires that security relevant information be recorded for review by the TOE administrator(s).

210 The O.Audit security objective also helps to counter the threats:

- T.UNDETECTED_ACTIONS because the objective requires that security relevant information be recorded for review by the TOE administrator(s).

211 O.CONFIDENTIALITY – This security objective is necessary to implement the P.INFO_ACCESS organizational policy. The policy is covered/implemented because the objective requires that the content of the information transferred between components of the TOE be protected.

212 O.CONNECT – This security objective by stipulating the only permitted connectivity between Consoles and Agents is determined by the PATROL System Administration implement the OSP:

- P.INFO_ACCESS because the TOE administrator defines the permitted connectivity.

213 O.ENTITY_IDENTIFICATION – This security objective, by requiring identification of entities before permitting connectivity to components and access to data helps implement the OSPs and counter the threats:

- P.ACCOUNTABLE because an entitiy is associated with the action, and
- P.AUTHORIZATION because it verifies the entity's authority to connect to the component and/or access the data.
- T.REPLAY and T.TRAFFIC_SPOOF because it requires the identification of entities to verify that permission for connection/access to TOE components, or data is authorized.

214 O.INTEGRITY – This security objective is necessary to implement the policies P.INFO_ACCESS and P.INTEGRITY. The policies covered/implemented because the objective requires that integrity protection is applied to all information it releases between components and to ensure that these protections are applied.

215 O.SEPARATE_ROLES – This security objective, by requiring the specification of administrator roles, helps implement the OSP:

- P.AUTHORIZATION because user access can be restricted based on role.

216 O.SEPARATE_ROLES is also countering the threats:

- T.UNAUTH_ACCESS_DATA because user access can be restricted based on role, and
- T_C.UNAUTH_COMMANDS, because user authority can be restricted based on role.

8.1.2 Rationale for IT Environment Security Objectives

217 This section shows that all threats and assumptions, associated with the IT Environment, are completely covered by security objectives for the IT Environment. In addition, Table 25 demonstrates that each IT Environment Security objective counters, or addresses, at least one threat, or assumption.

Table 25: Security Objectives for the IT Environment Rationale Mapping

IT Environment Security Objectives	Threats and Assumptions
OE.CERTIFICATE_SUPPORT	TE.UNAUTH_ACCESS_NETWORK TE.UNAUTH_DATA_MOD_NETWORK T_R.APPLICATION_SECRETS P.INFO_ACCESS A.CAOPERATE_CORRECT A_ECERTIFICATE_ AUTHORITY
OE.DISCRETIONARY_ACCESS	TE.UNAUTH_ACCESS TE.UNAUTH_USAGE A.ACCESS_CONTROL A.SAME_ADMIN A_C.AUTHORIZED
OE.INSTALL	A.MANAGE A. NO_EVIL A.PEER A_C.AUTHORIZED A_R.BENIGN
OE.PHYSICAL_PROTECTION	A.PHYSICAL_PROTECT A_R.BENIGN
OE.PLATFORM_SUPPORT	A.OPERATE_CORRECT

218 The following IT Environment objectives are sufficient to address the named threats and to help implement the named organizational policies and meet the named assumptions as described in Section 3 of the ST.

219 OE.CERTIFICATE_SUPPORT This objective is sufficient to address the environmental threats: TE.UNAUTH_ACCESS_NETWORK, TE.UNAUTH_DATA_MOD_NETWORK, T_R.APPLICATION_SECRETS; the OSP: P.INFO_ACCESS, and the assumptions: A_ECERTIFICATE_ AUTHORITY and A.CAOPERATE_CORRECT because it requires the IT environment provide reliable Certificate Authority functions.

220 OE.DISCRETIONARY_ACCESS – This objective is sufficient to address the threats TE.UNAUTH_ACCESS, TE.UNAUTH_USAGE, and the assumptions A.ACCESS_CONTROL, A.SAME_ADMIN and A_C.AUTHORIZED because it ensures that the host platform discretionary access control (DAC) mechanism will protect TOE data and operation.

221 OE.INSTALL – This objective is sufficient to address A.MANAGE, A.NO_EVIL, A.PEER, A_C.AUTHORIZED, and A_R.BENIGN because it ensures that the TOE is delivered, installed, managed, and operated in a secure manner by non-hostile individuals.

222 OE.PHYSICAL_PROTECTION – This objective is sufficient to address A.PHYSICAL_PROTECT and A_R.BENIGN because it ensures that the critical parts of the TOE are protected from physical attack.

