Bit9, Inc.
Parity™ 6.0.1
Security Target

Version 2.0
February 22, 2011

Prepared for:
Bit9 Inc.
266 Second Ave
Waltham, MA 02451

Prepared by:
Booz Allen Hamilton
Common Criteria Testing Laboratory
900 Elkridge Landing Road, Suite 100
Linthicum, MD 21090-2950

Table of Contents

List of Figures

List of Tables

1 Security Target Introduction

This chapter presents the Security Target (ST) identification information and an overview. An ST contains the Information Technology (IT) security requirements of an identified Target of Evaluation (TOE) and specifies the functional and assurance security measures offered by the TOE.

1.1 ST Reference

This section provides information needed to identify and control this ST and its Target of Evaluation.

1.1.1 ST Identification

ST Title: Bit9, Inc. Parity™ Version 6.0.1 Security Target

ST Version: 2.0

ST Publication Date: February 22, 2011

ST Author: Booz Allen Hamilton

1.1.2 Document Organization

Chapter 1 of this ST provides identifying information for the Bit9 Parity™ Version 6.0.1, from this point forward referred to Parity. It includes an ST Introduction, ST Reference, ST Identification, TOE Reference, TOE Overview, and TOE Type.

Chapter 2 describes the TOE Description, which includes the physical and logical boundaries.

Chapter 3 describes the conformance claims made by this ST.

Chapter 4 describes the Security Problem Definition as it relates to threats, Operational Security Policies, and Assumptions met by the TOE.

Chapter 5 identifies the Security Objectives of the TOE and the Operational Environment.

Chapter 6 describes the Extended Security Functional and Assurance Requirements.

Chapter 7 describes the Security Functional Requirements (SFRs).

Chapter 8 describes the Security Assurance Requirements (SARs).

Chapter 9 is the TOE Summary Specification (TSS), a description of the functions provided by Parity to satisfy the security functional and assurance requirements.

Chapter 10 is the TOE Summary Specification Rationale and provides a rationale, or pointers to a rationale, for security objectives, assumptions, threats, requirements, dependencies, and PP claims.

Chapter 11 is the Security Problem Definition Rationale and provides a rationale for the chosen EAL, any deviations from CC Part 2 with regards to SFR dependencies, and a mapping of threats to assumptions, objectives, and SFRs. It also identifies the items used to satisfy the Security Assurance Requirements for the evaluation.

1.1.3 Acronyms

The acronyms used throughout this ST are defined in Table 1-2: Acronym Definitions. This table is to be used by the reader as a quick reference guide for acronym definitions.

Acronym	Definition
AD	Active Directory
ADO	ActiveX Data Objects
ADSI	Active Directory Service Interfaces
CA	Certificate Authority
CCEVS	Common Criteria Evaluation and Validation Scheme
CLI	Command Line Interface
EAL	Evaluation Assurance Level
GSR	Global Software Registry
HTTPS	Hypertext Transfer Protocol Secure
IIS	Internet Information Services
IT	Information Technology
LDAP	Lightweight Directory Access Protocol
OS	Operating System
PE	Portable Executable
PP	Protection Profile
SAR	Security Assurance Requirements
SFR	Security Functional Requirements
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer
ST	Security Target
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security

TOE	Target of Evaluation
TSS	TOE Summary Specification
UTC	Universal Time Code

Table 1-1: Acronym Definitions

1.1.4 References

- Common Criteria for Information Technology Security Evaluation, CCMB-2009-07-004, Version 3.1 Revision 3, July 2009
- Bit9 Using Parity™ Version 6.0.1 (User's guide)
- Bit9 Installing Parity™ Version 6.0.1 (Installation guide)
- Operating Environment Guidelines Parity™ Version 6.0.1
- Evaluated Configuration for Bit9 Parity 6.0.1

1.2 TOE Reference

Bit9, Inc. Parity™ Version 6.0.1

1.3 TOE Overview

This Security Target (ST) defines the Information Technology (IT) security requirements for Parity. Parity is a policy-driven whitelisting solution for restricting the execution of applications and devices that runs on Windows PCs. Whitelisting technology allows end-users to install and run legitimate software and devices while providing IT groups with a way to prohibit anything unauthorized or known to be malicious from executing. The end result is granular control of Windows computers, dramatically improving security, preventing software drift, and managing the flow of information to portable storage devices.

Parity's management capabilities track portable executable (PE) and script files and monitor their prevalence and execution. Unidentified files that have just appeared on the network receive a pending status. A file keeps its pending status until it becomes approved or banned. A pending file also can be acknowledged, which removes it from the list of new pending files but does not change its underlying pending status. Once a file is approved, it is allowed to execute on all systems but continues to be tracked.

After a network is under Parity control, Administrators approve new applications or patches using the approval methods that best suit their organization's software rollout procedures. Parity features several automatic approval methods (trusted directory, trusted publisher, trusted user, and trusted updaters) that make it easy to approve new software without having to do it file-by-file. For example, a Console User can globally approve desktop software like Microsoft Office by putting it into a trusted deployment directory. Computers on the network would then be permitted to run PEs because the Parity Client recognizes these files as approved. Alternatively, Administrators can manually mark individual files as approved or banned.

Parity can perform the following capabilities:

- Block the use of unauthorized software to protect client workstations against exploitation
- Reduce the burden of compliance through streamlined audits, activity monitoring, violation notification, and policy enforcement
- Utilize the Parity Knowledge service to inventory and assess the software applications across computers
- Determine whether a file exists on a network, which computers on the network have that file, where and when the file first arrived in an environment. What is known about the source, category, trust level, and threat of the file, whether a file has propagated and, if so, renamed itself, and how the inventory of files on computers has changed over time.
- Determine whether certain USB storage devices exist on a network, when they first were discovered, and on what computer
- Prevent data theft and leakage by auditing and controlling the transfer of data to USB storage devices
- Prevent software drift to minimize risk, maintain compliance and reduce support costs
- Generate audit trails for activity

1.4 TOE Scope

Although Parity performs many functions, the key functionalities of the TOE include:

- Auditing of user's actions on the TOE
- Management of audit, alert, and access control policies
- Alerts of policy security violations
- Policy enforcement on TOE resources
- Policy enforcement to control access to files, processes, and registry values on remote workstations

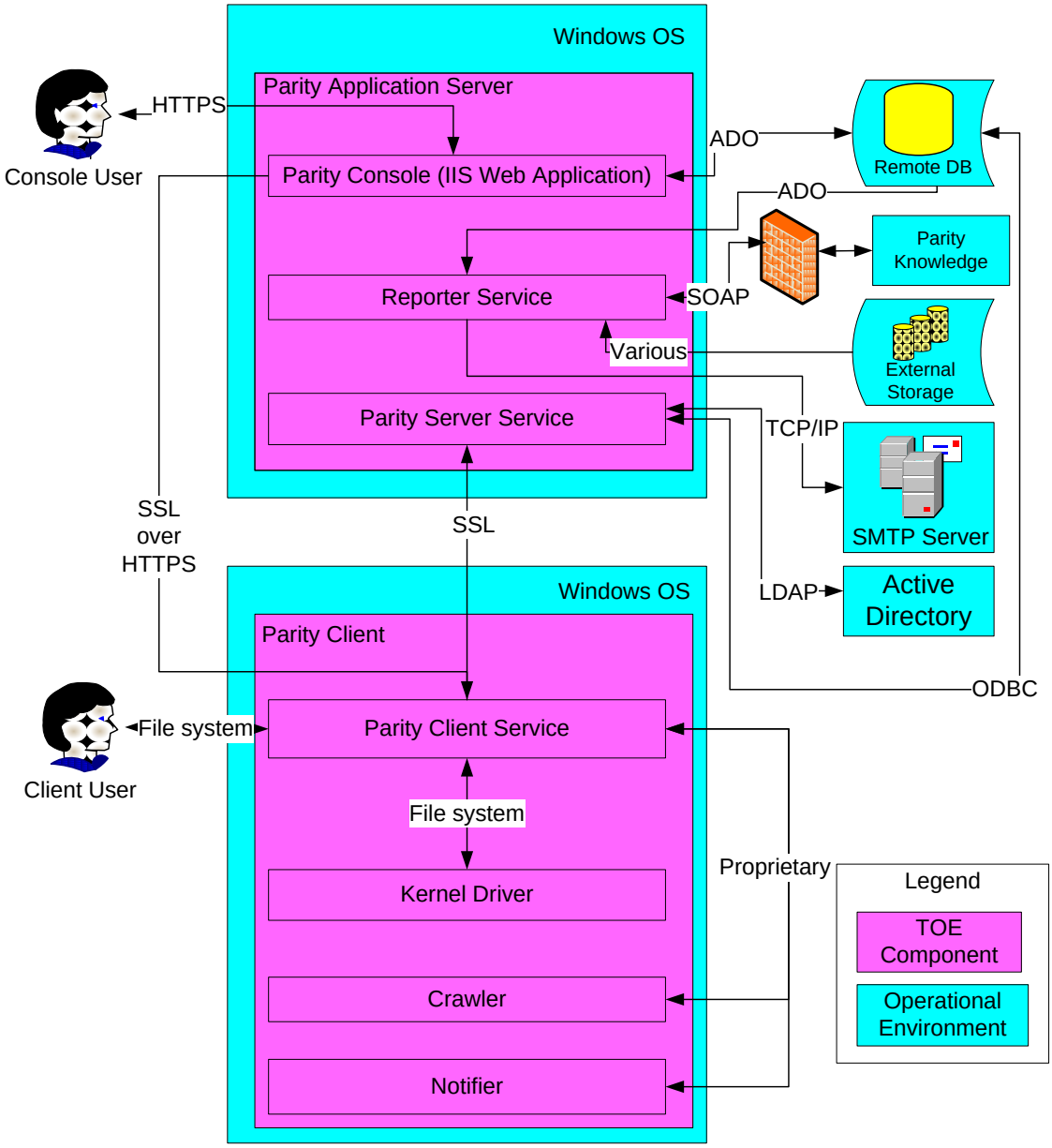
1.5 Product Architecture

The Parity architecture consists of the following components:

- Parity Application Server software, which provides central file security management, event monitoring, and a live inventory of files of interest on all Parity Client systems.
- Parity Client software, which runs on servers, desktops, and laptops, monitors files and either blocks or permits their execution based on security policy settings. It also reports new files to the Parity Application Server.

- Parity Knowledge, formally known as Global Software Registry™ (GSR), which is a service that compares new files introduced on computers running Parity Client to a database of known files, providing information on threat level, trust factor, and software categorization.
- Remote Database Server is an external storage device that maintains an up-to-date database of file-related events.
- External Storage is a remote database or syslog server that provides a backup to the Remote Database Server
- Remote SMTP Server will send e-mail notifications to their final destination when the Parity Server has detected that an alert condition has occurred and generates the e-mail.
- Remote Active Directory server that serves as a repository for user identification.

As shown in Figure 1-1, Parity is a Client-Server application. Parity Application Server, the server side of the system, contains the main application server and supported subsystems, as well as the web application for administration and reporting. The system requires a database server for storage which can reside on the same machine or a remote system. The Parity Client is installed on one or more endpoints in an organization and is responsible for enforcing the client access policies defined on the Parity Application Server. These policies contain rules regarding what files are allowed to execute, what registry operations are allowed, and what actions may be taken on removable devices. The Parity Client monitors all file activity and reports file metadata and activity events back to the Parity Application Server. Because Parity Knowledge is a repository that is maintained by the vendor, it is considered to be an environmental component.



Acronyms:
ADO - ActiveX Data Objects
SOAP - Simple Object Access Protocol
SMTP - Simple Mail Transfer Protocol
LDAP - Lightweight Directory Access Protocol
HTTPS - Hypertext Transfer Protocol Secure
SSL - Secure Sockets Layer
TCP/IP - Transmission Control Protocol/Internet Protocol

Figure 1-1: TOE Boundary

1.5.1 Parity Application Server

Parity Application Server software runs on standard Windows computers. It can be run on a dedicated system or on a virtual machine. Parity Application Server is used to manage policies, including software and device approvals and bans, and to provide visibility into events and file activity on computers running Parity Clients.

Using HTTPS, Console Users access the Parity Application Server and the TOE through a Web Browser. Console User access to the Parity Application Server is determined by the Parity Console (IIS Web App). The Parity Console (IIS Web App) performs the identification and authentication of all users of the Parity Application Server. All administration and reporting functions, to include access information, of Parity are communicated directly with the environmental database using ADO.

Console User authentication uses Parity Server Service and a secure LDAP connection with Active Directory to validate credentials and look up account memberships. Using ODBC, Parity Server Service logs record session information or errors to the database.

Once a user has successfully authenticated to TOE via the Parity Console, the Console User must then be authorized to perform actions on the TOE by the Parity Server Service. The most important action which can be authorized by the Parity Server Service is the ability to create policies. Examples of policies include but are not restricted to: (1) approving a discovered file by its hash value, (2) approving a discovered file by trusted publisher (digital certificate), (3) blocking registry changes, (4) banning a discovered file by its hash value, (5) approving a device for network use, or (6) trusting all content of a client directory.

After policy creation and when a connection between Parity Client and Parity Server is re-established after being disconnected, the Parity Server Service will propagate polices to Parity Clients using operational environment provided SSL. In addition to passing client policies to Parity Clients, the Parity Server Service also receives events using operational environment provided SSL.

Note that the product's SSL implementation is provided by the operational environment.

Additionally, the Reporter Service handles all scheduled and background tasks for the Parity system. It uses a mutually authenticated web service (SOAP over operational environment provided SSL) connection with Parity Knowledge (GSR) to retrieve enhanced metadata regarding all files known to the system. This enhanced metadata includes threat, trust, and categorization values. The Reporter Service accesses the Parity database directly using ActiveX Data Objects (ADO).

Note: Although only Console Users authenticated by the TOE may view audited events through database queries, alerts may be sent to any valid email address.

Lastly, Parity tracks executable files and monitors their prevalence and execution. Unidentified files that have just appeared on the network receive a *pending* status. A file keeps its *pending* status until it becomes *approved* or *banned.* A pending file also can be *acknowledged*, which removes it from the list of new pending files but does not change

its underlying pending status. Once a file is *approved*, it is allowed to execute on all systems but continues to be tracked.

Besides blocking unauthorized files, Console Users can use other Parity features to determine information such as the following:

- Whether a file exists on a computer on the network
- Which computers have the file
- Where and when the file first arrived in the network environment
- What is known about the source, category, trust level, and threat of the file
- Whether and when a file has executed, and on which computers
- How the inventory of files on computers has changed over time
- Whether a file has propagated and, if so, whether it has been renamed
- Whether certain USB storage devices exist on a network, when they first were discovered, and on what computer

1.5.2 Parity Client

Parity Client, also known as Parity Service Agent, software running on client computers monitors files, process and registry activity and communicates with the Parity Application Server when necessary. If the client is unable to connect to the Parity Server, it uses an offline policy to make decisions that allow for uninterrupted enforcement of access control. When a disconnected computer running the Parity Client reconnects, it receives updates from the server and communicates relevant file activity during the time it was disconnected from the network. The Parity Client runs silently in the background until it blocks a file, at which point it displays a message to the computer user that explains why the file was not permitted to execute. Depending on the file state and the Parity Client's security level, Parity may be configured to let the user on the client computer choose to run a blocked file.

The Parity Client Service is the main user space subsystem of the Parity Client. It is responsible for all communications with the Parity Application Server (via SSL over HTTPS or SSL). It contains a local storage mechanism for persisting all client access policies, and file and activity (event) data. It coordinates with all other subsystems on the Parity Client to perform its functionality.

When a decision requires user input (for example, "allow" or "block") or a rule has been triggered that specifies information that will be displayed to the Client User, the Notifier subsystem is activated. It receives input from the Parity Client Service using a proprietary protocol and may provide status information back to the Parity Client Service based on the action taken by the Client User.

1.5.3 Parity Knowledge

Parity Knowledge, formally known as Global Software Registry™ (GSR), is a web service hosted by Bit9 that helps identify and classify software discovered in a network

by comparing it to an extensive database of known files. Based on weighted analysis, Parity Knowledge service further assigns a threat level (malicious, potentially malicious, unknown, or clean) and a trust level (0-10 or unknown) to each file. Parity Application Server can include this information in its live file inventory so that Console Users immediately know the threat status and other key information about files on network systems.

Parity Knowledge implements ongoing updates to two different user configurable mechanisms which update the known database files.

- The first mechanism, also known as incremental changes, updates the database by adding new files every 10 minutes (this default setting; it may be changed by the Console User.) Once the new file is detected, it is reported to the server. Ten minutes after the file has been reported, it is added to the database.
- The second update mechanism, also known as a full refreshes, occurs once every three hours (this is the default setting; it may be changed by the Console User). Every three hours existing files are compared, when a change to an existing file is detected, the database updates to reflect the change.

Note that as a centrally maintained repository that is managed by the vendor, Parity Knowledge is considered to be a part of the Operational Environment. The data that resides within Parity Knowledge is transmitted to the TOE via a secure channel and with mutual authentication so that it can be trusted to affect the TSF's behavior.

1.5.4 Remote Database

Parity allows Console Users to copy event data to an external SQL Server. External logging gives Console Users the option of creating custom report implementations directly through SQL. Using an external server can also allow Console Users to fulfill compliance requirements for long-term event storage while maintaining events for a shorter period on the Parity Server database.

Parity automatically backs up its database to a specified backup location within two minutes of a critical change or once an hour, whichever comes first. Backups also include saved Find Files queries. Continuous automated backups ensure that the server and connected computers remain synchronized.

Parity Application Server can be configured to export a copy of the events (as they are generated) to an external Microsoft SQL Server or Server SQL Express database. Note that this database is not maintained by the TOE; as such, whatever database is chosen will reside in the Operational Environment.

1.5.4.1 Standard SQL Server

When using a Microsoft SQL Server database, there is no maximum allowable size for the audit storage (events table). The maximum is determined by the space available on the database server, as well as any size/growth restrictions placed on the database or

tables by the SQL Server administrator. In the evaluated configuration NT Authentication will be used for connection between the TOE and SQL Server/SQL Express.

1.5.4.2 Server SQL Express Database

When using a SQL Express database, there is a total maximum database size limit of 4GB. When the total database size reaches 3GB, an Alert event is automatically generated once a day. An optional email alert can also be enabled for this event.

There are two options provided to control when (or if) events will be pruned (deleted): based on date/time (e.g. events older than X days), and based on total event count (e.g. when the total number of events is greater than X). Either of these pruning options can be defined by a Parity Administrator from the Parity Console. When one of these conditions occurs, the administrator can specify how much (as a percentage of the total events) should be deleted.

1.5.5 External Storage

In addition to using the Remote Database, the Parity Server can be configured to export all events to a Syslog server or secondary backup database. The IP address and port of the Syslog server is configured by the customer from within the Parity Console. If a database is used, the same configuration information applies as in the previous section.

External Storage, regardless of the implementation chosen, is considered to be part of the Operational Environment.

1.5.6 Remote SMTP Server

In order for the TOE to send emails to administrators when alert conditions are triggered, it must have the ability to contact an SMTP server. The SMTP server resides in the Operational Environment and is used to relay emails at the request of the TOE.

1.5.7 Active Directory

An Active Directory (AD) server is deployed in the Operational Environment and used to define subjects against which Client Access Policies can be defined. In other words, if an organization already utilizes Active Directory, the TSF does not need to define additional subjects because it can utilize this legacy data. In addition, this can also optionally be used to define subject identities for Console Access Policies.

1.6 TOE Type

The TOE type Bit9 Parity ™ version 6.0.1 provides the following: System Access Control. System Access Control is defined by CCEVS as "A technique used to define or restrict the rights of individuals or application programs to obtain data from, or place data onto, a storage device. The definition or restriction of the rights of individuals or application programs is to obtain data from, or place data into, a storage device limiting access to information system resources only to authorized users, programs, processes, or other systems." The TOE is considered to be an example of a System Access Control device because it uses whitelisting to define the rights of individuals to modify the

runtime state of remote workstations. It can also be used to define the ability to place data onto remote storage devices.

1.7 Terminology

This section defines the terminology used throughout this ST. The terminology used throughout this ST is defined in Table 1-1: Terminology Definitions. This table is to be used by the reader as a quick reference guide for terminology definitions.

Terminology	Definition
.NET	A software framework that can be installed on computers running Microsoft Windows operating systems. It includes a large library of coded solutions to common programming problems and a virtual machine that manages the execution of programs written specifically for the framework. The .NET Framework is a Microsoft offering and is intended to be used by most new applications created for the Windows platform.
Active Directory (AD)	Technology created by Microsoft that provides a variety of network services, including: • Lightweight Directory Access Protocol (LDAP)-like directory services • Kerberos-based authentication • DNS-based naming and other network information • Central location for network administration and delegation of authority • Information security and single sign-on for user access to networked based resources • Central storage location for application data • Synchronization of directory updates amongst several servers
ActiveX Data Objects (ADO)	A set of Component Object Model (COM) objects for accessing data sources. A part of Microsoft Data Access Component (MDAC), it provides a layer between programming languages and Object Linking and Embedding (OLE) Database (a means of accessing data stores, whether they be databases or otherwise, in a uniform manner).
Administrator	Monitors file activity and configures all aspects of the system, including creating policies, setting alerts, and creating all types of user accounts
Alert	A tool that provides notifications in the Parity Console and email notifications to a list of subscribers when a specific event an Administrator or PowerUser may choose occurs. Alerts can be made policy-specific.
Baseline	A reference point that can be used to determine drift of computers running Parity Client from the reference, and thus potential risk for those computers. A baseline can be a named table of files or the current set of files on a reference computer.
Basic Authentication	Authentication to the TOE through the use of any valid user interface. Data for credentials are stored on the database rather than Active Directory.