223 OE.PLATFORM_SUPPORT – This objective is sufficient to address A.OPERATE_CORRECT because it ensures that the underlying hardware and software operate correctly, and that reliable system time is available to the TOE.

8.2 Security Functional Requirements Rationale

224 The security requirements rationale section is provided to demonstrate that the set of security requirements is suitable to meet and traceable to the security objectives.

8.2.1 Traceability and Suitability

225 The following Table 26 provides the correspondence mapping between security objectives for the TOE and the requirements to satisfy them.

Table 26: TOE Requirements to Security Objectives Mapping

Requirement	O.ADMIN	O.AUDIT	O.CONNECT	O.CONFIDENTIALITY	O.INTEGRITY	O.ENTITY_IDENTIFICATION	O.SEPARATE_ROLES
FAU_GEN.1		X					
FDP_ACC.2			X			X	
FDP_ACF.1			X			X	
FIA_ATD.1						X	
FIA_UAU.3						X	
FIA_UID.2						X	
FIA_USB.1						X	
FMT_MOF.1 (1)	X						X
FMT_MOF.1 (2)	X						X
FMT_MSA.1	X						X
FMT_MSA.3	X						X

Requirement	O.ADMIN	O.AUDIT	O.CONNECT	O.CONFIDENTIALITY	O.INTEGRITY	O.ENTITY_IDENTIFICATION	O.SEPARATE_ROLES
FMT_MTD.1 (1)	X						X
FMT_MTD.1 (2)	X						X
FMT_SMR.1							X
FCL_SSL_EXP.1			X	X	X	X	

226 The suitability of the TOE security functional requirements to meet the named objectives is described below:

227 O.ADMIN –, FMT_MOF.1.1 (1), FMT_MOF.1.1 (2), FMT_MSA.1, FMT_MSA.3; FMT_MTD.1 (1), FMT_MTD.1 (2) require the TOE to provide management functionality to administer the TSF security services.

228 O.AUDIT - FAU_GEN.1 require the TOE to generate audit events and provide management support for these functions.

229 O.CONNECT –FDP_ACC.2, FDP_ACF.1, FCL_SSL_EXP.1 require that the TOE only allow connectivity between Consoles and Agents.

230 O.CONFIDENTIALITY - FCL_SSL_EXP.1 requires that the TOE provide SSL standard protocol protection to ensure confidentiality.

231 O.INTEGRITY - FCL_SSL_EXP.1 requires that the TOE provide SSL standard protocol protection to ensure integrity.

232 O.SEPARATE_ROLES –FMT_ MOF.1.1 (1), FMT_MOF.1.1 (2), FMT_MSA.1, FMT_MSA.3; FMT_MTD.1 (1), FMT_MTD.1 (2) require that the TOE provide the capability to limit the extent of access control and user authorizations by the definition of roles, the user privileges, and security relevant authorizations and attributes.

233 O.ENTITY_IDENTIFICATION –FDP_ACC.2, FDP_ACF.1, FIA_ATD.1, FIA_UAU.3, FIA_UID.2, FIA_USB.1, require that the TOE identify all entities prior to interaction with that entity.

234 Table 27 provides the mapping between security objectives for the IT Environment and the requirements to satisfy them.

Table 27: IT Environment Security Functional Requirements to Security Objectives Mapping

Requirement	OE.CERTIFICATE_SUPPORT	OE.DISCRETIONARY_ACCESS	OE.INSTALL	OE.PHYSICAL_PROTECTION	OE.PLATFORM_SUPPORT
FDP_ACC.1		X	X		
FDP_ACF.1 (1)		X	X		
FIA_ATD.1					
FIA_UAU.2	X				
FIA_UID.2	X				
FIA_USB.1	X				
FPT_STM.1					X
Assumption					
A.PHYSICAL_PROTECT				X	

235 The suitability of the IT security functional requirements to meet the named objectives is described below:

236 **NOTE:** Those objectives above that are not mapped to SFRs for the IT Environment, are mapped to assumptions.

237 FPT_STM.1 requires that the IT Environment provide reliable time stamp.

238 OE.CERTIFICATE_SUPPORT – FIA_UAU.2 requires the IT Environment authenticate entities/users before allowing any TSF-related action; FIA_UID.2 requires the IT Environment identify entities/users before allowing any TSF-related action, and FIA_USB.1 requires the IT Environment be able to associate user attributes with subjects acting on behalf of the user.

239 OE.DISCRETIONARY_ACCESS – FDP_ACC.1 and FDP_ACF.1 (1) require that the IT Environment enforce the discretionary access control policy (DAC).

240 OE.INSTALL – FDP_ACC.1 and FDP_ACF.1 requires that the IT Environment provide subset access control and security based access control; those responsible for the TOE must ensure that the TOE is delivered, installed, managed, and operated in a manner that maintains IT security objectives – environmental access control components must be installed such that IT security objectives are maintained.

241 OE.PLATFORM_SUPPORT – FPT_STM.1 requires that the IT Environment provide reliable time stamps.

8.2.2 Rationale For Explicitly Stated Requirements

242 The explicitly stated requirement FCL_SSL_EXP.1 address the need for SSL standard protocol in the TOE for determining host to host access for identification and authentication and user data protection between TOE components.

8.2.3 Rationale For Assurance Requirements

243 The chosen assurance requirements identified in this ST are drawn from the CC EAL 2 assurance package. This ST has been developed for a generalized environment where there is a low level of risk to the assets. The Security Objectives were reviewed and EAL 2 was found sufficient to address them through the developer testing, vulnerability analysis, and the required independent testing.

8.2.4 Requirement Dependency Rationale

244 Table 28 illustrates that all of the functional requirement dependencies have been satisfied with the exception of FDP_MSA.3 for the host platform. The assumption A.SAME_ADMIN assume the administration of the host platform is the same as that of the PATROL Agent and Console. Satisfaction of the FDP_MSA.3 SFR by the TOE represents satisfaction of the SFR by the host platform as well.

Table 28: Security Functional Requirement Dependency Mapping

SFR Specified in the ST	Dependencies	Reference Number of Dependency
FAU_GEN.1	FPT_STM.1	Satisfied
FDP_ACC.2	FDP_ACF.1	Satisfied
FDP_ACF.1	FDP_ACC.1 FDP_MSA.3	Satisfied (FDP_ACC.2 is hierarchical to FDP_ACC.1) Satisfied
FDP_ACC.1 (host)	FDP-ACF.1	Satisfied
FDP_ACF.1 (host)	FDP_ACC.1 FDP_MSA.3	Satisfied No (See rationale above)
FIA_ATD.1	None	
FIA_UAU.2	FIA_UID.1	Satisfied (FIA_UID.2 is hierarchical to FIA_UID.1)
FIA_UAU.3	None	
FIA_UID.2	None	
FIA_USB.1	FIA_ATD.1	Satisfied
FMT_MOF.1.1 (1-2)	FMT_SMR.1	Satisfied
FMT_MSA.1 (1-2)	[FDP_ACC.1 or FDP_IFC.1], FMT_SMR.1	(FDP_ACC.2 is hierarchical to FDP_ACC.1) Satisfied
FMT_MSA.3	FMT_MSA.1 FMT_SMR.1	Satisfied Satisfied
FMT_MTD.1 (1-2)	FMT_SMR.1	Satisfied

SFR Specified in the ST	Dependencies	Reference Number of Dependency
FMT_SMR.1	FIA_UID.1	Satisfied (FIA_UID.2 is hierarchical to FIA_UID.1)

8.2.5 Mutually Supportive

245 The set of security requirements provided in this ST form a mutually supportive and internally consistent whole as evidenced by the following:

246 The choice of security requirements is justified as shown in Sections 8.2.1 and 8.2.2. The choice of SFR and SARs were made based on the assumptions about, the objectives for, and the threats to the TOE and the security environment. This ST provides evidence that the security objectives counter threats to the TOE (Table 24).

247 The SOF claim is valid with the threat environment described in Section 3. The rationale for the chosen level of SOF-basic is based on the minimum attack potential of the threat agents identified in this Security Target. The SOF claim is commensurate with the EAL 2 level of assurance.

248 The SARs are appropriate for the assurance level of EAL 2 and are satisfied as shown in Section 6.2.

249 The statement of requirements is written using consistent language and do not contain internal contradictions in presenting the security functionality of the TOE.

8.2.6 Rationale for Strength of Function

250 The rationale for the chosen level of SOF-basic is based on the minimum attack potential of the threat agents identified in this security target. The CC associates a SOF-Basic as being resistant to threats possessing low attack potential. Additionally, the level of SOF-basic is valid for the TOE Security Functions and Assurance Measures because they support the SFRs and SARs as demonstrated in Sections 8.3.1 and 8.3.2.

8.3 Rationale for TOE Summary Specification

251 This section in conjunction with Section 6 demonstrates that the TOE security functions and assurance measures are suitable to meet the TOE security requirements.