Blacklist	A list or collection of software that, for one reason or another, is expressly forbidden.
Client Access Policies	The union of Parity Policies (see "Policy), Policy Settings (see "Policy Settings"), and Policy Rules. Client Access policies include the information contained within the Policy Setting definition below as well as the more granular approval/ban rules covering individual devices, files, publishers, processes, applications, and users.
Client Users	Users who interact with the TOE via the Parity Client. Their requests to interact with their local system are mediated by the Client Access Policy.
Computer	Windows-compatible computer that runs the Parity Client. Each computer protected by Parity communicates with the Parity Application Server when it is on the same network.
Computer initialization	File initialization process for new computers that come online to the Parity system. During initialization, each file on the hard-drive of the new machine is evaluated and classified by the Parity Application Server.
Console Access Policies	The union of Parity Policies (see "Policy), Policy Settings (see "Policy Settings"), and Policy Rules. Console Access policies include information regarding the user's identification along with group association to determine what information can be accessed.
Console Users	Term that refers to the collection: Administrators, PowerUser, and ReadOnly. Collectively, these are the users which are permitted to access the management interface of the TOE.
Dashboard	Interface that graphically displays network environment information through a series of compact "portlets." Through a Dashboard, Parity Server users can drill down to more detailed information about network elements such as files, computers, and alerts.
Device Rules: Approvals	Console Users can choose to approve file activity on specific, detected USB devices, regardless of the security policy in force for a computer.
Drift	Information that can help determine to what extent one or more computers have changed from a baseline of files. This can help determine level of compliance with company policies on acceptable files, and also identify files that should be approved and added to an updated baseline.
Event	An event is informational message resulting from Parity activities that can be filtered and presented as custom reports. Events include files blocked, pending files executed, and changes made to the system by console users.
Executable	An executable is any file that contains executable code. Parity examines the *content* of each unknown file that appears on a computer in its network, determines whether it contains executable code, and if so, categorizes it according to executable type. Scripts are included in this process.
Explicit Group Assignment	Assignment to restrict/allow performable functions of the TOE. Falls under the different roles types as defined by the TOE.
File Rule	The Files tab of the Software Rules page shows all of the approvals and bans created at a site for individual files. From this tab, a Console User can manage rules. The Console User can check to see whether a particular file has any rule,

	file or ban, affecting it, and the Console User can remove rules from one or more checked files.
File state	Parity classification that determines how executables are tracked and permitted or not permitted to be run. File state includes approved, banned, and pending states.
Files	Display of the Files page, which includes tabbed lists of networked files, including: File Catalog, a searchable archive of all unique files hashed on the Parity server, and Files on Computers, a list of all files running on all of computers running Parity Clients.
Group	A set of users with a selection of permissions that is stored either in the Active Directory or a Parity-defined group.
Hypertext Transfer Protocol Secure (HTTPS)	A combination of the Hypertext Transfer Protocol with the SSL/TLS protocol to provide encryption and secure (website security testing) identification of the server.
Internet Information Services (IIS)	Set of Internet-based services for servers using Microsoft Windows
Lightweight Directory Access Protocol (LDAP)	An application protocol for querying and modifying data using directory services running over TCP/IP. A directory is a set of objects with attributes organized in a logical and hierarchical manner. A simple example is the telephone directory, which consists of a list of names (of either persons or organizations) organized alphabetically, with each name having an address and phone number associated with it.
Manifest file	A document that lists the contents and attributable file information that uniquely identifies each file contained in the document. This information may be used to automatically generate a set of approval policies.
Meter	A software tool that enable a Console User to monitor the number of executions of files an Administrator or PowerUser may specify, and the users and computers executing them.
Modes	Active Parity Clients can be operated in one of two modes: Visibility Only provides the file and event tracking features of Parity, but does not enforce file or device bans or other security restrictions. Visibility and Control mode blocks banned files and allows a Console User to choose one of three SecCons to determine how pending files are treated. Visibility and Control policies can be configured to enforce other file and device security rules.
Open Database Connectivity (ODBC)	Standard software API method for using database management systems (DBMS). The designers of ODBC aimed to make it independent of programming languages, database systems, and operating systems.
Parity Application Server	Windows-compatible computer running the Bit9 Parity Application Server software.
Parity Client	Agent software installed on computers on a network; the agent runs independently but reports to the Parity Application Server.

Parity Console	The console, which can be displayed remotely with a web browser, is the user interface and management center for all Parity management activities.
Policy	Each computer protected by Parity is associated with a policy that defines its security characteristics. Computers with the same security requirements can share the same policy. Note that users of computers running the Parity Client do not need Parity accounts. The server requires no direct interaction with users of computers Parity is monitoring.
Policy Settings	Policy settings define the way a Parity user manages a particular group of computers. There are three categories of settings: basic policy definitions, device settings, and advanced settings. • Basic policy definitions include the policy name and other descriptive information, whether computers in this policy allow Parity Client upgrades, whether live file inventory is activated for these computers, and the basic security level (the Mode and SecCon) for the policy. • Device settings control the way a Parity policy treats removable devices. Parity Administrators or PowerUsers can make different rules to control read, write, and execute operations on devices, and can designate approved devices to be treated differently than non-approved devices. • Advanced policy settings control whether computers in a policy have certain file types blocked, whether files installed by specially designated "trusted" users are allowed to execute, and whether special treatment of certain directories is enabled. The possible values are Active, Off, and Report Only.
Portlet	A Dashboard element that displays information such as the number and types of computers managed by Parity, the number and type of security policies enforced, the drift of files on one or more computers away from a reference point, the types of software on a network, and whether the files identified by Parity are compatible with Vista. Although there is variation in the specific information displayed, their structure is generally similar. Each portlet has a banner with its name in the top left and a series of buttons in the top right.
PowerUser	A type of Console User which monitors file activity and can set policy. PowerUsers have limited user-account creation privileges, including creating ReadOnly and Unauthorized user accounts but not Administrator and other PowerUser accounts. Some system settings cannot be changed by PowerUsers.
ReadOnly	A type of Console User which monitors file activity and views previously executed Find Files search results. ReadOnly users cannot change any aspect of the Parity system configuration, and cannot create or edit users, policies, bans or approvals.
Role	A set of access control permissions within the system.

Saved Views	A menu to further specify the files that an Administrator or PowerUser may want to see, including Banned Files, New Pending Files, Malicious Files, Categorized Files, and Trusted Packages. Administrators or PowerUser also can Approve files (both globally and locally) and Ban files here.
SecCon (Security Condition)	The protection level applied to computers running Parity Client. A range of levels from Lockdown (most protective) to Agent Disabled (least protective) enable a Console User to specify the level of file blocking required. Additionally, there is an "offline seccon" which applies when the Parity Server cannot be reached.
Secure Sockets Layer (SSL)	Cryptographic protocol that provide security for communications over networks such as the Internet. SSL encrypts the segments of network connections at the Transport Layer end-to-end.
Simple Mail Transfer Protocol (SMTP)	Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks. SMTP was first defined in RFC 821 (STD 15) (1982), and last updated by RFC 5321 (2008).
Simple Object Access Protocol (SOAP)	A protocol specification for exchanging structured information in the implementation of Web Services in computer networks.
Snapshot	A clean baseline for network files computer that Administrator or PowerUser have created for use in baseline drift analysis.
Software (File) Rules: Bans	Bans enable Console Users to specify files (by name or hash) to be blocked for particular groups of computers (i.e., by policy) or all computers at a site. Parity can ban files individually, and also can ban all files identified on a list of hashes the Console User provides.
Software Rules: Approvals	Several complementary software approval methods enable Console Users to approve legitimate software to run on all computers, on groups of computers (i.e., by policy) or to locally approve software to run on a single computer. Registry Rules Console Users can specify rules to protect specific registry key/value patterns from modification.
Transport Layer Security (TLS)	Cryptographic protocol that provide security for communications over networks such as the Internet. TLS encrypts the segments of network connections at the Transport Layer end-to-end.
Trusted Directory	When a Parity Client contains a Trusted Directory, the contents (or changes) to that folder are sent from the client system to the Parity Console. The Parity Console then notifies the Parity Server Service. The Manifest Processing component of the Parity Server Service is responsible for validating the manifest file, importing it into the system, and then updating the client access policies appropriately.
User	A user can refer to either a Console User or a Client User.
Users	Console Users and Client Users
Whitelist	A list or collection of software or entities that are known, trusted, or explicitly permitted.

Table 1-2: Terminology Definitions

2 TOE Description

This section provides a description of the TOE in its evaluated configuration. This includes the physical and logical boundaries of the TOE.

In addition to the items described in the subsections below, the TOE includes the following user documentation:

- Bit9 Using Parity™ Version 6.0.1 (User's guide)
- Bit9 Installing Parity™ Version 6.0.1 (Installation guide)
- Operating Environment Guidelines Parity™ Version 6.0.1
- Evaluated Configuration for Bit9 Parity 6.0.1

The 'Evaluated Configuration for Bit9 Parity 6.0.1' document is a security guide that explains how to set up the evaluated configuration and provides information to administrators and users to ensure the secure operation of the system. Bit9 provides documentation on their support website. Included within this documentation is the 'Evaluated Configuration for Bit9 Parity 6.0.1' which must be referenced to place the product within the CC evaluated configuration.

2.1 Evaluated Components of the TOE

The scope and requirements for the evaluated configuration are summarized in Table 2-1. The items within the component column are a modular level within the TOE. These items are described within the definitions column. They also belong to one of two of the software packages defined in the software package column. These software packages are bundled together for installation but contain multiple executables as defined in Section 2.3 of the ST. Both software packages are required for the functionality of the TOE.

Component	Software Package	Definition
Crawler	Parity Client 6.0.1	The Crawler is responsible for file and directory analysis. When a client access policy defines a Trusted Directory on a client machine, the Parity Client Service executes the Crawler module to analyze files within that directory.
Kernel Driver	Parity Client 6.0.1	The Kernel Driver module is the Parity kernel driver that interfaces with the Parity Client's operating system.
Notifier	Parity Client 6.0.1	The Notifier is the engine that creates a pop-up message that a Client User sees when Parity blocks file execution if the client has been configured to do so.
Parity Server Service	Parity Server 6.0.1	The Parity Server Service is where the main application and business logic resides. This module is responsible for receiving connections from all clients, managing and broadcasting client access policies to each client, performing administrative functions, and processing events and data from all clients.
Parity Client	Parity Client 6.0.1	The Parity Client Service is the main user space module of the Parity Client. It is responsible for all communications with the

Service		Parity Application Server. It contains a local storage mechanism for persisting all client access policies, and file and activity (event) data. It coordinates with all other subsystems on the Parity Client to perform its functionality.
Parity Console (IIS Web Application)	Parity Server 6.0.1	The Parity Console module is a web application that runs under Microsoft IIS. All administration and reporting functions of Parity are accessed via Console Users logging into the Parity Console.
Reporter Service	Parity Server 6.0.1	The Reporter Service handles all scheduled and background tasks for the Parity system. It uses a mutually authenticated web service connection with Parity Knowledge (GSR) to retrieve enhanced metadata (threat, trust, and categorization) regarding all files known to the system. It periodically exports audit data to specified targets. It also monitors the audit data for alertable events and sends email.

Table 2-1: Component Definitions

2.2 Components and Applications in the Operational Environment

Each of the evaluated Operational Environment components is defined below:

Component	Definition
Active Directory (AD)	Assuming users, computers, and groups by using Microsoft Active Directory have been defined and named, Parity can use Active Directory environment to set access privileges for users of the Parity Console, assign security policies to computers, provide user and computer metadata, and designate certain groups or users to be able to install software (and have it automatically approved) on Parity-managed computers.
Database	Parity keeps an up-to-date database of file-related events. From this data, a Console User can view predefined or custom reports that reveal activity of interest so that a Console User can monitor environmental changes and significant Parity operations. Additionally, Console Users can export Parity events to Syslog and to CSV files.
Email Server	The Email Server sends information regarding alerts (notifications) and is used to propagate those alerts to the users of the TOE.
Operating System	Both the Parity Server and Parity Client run on Microsoft Windows. Actions taken against Microsoft Windows on a client system can be mediated by the TOE.
Parity Knowledge (GSR)	Parity Knowledge, formally known as Global Software Registry™ (GSR), is a web service hosted by Bit9 that helps identify and classify software discovered in a network by comparing it to an extensive database of known files. Based on weighted analysis, Parity Knowledge service further assigns a threat level (malicious, potentially malicious, unknown, or clean) and a trust level (0-10 or unknown) to each file. Parity Application Server can include this information in its live file inventory so that Console Users immediately know the threat status and

	other key information about files on network systems. Parity Knowledge requires a direct internet connection for use. It is a strictly optional component that facilitates convenient administration, so the inability to use it in a secure processing environment with an isolated network does not have a security impact.
Syslog Server	The Syslog Server is an optional external storage device that stores, reports, and analyzes messages from the Reporter Service in the Parity Application Server.
Backup Server	An external storage device used to store an up-to-date database of file-related events.
Web Browser	The User Interface is a browser based GUI. Using the Web Browser a Console User can manage the TOE. All administration and reporting is done via the Web Browser.
Web Server	The Web Server provides Parity Console content to administrators.
Windows OS	The Parity Application Server and Parity Client must be specifically installed on a Microsoft Windows Operating System.

Table 2-2: Component Definitions for the OE

2.3 Physical Boundary

Parity is a software-only TOE which is intended to be installed on off-the-shelf servers and on client workstations. The physical boundary of evaluation does not include any product hardware, operating systems, or dependent software.

The following software versions were in the evaluated configuration:

- Parity Server 6.0.1 – includes ParityServer and ParityReporter executables
- Parity Client 6.0.1 – includes Parity, Crawler, Notifier, and TimedOverride executables

In the evaluated configuration, the TOE is defined to be one instance of the Parity Server and one or more instances of the Parity Client, all of which are installed on separate systems.

The following Operational Environment software is considered to be outside the evaluation boundary.

- Microsoft Internet Information Server (IIS) 6.0 or higher
- Microsoft Internet Explorer 7.0 or higher
- Microsoft Windows XP SP2 or higher (for clients)
- Microsoft Windows Server 2003 or 2008 SP2 (for the Application Server)
- Mozilla Firefox 3.0 or higher
- SMTP Server – any server that allows unauthenticated SMTP connections
- Active Directory W2K3 SP2 or higher

- Syslog server – any server that allows unauthenticated Syslog connections
- Microsoft SQL Server (regular or Express) version 2005 r2 or higher
- VMware ESX Server 4.0
- LDAP server (for Active Directory) – version that is compatible with Active Directory W2K3 SP2 or higher
- DNS server – any DNS server is acceptable
- Parity Knowledge (unversioned, the TOE will automatically connect to the most up-to-date server)

The specific versions of Microsoft IIS and SQL server are specified in section 2.3.1. For the other items in this list which are not identified by version number, specific versions are not required.

There is no specific required deployment of Operational Environment software. Different software may co-located or installed on separate physical machines. Note that if distributed machines are used, it's expected that these components will be configured to use their native mechanisms for encrypted communications where applicable in order to mitigate risk of eavesdropping.

2.3.1 Parity Server System Requirements

The Parity Server software runs on dedicated server hardware. System requirements vary according to system load, which is determined by how many computers running Parity Client are supported by the server. Parity Application Server supports up to 50,000 Parity Client systems. For adequate performance, hardware should meet the specification shown in Table 2-4.

The following are recommended hardware requirements that a machine must meet or exceed to run Parity Application Server:

Parity Client Load	Less than 300 Clients	300 to 50000 Clients
Processor	Dual Core Server Class	Dual Core or Dual Processor Server Class
Disk space	40 GB	40GB for Parity, 72GB for SQL
RAM	2-4 GB	4 GB
Network	1 GB NIC	1 GB NIC
IP address	Fixed IP address or (preferably) a fully qualified DNS name. Computers running the Parity Client recognize the server by either its fixed IP address or DNS-name lookup.	
Operating System	• Microsoft Windows Server 2003 - SP2 32-bit, with Microsoft Internet Information Services (IIS) version 6.0, with latest patches. • Microsoft Windows Server 2008 SP2 32-bit or 64-bit, with Microsoft Internet Information Services (IIS) version 7.0, with any patches. • For both Server 2003 and 2008, install .NET 3.5, with latest patches.	
SQL Server	• SQL Server 2005 Express SP2 for <300 Client computers • SQL Server 2005 SP2 or above Standard/Enterprise OR SQL Server 2008 SP1 or above for >300 Client Computers	

Table 2-3: Parity Application Server System Requirements

2.3.2 Parity Application Server Virtualization

To run VMware ESX Server v.4.0+ to create a virtualized environment for Parity:

- memory meeting the configurations must be allocated as 'reserved'
- minimum dual virtual processors are required for all configurations

The hardware requirements for the machine hosting the virtual environment should allow the virtual environment to be capable of the same level of performance as a hardware based installation.

2.3.3 Parity Client System Requirements

The following are recommended hardware requirements to run on each Parity Client:

OS	Processor	CPU	RAM	Disk Space
Windows XP 32-bit, SP2 & SP3	Intel Pentium 4 650MHz (or equivalent processor)	650 MHz	Any configuration that enables standard desktop applications to run with good performance, preferably at least 768M.	Approximately 65 MB, depending on the number of applications installed on the system
Windows 2003 Server 32-bit & 64-bit & R2, SP1				
Windows Vista 32-bit & 64-bit, SP1 & SP2				
Windows 2008 Server 32-bit & 64-bit, Windows 2008 Server R2 64-bit				
Windows 7 32-bit & 64-bit				

Table 2-4: Parity Client System Requirements

2.3.4 Global Software Registry

Although the TOE can operate without an internet connection, the GSR requires an Internet connection to maintain connection GSR Server at Bit9 headquarters. Because the GSR's information is maintained by the vendor and resides at a static location, it is not under the control of the customer and is therefore not considered to be inside the physical boundary.

2.3.5 Network Boundary

If the TOE is deployed with an internet connection, it is expected that the network on which it is deployed is secured with a firewall to reduce the TOE's exposure to potential outside threats. This firewall is not within the physical boundary of the TOE.

2.4 Logical Boundary

The logical boundary of the TOE is broken down into the following security classes: Security Audit, Encrypted Communications, User Data Protection, Identification and

Authentication, Security Management, Protection of the TSF, TOE AccessTOE Access, and Trusted/Path Channel. Listed below are the security functions with a listing of the capabilities associated with them:

2.4.1 Security Audit

The TOE collects, aggregates, and reports on IT activity and generates alerts when file/device information changes. "IT activity" refers to the content and behavior of systems which reside in the Operational Environment. Auditing functions are performed by the TOE by collecting the logs via clients and servers, using a database to store the logs over an established timeframe, and presenting the logs via reports and queries. Note that the TOE's client-server architecture has implications on the location of this database. When clients (which reside on environmental systems) collect data, it is stored in an internal cache until communications with the server can be established. Once the server can be reached, the data is sent to it and the server stores it in a SQL database in the Operational Environment. In addition, the TOE generates audit reports for its own startup and shutdown and all user actions on the TOE. Authorized users are able to select the notification mechanism for all auditable events. The TOE monitors these for events and notifies (alerts) users when a predefined condition is met. Alerts can be sent via email, which requires mediation by an environmental SMTP server. The TOE relies on the host operating system to provide reliable timestamps for audit records.

As soon as the Parity Client is installed on a system, the file inventory process begins. The software takes an inventory of all executable files on the client computer's fixed disks (but not removable drives) and creates three hashes of each file. When a computer with a Parity Client first connects to the server, it sends each collection of hashes to the Parity Server to determine its status and update the server's file inventory. This file inventory is then used as a basis for access control on the client system.

Parity can detect a variety of removable devices in the Operational Environment that contain file systems. The Device Rules page provides an inventory of all removable devices detected by Parity Clients running on managed computers. Console Users can approve any device in the table, and Console Users can remove approval from approved devices.

Parity requires access to an SMTP (Simple Mail Transport Protocol) server to send messages to a list of subscribers when alert conditions are met. In addition, since this can be used to send messages to any valid e-mail address, it is the responsibility of the administrator to ensure that these messages are only being sent to appropriate recipients.

As with file inventory, Parity keeps an up-to-date external database of file-related events. From this data, Users can view predefined or custom reports that reveal activity of interest. From these reports, Console Users can monitor environmental changes or significant Parity operations

2.4.2 Encrypted Communications

Remote users establish a session with the Parity Server using a web-based GUI that is secured via HTTPS. Cryptography for this is provided by the environmental web server and Operating System. This secured path is used for user authentication and management of the TOE by authorized users. The Operational Environment generates cryptographic keys during communication with remote users, the ODBC client, and Active Directory. This is accomplished by the TOE requesting that these environmental components use their native cryptographic facilities. All communications between the Parity Server and Parity Clients and between the Parity Server and the GSR are protected using imported certificates.

The only cryptographic function provided by the TSF is the ability to hash files for the purpose of access control checking.

The cryptography used in this product has not been FIPS-certified, nor has it been analyzed or tested to conform to cryptographic standards during this evaluation. All cryptography has only been asserted as tested by the vendor.

2.4.3 User Data Protection

Parity determines whether a Client User or Console User can perform an authorized action. Console Users cannot perform any action on a Parity controlled system or object unless the user has been authenticated by the TOE or through an Operational Environment authentication mechanism, and they are authorized access by Parity. Client Users cannot perform any action on a Parity controlled system or object unless they have first authenticated to the OS which resides in the Operational Environment. Parity then authorizes them access to various objects on the OS using their validated identity.

The Active Directory environment sets access privileges for users of the Parity Console and assigns security policies to computers which provide the user and computer metadata. Additionally, it designates roles on Parity managed computers. These rules can be written to determine allowed files, processes, and registry changes and then mapped to policies.

When the policy is mapped, a SecCon, or protection level, is applied to computers running Parity Clients. The SecCon level defined in a policy determines how Parity reacts when a resource that is not whitelisted is requested. Whenever a Console User creates a policy, the Console User specifies a SecCon level to control computers in that policy. SecCon levels, which vary in restrictiveness, affect how file execution is controlled for policy settings. Files blocking and other control functions in Parity depend on both the SecCon level and on more specific policy settings in effect, including policy-specific bans. In Visibility and Control mode, the Console User chooses SecCon 20, 30, or 40 from a menu. The other modes automatically designate SecCon: 60 for Visibility Only mode and 80 for Agent Disabled.

Additionally, Parity provides two sets of features to enable a Console User to define operations that users can perform over removable devices. Device control settings affect

read, write, and execute operations on a per policy basis. To further fine-tune network control of removable devices, a Console User can explicitly approve certain removable devices so that while unknown or unapproved devices are restricted by policy settings, devices in the approved table can be written to, and/or files from them can be executed.

Parity is protecting the resources of both an environmental computer system and of the TSF itself.

User account information is stored on the Parity Server system and is not protected by the TSF. This particular information is maintained by the Operational Environment. Information protected by the Parity Client system includes the registry data that is acted upon by the policies. The user account information is maintained within the Active Directory component of the Operational Environment and as discussed in Section 2.4.4 below should be configured to use an LDAP with Kerberos or NTLM authentication.

2.4.4 Identification and Authentication

There are four types of users of the TOE: Administrators, PowerUsers, and ReadOnly and Client Users. Administrators, PowerUsers, and ReadOnly are collectively referred to as Console Users.

Console Users manage the TOE remotely through a Web Browser. They are identified and authenticated with username and password. This authentication data can be maintained within the TSF or Active Directory integration can be performed to allow an existing organizational user to access the TOE using credentials maintained by the Operational Environment. If the Operational Environment is used, username/password validation will be done with LDAP. In order to protect these communications, LDAP should be configured to use Kerberos or NTLM to authenticate the Console User.

Client Users access the TOE indirectly by using their own local machines and are identified and authenticated by the underlying Operating System. These OS validated identities are then received by the TOE for identification by the TOE through the users' stored LDAP information when the Client Users perform actions mediated by the TOE. Runtime modifications to their machines are mediated by the TOE but they have no visible interaction with it except for the receipt of pop-up messages when an operation has been blocked. In the evaluated configuration, Client Users are identified by the user account they use to log in to their system which is derived from Active Directory. If guest account access is allowed, policies can be written which apply based on the machine name of the system which is being used. In other words, the TOE's policy enforcement is based on subject identification information which is collected by the Operational Environment.

There is one condition where Console Users can perform a TSF-mediated action against the Parity Client, known as timed override. This feature is used to change the security posture of a client when it is in an isolated environment and cannot communicate with the server. To perform this action, a code is generated on the Parity Console. It must then be entered to the Parity Client within a certain period of time. No authentication data is required to be provided when this feature is used.

In addition to authenticating Console Users, the TOE performs mutual authentication with the environmental Parity Knowledge service. By mutually exchanging certificates, the TOE and this remote entity are able to determine each others' authenticity before exchanging file metadata.

2.4.5 Security Management

There are four default roles for the TOE: Administrators, PowerUsers, ReadOnly, and Client Users. Administrators, PowerUsers, and ReadOnly are all types of Console Users. The role given to a user determines what operations or management functions they can perform on the TOE. Restrictions can be set on Client Users based on policy; permissions for Console Users are static.

Additionally, Parity Server can take advantage of an Active Directory environment to set access privileges for users of the Parity Console, assign security policies to computers, provide user and computer metadata, and designate certain roles or users to be able to install software (and have it automatically approved) on Parity managed computers.

The major security management functions of the TOE are the ability to review audit data and the ability to configure how the client access control policy is enforced. Policy data is propagated to clients and stored internally. The Parity Console, as a web-based application, requires the use of an environmental DNS server if it is to be identified by a qualified domain or host name as opposed to an IP address.

The environmental Parity Knowledge server is a repository maintained by the vendor which contains pre-defined classifications of known files and processes, both legitimate and malicious. This data can be useful to aid in administration but is not required in order to manage policies.

2.4.6 Protection of the TSF

To ensure Inter-TSF confidentiality during transmission between the client and server sides of the parity system, the Parity Application Server requests the Operational Environment to use a mutual authenticated web service (SOAP over SSL) to retrieve enhanced metadata (threat, trust, and categorization) regarding all files known to the system (all files stored by sets of cryptographic hash values in the Parity database) from client workstations. Inter-TSF confidentiality during transmission between the Client and Server is accomplished via a secure binary protocol sent over an SSL connection. These communications are between the Parity Server and the environmental Parity Knowledge repository and established by Parity Knowledge after the TSF has requested to communicate with it.

Secure communications are provided by the Operational Environment. The TSF requests their use but does not implement this capability internally.

2.4.7 TOE Access

Before a session begins, a warning will be displayed alerting the Console User that unauthorized access to the TOE is prohibited.

2.4.8 Trusted Path/Channel

Console Users, who are logically distinct from other communication paths, use a trusted path to provide assured end point identification and to protection of the communicated data from modification and disclosure. The trusted path is from the Console User's remote browser to the TOE via the Operational Environment's IIS server. The cryptographic functionality required for the establishment of this trusted path is performed by IIS when an access request is initiated by a Console User.

2.4.9 Features Excluded from TOE

The DAS CLI is excluded from the evaluation. It is used only for diagnostics and status and is not used during normal Parity operation. Its use is reserved for the vendor only and can be completely disabled and removed. It provides no added security related functionality, is not required for use in the TOE, and does not satisfy any of the SFRs. It should be noted that the DAS CLI and CLI are the same.

Windows Embedded for Point of Service (WEPOS) is excluded from the evaluation. WEPOS is excluded because any system on which it is being deployed is insufficient to maintain physical and logical security aspects of the TOE.

The Live Inventory SDK is a series of static read-only views into the external database which allow for integration into legacy systems such as existing help desk implementations. This is an alternate means of looking at the same data which is made available at the Parity Console via the FAU_SAR requirements. In addition, it provides direct access to the external database without any mediation by the TSF. Because it circumvents the Console Access Policy, this feature has been excluded from evaluation.

3 Conformance Claims

3.1 CC Version

This ST is compliant with *Common Criteria for Information Technology Security Evaluation*, CCMB-2009-07-004, Version 3.1 Revision 3, July 2009

3.2 CC Part 2 Extended

This ST and Target of Evaluation (TOE) is Part 2 extended for EAL2 to include all applicable NIAP and International interpretations through 22 February 2011.

3.3 CC Part 3 Augmented Plus Flaw Remediation

This ST and Target of Evaluation (TOE) is Part 3 augmented plus flaw remediation and ASE_TSS.2 for EAL2 to include all applicable NIAP and International interpretations through 22 February 2011.

3.4 PP Claims

This ST does not claim Protection Profile (PP) conformance.

3.5 Package Claims

This TOE has a package claim of EAL2.

3.6 Package Name Conformant or Package Name Augmented

This ST and Target of Evaluation (TOE) is conformant to EAL package claims augmented with ALC_FLR.1 and ASE_TSS.2.

3.7 Conformance Claim Rationale

There is no Conformance Claim rationale for this ST.

4 Security Problem Definition

4.1 Threats

The threats which exist for this evaluation represent both threats against the TOE directly as well as threats against the environmental resources to which the TOE controls access. These threats are enumerated below.

T.ACCESS: Unauthorized users could gain local or remote access to protected objects that they are not authorized to access.

T.ADMIN_ERROR: An administrator may incorrectly install or configure the TOE or install a corrupted TOE, resulting in ineffective security mechanisms.

T.MASK: A malicious user or process may view audit records, cause audit records to be lost or modified, or prevent future audit records from being recorded, thus masking a user's action.

T.MASQUERADE: A malicious user or process may impersonate the GSR or Parity Application Server in order to intentionally provide inaccurate configuration information or metadata to the TOE.

T.REVERSE: A malicious or ignorant user may acquire and configure a reverse-engineered version of the TOE that bypasses or subverts access control to protected resources.

T.UNAUTH: Malicious or non-malicious users could gain unauthorized access to the console by bypassing identification and authentication countermeasures.

4.2 Organizational Security Policies

The TOE addresses the organizational security policy described below.

P.ACCESS_BANNER The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the system.

4.3 Assumptions

In order to provide assurance that the TOE and its Operational Environment are configured in a manner that reduces their risk posture, it is necessary to make assumptions regarding their deployment. The assumptions can be grouped into the following: personnel assumptions, physical assumption, and connectivity assumptions.

4.3.1 Personnel Assumptions

A.ADMIN: One or more authorized administrators will be assigned to install, configure and manage the TOE.

A.NOEVIL: Administrators and PowerUsers of the TOE are not careless, willfully negligent, or hostile and will abide by the instructions provided by applicable guidance documentation.

A.PATCHES: Administrators exercise due diligence to update the Operational Environment with the latest patches in order to remove the risk of compromise via known and preventable exploits.

A.PASSWORD: Console Users will either choose strong passwords as defined by their organizational guidance or, if Active Directory integration is used for the Parity Console, that the Active Directory enforces strong password policies.

4.3.2 Physical Assumption

A.LOCATE: The TOE server and remote database will be located within controlled access facilities that will prevent unauthorized physical access.

4.3.3 Connectivity Assumptions

A.CONNECT: The TOE will be deployed in an environment where external data stores reside on a trusted network and client systems have the capability to communicate with the Parity Application Server intermittently if not persistently.

A.CONTEXT: Clients deployed on remote systems will be installed in a context that prevents Client Users from disabling, removing, altering, or reconfiguring the client.

A.CLIENTID: Client Users are identified to the TOE via the host name of their workstation and/or the Active Directory credentials used to authenticate to it.

A.CRYPTOGRAPHY: Data can be encrypted or decrypted using secure algorithms at the request of the TSF.

5 Security Objectives

5.1 Security Objectives for the TOE

The following security objectives are capabilities provided by the TOE in order to mitigate the threats defined against it and the threats against which it has been designed to protect

O.ACCESS: The TOE will provide measures to authorize users to access TSF data or resources protected by the TOE once the user has been authenticated. User authorization is based on access rights configured by the authorized users of the TOE and the binding of external attributes to subjects recognized by the TSF.

O.ALERT: The TOE will provide measures for defining audit events which represent noteworthy violations and will send notifications when these events occur.

O.AUDIT: The TOE will provide measures for recording security relevant events that will assist Console Users in detecting misuse of the TOE or activity on client systems.

O.AUTH: The TOE will provide mechanisms to identify and authenticate Console Users requesting access to the TSF prior to allowing any TSF-mediated actions except for the receiving of alerts.

O.CRYPTOGRAPHY: The TOE will provide a mechanism to hash files which reside in the Operational Environment.

O.DISPLAY_BANNER: The TOE will display an advisory warning regarding use of the TOE.

O.EAVESDROPPING: The TOE will use its environment to protect TSF data in transit using certificates imported from a certificate authority.

O.MANAGE: The TOE will provide authorized administrators with the resources to manage and monitor user accounts, resources, and security information relevant to the TOE.

O.MUTUAL: The TOE will provide a mechanism for establishing mutual authentication between itself and the GSR as a prerequisite for allowing for the transfer of TSF data.

O.PROTECT: The TOE will provide measures for the server to validate the integrity of a client and for the client to validate the integrity of a server.

O.WHITELIST: The TOE will provide access control enforcement for files, processes, registry values, and devices which reside on client systems to prevent unauthorized access to or modification of system assets.

5.2 Security Objectives for the Operational Environment of the TOE

The following security objectives for the Operational Environment must be satisfied in order for the TOE to fulfill its security objectives.

OE.ADMIN: One or more authorized administrators will be assigned to install, configure and manage the TOE.

OE.CLIENTID: Client Users are identified to the TOE via the host name of their workstation and/or the Active Directory credentials used to authenticate to it.

OE.AUTH: The Operational Environment will provide measures to uniquely identify Client Users and will authenticate their claimed identity prior to granting a user access to the resources protected by the TOE.

OE.ATTRIBUTES: The Operational Environment will provide an external directory of users which defines attributes based on existing organizational structure.

OE.CRYPTOGRAPHY: The environment will encrypt TSF data in transit using imported certificates from a certificate authority.

OE.FILESYS: The file systems of machines in the Operational Environment protect the binaries which comprise the TOE and the external data stores which it uses to enforce the SFRs.

OE.NOEVIL: Administrators and PowerUsers of the TOE are not careless, wilfully negligent, or hostile, and they will abide by the instructions provided by applicable guidance documentation.

OE.PASSWORD: Console Users will either choose strong passwords as defined by their organizational guidance or, if Active Directory integration is used for the Parity Console, that the Active Directory enforces strong password policies.

OE.LOCATE: The TOE server and remote database will be located within controlled access facilities that will prevent unauthorized physical access.

OE.SYSTIME: The operating environment will provide reliable system time.

6 Extended Security Functional Requirements Definition

6.1 Extended Security Functional Requirements Definition

The following table provides a summary of the Extended Security Functional Requirements implemented by the TOE.

Security Functional Class	Security Functional Component
Security audit (FAU)	FAU_GEN_EXT.1.1 File Inventory
	FAU_GEN_EXT.1.2 Detect Devices
	FAU_STG_EXT.3 Action in Case of Possible Audit Data Loss
Identification and Authentication (FIA)	FIA_UAU_EXT.8 Generation of Authentication Credentials
	FIA_UID_EXT.2 External entity identification before any action
	FIA_UAU_EXT.2 External entity authentication before any action

Table 6-1: Extended Security Functional Requirements for the TOE

6.1.1 Security Audit (FAU)

The FAU_GEN_EXT family defines requirements for recording the occurrence of security relevant events that take place in the Operational Environment but are observed by the TSF. This family identifies the level of auditing, enumerates the types of events that shall be auditable by the TSF, and identifies the minimum set of audit-related information that should be provided within various audit record types. The FAU_GEN_EXT.1 requirements are being added because CC part 2 lacks the ability to detect whether or not a file/device has been introduced. The generation of these types of records is specific to the TOE and does not refer to the generic audit record generation as described in CC Part 2.

FAU_GEN_EXT.1.1 The TSF shall be able to report whether a file has been identified on the network based on the following logical elements: **[*assignment: logical elements*]**

FAU_GEN_EXT.1.2 The TSF shall be able to report whether a device has been identified on the network based on the following physical or logical elements: **[*assignment: physical or logical elements*]**

Management: FAU_GEN_EXT.1

There are no management activities foreseen.

Audit: FAU_GEN_EXT.1

There are no management activities foreseen.

The FAU_STG_EXT family defines the requirements for the TSF to be able to create and maintain a secure audit trail. Stored audit records refers to those records within the audit trail, and not the audit records that have been retrieved (to temporary storage) through selection. The FAU_STG_EXT.3 requirement is being added because CC Part 2 lacks an audit storage requirement that demonstrates the ability of the TSF to write audit data when a storage threshold has been reached. The generation of these types of records is specific to the TOE and does not refer to the generic audit record generation as described in CC Part 2.

FAU_STG_EXT.3.1 The TSF shall clone audit records to [***assignment: location***] when the audit trail is increased.

Management: FAU_STG_EXT.3

There are no management activities foreseen.

Audit: FAU_STG_EXT.3

There are no auditable events foreseen.

6.1.2 Identification and Authentication (FIA)

The FIA_UAU_EXT family defines the types of user authentication mechanisms supported by the TSF. The FIA_UAU_EXT extended requirements were created to specifically address the ability of the TOE to facilitate the transfer of user credentials.

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

FIA_UAU_EXT.8.1 The TSF shall generate [***assignment: authentication data***] to support user authentication.

Management: FIA_UAU_EXT.8

There are no management activities foreseen.

Audit: FIA_UAU_EXT.8

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

a) Not specified: All reauthentication attempts.

The FIA_UID_EXT family defines the conditions under which each external entity shall be required to identify itself before performing any other actions that are to be mediated by the TSF and which require user identification.

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

Management: FIA_UID_EXT.2

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

a) the management of the external entity identities.

Audit: FIA_UID_EXT.2
The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Not specified: All use of the external entity identification mechanism, including the external entity identity provided.

6.1.3 Protection of the TSF (FPT)

The FPT_ITC_EXT family defines requirements of the TSF to utilize its operational environment to provide inter-TSF confidentiality between the TOE and trusted IT products.

FPT_ITC_EXT.1.1 The TSF shall leverage third-party cryptographic suites to protect all TSF data transmitted from the TSF to another trusted IT product from unauthorised disclosure during transmission.

Management: FPT_ITC_EXT.1

There are no management activities foreseen.

Audit: FPT_ITC_EXT.1

There are no auditable events foreseen.

The FPT_ITT_EXT family defines requirements of the TSF to utilize its operational environment to provide inter-TSF confidentiality between separate parts of the TOE.

FPT_ITT_EXT.1.1 The TSF shall leverage third-party cryptographic suites to protect TSF data from [selection: *disclosure, modification*] when it is transmitted between separate parts of the TOE.

Management: FPT_ITT_EXT.1

There are no management activities foreseen.

Audit: FPT_ITT_EXT.1

There are no audit activities foreseen.

6.1.4 Trusted Path/Channels (FTP)

The FTP_TRP_EXT family defines requirements of the TSF to utilize its operational environment to provide confidentiality between the TOE and users.

FTP_TRP_EXT.1 Trusted Path through the OE

Hierarchical to:	No other components.
FTP_TRP_EXT.1.1	The TSF shall leverage third-party cryptographic suites to provide a communication path between itself and [selection: *remote, local*] users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [selection: *modification, disclosure, [assignment: other types of integrity or confidentiality violation]*].
FTP_TRP_EXT.1.2	The TSF shall permit [selection: *the TSF, local users, remote users*] to initiate communication via the trusted path.
FTP_TRP_EXT.1.3	The TSF shall require the use of the trusted path for [initial user authentication, [selection: *initial user authentication, [assignment: other services for which trusted path is required]*].

Management: FTP_TRP_EXT.1

There are no management activities foreseen.

Audit: FTP_TRP_EXT.1

There are no auditable events foreseen.

6.2 Extended Security Assurance Requirements Definition

There are no extended Security Assurance Requirements in this ST.

7 Security Functional Requirements

7.1 Security Functional Requirements for the TOE

Security Function	
Security Audit (FAU)	FAU_ARP.1 Security Audit Automatic Response FAU_GEN.1 Audit Data Generation FAU_GEN.2 User Identity Association FAU_GEN_EXT.1 Object Inventory FAU_SAA.1 Security Audit Analysis FAU_SAR.1 Audit Review FAU_SAR.3 Selectable Audit Review FAU_STG.3 Action in Case of Possible Audit Data Loss FAU_STG_EXT.3 Action in Case of Possible Audit Data Loss
Cryptographic Support (FCS)	FCS_COP.1 Cryptographic Operation
User Data Protection (FDP)	FDP_ACC.1(1) Subset Access Control FDP_ACC.1(2) Subset Access Control FDP_ACF.1(1) Security Attribute Based Access Control FDP_ACF.1(2) Security Attribute Based Access Control
Identification and Authentication (FIA)	FIA_ATD.1 User Attribute Definition FIA_UAU.1 Timing of authentication FIA_UAU.5 Multiple Authentication Mechanisms FIA_UAU_EXT.2 External Entity Authentication Before Any Action FIA_UAU_EXT.8 Generation of Authentication Credentials FIA_UID.2 User Identification Before Any Action FIA_UID_EXT.2 External Entity Identification Before Any Action FIA_USB.1 User-Subject Binding
Security Management (FMT)	FMT_MOF.1 Management of Security Functions Behavior FMT_MTD.1 Management of TSF Data FMT_SMF.1 Specification of Management Functions FMT_SMR.1 Security Roles
Protection of the TSF (FPT)	FPT_ITC_EXT.1 Inter-TSF confidentiality during transmission FPT_ITT_EXT.1 Internal TOE TSF data transfer
TOE Access (FTA)	FTA_TAB.1 Default TOE Access Banners
Trusted Path/Channels (FTP)	FTP_TRP_EXT.1 Trusted Path through the OE

Table 7-1: Security Functional Requirements for the TOE

7.1.1 Class FAU: Security Audit

FAU_ARP.1 Security Audit Automatic Response

Hierarchical to: No other components.

FAU_ARP.1.1 The TSF shall take **[*the following action: send SMTP message***] upon detection of a potential security violation.

Dependencies: FAU_SAA.1 Potential violation analysis

Application Note: *The users to be messaged can include a user who receives all alerts and one or more users who receive alerts based on type. The SMTP message can be sent to any valid email address and is not necessarily bound to a Console User.*

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 following event types:***

Server's auditable events: Server Management, Session Management, Discovery, Computer Management, Policy Management, Policy Enforcement, Parity Knowledge, General Management

Client's auditable events: Sever Management, New Files, Security, Software Approval, Console User, Computer Management, Alert, Internal Events, Software Banning, Executed Files, Custom Rules, Device Control, Error, Parity Knowledge, Baseline Drift, Parity Reporter, Registry Rules, Memory Rules].

Dependencies: FPT_STM.1 Reliable time stamps

Application Note: *Startup and shutdown of the audit functions is synonymous with startup and shutdown of the TOE.*

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

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [***event subtype, description, priority, object identity if applicable***].

c) [***the specific data types per event that should be captured are defined in Tables 9-3 and 9-4***].