8.3.1 TOE Security Functions Satisfy Security Functional Requirements

252 The specified TOE security functions work together so as to satisfy the TOE security functional requirements. Section 6 includes in the descriptions of security functions a mapping of the security functional requirements to show that each security function is traced to at least one SFR. Table 29 demonstrates that each SFR is covered by at least one security function. The security functions and assurance measures described in the TOE Summary Specification and indicated below are all necessary for the required security functionality claimed for the TOE.

Table 29: Correspondence of SFRs to TSFs

Requirement	Security Audit	User Data Protection	Identification and Authentication	Security Management	Protection of Security Functions
FAU_GEN.1	X				
FDP_ACC.2		X			
FDP_ACF.1		X			
FIA_ATD.1			X		
FIA_UID.2			X		
FIA_USB.1			X		
FMT_MOF.1 (1-2)				X	
FMT_MSA.1				X	
FMT_MSA.3				X	
FMT_MTD.1 (1)				X	
FMT_MTD.1 (2)				X	
FMT_SMR.1				X	
FCL_SSL_EXP.1		X			
SFRs Allocated to the IT Environment					
FDP_ACC.1		X			
FDP_ACF.1		X			
FIA_ATD.1			X		
FIA_UAU.2			X		
FIA_UAU.3			X		
FIA_UID.2			X		
FIA_USB.1			X		
FPT_STM.1					X

253 The SFR FAU_GEN.1, Audit data generation, is implemented under the Security Audit security function. Audit records will be generated for startup and shutdown of audit functions, and the list of events specified in Table 14.

254 The SFR FDP_ACC.2, Complete access control, is implemented under the User Data Protection security function. The access control SFP will be enforced based on the ACL.

255 The SFR FDP_ACF.1, Security attribute based access control, is implemented under the User Data Protection security function. The access control SFP will be based on the Console's authorization to connect based on the Agent's access control list.

256 The SFR FIA_ATD.1, User attribute definition, is implemented under the Identification and Authentication security function. The defined role, user-id, and hostname(s) attributes belonging to users will be maintained by PATROL. The host platform will maintain the user-id, group membership, and user name.

257 The SFR FIA_UAU.2, User authentication before any action, is implemented under the Identification and Authentication security function. Each user must be successfully authenticated before allowing any actions on behalf of that user by SSL functionality and the Certificate Authority's including credentials associated with X.509 certificates, signature verification, and valid key exchange.

258 The SFR FIA_UID.2, User identification before any action, is implemented under the Identification and Authentication security functions. Each user will be required to identify itself via the host platform, the verification by SSL and the Certificate Authority, and through PATROL's ACL before any actions on behalf of that user are allowed.

259 The SFR FIA_USB.1, User-subject binding, is implemented under Identification and Authentication security function. The appropriate security attributes, as maintained in PATROL's ACL, the SSL functionality and Certificate Authority's including credentials associated with X.509 certificates will associated with each user.

260 The SFR FMT_MOF.1 (1), Management of security functions behavior, is implemented under the Security Management security function. Only the PATROL® System Administrator is able to disable, enable, and/or modify the behavior of: security functions (e.g., audit), Agent configuration files and databases, and Knowledge modules.

261 The SFR FMT_MOF.1 (2), Management of security functions behavior, is implemented under the Security Management security function. Only the User Role Administrator is able to grant or remove permissions of users to perform specific console operations.

262 The SFR FMT_MSA.1 Management of security attributes, is implemented under the Security Management security function. Only the PATROL® System Administrator is able to query, modify, or delete rule sets.

263 The SFR FMT_MSA.3, Static attribute initialization, is implemented under the Security Management security functions. The use of restrictive default values for the enforcement of DAC SFP security values will be enforced.

264 The SFR FMT_MTD.1 (1), Management of TSF data, is implemented under the Security Management security function. The ability to set the time and date used for audit trail time stamps, management of user identities, and the management of access control lists will be restricted to the PATROL® System Administrator. Additionally, the user in the role of User Role Administrator will be able to manage user identities.

265 The SFR FMT_MTD.1 (2), Management of TSF data, is implemented under the Security Management security function. The ability to manage the audit trail will be restricted to the PATROL® System Administrator.

266 The SFR FMT_SMR.1, Security Roles, is implemented under the Security Management security function. The following roles associated with human users are defined for the system: PATROL® System Administrator and User Role Administrator.