Application Note: *Subject identity can refer to any or all of the following: computer name, IP address, username*

FAU_GEN.2 User identity association

Hierarchical to: No other components.

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

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

FAU_GEN_EXT.1 Object Inventory

Hierarchical to: No other components.

FAU_GEN_EXT.1.1 The TSF shall be able to report whether a file has been identified on the network based on the following logical elements:

[***First seen name, First seen date, Last updated, First seen path, First seen computer, Extension, Global state, Global flags, Installer/Updater, File Prevalence, Publisher, Company, Product Name, Product Version, Description, File Type, SHA-256, MD5, SHA-1, Trust rating, Threat level, Category, Policy Specific States, History***]

FAU_GEN_EXT.1.2 The TSF shall be able to report whether a device has been identified on the network based on the following physical or logical elements:

[***Vendor, Device Name, Device State, First seen host, First seen date, Last modified by, Date modified***]

Dependencies: FPT_STM.1 Reliable Time Stamps

FAU_SAA.1 Security Audit Analysis

Hierarchical to: No other components.

FAU_SAA.1.1 The TSF shall be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the enforcement of the SFRs.

FAU_SAA.1.2 The TSF shall enforce the following rules for monitoring audited events:

a) Accumulation or combination of [***a configurable number or percentage of monitored assets by time range of file propagations and/or file blocks***] known to indicate a potential security violation;

b) ***[The occurrence of auditable events listed in FAU_GEN.1.1 if configured to do so].***

Dependencies: FAU_GEN.1 Audit data generation

FAU_SAR.1 Audit review

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation

FAU_SAR.1.1 The TSF shall provide [***all console users via dashboard or Portlet functionality***] with the capability to read [***all information collected by FAU_GEN.1***] from the audit records.

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

FAU_SAR.3 Selectable Audit Review

Hierarchical to: No other components.

FAU_SAR.3.1 The TSF shall provide the ability to apply [***selective display***] of audit data based on [***time range, top X events***].

Dependencies: FAU_SAR.1 Audit review

FAU_STG.3 Action in Case of Possible Audit Data Loss

Hierarchical to: No other components.

FAU_STG.3.1 The TSF shall [***delete a configurable percentage of the oldest records***] if the audit trail exceeds [***a configurable age or number of records value***].

Dependencies: FAU_STG.1 Protected audit trail storage

FAU_STG_EXT.3 Action in Case of Possible Audit Data Loss

Hierarchical to: No other components.

FAU_STG_EXT.3.1 FAU_STG_EXT.3 The TSF shall clone audit records to ***[backup source]*** when the audit trail is increased.

Dependencies: FAU_STG.1 Protected audit trail storage

Application Note: The intent of this iteration is to issue the claim that if desired, the recording of audit data can be mirrored to an additional source. The recording of audit data to the additional source is not dependent on the current number of records. Once external auditing has been enabled, all events are mirrored.

7.1.2 Class FCS: Cryptographic Support

FCS_COP.1 Cryptographic Operation

Hierarchical to: No other components.

FCS_COP.1.1 The TSF shall perform [***hashing***] in accordance with a specified cryptographic algorithm [***SHA-1, SHA-256***] and cryptographic key sizes [***160-bit, 256-bit***] that meet the following: [***no standard***].

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]

Application Note: The TSF also performs the non-cryptographic MD5 hashing function.

7.1.3 Class FDP: User Data Protection

FDP_ACC.1(1) Subset Access Control

Hierarchical to: No other components.

FDP_ACC.1.1(1) The TSF shall enforce the [***client access policy***] on [***Client Users or machine***].

Dependencies: FDP_ACF.1 Security attribute based access control

FDP_ACC.1(2) Subset Access Control

Hierarchical to: No other components.

FDP_ACC.1.1(2) The TSF shall enforce the [***console access policy***] on [***Console Users***].

Dependencies: FDP_ACF.1 Security attribute based access control

FDP_ACF.1(1) Security Attribute Based Access Control

Hierarchical to: No other components.

FDP_ACF.1.1(1) The TSF shall enforce the [***client access policy***] to objects based on the following: [***object identity and attributes as defined in table 7-4 below***].

Application Note: Subject identity refers to component of user DN, local SID on client system, or security group.

Object	Attributes	Comments
File	- Object (target) filename, path, hash, publisher - Subject (process) filename, path, hash, publisher, user, approval state - Operation: read, modify, execute - State (pending, approved, blocked, acknowledged)	- Hash = Cryptographic hash of file contents, are MD5, SHA1, and SHA256 - Publisher = Digital Signature on file - User = is interpreted as user name, SID and group membership - Approval State = Whether subject is approved, banned or pending, and whether subject has been promoted (i.e. is authorized to create new content) - State = files exist in the following states: • Pending – used to categorize files which have just been discovered and are neither approved nor blocked • Approved – used to categorize files which are allowed to be executed • Blocked – used to categorize files which are not allowed to be executed • Acknowledged – used to acknowledge the presence of a pending file while deferring judgment on whether it should be approved or blocked [The above descriptions apply to entire table] - Modify operation encapsulates: create, delete, write

Process	- Object (target) filename, path, hash, publisher - Subject (process) filename, path, hash, publisher, user, approval state - Operation: terminate, suspend/resume, impersonate, modify, create	- The object here is "memory", as in memory protection - Therefore a memory rule is defined for a process * Note: Memory rules are not supported by Windows 2003 because of OS limitations
Registry	- Object (target) path - Subject (process) filename, path, hash, publisher, user, approval state - Operation: modify	- Path = For registry, this is: Root key, subkey and value - Modify operation encapsulates: create, delete, write
Device	- Object (target) device vendor, device name - Operation: read, write, execute	- Only applies to USB devices with a file system

Table 7-2: Client access policy attributes

FDP_ACF.1.2(1) The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [***the subject and the operation they are attempting to perform against the object are compared to applicable file rules in a policy which applies to the subject or software and device rules which apply to all policies. The outcome of these rule decisions is one of the following states: approved, banned, pending, prompt, and ignore. The SecCon level applied to the client is defined by the policy and determines what action to take based on the state***].

Application Note: *The SecCon level applied to the client can differ depending on whether or not the client is able to connect to the server.*

Application Note: *Software and Registry Rules are ranked according to priority. If two rules contradict, the higher-ranked rule is the one which applies.*

FDP_ACF.1.3(1) The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [***if the SecCon level applied to the client is "Agent Disabled", the policy is not invoked***].

FDP_ACF.1.4(1) The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [***no additional rules***].

Dependencies: FDP_ACC.1 Subset access control

FDP_ACF.1(2) Security Attribute Based Access Control

Hierarchical to: No other components.

FDP_ACF.1.1(2) The TSF shall enforce the [***console access policy***] to objects based on the following: [***subject identity and (AD group membership or explicit group assignment)***].

FDP_ACF.1.2(2) The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [***group is determined by either AD group membership or explicit group assignment. The operations allowed by each role are statically defined***].

FDP_ACF.1.3(2) The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [***no additional rules***].

FDP_ACF.1.4(2) The TSF shall explicitly deny access to objects based on the following additional rules: [***no additional rules***].

Dependencies: FDP_ACC.1 Subset access control

7.1.4 Class FIA: Identification and Authentication

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: [***Console Users: username, password, role***].

Dependencies: No dependencies.

FIA_UAU.1 Timing of authentication

Hierarchical to: No other components.

FIA_UAU.1.1 The TSF shall allow [***alerts to be received***] on behalf of the user to be performed before the user is authenticated.

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

Dependencies: FIA_UID.1 Timing of Identification.

FIA_UAU_EXT.2 External entity authentication before any action

Hierarchical to: No other components.

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

Dependencies: FIA_UID_EXT.2 External entity identification before any action.

FIA_UAU.5 Multiple Authentication Mechanisms

Hierarchical to: No other components.

FIA_UAU.5.1 The TSF shall provide [***Console Users with basic authentication, Active Directory integration, Timed Override authentication***] to support user authentication.

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the [***following rules: Console Users are authenticated either with basic authentication or using Active Directory authentication credentials, depending on whether or not the Parity Console is configured to use Active Directory integration. Timed Override authentication is only used for direct temporary access to the Parity Client itself***].

Dependencies: No dependencies.

FIA_UAU_EXT.8 Generation of Authentication Credentials

Hierarchical to: No other components.

FIA_UAU_EXT.8.1 The TSF shall generate [***timed override token***] to support user authentication.

Dependencies: FIA_UAU.5 Multiple Authentication Mechanisms

Application Note: The purpose of this authentication key is to authenticate Console Users to client systems through use of the Timed Override utility.

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 be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Dependencies: No dependencies.

FIA_UID_EXT.2 External entity identification before any action

Hierarchical to: No other components.

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

Dependencies: No dependencies.

FIA_USB.1 User-Subject Binding

Hierarchical to: No other components.

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: [***hostname or a Client User's defined DN component or for Console Users an AD group***].

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [***for Console Users, Active Directory Policy Mappings are defined in the console to map AD attributes. For Client Users, hostname, an individual instance of the client access policy can be applied to individual computers in the console***].

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [***if the "set automatic policy for existing computers" check box is selected for an instance of the client access policy, the Console Users' AD mapping overrides the hostname mapping if both are present***].

Dependencies: FIA_ATD.1 User attribute definition

7.1.5 Class FMT: Security Management

FMT_MOF.1 Management of Security Functions Behavior

Hierarchical to: No other components.

FMT_MOF.1.1 The TSF shall restrict the ability to [***modify the behavior of***] the functions [***rules which comprise the client access policy***] to [***Administrators and PowerUsers***].

Dependencies: FMT_SMR.1 Security roles
FMT_SMF.1 Specification of Management Functions

FMT_MTD.1 Management of TSF data

Hierarchical to: No other components.

FMT_MTD.1.1 The TSF shall restrict the ability to [***change_default, query, modify, delete, create***] the [***objects in table 7-5 below***] to [***the groups specified in table 7-5***].

Dependencies: FMT_SMR.1 Security roles
FMT_SMF.1 Specification of Management Functions

Application Note: PowerUsers cannot create Administrators

Object	Operation(s)	Administrator	PowerUser	ReadOnly	Comments
Dashboard	Query, Modify, Delete, Create	X	X	X	None
Dashboard	Change Default, Share	X	X		Only Administrators and PowerUsers can Modify, Delete, Create shared dashboards, or Change Default for global default dashboard
Portlet	Query	X	X	X	None
Portlet	Modify, Delete, Create	X	X		None
Baseline Drift	Query	X	X	X	None
Baseline Drift	Modify, Delete, Create	X	X		None
System Backup	BackUp, Restore	X			The TOE can be fully backed up and restored, including the configuration, system settings, file database, and event log.
Snapshot	Query	X	X	X	None
Snapshot	Modify, Delete, Create	X	X		None
Saved Views	Query	X	X	X	Saved Views apply to: Baseline Drift, Events, and Files

Saved Views	Modify, Delete, Create	X	X		None
Events	Query	X	X	X	None
Files	Query	X	X	X	None
Computers	Query	X	X	X	Advanced options are not visible to ReadOnly users
Computers	Modify, Delete	X	X		A computer cannot be "created"; it appears automatically when it registers with the server.
Client Access Policies	Query	X	X	X	1. Client Access Policies include the following options in the Parity Console: "Policies", "Software Rules", "Registry Rules" "Device Rules". 2. Client Access Policies include the ability to define polices, rules, and trusted directories, users, groups, publishers, and updaters.
Client Access Policies	Change Default, Modify, Delete, Create	X	X		Change Default only applies to Parity "Policies", not any other Client Access Policy. The assignment of policies to Active Directory objects is considered a "Modify" operation.

Meters	Query	X	X	X	None
Meters	Modify, Delete, Create	X	X		None
Alerts	Query	X	X	X	None
Alerts	Modify, Delete, Create	X	X		None
Console Users	Query, Modify, Delete, Create	X	X		Shown as "Login Accounts" in the Parity Console. Note: PowerUsers can only create/modify/delete ReadOnly users. Note: Active Directory users and their Console Access Policies are managed outside of the TOE.
System Administration	Query, Modify	X			This includes: All server configuration options, all event (audit) pruning and exporting options, default alert email options, security settings and, licensing
Own password	Modify	X	X		None
Timed Override	Modify	X	X		This is a Console User performing an action using the client.

Table 7-3: Policy attributes for groups

FMT_SMF.1 Specification of Management Functions

Hierarchical to: No other components.

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [***see table 7-5***].

Dependencies: No dependencies.

Application Note: *Based upon the configuration of the TOE for actions specified in table 7-5, the TOE will perform functionality without additional user interaction. This is most notable with the TOE's backup functionality.*

FMT_SMR.1 Security Management Roles

Hierarchical to: No other components.

FMT_SMR.1.1 The TSF shall maintain the roles [***Console Users (Administrators, PowerUsers, ReadOnly), Client Users***].

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

Dependencies: FIA_UID.1 Timing of identification

7.1.6 Class FPT: Protection of the TSF

FPT_ITC_EXT.1 Inter-TSF confidentiality during transmission through the OE

Hierarchical to: No other components.

FPT_ITC_EXT.1.1 The TSF shall leverage third-party cryptographic suites to protect all TSF data transmitted from the TSF to another trusted IT product from unauthorised disclosure during transmission.

Application Note: *The remote trusted IT products are as follows: Active Directory, Parity Knowledge, and SQL database. The assurance provided for Parity Knowledge is mutual certificate-based authentication.*

Application Note: *The functionality provided to the TOE for this requirement is derived from the operational environment. When data transmission occurs, the TOE reaches out to the operational environment to provide the protected communication.*

Dependencies: No dependencies.

FPT_ITT_EXT.1 Basic Internal TSF Data Transfer Protection through the OE

Hierarchical to: No other components.

FPT_ITT_EXT.1.1 The TSF shall leverage third-party cryptographic suites to protect TSF data from [disclosure, modification] when it is transmitted between separate parts of the TOE.

Dependencies: No dependencies.

Application Note: *The functionality provided to the TOE for this requirement is derived from the operational environment. When data transmission occurs, the TOE reaches out to the operational environment to provide the protected communication.*

7.1.7 Class FTA: TOE Access

7.1.7.1 FTA_TAB.1 Default TOE access banners

Hierarchical to: No other components.

FTA_TAB.1.1 Before establishing a user session, the TSF shall display an advisory warning message regarding unauthorised use of the TOE.

Dependencies: No dependencies.

Application Note: *The access banner applies whenever the TOE will provide a prompt for identification and authentication (e.g., administrators). The intent of this requirement is to advise users of warnings regarding the unauthorized use of the TOE and to provide the Security Administrator with control over what is displayed (e.g., if the Security Administrator chooses, they can remove banner information that informs the user of the product and version number).*

7.1.8 Class FTP: Trusted Path/Channels

FTP_TRP_EXT.1 Trusted Path through the OE

Hierarchical to: No other components.

FTP_TRP_EXT.1.1 The TSF shall leverage third-party cryptographic suites to provide a communication path between itself and [***remote***] users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [***disclosure, modification***].

Application Note: *Remote users refers to remote Console Users in this instance.*

FTP_TRP_EXT.1.2 The TSF shall permit [remote users] to initiate communication via the trusted path.

FTP_TRP_EXT.1.3 The TSF shall require the use of the trusted path for [initial user authentication, [***management of the TSF***].

Dependencies: No dependencies.

Application Note: *The functionality provided to the TOE for this requirement is derived from the operational environment. When data transmission occurs, the TOE reaches out to the operational environment to provide the protected communication.*

7.2 Operations Defined

The notation, formatting, and conventions used in this security target (ST) are consistent with version 3.1 of the Common Criteria for Information Technology Security Evaluation. All of the components in this ST are taken directly from Part 2 of the CC except the ones noted with "_EXT" in the component name. Font style and clarifying information conventions were developed to aid the reader.

The CC permits four functional component operations: assignment, iteration, selection, and refinement to be performed on functional requirements. These operations are defined in Common Criteria, Part 1 as:

7.2.1 Assignments Made

An assignment allows the specification of parameters and is specified by the ST author in [***italicized bold text***].

7.2.2 Iterations Made

An iteration allows a component to be used more than once with varying operations and is identified with the iteration number within parentheses after the short family name, FAU_GEN.1(1), FAU_GEN.1(2).

7.2.3 Selections Made

A selection allows the specification of one or more items from a list and is specified by the ST author in [underlined text].

7.2.4 Refinements Made

A refinement allows the addition of details and is identified with "Refinement:" right after the short name. ~~The old text is shown with a strikethrough~~ and ***the new text is specified by italicized bold and underlined text***.

8 Security Assurance Requirements

This section identifies the Security Assurance Requirement components met by the TOE. These assurance components meet the requirements for EAL2 augmented with ALC_FLR.1 and ASE_TSS.2.

8.1 Security Architecture

8.1.1 Security Architecture Description (ADV_ARC.1)

ADV_ARC.1.1D: The developer shall design and implement the TOE so that the security features of the TSF cannot be bypassed.

ADV_ARC.1.2D: The developer shall design and implement the TSF so that it is able to protect itself from tampering by un-trusted active entities.

ADV_ARC.1.3D: The developer shall provide a security architecture description of the TSF.

ADV_ARC.1.1C: The security architecture description shall be at a level of detail commensurate with the description of the SFR-enforcing abstractions described in the TOE design document.

ADV_ARC.1.2C: The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.

ADV_ARC.1.3C: The security architecture description shall describe how the TSF initialization process is secure.

ADV_ARC.1.4C: The security architecture description shall demonstrate that the TSF protects itself from tampering.

ADV_ARC.1.5C: The security architecture description shall demonstrate that the TSF prevents bypass of the SFR-enforcing functionality.

ADV_ARC.1.1E: The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

8.1.2 Functional Specification with Complete Summary (ADV_FSP.2)

ADV_FSP.2.1D: The developer shall provide a functional specification.

ADV_FSP.2.2D: The developer shall provide a tracing from the functional specification to the SFRs.

ADV_FSP.2.1C: The functional specification shall completely represent the TSF.

ADV_FSP.2.2C: The functional specification shall describe the purpose and method of use for all TSFI.

ADV_FSP.2.3C: The functional specification shall identify and describe all parameters associated with each TSFI.

ADV_FSP.2.4C: For each SFR-enforcing TSFI, the functional specification shall describe the SFR-enforcing actions associated with the TSFI.

ADV_FSP.2.5C: For SFR-enforcing TSFIs, the functional specification shall describe direct error messages resulting from processing associated with the SFR-enforcing actions.

ADV_FSP.2.6C: The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

ADV_FSP.2.1E: The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.2.2E: The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

8.1.3 Architectural Design (ADV_TDS.1)

ADV_TDS.1.1D: The developer shall provide the design of the TOE.

ADV_TDS.1.2D: The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

ADV_TDS.1.1C: The design shall describe the structure of the TOE in terms of subsystems.

ADV_TDS.1.2C: The design shall identify all subsystems of the TSF.

ADV_TDS.1.3C: The design shall describe the behavior of each SFR-supporting or SFR-non-interfering TSF subsystem in sufficient detail to determine that it is not SFR-enforcing.

ADV_TDS.1.4C: The design shall summarise the SFR-enforcing behavior of the SFR-enforcing subsystems.

ADV_TDS.1.5C: The design shall provide a description of the interactions among SFR-enforcing subsystems of the TSF, and between the SFR-enforcing subsystems of the TSF and other subsystems of the TSF.

ADV_TDS.1.6C: The mapping shall demonstrate that all behavior described in the TOE design is mapped to the TSFIs that invoke it.

ADV_TDS.1.1E: The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_TDS.1.2E: The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

8.2 Guidance Documents

8.2.1 Operational User Guidance (AGD_OPE.1)

AGD_OPE.1.1D The developer shall provide operational user guidance.

AGD_OPE.1.1C The operational user guidance shall describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

AGD_OPE.1.2C The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.

AGD_OPE.1.3C The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

AGD_OPE.1.4C The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_OPE.1.5C The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.

AGD_OPE.1.6C The operational user guidance shall, for each user role, describe the security measures to be followed in order to fulfill the security objectives for the operational environment as described in the ST.

AGD_OPE.1.7C The operational user guidance shall be clear and reasonable.

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

8.2.2 Preparative Procedures (AGD_PRE.1)

AGD_PRE.1.1D The developer shall provide the TOE including its preparative procedures.

AGD_PRE.1.1C The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.

AGD_PRE.1.2C The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of

the operational environment in accordance with the security objectives for the operational environment as described in the ST.

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

AGD_PRE.1.2E The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.

8.3 Lifecycle Support

8.3.1 Authorization Controls (ALC_CMC.2)

ALC_CMC.2.1D: The developer shall provide the TOE and a reference for the TOE.

ALC_CMC.2.2D: The developer shall provide the CM documentation.

ALC_CMC.2.3D: The developer shall use a CM system.

ALC_CMC.2.1C: The TOE shall be labeled with its unique reference.

ALC_CMC.2.2C: The CM documentation shall describe the method used to uniquely identify the configuration items.

ALC_CMC.2.3C: The CM system shall uniquely identify all configuration items.

ALC_CMC.2.1E: The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

8.3.2 CM Scope (ALC_CMS.2)

ALC_CMS.2.1D: The developer shall provide a configuration list for the TOE.

ALC_CMS.2.1C: The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; and the parts that comprise the TOE.

ALC_CMS.2.2C: The configuration list shall uniquely identify the configuration items.

ALC_CMS.2.3C: For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.

ALC_CMS.2.1E: The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

8.3.3 Delivery Procedures (ALC_DEL.1)

ALC_DEL.1.1D The developer shall document procedures for delivery of the TOE or parts of it to the consumer.

ALC_DEL.1.2D The developer shall use the delivery procedures.

ALC_DEL.1.1C The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to the consumer.

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

8.3.4 Flaw reporting procedures (ALC_FLR.1)

ALC_FLR.1.1D The developer shall document flaw remediation procedures addressed to TOE developers.

ALC_FLR.1.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.1.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.1.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.1.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users. Evaluator action elements:

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

8.4 Security Target Evaluation

8.4.1 Conformance Claims (ASE_CCL.1)

ASE_CCL.1.1D The developer shall provide a conformance claim.

ASE_CCL.1.2D The developer shall provide a conformance claim rationale.

ASE_CCL.1.1C The conformance claim shall contain a CC conformance claim that identifies the version of the CC to which the ST and the TOE claim conformance.

ASE_CCL.1.2C The CC conformance claim shall describe the conformance of the ST to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.

ASE_CCL.1.3C The CC conformance claim shall describe the conformance of the ST to CC Part 3 as either CC Part 3 conformant or CC Part 3 extended.

ASE_CCL.1.4C The CC conformance claim shall be consistent with the extended components definition.

ASE_CCL.1.5C The conformance claim shall identify all PPs and security requirement packages to which the ST claims conformance.

ASE_CCL.1.6C The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.

ASE_CCL.1.7C The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PPs for which conformance is being claimed.

ASE_CCL.1.8C The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PPs for which conformance is being claimed.

8.4.2 Extended Components Definition (ASE_ECD.1)

ASE_ECD.1.1D The developer shall provide a statement of security requirements.

ASE_ECD.1.2D The developer shall provide an extended components definition.

ASE_ECD.1.1C The statement of security requirements shall identify all extended security requirements.

ASE_ECD.1.2C The extended components definition shall define an extended component for each extended security requirement.

ASE_ECD.1.3C The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.

ASE_ECD.1.4C The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.

ASE_ECD.1.5C The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements can be demonstrated.

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

ASE_ECD.1.2E The evaluator shall confirm that no extended component can be clearly expressed using existing components.

8.4.3 ST Introduction (ASE_INT.1)

ASE_INT.1.1D The developer shall provide an ST introduction.

ASE_INT.1.1C The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.

ASE_INT.1.2C The ST reference shall uniquely identify the ST.

ASE_INT.1.3C The TOE reference shall identify the TOE.

ASE_INT.1.4C The TOE overview shall summarize the usage and major security features of the TOE.

ASE_INT.1.5C The TOE overview shall identify the TOE type.

ASE_INT.1.6C The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.

ASE_INT.1.7C The TOE description shall describe the physical scope of the TOE.

ASE_INT.1.8C The TOE description shall describe the logical scope of the TOE.

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

ASE_INT.1.2E The evaluator shall confirm that the TOE reference, the TOE overview, and the TOE description are consistent with each other.

8.4.4 Security Objectives (ASE_OBJ.2)

ASE_OBJ.2.1D The developer shall provide a statement of security objectives.

ASE_OBJ.2.2D The developer shall provide a security objective rationale.

ASE_OBJ.2.1C The statement of security objectives shall describe the security objectives for the TOE and the security objectives for the operational environment.

ASE_OBJ.2.2C The security objectives rationale shall trace each security objective for the TOE back to threats countered by that security objective and OSPs enforced by that security objective.

ASE_OBJ.2.3C The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.

ASE_OBJ.2.4C The security objectives rationale shall demonstrate that the security objectives counter all threats.

ASE_OBJ.2.5C The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.

ASE_OBJ.2.6C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

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

8.4.5 Security Requirements (ASE_REQ.2)

ASE_REQ.2.1D The developer shall provide a statement of security requirements.

ASE_REQ.2.2D The developer shall provide a security requirement rationale.

ASE_REQ.2.1C The statement of security requirements shall describe the SFRs and the SARs.

ASE_REQ.2.2C All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.

ASE_REQ.2.3C The statement of security requirements shall identify all operations on the security requirements.

ASE_REQ.2.4C All operations shall be performed correctly.

ASE_REQ.2.5C Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.

ASE_REQ.2.6C The security requirements rationale shall trace each SFR back to the security objectives for the TOE.

ASE_REQ.2.7C The security requirements rationale shall demonstrate that the SFRs meet all security objectives for the TOE.

ASE_REQ.2.8C The security requirements rationale shall explain why the SARs were chosen.

ASE_REQ.2.9C The statement of security requirements shall be internally consistent.

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

8.4.6 Security Problem Definition (ASE_SPD.1)

ASE_SPD.1.1D The developer shall provide a security problem definition.

ASE_SPD.1.1C The security problem definition shall describe the threats.

ASE_SPD.1.2C All threats shall be described in terms of a threat agent, an asset, and an adverse action.

ASE_SPD.1.3C The security problem definition shall describe the OSPs.

ASE_SPD.1.4C The security problem definition shall describe the assumptions about the operational environment of the TOE.

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

8.4.7 TOE Summary Specification (ASE_TSS.2)

ASE_TSS.2.1D The developer shall provide a TOE summary specification.

ASE_TSS.2.1C The TOE summary specification shall describe how the TOE meets each SFR.

ASE_TSS.2.2C The TOE summary specification shall describe how the TOE protects itself against interference and logical tampering.

ASE_TSS.2.3C The TOE summary specification shall describe how the TOE protects itself against bypass.

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

ASE_TSS.2.2E The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.

8.5 Tests

8.5.1 Analysis of Coverage (ATE_COV.1)

ATE_COV.1.1D: The developer shall provide evidence of the test coverage.

ATE_COV.1.1C: The evidence of the test coverage shall show the correspondence between the tests in the test documentation and the TSFIs in the functional specification.

ATE_COV.1.1E: The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

8.5.2 Functional Tests (ATE_FUN.1)

ATE_FUN.1.1D The developer shall test the TSF and document the results.

ATE_FUN.1.2D The developer shall provide test documentation

ATE_FUN.1.1C The test documentation shall consist of test plans, expected test results and actual test results.

ATE_FUN.1.2C The test plans shall identify the tests to be performed and describe the scenarios for performing each test. These scenarios shall include any ordering dependencies on the results of other tests.

ATE_FUN.1.3C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE_FUN.1.4C The actual test results shall be consistent with the expected test results.

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

8.5.3 Independent Testing (ATE_IND.2)

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

ATE_IND.2.1C The TOE shall be suitable for testing.

ATE_IND.2.2C The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

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

ATE_IND.2.2E The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

ATE_IND.2.3E The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

8.6 Vulnerability Assessment

8.6.1 Vulnerability Analysis (AVA_VAN.2)

AVA_VAN.2.1D The developer shall provide the TOE for testing.

AVA_VAN.2.1C The TOE shall be suitable for testing.

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

AVA_VAN.2.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA_VAN.2.3E The evaluator shall perform an independent vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design and security architecture description to identify potential vulnerabilities in the TOE.

AVA_VAN.2.4E The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

9 TOE Summary Specification

Bit9 Parity is comprised of multiple applications that span client-server architecture which combine to monitor the Operational Environment tracking executable files, their prevalence, and execution. The TOE also monitors the identification and authentication, authorization, and management activities of the Operational Environment as well as the management events of the TOE.

9.1 TOE Summary Functions

This section describes the security functions provided by the TOE.

9.1.1 Security Audit

The TOE collects, aggregates, and reports on IT activity, and generates alerts when file/device information changes. These functions are performed by the TOE by collecting log data, using a database to store the logs over an established timeframe, and presenting the logs via reports and queries. In addition, the TOE generates audit records for its own startup and shutdown and all user actions on the TOE. Authorized users are able to select the notification mechanism for all auditable events. The TOE can be configured to send an email notification if a threshold of events is exceeded. A threshold is based on a combination of number of events and timeframe (for example, 100 events over a 30 minute timeframe). For events generated by users, the user's unique identification is also stored within the logged audit.

Audit data is maintained in both the Parity Client and the Parity Application Server. At predefined times or when conditional events require, information is shared between the Parity Application Server and the Parity Clients. The following sections provide examples of information that is collected.

9.1.1.1 Drift

Parity's Live Inventory of files on a network gives Console Users the ability to measure baseline drift, the difference between a baseline of files and the current files on a specified target. Differences may also be assessed against physical and logical devices. These differences are available as a baseline drift report that a Console User can view either in detail in dynamic tables or as graphic charts on a Parity dashboard. Baseline drift reports provide not only simple numbers of file differences but also risk analyses related to those changes.

Once it is set up, a drift report runs automatically every few hours, giving Console Users an up-to date record of changes in a file inventory. Console Users can create different baseline drift reports for different targets and baselines, and Bit9 provides some reports pre-configured for use. Only Power Users and Administrators can create, modify and delete reports.

9.1.1.2 Audit Data Generation in Parity Application Server

Because the Parity Application Server and Parity Reporter components are Windows Services, any start/stop/restart of the service will generate an event in the Windows Application Event Log. The Parity Console is an IIS web site and any start/stop/restart of the IIS web service will generate an event in the Windows Application Event Log. Parity's audit capabilities cannot be disabled so the startup and shutdown of the TOE is considered to be synonymous with startup and shutdown of the audit capabilities. All auditable client events are cached on each client, so stopping the Parity Server Service does not stop the auditing of Parity Client activity.

Additionally, the Parity Server maintains its own event table for tracking audit, security, and functional information. The following table provides a list of the event types which are stored in this table.

Event Type	Event Subtype
Server Management	Server shutdown
	Server restart
	Server config modified
	Server backup started
	Server backup failed
	Server backup missed
	Database server reached specified limit
	Old events were deleted
	Database verification error
	File tracking disabled
	Server backup stopped
	Server upgrade succeeded
	Server upgrade failed
	Server config list error
	AD lookups are slow
	License added
	License error
	License warning
	New certificate generated
	New certificate generation failed
	Certificate imported
	Certificate import failed
	Strong SSL communications changed
	Certificate expiring
	Certificate expired
	Agent certificate expired
	Agent communications disabled
	Common Name mismatch

	Database error Communication error System error Parity Knowledge connection lost Parity Knowledge connection restored Parity Knowledge proxy set Parity Knowledge proxy cleared Reporter error Reporter task execution
Discovery	New publisher found File group created New pending file to computer Banned file written to computer New file on network First execution on network New device found Device attached Device detached

Computer Management	Computer added Computer deleted Computer modified CLI password reset Agent shutdown Agent restart Agent policy changed Agent SecCon changed Agent policy updated Agent upgraded Agent synchronization started Agent synchronization finished Agent bulk state change finished Agent bulk state change requested Agent deleted events Agent requires upgrade Cache check start Cache check complete Cache check error Agent synchronization requested Temporary SecCon override Temporary SecCon restore Agent uninstalled File receive error File process error Installer rescan requested Automatic synchronization scheduled CLI executed Unauthorized computer registration
General Management	Alert created Alert deleted Alert modified Alert triggered Alert reset Meter created Meter deleted Meter modified Baseline drift report created Baseline drift report deleted Baseline drift report modified Baseline drift report generated Baseline drift report generation is slow

	Snapshot created Snapshot deleted Snapshot modified
Policy Management	AD rules changed AD rules loaded Custom rule created Custom rule deleted Custom rule modified Device approval created Device approval removed File approval created File approval modified File approval deleted File ban created File ban deleted File ban modified File group local approval File local approval File remove local approval Group remove local approval Install package created Memory rule created Memory rule deleted Memory rule modified Policy created Policy deleted Policy file tracking changed Policy modified Publisher approval created Publisher approval removed Publisher modified Registry rule created Registry rule deleted Registry rule modified Trusted directory check Trusted directory created Trusted directory deleted Trusted directory import Trusted directory modified Trusted directory scan

	Trusted User added Trusted User deleted Updater disabled Updater enabled
Parity Knowledge	Potential risk file detected Malicious file detected
Policy Enforcement	Access approved (memory rule) Access block (memory rule) Execution allowed (custom rule) Execution allowed (trusted user) Execution block (banned file) Execution block (custom rule) Execution block (network file) Execution block (pending file) Execution block (removable media) Execution block (still analyzing) File access error File approved (block and ask) File approved (local approval) File approved (publisher) File approved (trusted user) File approved (updater) Metered execution Process demoted Read block (removable media) Report access (memory rule) Report execution (removable media) Report execution block (banned file) Report read (removable media) Report write (custom rule) Report write (registry rule) Report write (removable media) Self protection block Write approved (custom rule) Write block (custom rule) Write block (registry rule) Write block (removable media)

Session Management	Console user created Console user deleted Console user login Console user logout Console user modified Multiple failed logins

Table 9-1: Auditable events for Application Server

9.1.1.3 Alerts Generated from the Parity Server

Any alert shown on the Alerts page, whether provided with Parity or created by Parity Administrator or PowerUser, can be considered an alert class. Each time conditions exist that meet the triggering condition of that alert class, an alert instance occurs. For some alert classes, it is only possible to have one instance. For example, there is only one Parity database, and so Parity Database Limit Alert can have only one instance at a time. For other classes, there can be many instances simultaneously. For example, there might be multiple malicious files on a network, and so there could be multiple Malicious File Detected alert instances.

When any triggered instances of an alert class exist, Parity sends an alert email for every alert instance. Instances are defined as distinct cases that match the alert conditions. In the case of malicious files, for example, if the same malicious file shows up 20 times before an alert is reset, it only counts as one instance. But if 20 different malicious files appear before the alert notification is reset, each one counts as an instance and each one generates a new email message to alert subscribers.

9.1.1.4 Audit Data Generation in Parity Client

Since the Parity Client is a Windows Server, any start/stop/restart of the service will generate an event in the Windows Application Event Log.

Additionally, the Parity Client maintains its own event table for tracking audit, security, and functional information. The following table provides a list of the event types which are stored in this table.

Event Type	Event Subtype
Computer Management	Agent shutdown Agent restart Cache check complete Cache check error Temporary SecCon Override Temporary SecCon Restore Agent uninstalled File receive error File process error Agent resync requested

	Cache check start CLI executed
Discovery	Device attached Device detached New device found
Policy Enforcement	Access approved (memory rule) Access block (memory rule) Execution allowed (trusted user) Execution block (banned file) Execution block (custom rule) Execution block (network file) Execution block (pending file) Execution block (removable media) Execution block (still analyzing) Execution allowed (custom rule) File access error File approved (block and ask) File approved (local approval) File approved (publisher) File approved (trusted user) File approved (updater) Metered execution Process demoted Read block (removable media) Report access (memory rule) Report execution (removable media) Report execution block (banned file) Report read (removable media) Report write (custom rule) Report write (registry rule) Report write (removable media) Self protection block Write approved (custom rule) Write block (custom rule) Write block (registry rule) Write block (removable media)
Policy Management	Trusted directory check Trusted directory scan

Table 9-2: Auditable events for Parity Client

9.1.1.5 Audit Storage in Parity Application Server

Audit Storage for the TOE is handled by a Microsoft SQL Server or SQL Express database. This database must be acquired and configured separately from the process of acquiring the TOE because it resides in the Operational Environment. In the evaluated configuration NT Authentication will be used for connection between the TOE and SQL Server/SQL Express.

When using a Microsoft SQL Server database, there is no maximum allowable size for the audit storage (events table). The maximum is determined by the space available on the database server, as well as any size/growth restrictions placed on the database or tables by the SQL Server administrator.

When using a SQL Express database, there is a total maximum database size limit of 4GB. When the total database size reaches 3GB, an Alert event is automatically generated once a day. An optional email alert can also be enabled for this event.

There are two options provided to control when (or if) events will be pruned (deleted): based on date/time (e.g. events older than X days), and based on total event count (e.g. when the total number of events is greater than X). Either of these pruning options can be defined by an Administrator from the Parity Console. When one of these conditions occurs, the Administrator can specify how much (as a percentage of the total events) should be deleted.

- In Standard SQL Server configuration, 10% of the events will be pruned when the total number of events reaches 10 million.
- In a SQL Express configuration, 10% of the events will be pruned when the total number of events reaches 1 million.

When events are automatically deleted due to the specified pruning options, an "old events were deleted" event is generated with "Notice" priority. This event details how many events were deleted.

In addition, the customer can choose to maintain periodic SQL Server backups of the Parity database, using options available to SQL Server and/or third-party software. These backups could be of the complete database or just the events table. Audit logging can also be mirrored to a remote syslog server or secondary SQL database for redundancy purposes or if the TOE's pruning capabilities are inconsistent with an organization's data retention policies.

9.1.1.6 Audit Storage in Parity Client

All Parity Client generated events are cached locally by the Parity Client until it can connect to the Parity Application Server. Once sent to the server, the event is removed from the Client.

The client event table can grow large only if a client remains disconnected from the server for an extended period of time (or indefinitely). As with the server events, there

are two options provided to control when (or if) events should be pruned on the Client: based on date/time (e.g. events older than X seconds), and based on total event count (e.g. when the total number of events is greater than X). There is also a property to control how many events should be deleted upon one of these triggers (as a percentage of the total events).

Whenever events are deleted from a client queue, based on the user specified options, an "Agent deleted events" event with "Notice" priority will be generated denoting how many events have been deleted.

There are also options to set event-type specific auto-deletion (aging) rules. For example, "delete all 'Agent restarted' events after 60 seconds if not connected to the server", in addition to having general aging rules that apply to all events. All of the Client side event pruning options can be specified globally (for all Clients) and/or per machine.

9.1.1.7 Selectable Audit Review

The primary mechanism for reviewing audit data within the Parity Console is through the use of Dashboards. A Dashboard is a webpage displayed on the Parity Console which consists of a series of portlets, each of which provides summary information that can help Console Users manage the security of networked computers and the files on them. Most portlets display a specific type of information from the Parity database, but they also might display news feeds or other information from an outside URL. Portlets are organized by information type as follows:

- **Events:** These portlets display event information from the Parity Server database, such as the number of blocked file blocked executions over a period of time or alerts that have been triggered
- **Baseline Drift**: These portlets display the results of baseline drift analysis in Parity, such as daily drift of software from a baseline or the computers with the most deviation from the baseline.
- **Computers:** These portlets display information available in Parity about the computers on your system, such as the number of computers in each operating system or the number of computers at each SecCon level.
- **Files:** These portlets show information about the files on Parity-managed computers, such as the number of newly seen files over time or the category (browsers, utilities, messaging, etc.) of the files on the system.
- **Other:** These portlets may be a display of security news from a website, system-created "action" portlets such as the emergency lockdown button, or combinations of different types of information from the Parity database.

All Console Users can view the Events view from the Parity Console. The Events view contains both audit and product-level information. Parity Console access can be granted

based on Active Directory settings or by explicitly created login accounts from the console.

Administrators and PowerUsers have the ability to define their own custom Dashboards and portlets. A custom Dashboard is defined by determining the layout of the page and what portlets will appear on them. Custom portlets allow for different ways to represent the default data. For example, if a portlet defines a chart view for data like baseline deviations over time, a custom portlet may be used to change the length of time to be represented on the x-axis. In addition, custom portlets may include HTML pages or RSS feeds. However, Console Users should be aware that accessing any third-party web content may pose a security risk.

Events cannot be edited by any user. The Events view from within the Parity Console can be filtered and set to display only specified fields, but it cannot be restricted. Any user with access to the console can display any available field or record. Moreover, any user may selectively display auditable data. The selections may be based on a time range or position in the events queue.

Access to either of the two external formats – the external SQL database and the syslog server – is restricted based on the administration options provided by Microsoft or third-party administration tools. Because these are components of the Operational Environment, they are not protected by the TSF.

9.1.1.8 Data Contained in a Parity Event Log

Parity event contains the following information:

Field Information	Definition
Date/Time	Stored in UTC format, taken from the machine where the event originated and displayed in local time. This is based on the time settings of the Parity Application Server.
Source (e.g. Computer Name) and IP Address, if applicable	Source is stored internally as an ID, displayed as a string with a link to the full computer details
User Name, if applicable	User name is stored internally as an ID, displayed as a full string (e.g. DOMAIN\USER) with a link to the user's Active Directory details (if available)
Description	A formatted text string explaining the event and any action taken. The explanation of the event includes its type and subtype. The list of these events can be found in Table 9-1 and 9-2.
Priority	One of the following possible values (in rough order of severity): • Notice • Info • Warning • Error

	• Critical

Table 9-3: Information contained in Parity event log

In addition, events may contain other fields as appropriate.

Field Information	Definition
File Path/File Name	The path and filename of the event (e.g. a blocked execution or a new file discovered on the network)
File Hash	The SHA256 hash of the file referred to by the event
Policy	The policy currently assigned to the Source, if the source is a computer
Process	The full path and filename of the process running that triggered the event
Installer/Root Hash	The filename and SHA256 hash of the parent or root file of the file referred to by the event

Table 9-4: Additional information for security related events in Parity event log

9.1.2 Encrypted Communications

Remote users establish a session with the Parity Application Server using a web-based GUI that is secured via HTTPS. This secured path is used for user authentication and management of the TOE by authorized users.

The Parity Server utilizes certificates generated by trusted CAs in order to protect communications between itself and the Parity Client. For communications to components in the Operational Environment, the TOE relies on their cryptographic facilities.

Additionally, communication between the TOE and remote users is protected from modification or disclosure via HTTPS. This trusted path is established when a Console User accesses the Parity Console.

In all three data transmission instances described above, the TOE will reach out to the operational environment to provide the protected communication.

9.1.2.1 From Parity Application Server to Parity Clients

All communications between the Parity Application Server and the Parity Clients occur over a secure and encrypted TCP/IP port (default is 41002). The communications are encrypted and authenticated using a server-side certificate. A Parity Administrator can import any valid certificate as a PKCS#12 file for use in server-client communications, using the Parity Console.

The server will not accept any incoming connections if the Parity Client does not have a valid certificate. In addition, the client will not attempt a connection to the server unless the server name matches the Common Name of the public portion of the certificate transmitted by the server.