267 The SRE FCL_SSL_EXP.1, Basic data exchange confidentiality, is implemented under the User Data Protection security functions. PATROL via the SSL functionality will enforce the ability to transmit objects in a manner that will protect it from unauthorized disclosure. The Data exchange integrity, is implemented under the User Data Protection security function. PATROL via the SSL functionality will enforce the ability to transmit user data in a manner that will protect it from modification, deletion, insertion, and replay, and will also provide the ability to detect the aforementioned. The SSL functionality also provides unforgeable authentication. The use of authentication data that has been forged or copied by any Console or Agent will be prevented by the SSL and the Certificate Authority's signature verification and valid key exchange. Basic internal TSF data transfer protection is implemented via the SSL functionality. Data transmitted between the TOE components will be protected from disclosure and modification. Replay protection is implemented via the SSL functionality and the Certificate Authority. Replays of connection requests, service requests, and service responses will be detected. Through the Certificate Authority's use of SSL, a communication channel between components will be provided that is logically distinct , provides assured identification of its endpoints, and provides protection of data from modification or disclosure.

8.3.2 Assurance Measures Comply with Assurance Requirements

268 Section 6.2 of this document identifies the Assurance Measures implemented by BMC to satisfy the assurance requirements of EAL2 as delineated in the table in Annex B of the CC, Part 3. Table 30 maps the Assurance Requirements with the Assurance Measures as stated in Section 5.2.

Table 30: Assurance Compliance Matrix

Assurance Measure	Configuration Management	Delivery and Operation	Development	Guidance	Test	Vulnerability Assessment
ACM_CAP.2	X					
ADO_DEL.1		X				
ADO_IGS.1		X				

Assurance Measure	Configuration Management	Delivery and Operation	Development	Guidance	Test	Vulnerability Assessment
ADV_FSP.1			X			
ADV_HLD.1			X			
ADV_RCR.1			X			
AGD_ADM.1				X		
AGD_USR.1				X		
ATE_COV.1					X	
ATE_FUN.1					X	
ATE_IND.2					X	
AVA_SOF.1						X
AVA_VLA.1						X

269 ACM: Configuration Management

270 BMC documentation verifies that BMC has implemented a CM Plan that uniquely identifies each version of the TOE. BMC also maintains a configuration list of each TOE version that describes the configuration items that comprise the TOE and the method used to uniquely identify them.

271 ADO: Delivery and Operation

272 BMC satisfies the Delivery and Operation (ADO) assurance requirements because BMC personnel are responsible for Patrol® Classic from development through delivery. Documentation that system administrator personnel reference is listed below and found to be sufficient to ensure that the installation, generation, and start-up procedures will result in a secure configuration. The BMC Software Patrol Classic product security patch is delivered to the BMC System Engineer/Representative and not to the customer. The BMC System Engineer/Representative installs the security patch using the installation instructions listed below.

- BMC Software PATROL® Classic, PATROL® Enterprise Manager and PATROL® Perform/Predict Product Packaging and Delivery Procedures
- BMC Software PATROL® Version 3.4.11 Security Target under NIAP Common Criteria EAL2 Installation Instructions
- PATROL Security Technical Bulletin mmddyy

273 ADV: Development

274 The FSP identifies the TSF and its externally visible interfaces, and provides details of the effects, error messages and exceptions of each interface.

275 The BMC provided HLD describes the TSF in terms of subsystems. The HLD describes the security functionality of each subsystem, their interfaces, and which of those interfaces are externally visible. It identifies the 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.

276 The BMC provided RCR document provides a table showing the relationship between the Defined Security Function, SFRs, FSP Security Function, FSP Interfaces and the Rationale between the ST and FSP and the FSP and HLD. It also includes the relationships to the subsystems described in the HLD.

277 AGD: Guidance Documents

278 BMC provides a series of guidance manuals that contain the information needed to satisfy the Guidance Document assurance requirements. These manuals describe the administrative security functions and how to implement them in a secure manner. The PATROL Security Technical Bulletin also provides guidance for the proper secure operation of the software.

279 ATE: Tests

280 BMC documentation contains satisfactory evidence that the TSF as described was successfully tested. The evaluator will also conduct further testing as well as reproduce the developer's test to ensure that the TSF operates as described.

281 AVA: Vulnerability Assessment

282 Section 8.3.3 discusses strength of function of the TOE as SOF-basic. BMC has developed a Vulnerability Analysis document that addresses obvious weaknesses that could be exploited by an attacker attempting to violate the TSP.

8.3.3 TOE SOF Claims Rationale

283 The overall TOE SOF claim is SOF-basic because this SOF is sufficient to resist the threats identified in Section 3.2. Section 8.1 provides evidence that demonstrates that TOE threats are countered by the TOE security objectives. Sections 8.2.1 and 8.2.2 demonstrate that the security objectives for the TOE and the IT Environment are satisfied by the security requirements. The SOF-basic claim for the TOE applies because the TOE protects against an attacker of limited ability with no special tools from accessing the TOE.