There is also an "Enable Verification" option that can be enabled by a Parity Administrator on the Parity Console to enforce strong SSL communications. When checked, each client will verify the authenticity of the server certificate (using WinVerifyCert) before attempting communications (i.e. it must be a valid, non-expired certificate that has been added to the client's trusted root certificates).

9.1.2.2 From Parity Clients to Parity Console

Manifests, as used in Parity, are created by a Parity Client enforcing a Trusted Directory. They are then transported to the Application Server and automatically imported into the Server subsystem.

When a Parity Client has been assigned a directory to monitor, it generates manifest files that are uploaded to the Parity Console via a secure HTTPS connection. Once uploaded, the Parity Application Server will read the contents of the manifest. Manifests are wrapped in a digest header containing the SHA256 checksum of its contents. This header is validated by the server before importing. The manifest also contains the unique client ID from the machine on which it was generated. If that ID is not known to the Parity Application Server, the file will be rejected.

9.1.2.3 From Parity Reporter to Parity Knowledge (GSR)

The Parity Reporter communicates with the Parity Knowledge web service through dual certificates (mutual authentication). When a connection is established, the Parity Reporter passes in its Activation Key and its Server ID. If either is invalid or do not match the data stored internally on the Bit9 servers, the connection will be rejected.

When the Parity Reporter service is installed, it installs a certificate into Trusted People section of the Local Machine certificate store.

This client side certificate is used to authenticate Parity Reporter with the Parity Knowledge (GSR) web service at services.bit9.com.

The Reporter certificate is used to negotiate symmetric encryption for all traffic between the servers.

9.1.2.4 From Parity Application Server to Active Directory

When authenticating any Parity Console User, or assigning Parity policies to any Parity Client (if Active Directory Policy Mappings are used), the Parity Application Server uses LDAP to query the Active Directory services. The account under which Parity Application Server is running must have at least read permissions to all domains that will be used for either console logins or policy assignments.

The user/password validation with LDAP occurs with ADS_SECURE_AUTHENTICATION and ADS_USE_ENCRYPTION. Active Directory will use Kerberos, and possibly NTLM, to authenticate the client; and encryption will be used if supported by the ADSI provider.

9.1.2.5 Administration and Reporting via Web Console

All administration and reporting is done via a browser based web console. The console, in turn, communicates to the Parity Server Service (for authentication and administration), and directly to the Parity database (for reporting).

Access to the Parity Console is done via a browser over a secure HTTPS connection.

If desired, an administrator can apply their own certificate of any key size and type to the Parity Console web site using the administrative features of IIS or command-line utilities provided by Windows, such as certutil.exe and netsh.exe.

9.1.2.6 Internal TOE TSF data transfer

All files analyzed by Parity Clients across the system are hashed for identification and tracking purposes. All files are hashed using three distinct hashing algorithms and compared using the same distinct algorithms that the file was hashed against. The three distinct hashing algorithms are:

- MD5 – Using Microsoft's Crypt API, using the CALG_MD5 algorithm provided by the Microsoft Base Cryptographic Provider.
- SHA1 – Uses Microsoft's Crypt API, using the CALG_SHA1 algorithm provided by the Microsoft Base Cryptographic Provider.
- SHA256 – If Windows XP SP3 or later (W2K, W2K3, Vista, W2K8, W7), using Microsoft's Crypt API, using the CALG_SHA_256 algorithm provided by the Microsoft Base Cryptographic Provider. If earlier than XP SP3, an algorithm taken from the FIPS 180-3 document, http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf is used.

Files are hashed so that information about attempted file accesses from the Parity Clients can be efficiently transmitted to the Parity Server. The use of all three hashing algorithms allows for multiple verifications of the identity of the file being tracked.

The cryptography used in this product has not been FIPS-certified, nor has it been analyzed or tested to conform to cryptographic standards during this evaluation. All cryptography has only been asserted as tested by the vendor.

9.1.3 User Data Protection

The User Data Protection function of the TOE, specifically the definition and enforcement of Client Access Policies, represents the primary functionality of the TOE. It works in the following manner:

- Policies are defined which apply to specific users, groups, and/or computers in the Operational Environment
- Policies define the overall security posture of a system and contain a collection of rules which explicitly allow or deny certain operations

- Parity Clients are installed on all necessary systems
- When actions occur on these systems, the Parity Clients will examine the subject, object, and operation and compare them against their defined policies in order to determine if these actions should be allowed or denied

The following sections provide additional details regarding the specifics of the User Data Protection functionality of the TOE.

9.1.3.1 Creating Policies

Parity policies are named groups of protection rules shared by targeted groups of computers running the Parity Client — every computer running a Parity Client must belong to a policy. A Console User may create policies based on security and organizational requirements.

Each installer automatically assigns a policy to each client it installs. The Parity Server may also assign a policy based on Active Directory data for the user and/or computer running the client each time the computer with the Parity Client connects to the server.

Policies enable Console Users to organize computers running Parity Client into groups with common security requirements and member characteristics. For example, Console Users can create policies based on departmental affiliations like sales, marketing, or other organizational relationships. A single policy may be appropriate if Console Users are setting a single, company-wide operating standard for all computers on a network, but typically Console Users will create multiple policies. Policies normally are assigned to computers, not users, although Active Directory information can be used to vary policy by user. Any computer has only one policy at a time, regardless of the number of logged on users. Policies have the following parameters: device control setting (for how device rules are handled)**,** SecCon, Active Directory mapping, and computer mapping.

9.1.3.2 Policies on Client Computers

Client computer systems become visible to the Parity Server when Console Users install and run the Parity Client on them. When Console Users download and install the client, an initialization process begins, delivering information about the computer and its files to the Parity Server.

For each security policy Console Users create, Parity automatically generates a common *client installer*. The installer includes the Parity Client itself and also specifies the policy assigned to the computer and the Parity Server address. If Console Users have not chosen to use AD-based policy assignment, this same client installer is used for every computer controlled by the policy. If Active Directory mapping is used, policies can be tailored to apply to specific users, groups, and computers as already defined in the environmental Active Directory implementation. If this is the case, different client installers will be generated based on the users or groups that will utilize each client or, in the case of computer mapping, different client installers will be generated for each computer.

As soon as the Parity Client software is installed, file initialization begins. The client software takes an inventory of all executable files on the client computer's fixed disks (but not removable drives) and creates a set of three hashes of each file. When a computer with the Parity Client first connects to the server, the client sends each set of hashes to the Parity Server to determine its status and update the server's file inventory. Files on a computer at initialization receive a *local* state of Approved unless they already have been identified and globally banned or banned by policy on the Parity Server.

Both during and after initialization, files not in the File Catalog (i.e., not seen before) on a Parity Server are assigned a *global* state of Pending and added to the catalog. After initialization, newly discovered files get an initial *local* status of Pending, and are included in the New Pending Files list, a Saved View on the Files page.

9.1.3.3 Rules

The Parity Server maintains a central database of unique files (determined by hashes) for all executable files or devices tracked on network computers running the Parity Client. Each file/device has a Global State, shown below, which indicates how it is to be treated on Parity-managed computers. The next sections will elaborate on rules for: software and registry, devices, and files.

State	Description
Approved	Allowed to execute on all computers.
Banned	Banned by hash, and not allowed to execute on any computer running in Visibility and Control mode.
Approved by policy	Allowed to execute on all computers in one or more policies.
Banned by policy	Banned by hash from execution on all computers in one or more policies (in Visibility and Control mode).
Pending	Not Approved or Banned (globally or by policy). Parity blocks or permits execution of a pending file depending on the SecCon level of the Policy of the computer attempting the execution.

Table 9-5: Global File State in the Parity database

Several key feature groups work together in Parity to secure computers on a network. At the heart of this security capability is the ability to classify files according to their *state.* Groups of security rules, called policies, can control how different groups of computers treat files in different states. For the file rules and software and registry rules, the rules themselves define to what policy they belong; device rules are global and not mapped to a policy.

Note: The rules defined for a process (memory rules) are not supported by Windows 2003 because of OS limitations.

9.1.3.3.1 Software and Registry Rules

Software approval ensures that users of computers running Parity Client can freely install and run known-good applications regardless of the Parity security settings and SecCon enforcement level in effect. Parity supports several complementary methods for approving software on computers. Based on the method(s) Console Users select, Parity permits installation of approved software on all computers, on computers in selected policies, or on individually selected computers.

Software Rules can whitelist files and binaries based on the following conditions:

- Trusted Directory – all files in a specific directory are allowed to be executed. Parity generates manifest files of the directory's contents, and all matching files in the directory are whitelisted. This is useful for allowing periodic rollouts from a trusted deployment server.
- Trusted User/Group – a user or group (either as defined by Active Directory or the Windows OS) is trusted to run anything on that system.
- Trusted Publisher – if a trusted publisher is specified, all files digitally signed by that publisher are automatically whitelisted.
- Trusted Updater – if a trusted updater is specified, all files provided by that updater are automatically whitelisted. Trusted updaters include:
    - Adobe: Acrobat Reader 8.0, Acrobat Reader 9.0, FrameMaker, Illustrator, InCopy, InDesign, PageMaker, Photoshop, Premiere Pro
    - Allow Printer Installations
    - BigFix Enterprise Client 7.0
    - CA ITM
    - Google Desktop Search
    - Java
    - LanDesk 8
    - McAfee: ePO 4.0/4.5, VirusScan Enterprise 8.5
    - Microsoft: .NET 2.0/3.0/3.5, SCCM
    - Mozilla: Firefox, Thunderbird
    - SMS Software Approval
    - Sophos Anti-Virus 7.0
    - Spybot – Search&Destroy 1.4
    - Symantec – Antivirus 10.0/11.0
    - Trend Micro OfficeScan 8.5
    - WebEx for Firefox
    - WebEx for Internet Explorer
    - Windows Defender
    - Windows Update

- Local Approval – local approval allows for a per-machine override to a policy for specific exemptions. For example, a specific computer may need to install a specific application which has not been whitelisted elsewhere.
- File Approvals – file approvals (and file bans) allow Administrators and PowerUsers to specify specific approved files which apply to the policy based on hash values.

Parity provides a related capability for a Console User to create Registry Rules, which control what happens when there is an attempt to make any changes at specified registry paths, and if a Console User chooses, by specified users and/or processes.

Software and Registry rules are ranked on the Parity Console in their order of priority. A Console User can change the rankings on the page which displays the rules. Rules are evaluated based on this order. If two rules contradict (for example an allow rule and a block rule for the same file), the higher-ranked rule is the one which is enforced.

9.1.3.3.2 Device Rules

Each Parity Client can detect a variety of removable devices that contain file systems. The Device Rule, which are global and not mapped to a policy are manipulated from the Device Rules page; the page provides an inventory of all removable devices detected by Parity Clients running on managed computers. Console Users can approve any device in the table, and Console Users can remove approval from approved devices.

In addition to its approval state, each device in the devices tables includes device Vendor and Name (if available), the date/time it was first seen on the network, and the name of the computer on which it was first seen. Console Users also can add columns for the date the last modifications were made to files on the device, and who made the changes.

9.1.3.3.3 File Rules

The Files tab of the Software Rules page shows all of the approvals and bans created at a site for individual files. Approvals and bans can be global, applying to all computers, or they can be applied (or mapped) to computers in selected policies. Bans block file executions for affected computers in Visibility and Control mode, report an event for computers in Visibility Only mode, and do nothing for computers in Agent Disabled mode.

In addition to explicit approvals and bans, Parity allows a Console User to define Custom Rules for allowing or blocking file execution or writing at specified locations (trusted directories), and if a Console User chooses, by specified users and/or processes (trusted publishers and updaters). These Custom Rules may be used for performance optimizations, file integrity control, and creation of a trusted file path for software distribution.

9.1.3.4 Subset Access Control

Parity policies are named groups of protection rules shared by designated computers running the Parity Client -- every computer running a Parity Client must belong to a policy. A Console User creates policies based on security and organizational requirements. For example, policy membership might be based on functional group (marketing, customer service); location; or type of computer (laptop, desktop, server).

Each policy has its own Parity Client installer, which is automatically generated on the server when the policy is created. Each installer automatically assigns a policy to each client it installs, and the policy can be reassigned once the Parity Client connects to the Parity Application Server.

9.1.3.5 Security Attribute Based Access Control

On the Parity Client, the "SecCon", or security condition, is an attribute that is part of the Parity policy assigned to each client. SecCon determines how the Parity Client should behave when unknown files are encountered. The following are the SecCons:

SecCon Level	Name	Description
80	Agent Disabled	Parity no longer tracks any device, file, process, or registry activity. It continues to run, and can communicate with the server, but is effectively disabled. This is the only SecCon where the agent can be uninstalled without authorization.
60	Visibility Only	Parity will track all operations, but will not block any operation other than its own self-protection.
40	Monitor Mode	Parity will track all operations and will enforce (block) any file bans, device block rules, and other rules that explicitly block an operation. Unknown (or pending) files are allowed to execute.
35	Local Approval Mode	This is a special SecCon that cannot be assigned to a policy; it can only be applied on a machine-by-machine basis. When in Local Approval Mode, all files that appear (are written locally) on the machine will be automatically locally approved (and therefore allowed to execute), unless they have been explicitly banned by a rule.
30	Block and Ask	Parity will track all operations, enforcing all policies. When an unknown (or pending) file is executed, a dialog box (the "Notifier") is presented to the user, where they can choose to Block or Allow the operation. If they choose to allow it, that file becomes locally approved (if it is a local file).
20	Lockdown	Parity will track all operations, enforcing all policies. Unknown (or pending) files are blocked from executing.

Table 9-6: SecCons

Table 9-7 below provides a glance view of what operations are allowed or denied based on SecCon level.

Active Policy Settings	SecCon Levels				
	80 – Agent Disabled	60 – Visibility Only	40 - Monitor	30 – Block and Ask	20 - Lockdown
Block unanalyzed scripts and executables	off	permit	block	block	block
Block unapproved scripts	off	permit	permit	block and ask	block
Block unapproved executables	off	permit	permit	block and ask	block
Deny executions from removable devices	off	permit	block	block	block
Deny writes to removable devices	off	permit	block	block	block
Report reads from removable devices	off	permit	permit and report	permit and report	permit and report
Block executables run from a network drive	off	permit	block	block	block
Block banned file hashes	off	permit	block	block	block
Block banned file names	off	permit	block	block	block

Table 9-7: Operations by SecCon

Every policy additionally contains an "offline SecCon." If the Parity Client cannot communicate with the Parity Console, the Parity Client independently enforces access at the level specified by the offline SecCon.

When enforcing access policies on the client, Parity determines the access allowed to objects based on the user DN, local SID, or security group along with process attributes, file attributes, registry location, device name, requested operations, and SecCon. Based on the defined policy, a client may access an object if the above attributes are defined and authorized by the TOE. Similarly, console access policies determine access to objects based on subject identity and group membership/assignment. The group is determined by either AD group membership or an explicit assignment. Operations are statically defined.

9.1.3.6 File Tracking

Parity tracks executable files and monitors their prevalence and execution. Unidentified files that have just appeared on the network receive a *pending* status. A file keeps its *pending* status until it becomes *approved* or *banned.* A pending file also can be *acknowledged,* which removes it from the list of new pending files but does not change its underlying pending status. Once a file is *approved,* it is allowed to execute on all systems but continues to be tracked.

After a network is under Parity control, Administrators and PowerUsers can approve new applications or patches using the approval methods that best suit an organization's software rollout procedures. Parity features several automatic approval methods (trusted directories, approved publishers, trusted users, and enabled updaters) that make it easy to approve new software without having to do it file-by-file. For example, Console Users can globally approve desktop software like Microsoft Office by putting it into a trusted deployment directory. Computers on the network would then be permitted to run Microsoft Office executables because the Parity Client recognizes these files as approved. Alternatively, Console Users can manually mark individual files as approved or banned.

Besides blocking unauthorized files, as shown in Table 9-7, Console Users can determine information about files such as the following:

- Whether a file exists on a computer on the network
- Which computers have the file
- Where and when the file first arrived in the network environment
- What is known about the source, category, trust level, and threat of the file
- Whether and when a file has executed, and on which computers
- Whether a file has propagated and, if so, whether it renamed itself
- How the inventory of files on computers has changed over time
- Whether certain USB storage devices exist on a network, when they first were discovered, and on what computer

By tracking information about files throughout the enterprise, Console Users can gain visibility into the activities of organizational units. This can be used for incident handling, proactive avoidance of malicious activity, or compliance management.

Field /Button	Description
First seen name	File name of the first file observed by Parity to have this signature.
First seen date	Exact time the first file with this signature was observed on a network computer, displayed in month-day-year-hour: minute format.
Last updated	Exact time this file last changed file status, displayed in month day-year-hour: minute format. Files change status whenever they are manually or implicitly approved or banned. The initial last-update time is recorded when a new file is first observed and placed in the pending state.
First seen path	Path of the first file observed by Parity to have this signature.

Field /Button	Description
First seen computer	Name of the computer on which the file was first observed. If a Console User subsequently deletes the first-seen computer from the system, it is no longer associated with the file and this field is blank. A Console User can click on this name to get the Computer Details page for this computer.
Extension	File extension of the first file observed by Parity to have this signature.
Global state	The current state of the file (Pending, Approved, Banned, Approved by policy or Banned by policy). If the file is Pending, a Console User can choose [Approve] or [Ban] it to change its global state. If it is Approved, a Console User can [Remove approval]; if it is Banned, a Console User can [Edit ban] to change the ban definition or [Remove ban].
Global flags	File-state metadata for use by Bit9 support engineers.
Installer/Updater	During file analysis, Parity determines whether the file is likely to be an installer or updater: Yes - File expands to create more files that require pre-approval or auto-approval (depending on whether the trusted top-level installer is approved for installation on a client computer or the Parity Server). A Console User can click [Mark as not installer] to change the value to No. No - File is non-expandable. A Console User can click [Mark as installer] to change the value to Yes.
File Prevalence	The number of computers on which this file exists. Three action links are associated with this field: [Find all instances] - Opens the Find Files page with the hashes for this file already filled in as the search criteria and shows all instances of the file in the Find Files results. [Add meter] - Opens the Add Software Meter page with the name and hashes of this file already filled in. A Console User can create a meter that will record the number of times this file executes on computers running Parity Clients. If a meter already exists for this file, the link changes to [Edit meter]. [Add prevalence alert] - Opens the Add Alert page with the Alert pre-named as “Prevalence of <filename>” and the file’s hashes used as the file specification. If the alert already exists for this file, the link changes to [Edit alert].
Publisher	If the file is digitally signed or was included in a digitally signed package, Parity displays the publisher (software manufacturer) of the associated application.
Company	The Company name (if provided) in the file metadata.
Product Name	The Product Name (if provided) in the file metadata.
Product Version	The Product Version (if provided) in the file metadata.
Description	The Description (if provided) in the file metadata.
File Type	One of the following: Application - Any executable (e.g. .exe or .com) except for Packages Supporting File - Any library loaded by an executable (e.g., .dll, .ocx, .sys) Package - Any installer (.exe with contents, such as a self-extracting zip or setup program) Script File - Any script or batch file (e.g., .bat, .vbs, .wsf) Other - Reserved for future types Unknown - Files reported by older Parity Clients that don’t provide file type information

Field /Button	Description
SHA-256	Hash (data signature) of the file created using Bit9's SHA-256 algorithm. SHA-256 is used internally as the preferred data signature for files tracked by Parity. SHA-256 hashes created in Parity may be identical to those created by other means. However, some files change their hash every time they are installed because they include date, location, or other context-specific information not relevant for tracking purposes. For files known to do this, Parity uses a special fuzzy hashing algorithm that eliminates this extraneous variation, and so shows every instance of such files on computers running Parity Clients to be identical. When this algorithm has been used, the hash is identified as "SHA-256". A Console User can search for files by hash using filters on the Files page, or on the Find Files page. The [Find all instances] link provides a way to do this directly from the File Details page.
MD5	MD5 is a widely used hashing algorithm. Bit9 provides this alternate hash in case a Console User or the system needs to identify the file against a list of published MD5 signatures.
SHA-1	SHA1 is another widely used hashing algorithm. Bit9 provides this alternate hash in case a Console User or the system needs to identify the file against a list of published SHA1 signatures.
Trust rating	Indicates the level of trust for the file based on Parity Knowledge service information such as file source, signatures. The trust rating is showing on a scale of 0 (none) to 10 (most trusted), along with a graphic meter reflecting this rating. The value of this field is a subjective assessment of the file's integrity. As an indication of whether the file appears to be safe based on information derived from Parity Knowledge service analysis, the trust value does not signify actual approval on the Parity server. However, files that are approved via the Parity trusted publisher mechanism are automatically flagged as trusted.
Threat level	If a Console User has configured Parity Knowledge service analysis, Parity automatically submits discovered files for threat analysis. Parity Knowledge service flags known malware with a red x icon. No flag indicates that the file was not recognized as malware, not necessarily that it is safe. Threat levels include: 0 - Clean 1 - Potentially malicious 2 - Malicious Unknown - Not identified
Category	If Console User has configured Parity Knowledge service analysis, this field shows the category this file is in (e.g., Entertainment, Hacking Tools, Instant Messaging, and Media Players). In some cases, the category is unknown.
Policy Specific States	Indicates ways in which the file is treated differently in particular policies. For example, if the file is under a policy-specific hash ban or approval, the policy name is shown here. If there is no policy specific treatment of the file, this section of the File Details page does not appear.
History	Indicates whether the file was identified on the first-seen computer during initialization or detected after initialization. Also indicates any approvals or bans applied to the file. Files detected after initialization are tracked as pending files until approved or banned, and may be viewed in the New Pending view on the Files page File Catalog tab.

Table 9-8: File Details

9.1.4 Identification and Authentication

9.1.4.1 User Attribute Definition

Client Users are authenticated by the underlying Operating System before they are allowed to access the TOE. The TOE requires these users to be identified before they perform any security relevant actions.

Although all TOE users are identified and authenticated with usernames and passwords, security attributes are not necessarily maintained within Parity. For the Active Directory accounts, the security attributes of username, password, and membership are maintained outside of Parity. For built-in Parity users, the security attributes username, password, and group may be stored within Parity. The TOE is capable of establishing a mapping between these attributes so that subjects in the environmental Active Directory can be bound to users of the TOE.

During a user-subject bind, the hostname or Client User's DN component or AD group is associated with subjects acting on behalf of a user. (The association uses the timed override token.) If the "set automatic policy for existing computers" check box is selected for an instance of the client access policy, the AD mapping overrides the hostname mapping if both exist.

9.1.4.2 Timing of Authentication

The only Parity Console page that can be accessed by a user prior to authentication is https://[server]/hostpkg page, which provides access to the Parity Client installation programs. No other page can be accessed and no action can be taken without user authentication. However, alerts may be sent to any valid email address. Because of this, the act of receiving an alert can be performed without authenticating to the TOE. Also note that since alerts can be sent to any valid email address, the recipient does not necessarily need to be a Console User. In this particular instance there is no mechanism to bind the user to the target of the alert.

9.1.4.3 Parity Console (from client browser)

When logging into the Parity Console, users are authenticated by one of two possible means (in order):

1. Against the user table in the Parity database (manually created users)
2. Through Active Directory authentication and membership

In both cases, the user is verified against their password.

In the latter case, the user must be a member of one of the defined Active Directory Bit9 permission groups.

9.1.4.4 Parity Knowledge

In order for the Parity Server to receive updates from Parity Knowledge, Parity Knowledge must first identify itself to the TOE and provide valid authentication

credentials. This is performed through a mutual exchange of certificates which were generated by a trusted CA and provided by the vendor. Once each server has validated the other's credentials, there is sufficient trust to exchange data.

9.1.5 Security Management

Bit9 Parity shall be able to associate a Console User with a level of authority on the Parity Console. The following are the privileges available to Console Users:

- Administrator,
- PowerUser
- ReadOnly

Parity "policies" and "rules" are the assets that define how Parity Clients will enforce (or not) security regarding file and registry access, modification, and execution. Both Administrators and PowerUsers can create, modify and delete any policy or rule. ReadOnly users cannot. All users can view details of audit information (event) via alerts.

The Parity Console menu bar facilitates security by linking features for monitoring network computers and the files on them, managing users, approving/banning software, approving some detachable devices, and configuring the Parity server.

As shown in Table 7-5, the permissions associated with the security functions in each group (role) are not editable. Privileges are not scoped, with the exception of Dashboards. A user can always view the details (definition) of their own dashboards, but a ReadOnly user cannot view the details of any other dashboards.

For all other assets, if a user can view one asset of that type, he/she can view all assets of that type.

The functionality available to the different roles is defined below.

User	Functionality
ReadOnly user	• Can view all assets (in grid/report view) in the system except for Login Accounts • Can only view the details of the following assets: Files, Computers, Users, Policies and Devices (and Dashboards created by that user) • (Details pages are used to create or edit different assets, or view the detailed properties of an asset.) • Can create their own dashboards, but can only place on them portlets that have already been defined. Their personal dashboards are the only asset in the system that they can edit or delete. No other asset (including Saved Views, Portlets, Policies, Rules and everything else) can be edited or deleted. • Has no access to any System Administration or configuration pages
PowerUser	Can do everything in the system except: • Cannot create, edit or delete Administrator or PowerUser login accounts

	(except their own account) • Has no access to any System Administration or configuration pages
Administrator	Can do everything in the system

Table 9-9: Role-based functionality

Note: PowerUsers and Administrators can perform modifications to the client access policy rules.

9.1.5.1 Timed Override

Occasionally, a system that is not connected to the network may need to be temporarily placed in a relaxed security state in order to make some critical update to that system. Because the system is not connected, it cannot be managed directly by the Parity Console. For these situations, the TSF includes a feature known as Timed Override.

The Timed Override utility is a program on the Parity Client that allows for a temporary local modification of a SecCon value. In order to use Timed Override, a Console User must specify a machine, duration, SecCon value, and override length. The Parity Console generates an authentication credential valid only for the target machine for the specified duration. If the Console User runs the TimedOverride.exe program on that machine and enters the credential before time expires, that machine's SecCon will be changed to the specified value for the specified length of the override.

9.1.6 Security Architecture

Through policies, SecCon and modes, and session cookies, Parity protects itself from tampering and bypass.

Parity Administrators and PowerUsers can create policies that protect specified directories from unauthorized tampering. For example, to prevent users from uninstalling an application, a Parity Administrator or PowerUser can specify a write policy that instructs Parity to block any changes to the application directory.

Parity Administrators and PowerUsers can create a directory policy that applies to a particular directory only when a particular process attempts to write or execute files there. In addition, Parity Administrators and PowerUsers can choose to apply a directory policy to all computers on a network, or only to computers within associated policies the Parity Administrators and PowerUsers select. When applied to a policy controlling a group of computers, a directory policy can override certain other settings.

Parity users are not permitted to uninstall the Parity Client while it is running.

9.1.6.1 SecCons and Modes

To protect Parity clients from attempted bypass by disconnecting the client machine from the Parity network, specific software installation procedures must be followed, as disconnected client machines are under lockdown (SecCon 20) protection.

To permit new applications to be installed on a selected computer under lockdown (SecCon 20) protection, Administrators or PowerUsers may temporarily relax protection. This is done by moving the computer into the predefined Local Approval policy for as long as it takes to complete software installation.

Because disconnected computers cannot be controlled directly from the Parity Server, Administrators or PowerUsers need a different way to instruct the client to make the transition to another SecCon. Parity provides a feature that generates a special code that can be entered on the client for a computer to switch its SecCon for a specified amount of time. The code is specific to one client, and it can be used only once. Administrators or PowerUsers can generate codes to switch a computer into any SecCon except *80 - Agent Disabled*, although this feature is primarily useful for temporary transitions to Local Approval mode.

Once the specified time for the override has elapsed, the computer is automatically restored to its original policy, at which point it continues to be able to run all files installed while it was at the relaxed SecCon level. Files run or installed while the computer was in local-approval mode are locally approved on the computer but continue to have a *global* state of pending.

9.1.6.2 Session Cookies that Prevent Spoofing

The client passes in a cookie when it first registers with the server. Every Parity Application Server is assigned a random unique ID, a 16-byte (128-bit) number that is converted to a base64 string with a token prefix - the result is a 48 character string. This server ID string is embedded into each Parity Client setup program and stored on each client.

When client registers with the server, it passes in a cookie. This cookie is the client's unique ID encoded with the server ID. This cookie can only be deciphered by a server using the same server ID string. If the cookie cannot be deciphered or is invalid, the connection is rejected by the server and a security warning event is logged.

9.1.6.3 Boot Protection

The Parity kernel driver loads after the first phase of Windows kernel initialization and after the file system driver for the boot volume is loaded. The specific ordering is based on other filter drivers registered as boot drivers and their associated load order group. Filter drivers in lower-altitude groups are loaded first. This all occurs before Ntoskrnl.exe is even loaded and before there are any other non-boot drivers or processes present.

The Parity kernel driver is officially assigned an altitude of 80800 and is in the Security Enhancer load order group (details are available at http://msdn.microsoft.com/en-us/library/ff549689(VS.85).aspx). As such, it loads early and it subsequently lies below most other filters (Antivirus, Backup, Virtualization, Encryption, etc.) and just above file system drivers. By definition, a filter installed at a higher altitude affects the operations

processed by Parity; a filter installed at a lower altitude affects the operations Parity generates.

Bypassing the invocation of the kernel component would require the presence of another boot driver or compromised kernel or an offline boot (such as Safe Mode, alternate boot volume, or recovery CD), each of which can be separately managed outside the mechanisms provided in Parity.

9.1.6.4 Runtime Protection

By default, regardless of the access control policy being implemented, the TOE will implement policy rules which prevent a user from modifying or disabling the TOE. File protection prevents write operations to the files and containing directory for the Parity installation directory (e.g. C:\Program Files\Bit9\Parity Agent) and the Parity data directory (e.g. C:\ProgramData\Bit9\Parity Agent). Registry protection prevents modification to the contents within the Parity and ParityDriver (e.g. HKLM\System\CCS\services\Parity and ParityDriver) services keys and the Parity system-wide software key (e.g. HKLM\Software\Bit9\Parity Agent). Process protection applies to the Parity.exe service process.

9.2 TOE Summary Specification Rationale

This section identifies the security functions provided by the TOE mapped to the security functional requirement components contained in this ST. This mapping is provided in the following table.

Security Function	
Security Audit (FAU)	FAU_ARP.1 Security Audit Automatic Response FAU_GEN.1 Audit Data Generation FAU_GEN.2 User Identity Association FAU_SAA.1 Security Audit Analysis FAU_SAR.3 Selectable Audit Review FAU_STG_EXT.3 Action in Case of Possible Audit Data Loss
Cryptographic Support (FCS)	FCS_COP.1 Cryptographic Operation
User Data Protection (FDP)	FDP_ACC.1(1) Subset Access Control FDP_ACC.1(2) Subset Access Control FDP_ACF.1 (1) Security Attribute Based Access Control FDP_ACF.1 (2) Security Attribute Based Access Control

Identification and Authentication (FIA)	FIA_ATD.1 User Attribute Definition FIA_UAU.1 Timing of authentication FIA_UAU.5 Multiple Authentication Mechanisms FIA_USB.1 User-Subject Binding FIA_UID.2 User Identification Before Any Action
Security Management (FMT)	FMT_MOF.1 Management of Security Functions Behavior FMT_MTD.1 Management of TSF Data FMT_SMF.1 Specification of Management Functions FMT_SMR.1 Security Roles
TOE Access (FTA)	FTA_TAB.1 Default TOE Access Banners
Protection of the TSF (FPT)	FPT_ITC_EXT.1 Inter-TSF confidentiality during transmission through the OE FPT_ITT_EXT.1 Basic Internal TSF Data Transfer Protection through the OE
Trusted Path/Channels (FTP)	FTP_TRP_EXT.1 Trusted Path through the OE

Table 9-10: Security Functional Requirements

9.2.1 Security Audit

The Security Audit function of the TOE enforces the FAU_ARP.1, FAU_GEN.1, FAU_GEN.2, FAU_SAA.1, FAU_SAR.3, FAU_GEN_EXT.1, and FAU_STG_EXT.3 requirements.

Parity collects security and system audit information. Administrators with proper privileges are able to monitor, alert, and report information about user activity.

The examination of the TSS showed that each of these requirements was successfully mapped to the SFRs listed above the information provided in the INT section of the ST.

The generation of audits (FAU_GEN.1) is provided in section 2.3.1.1 as well as in the TSS, section 9.1.1, 9.1.1.1, and 9.1.1.2. In addition to the generation of audits, audit privilege, more specifically an explanation in the use of the privilege in order to view the audited information is discussed in sections 9.1.1.5, 9.1.2.1, and 9.1.2.2. FAU_GEN.1.2 is then fulfilled in section 9.1.1.2 and 9.1.1.6 with the mapping of information audited in relation to the event that is occurring. Section 2.5.1 of the INT covers this information as well but in less detail. Sections 9.1.1.1 and 9.1.1.2 also discusses the information that can be audited based on event with the example of start up and shutdown of the TOE.

FAU_GEN_EXT.1 is discussed in sections 9.1.1.1, 9.1.3.3.1, and 9.1.3.6. Information in these sections pertains to the ability of the TOE to gather an inventory of objects in the Operational Environment.

FAU_STG_EXT.3 is covered in the TSS through sections 9.1.1.3 and 9.1.1.4. Sections 9.1.1.3 and 9.1.1.4 discuss deletion of the configurable percentage of the oldest records from the database and writing the newest records to a remote database.

FAU_SAA.1 and FAU_SAR.3 are covered in the TSS through sections 2.3.1.1, 9.1.1, 9.1.1.1, 9.1.1.2, 9.1.1.5 and 9.1.1.6. These sections discuss the types of reports/logs that are provided in the TOE. These sections demonstrate the use of scoping to apply restrictions on auditing. Additionally, Section 9.1.1.1 outlines the various forms of reports that a security administrator with the appropriate privilege can view or generate.

9.2.2 Encrypted Communications

The Cryptographic Support function of the TOE enforces the FCS_COP.1, FPT_ITC_EXT.1, FPT_ITT_EXT.1, and FTP_TRP_EXT.1 requirements.

FCS_COP.1 is discussed in section 9.1.2.6 and addresses the ability of the TOE to generate hashes of files which reside in the Operational Environment.

FPT_ITC_EXT.1 and FPT_ITT_EXT.1 are covered in the ST in section 2.3.1.6. FPT_ITC_EXT.1 is covered in the TSS through section 9.1.4.3.1. FPT_ITT_EXT.1 is covered in the TSS through section 9.1.4.3.2.

FTP_TRP_EXT.1 is covered in the ST through sections 2.3.1.7 and 9.1.5.1.

When the Parity Reporter service is installed, it installs a certificate into Trusted People section of the Local Machine certificate store (extracted from its own resource file).

All communications between the Parity Application Server and the Parity Clients occur over a secure and encrypted TCP/IP port (default is 41002). The communications are encrypted and authenticated using a server-side certificate. A Parity Administrator can import any valid certificate as a PKCS#12 file for use in server-client communications, using the Parity Console.

Parity Reporter communicates with Parity Knowledge using dual certificates (mutual authentication).

Parity protects TSF data. The TOE maintains and controls individual sessions for Console Users and Client Users. The TSF, when invoked by the underlying host OS, ensures that TSP enforcement functions are invoked and succeed before each function is allowed to proceed. The TSF maintains a security domain for its own execution that protects it from interference and tampering by untrusted subjects initiating actions through its own TSFI.

The Trusted Channel function of the TOE enforces the FPT_ITC_EXT.1 requirement. The TSF shall provide a channel for communication between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure. The TSF shall allow the TSF to initiate communication via the trusted channel, and shall use the trusted channel for the transfer of data between the Parity Application Server, the parity client, and the GSR.

9.2.3 User Data Protection

The User Data Protection function of the TOE enforces the FDP_ACC.1(1), FDP_ACC.1(2), FDP_ACF.1 (1), and FDP_ACF.1 (2) requirements.

FDP_ACC.1(1) and FDP_ACC.1(2) are covered in the TSS through sections 2.3.1.3 and 9.1.3.1. FDP_ACF (1) and FDP_ACF (2) are covered in the TSS through sections 2.3.1.3 and 9.1.3.2.

When a Client User attempts to access a protected resource, the applicable policy will examine the request and determine if they are allowed to access it based on the applicable rules and the policy's SecCon. Each policy has its own Parity Client installer, which is automatically generated on the server when the policy is created. Each installer automatically assigns a policy to each client it installs.

The TSF enforces the Parity policy to objects based on username, groups, or object. If a Client User supplies the correct username or is on the group list for access to an object, than he is granted access to the object.

Components of the TOE interact to enforce access control. SecCon determines how the Parity Client behaves when unknown files are encountered.

9.2.4 Identification and Authentication

The Identification and Authentication function of the TOE enforces the FIA_ATD.1, FIA_UAU.1, FIA_UAU.5, FIA_UID.2, FIA_USB.1, FIA_UAU_EXT.2, FIA_UAU_EXT.8, and FIA_UID_EXT.2 requirements.

FIA_ATD.1, FIA_UAU.1, FIA_UAU.5, FIA_USB.1, and FIA_UID.2 are discussed in section 2.3.1.5. FIA_ATD.1 and FIA_UID.2 are discussed in section 9.1.4.1. FIA_UAU.1 is covered in the TSS through section 9.1.4.2. FIA_UAU.5 is covered in the TSS through section 9.1.4.2.

FIA_UAU_EXT.2 and FIA_UID_EXT.2 are discussed 9.1.2.3 and 9.1.4.4. These requirements discuss the ability of the TSF to require authentication information from Parity Knowledge before exchanging data with it.

FIA_UAU_EXT.8 is discussed in section 9.1.5.1 in the context of facilitating authentication for the Timed Override utility.

The TOE provides user identification, authentication and authorization through the use of user names and passwords for Console or Client Users. Without authentication, Console Users cannot take action. Console Users can be identified and authenticated either using basic authentication directly to the Parity Console or by using Active Directory credentials. The TOE is able to bind users in Active Directory to subjects requesting access to the TSF. Client Users do not authenticate to the TOE; instead, they are authenticated to their workstation which runs the Parity Client in the background whenever the workstation is in use.

9.2.5 Security Management

The Security Management function of the TOE enforces FMT_MOF.1, FMT_MTD.1, FMT_SMF.1, and FMT_SMR.1 requirements.

FMT_MOF.1, FMT_MTD.1, FMT_SMF.1, and FMT_SMR.1 are covered in the TSS in section 2.3.1.5. FMT_MTD.1 is covered in the TSS through section 9.1.4.1. FIA_MTD.1 is covered in the TSS through section 9.1.4.2. FMT_SMF.1 is covered in the TSS through section 9.1.5. FIA_MTD.1 is covered in the TSS through section 9.1.5.3. FMT_SMR.1 is covered in the TSS through section 9.1.5.3.

The TOE provides management capabilities through the Parity Console. The TSF shall provide the ability to manage its security functions including the configuration of policies and Console Users. The TOE defines roles for security relevant authority and associates these with users.

The web console provides account management functionality, allowing a Console User to enable and disable users and manage passwords for users.

9.2.6 TOE Access

Prior to establishing a session, the TOE displays an advisory warning message regarding unauthorised use of the TOE. The access banner applies whenever the TOE provides a prompt for identification and authentication (e.g., Console Users). This is to advise users of warnings regarding the unauthorized use of the TOE.

Based on the above information, the TOE enforces the FTA_TAB.1 requirements as stated in section 6.

10 Rationale

10.1 Security Objectives Rationale

The following table provides a mapping with rationale to identify the security objectives that address the stated assumptions and threats.

Assumption	Objective	Rationale
A.ADMIN One or more authorized administrators will be assigned to install, configure and manage the TOE.	OE.ADMIN One or more authorized administrators will be assigned to install, configure and manage the TOE.	OE.ADMIN maps to A. ADMIN in order to provide assurance that the TOE is deployed in a manner that minimizes the risk of incorrect installation or configuration.
A.CONNECT The TOE will be deployed in an environment where external data stores reside on a trusted network and client systems have the capability to communicate with the Parity Application Server intermittently if not persistently.	OE.CONNECT The TOE will be deployed in an environment where external data stores reside on a trusted network and client systems have the capability to communicate with the Parity Application Server intermittently if not persistently.	OE.CONNECT maps to A.CONNECT because it ensures that the TOE is resistant to attack from external network entities and that the Parity Client(s) can communicate with the Parity Server in order to receive policy information.
A.CONTEXT Clients deployed on remote systems will be installed in a context that prevents Client Users from disabling, removing, altering, or reconfiguring the client.	OE.FILESYS The file systems of machines in the Operational Environment protect the binaries which comprise the TOE and the external data stores which it uses to enforce the SFRs.	OE.FILESYS maps to A.CONTEXT because the inherent protection features of the Windows file system will ensure that if the client is installed in the appropriate context, it will not be able to be altered via normal usage.
A.LOCATE The TOE server and remote database will be located within controlled access facilities	OE.LOCATE The TOE server and remote database will be located within controlled access facilities	OE.LOCATE maps to A.LOCATE in order to ensure that the risk of untrusted users gaining

Assumption	Objective	Rationale
that will prevent unauthorized access.	that will prevent unauthorized physical access.	access to the TOE server or remote database is reduced.
A.NOEVIL Administrators and PowerUsers of the TOE are not careless, willfully negligent, or hostile and will abide by the instructions provided by applicable guidance documentation.	OE.NOEVIL Administrators and PowerUsers of the TOE are not careless, willfully negligent, or hostile and will abide by the instructions provided by applicable guidance documentation.	OE.NOEVIL maps to A.NOEVIL in order to ensure that there are no careless, willfully negligent, or hostile administrators of the TOE.
A.PATCHES Administrators exercise due diligence to update the Operational Environment with the latest patches in order to remove the risk of compromise via known and preventable exploits.	OE.ADMIN One or more authorized administrators will be assigned to install, configure and manage the TOE.	OE.ADMIN maps to A. PATCHES in order to ensure that the authorized administrators properly patch the Operational Environment in a manner that reduces the risk posture of each of them.
A.PASSWORD Console Users will either choose strong passwords as defined by their organizational guidance or, if Active Directory integration is used for the Parity Console, that the Active Directory enforces strong password policies.	OE.PASSWORD Console Users will either choose strong passwords as defined by their organizational guidance or, if Active Directory integration is used for the Parity Console, that the Active Directory enforces strong password policies.	OE.PASSWORD maps to A.PASSWORD in order to ensure that Console Users will either choose strong passwords as defined by their organizational guidance or, if Active Directory integration is used for the Parity Console, that the Active Directory enforces strong password policies.
A.CLIENTID Client Users are identified to the TOE via the host name of their workstation and/or the Active Directory credentials used to authenticate to it.	OE.CLIENTID Client Users are identified to the TOE via the host name of their workstation and/or the Active Directory credentials used to authenticate to it.	OE.CLIENTID maps to A.CLIENTID in order to ensure that Client Users are identified to the TOE via the host name of their workstation and/or the Active Directory credentials used to authenticate to it.

Assumption	Objective	Rationale
A.CRYPTOGRAPHY: Data can be encrypted or decrypted using secure algorithms at the request of the TSF.	OE.CRYPTOGRAPHY: The environment will encrypt TSF data in transit using imported certificates from a certificate authority.	In order for the assumption to be satisfied that the TSF can rely on an external source to encrypt and decrypt data, the Operational Environment must be capable of providing this functionality.

Table 10-1: Assumption to Objective Mapping

Threat/Policy	Objective	Rationale
T.ACCESS Unauthorized users could gain local or remote access to protected objects that they are not authorized to access.	O.ACCESS The TOE will provide measures to authorize users to access specified TOE resources once the user has been authenticated. User authorization is based on access rights configured by the authorized users of the TOE and the binding of external attributes to subjects recognized by the TSF.	O.ACCESS (FDP_ACC.1(1), FDP_ACF.1(1), FDP_ACC.1(2), FDP_ACF.1(2), FIA_ATD.1, FIA_USB.1) addresses T.ACCESS by providing authorized Console Users with the capability to specify access restrictions on the protected TOE resources to Client Users. Once these restrictions are specified, they are then enforced by the clients.

Threat/Policy	Objective	Rationale
	O.ALERT The TOE will provide measures for defining audit events which represent noteworthy violations and will send notifications when these events occur. O.CRYPTOGRAPHY The TOE will provide a mechanism to hash files which reside in the Operational Environment.	O.ALERT (FAU_ARP.1, FAU_SAA.1) helps address T.ACCESS by allowing for the capability to define events which should trigger alerts. Alerts can be used proactively to detect attempted misuse allowing for further attempts to be thwarted. O.CRYPTOGRAPHY (FCS_COP.1) helps address T.ACCESS by identifying files using hashing. This is how the TSF identifies files when performing access control decisions.
	O.WHITELIST The TOE will provide access control enforcement for files, processes, registry values, and devices which reside on client systems to prevent unauthorized access to or modification of system assets.	O.WHITELIST (FDP_ACC.1(1), FDP_ACF.1.1(1), FAU_GEN_EXT.1) addresses T.ACCESS by defining and enforcing the client access policy which controls access to files, processes, registry values, and devices on client systems.
T.ADMIN_ERROR An administrator may incorrectly install or configure the TOE, or install a corrupted TOE resulting in ineffective security mechanisms.	OE.NOEVIL Administrators and PowerUsers of the TOE are not careless, willfully negligent, or hostile and will abide by the instructions provided by applicable guidance documentation.	OE.NOEVIL helps to mitigate T.ADMIN_ERROR by ensuring that relevant guidance documentation is followed in good faith.
	O.MANAGE The TOE will	O.MANAGE

Threat/Policy	Objective	Rationale
	provide authorized users with the resources to manage and monitor user accounts, TOE resources and security information relevant to the TOE.	(FMT_MOF.1, FMT_MTD.1, FMT_SMF.1, FMT_SMR.1) addresses T.ADMIN_ERROR by ensuring only authorized Console Users can perform operations which affect the configuration or behavior of the TSF.
T.MASK A malicious user or process may view audit records, cause audit records to be lost or modified, or prevent future audit records from being recorded, thus masking a user's action.	O.AUDIT The TOE will provide measures for recording security relevant events that will assist Console Users in detecting misuse of the TOE or activity on client systems.	O.AUDIT (FAU_GEN.1, FAU_GEN.2, FAU_SAR.1, FAU_SAR.3, FAU_STG.3, , FAU_STG_EXT.3, helps to address T.MASK by providing authorized users with tools necessary to monitor user activity and to protect against unauthorized access.
	OE.FILESYS The file systems of machines in the Operational Environment protect the binaries which comprise the TOE and the external data stores which it uses to enforce the SFRs.	OE.FILESYS helps address T.MASK by controlling direct access to the environmental database where audit data resides.
	OE.SYSTIME The operating environment will provide reliable system time.	OE.SYSTIME helps to mitigate T.MASK by ensuring that data provided by O.AUDIT is labeled with the correct time.

Threat/Policy	Objective	Rationale
T.MASQUERADE A malicious user or process may impersonate the GSR or Parity Application Server in order to intentionally provide inaccurate configuration information or metadata to the TOE.	O.MUTUAL The TOE will provide a mechanism for establishing mutual authentication between itself and the GSR as a prerequisite for allowing for the transfer of TSF data.	O.MUTUAL (FPT_ITC_EXT.1, FIA_UAU_EXT.2, FIA_UID_EXT.2) helps to mitigate the threat of impersonation by providing mutual authentication between the TOE and a trusted IT entity external to it. Once authentication has been performed, the communications are protected to reduce the risk of disclosure.
	O.EAVESDROPPING The TOE will use its environment to protect TSF data in transit using certificates imported from a certificate authority.	O.EAVESDROPPING (FPT_ITC_EXT.1, FPT_ITT_EXT.1, FTP_TRP_EXT.1) helps to mitigate the threat by using trusted certificates to protect data in transit.
	OE.CRYPTOGRAPHY The environment will encrypt TSF data in transit using imported certificates from a certificate authority.	OE.CRYPTOGRAPHY helps to mitigate the threat of impression by validating certificates provided by a trusted CA.
T.REVERSE A malicious or ignorant user may acquire a reverse-engineered version of the client that bypasses or subverts access control to protected resources.	O.ACCESS The TOE will provide measures to authorize users to access TSF data or resources protected by the TOE once the user has been authenticated. User authorization is based on access rights configured by the authorized users of the TOE and the binding of external attributes to subjects recognized by the TSF.	O.ACCESS (FDP_ACC.1.1(1), FDP_ACF.1.1, FDP_ACC.1(2), FDP_ACF.1.2, FIA_ATD.1, FIA_USB.1) helps mitigate the threat of reverse engineering because the TSF unconditionally forbids

Threat/Policy	Objective	Rationale
		Client Users from altering its registry values, modifying its files, or terminating its processes.
	O.PROTECT The TOE will provide measures for the server to validate the integrity of a client and for the client to validate the integrity of a server.	O.PROTECT (ADV_ARC.1, FPT_ITT_EXT.1) mitigates the threat of a reverse engineered TOE affecting the configuration of the deployment because each component of the TOE can validate the integrity of the other.
	OE.FILESYS The file systems of machines in the Operational Environment protect the binaries which comprise the TOE and the external data stores which it uses to enforce the SFRs.	OE.FILESYS helps address T.REVERSE by restricting the ability to modify the files and binaries which comprise the TOE if they were created in the appropriate administrator context.
T.UNAUTH Malicious or non-malicious users could gain unauthorized access to the console by bypassing identification and authentication countermeasures.	O.AUTH The TOE will provide mechanisms to identify and authenticate Console Users requesting access to the TSF prior to allowing any TSF-mediated actions except for the receiving of alerts.	O.AUTH (FIA_ATD.1, FIA_UAU.1, FIA_UAU.5, FIA_UAU_EXT.8.1, FIA_UID_EXT.2.1, FIA_UAU_EXT.2.1, FIA_UID.2, FIA_USB.1) helps mitigate the threat of unauthenticated access by providing mechanisms to validate the claimed identity of Console Users prior to interacting with TSF data. It does this by integrating with the external user directory and binding user data in

Threat/Policy	Objective	Rationale
		that directory to subjects in the TSF.
	OE.ATTRIBUTES The Operational Environment will provide an external directory of users which defines attributes based on existing organizational structure.	OE.ATTRIBUTES helps address T.UNAUTH by providing the means to identify users based on attributes stored in an external directory.
	OE.AUTH The Operational Environment will provide measures to uniquely identify Client Users and will authenticate their claimed identity prior to granting a user access to the resources protected by the TOE.	OE.AUTH helps address T.UNAUTH by requiring that the Operational Environment provide some assurance to the TOE regarding the identity of Client Users.
P.ACCESS_BANNER The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the system.	O.DISPLAY_BANNER The TOE will display an advisory warning regarding use of the TOE.	O.DISPLAY_BANNER (FTA_TAB.1) satisfies this policy by ensuring that the TOE displays a Security Administrator configurable banner that provides all users with a warning about the unauthorized use of the TOE.

Table 10-2: Threat/Policy to Objective Mapping

10.2 EAL 2 Justification

The threats that were chosen are consistent with attacker of low attack potential, therefore EAL2 was chosen for this ST.

10.3 Requirement Dependency Rationale

All Security Functional Requirement component dependencies have been met by the TOE as defined by the CEM with the exceptions of FAU_STG.1, FCS_COP.1, and FPT_STM.1. They have been excluded for the following reasons:

- FAU_STG.1 – The TSF is capable of deleting audit records because it has appropriate privileges on the environmental database in which audit data is stored, which is why FAU_STG_EXT.3 has been claimed. FAU_STG_EXT.3 has a dependency of FAU_STG.1. This dependency is not met because the TSF does not have the ability to protect against modifications of the environmental database because it resides in a SQL environment on a trusted machine as opposed to being stored within the Parity Server itself. This requirement is considered to be enforced by OE.FILESYS.
- FCS_CKM.1 – The TSF is capable of utilizing cryptographic hashes with the SHA-256 algorithm. However, keys are not required for hashing algorithms to operate properly. Therefore, the dependency on FCS_CKM.1 is not applicable.
- FPT_STM.1 – The TSF is capable of creating audit records that include timestamps of when the associated audited events occur, which is why FAU_GEN.1 has been claimed. FAU_GEN.1 has a dependency of FPT_STM.1. This dependency is not met because the TSF does not keep its own clock; instead, it relies on the clock of the system on which the Parity Server has been installed. This requirement is considered to be enforced by OE.SYSTIME.

10.4 Security Functional Requirements Rationale

The following table provides a mapping with rationale to identify the security functional requirement components that address the stated TOE and environment objectives.

Objective	Security Functional Components	Rationale
O.ACCESS The TOE will provide measures to authorize users to access TSF data or resources protected by the TOE once the user has been authenticated. User authorization is based on access rights configured by the authorized users of the TOE and the binding of external attributes to subjects recognized by the TSF.	FDP_ACC.1(1) Subset access control	FDP_ACC.1(1) defines the client access policy which is used to authorize operations which are performed against client systems.
	FDP_ACC.1(2) Subset access control	FDP_ACC.1(2) defines the console access policy which is used to authorize operations which are performed against the TOE.
	FDP_ACF.1(1) Security attribute based access control	FDP_ACF.1(1) enumerates the mechanisms by which the client access policy is enforced.

	FDP_ACF.1(2) Security attribute based access control	FDP_ACF.1(2) enumerates the privileges which are available to each group within the console.
	FIA_ATD.1 User attribute definition	FIA_ATD.1 defines the attributes which belong to users so that they the client access policy can be enforced on them based on those attributes.
	FIA_USB.1 User-subject binding	FIA_USB.1 defines the binding between users as defined in the external user directory and subjects as defined by the TSF.
O.ALERT The TOE will provide measures for defining audit events which represent noteworthy violations and will send notifications when these events occur.	FAU_ARP.1 Security audit automatic response	FAU_ARP.1 defines how the TOE sends alerts once an alert has been triggered.
	FAU_SAA.1 Security audit response	FAU_SAA.1 defines what audit events and accumulations of audit events are considered to be conditions worthy of triggering alerts.
O.AUDIT The TOE will provide measures for recording security relevant events that will assist Console Users in detecting misuse of the TOE or activity on client systems.	FAU_GEN.1 Audit data generation	FAU_GEN.1 defines the behavior of the TSF which causes security relevant events to be generated and enumerates the data which is contained within these events.
	FAU_GEN_EXT.1 File Inventory	FAU_GEN_EXT.1.1 defines the behavior of the TSF which identifies and records data about files on client systems.

	FAU_GEN.2 User identity association	FAU_GEN.2 confirms that all relevant auditable events include subject identity for the purposes of accountability.
	FAU_SAR.1 Audit review	FAU_SAR.1 provides the ability for all Console Users to read audit data in either a tabular or graphical format.
	FAU_SAR.3 Selectable audit review	FAU_SAR.3 defines the ability of the TOE to selectively display audit data based on event type and either a time range or number of events.
	FAU_STG.3 Action in case of possible data loss	FAU_STG.3 defines the ability of the TOE to automatically prune the oldest audit records in the database when a configurable number of records has been exceeded.
	FAU_STG_EXT.3 Action in case of possible data loss	FAU_STG_EXT.3 describes the optional ability of the TOE to write a duplicate copy of audit records to a remote syslog server or database whenever a new record is generated.
O.AUTH The TOE will provide mechanisms to identify and authenticate Console Users requesting access	FIA_ATD.1 User attribute definition	FIA_ATD.1 defines the security-relevant attributes of all Console Users. This includes attributes related to authentication.

to the TSF prior to allowing any TSF-mediated actions except for the receiving of alerts.	FIA_UAU.1 Timing of authenticated	FIA_UAU.1 requires Console Users to authenticate to the TOE before any TSF-mediated actions are allowed but states that receiving alerts is an exception since those are sent via email to any valid address.
	FIA_UAU.5 Multiple authentication mechanisms	FIA_UAU.5 defines the mechanisms for authentication to the console as either basic authentication (username/password) or Active Directory integration.
	FIA_UID.2 User identification before any action	FIA_UID.2 requires Console Users to identify themselves to the TOE before any TSF-mediated actions are allowed.
	FIA_USB.1 User-subject binding	FIA_USB.1 defines the mapping between the Active Directory attributes of users and the identity of subjects attempting to access the TOE.
	FIA_UAU_EXT.8 Generation of authentication credentials	FIA_UAU_EXT.8 allows for the TSF to generate an authentication credential for a Console User who is using the Timed Override feature of the Parity Client.
O.EAVESDROPPING The TOE will use its environment to protect TSF data in transit using either its own certificates or those imported from a certificate authority.	FPT_ITC_EXT.1 Inter-TSF trusted channel	FPT_ITC_EXT.1 states that a trusted channel is established between the TOE and trusted IT entities that exist in the Operational Environment.
	FPT_ITT_EXT.1 Basic Internal TSF Data Transfer Protection through the OE	FPT_ITT_EXT.1 states that communications between the Parity Application Server and remote clients are encrypted.

	FTP_TRP_EXT.1 Trusted Path through the OE	FTP_TRP_EXT.1 defines the trusted path from the Console User to the TSF is protected from disclosure of information.
O.MANAGE The TOE will provide authorized administrators with the resources to manage and monitor user accounts, resources, and security information relevant to the TOE.	FMT_MOF.1 Management of security functions behaviour	FMT_MOF.1 restricts the ability to modify the client access policy to Administrators and PowerUsers, two of the groups to which a Console User can belong.
	FMT_MTD.1 Management of TOE data	FMT_MTD.1 defines the operations which can be performed on the console by each group of Console User.
	FMT_SMF.1 Specification of management functions	FMT_SMF.1 enumerates all of the security-relevant management functions which can be performed against the TSF.
	FMT_SMR.1 Security roles	FMT_SMR.1 defines the roles of Console User and Client User and the groups to which a Console User can belong.
O.MUTUAL The TOE will provide a mechanism for establishing mutual authentication between itself and the GSR as a prerequisite for allowing for the transfer of TSF data.	FPT_ITC_EXT.1 Inter-TSF confidentiality during transmission	FPT_ITC_EXT.1 states that a trusted channel is established between the TOE and trusted IT entities that exist in the Operational Environment, of which the GSR is one.
	FIA_UAU_EXT.2 External entity authentication before any action	FIA_UAU_EXT.2 states that the TSF will require each external entity to successfully authenticate before it can perform any TSF-mediated actions on behalf of that entity

	FIA_UID_EXT.2 External entity identification before any action	FIA_UID_EXT.2 states that the TSF will require each external entity to successfully identify itself before it can perform any TSF-mediated actions on behalf of that entity
O.PROTECT The TOE will provide measures for the server to validate the integrity of a client and for the client to validate the integrity of a server.	ADV_ARC.1 Security architecture description	ADV_ARC.1 defines the ability of the TOE to protect itself from attempted tampering and bypass.
	FPT_ITT_EXT.1 Basic Internal TSF Data Transfer Protection through the OE	FPT_ITT_EXT.1 defines the ability of the TOE to establish a trusted channel between distributed parts of the TOE so that the endpoints are assured to one another.
O.WHITELIST The TOE will provide access control enforcement for files, processes, registry values, and devices which reside on client systems to prevent unauthorized access to or modification of system assets.. for files, processes, registry values, and devices which reside on client systems.	FDP_ACC.1(1) Subset access control	FDP_ACC.1(1) defines the client access policy which is used to authorize operations which are performed against client systems.
	FDP_ACF.1(1) Security attribute based access control	FDP_ACF.1.1 enumerates the mechanisms by which the client access policy is enforced. Enforcement of the client access policy is also known as whitelisting.
	FAU_GEN_EXT.1 Object inventory	FAU_GEN_EXT.1 assists in whitelisting by providing the TSF an inventory of objects which reside in the Operational Environment which may have rules applied to them.
O.CRYPTOGRAPHY The TOE will provide a mechanism to hash files which reside in the Operational Environment.	FCS_COP.1 Cryptographic operation	FCS_COP.1 defines the method by which the TSF hashes files in order to identify them.

O.DISPLAY_BANNER The TOE will display an advisory warning regarding use of the TOE.	FTA_TAB.1 Default TOE Access Banners	FTA_TAB.1 meets this objective by requiring the TOE display a Security Administrator defined banner before a user can establish an authenticated session. This banner is under complete control of the Security Administrator in which they specify any warnings regarding unauthorized use of the TOE and remove any product or version information if they desire.

Table 10-3: Security Functional Requirements Rationale

10.5 Assurance Measures

This section identifies the documentation which contains the assurance measures provided by the developer in order to meet the security assurance requirement components for EAL2 augmented with ASE_TSS.2 and ALC_FLR.1. A description of each of the TOE's documents which contain the assurance measures follows in Table 11-4. The wording provided within the documents listed is the assurance measures for each Security Assurance Requirement.

Component	Document(s)	Rationale
ADV_ARC.1 Security Architecture Design	TOE Design Specification for Bit9 Parity 6.0.1 – v2.0	This document describes the security architecture of the TOE.
ADV_FSP.2 Functional Specification with complete summary	Functional Specification for Bit9 Parity 6.0.1 – v2.0	This document describes the functional specification of the TOE with complete summary.
ADV_TDS.1 Architectural Design	TOE Design Specification for Bit9 Parity 6.0.1 – v2.0	This document describes the architectural design of the TOE.
AGD_OPE.1 Operational User Guidance	Using Parity v6.0.1	This document describes the operational user guidance for.
AGD_PRE.1 Preparative Procedures	• Installing Parity v6.0.1 • Operating Environment Guidelines v 6.0.1 • Evaluated Configuration	This document describes the preparative procedures that need to be done prior to installing.

Component	Document(s)	Rationale
	for Bit9 Parity 6.0.1	
ALC_CMC.2 Authorizations Controls	• Bit9 - Source Code Management System v1.1 • svn-log-Trillian.txt • svn-log-Trillian-M1.txt • Trillian-6.0-Files.txt • Trillian-6.0-M1-Files.txt	This document describes the authorization controls for the TOE.
ALC_CMS.2 CM Scope	• Bit9 - Source Code Management System v1.1 • svn-log-Trillian.txt • svn-log-Trillian-M1.txt • Trillian-6.0-Files.txt • Trillian-6.0-M1-Files.txt	These documents describe the CM scope of the TOE.
ALC_DEL.1 Delivery Procedures	Bit9 - Delivery Evidence v1.4	This document describes product delivery for and a description of all procedures used to ensure objectives are not compromised in the delivery process.
ALC_FLR.1 Flaw reporting procedures	Bit9 - Incident (Flaw) Remediation v1.3	This document provides the policies for issuing new releases of the TOE as corrective actions.
ASE_CCL.1 Conformance Claims	Bit9 Parity v6.0.1 Security Target – v2.0	This document describes the CC conformance claims made by the TOE.
ASE_ECD.1 Extended Components Definition	Bit9 Parity v6.0.1 Security Target – v2.0	This document provides a definition for all extended components in the TOE.
ASE_INT.1 Security Target Introduction	Bit9 Parity v6.0.1 Security Target – v2.0	This document describes the Introduction of the Security Target.
ASE_OBJ.2 Security Objectives	Bit9 Parity v6.0.1 Security Target – v2.0	This document describes all of the security objectives for the TOE.
ASE_REQ.2 Security Requirements	Bit9 Parity v6.0.1 Security Target – v2.0	This document describes all of the security requirements for the TOE.

Component	Document(s)	Rationale
ASE_SPD.1 Security Problem Definition	Bit9 Parity v6.0.1 Security Target – v2.0	This document describes the security problem definition of the Security Target.
ASE_TSS.2 TOE Summary Specification	Bit9 Parity v6.0.1 Security Target – v2.0	This document describes the TSS section of the Security Target.
ATE_COV.1 Analysis of Coverage	• Common Criteria Test Plan – 6.0.1 • Common Criteria Test Plan (matrix) • Test Results.html	These documents comprise the vendor's functional test plan and the associated test results.
ATE_FUN.1 Functional Tests	• Common Criteria Test Plan – 6.0.1 • Common Criteria Test Plan (matrix) • Test Results.html	These documents comprise the vendor's functional test plan and the associated test results.
ATE_IND.2 Independent Testing	• Booz Allen Hamilton Independent Test Procedures • Booz Allen Hamilton Independent Test Report	These documents comprise the test laboratory's independent functional and vulnerability test plans
AVA_VAN.2 Vulnerability Analysis	Vulnerability Analysis Bit9, Inc.Parity™ 6.0.1 Version 3.0	This document describes the vulnerability analysis of the TOE.

Table 10-4: Assurance Requirements Evidence