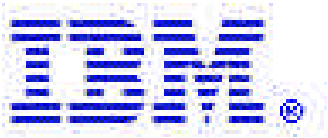
Tivoli software



Tivoli Access Manager for e-Business 4.1 with Fixpack 5

Security Target

CC Evaluation Document

Document Version Number 1.7

Document Creation Date: 12 November 2002

Document Update Date: 30 September 2003

Authors: Helmut Kurth

Owner: Bob Blakley

Status: Final version as used for certification

Table of Contents

Document Control Information

History

This track of document changes is a part of the standard document tracking process.

Version	Date	Summary of Changes
Tivoli Access Manager for e-Business 4.1 Security Target Draft 0.1	22 November 2002	First Draft.
Tivoli Access Manager for e-Business 4.1 Security Target Draft 0.2	4 December 2002	Completed chapters 3 to 6. Modified chapter 2 according to the comments from Anthony and I-Lung.
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.0	17 February 2003	Completed the document
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.1	19 March 2003	Updated document including comments from Tivoli team and evaluation report of Version 1.0
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.2	10 April 2003	Updated document especially on chapter 2 including clarifications on the evaluated configuration.
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.3	25 April 2003	Updated document in response to comments from BSI
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.4	08 May 2003	Updated document in response to comments from BSI, included RSA key generation as part of the TSF, adapted audit requirements
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.5	15 July 2003	Update document to include new platforms and address comments from BSI, the developer and evaluation team
Tivoli Access Manager for e-Business 4.1 Security Target Version 1.6	11 August 2003	Update document to address additional comments and remarks from BSI
Tivoli Access Manager for e-Business 4.1 with Fixpack 5	30 September 2003	Update document to address SP3

Security Target Version 1.7		

1. ST introduction

This document defines the Security Target for the Common Criteria Evaluation of Tivoli Access Manager for e-Business Version 4.1 with Fixpack 5 developed by IBM.

1.1. ST Identification

Title: Tivoli Access Manager for e-Business Version 4.1 with Fixpack 5 Security Target Version 1.7

Keywords: Access Control, ISO 10181-3, aznAPI

This document is the security target for the Common Criteria evaluation of Tivoli Access Manager for e-Business Version 4.1 with Fixpack 5 provided by IBM Inc. for a Common Criteria evaluation

1.2. ST Overview

IBM Tivoli Access Manager for e-Business is a specific implementation of the access control framework defined by the ISO 10181-3 [ISO 10181-3] standard and the Authorization API (aznAPI) [AZNAPI].

IBM Tivoli Access Manager for e-Business is a complete authorization solution for corporate Web, client/server, Tivoli Access Manager applications, and legacy (pre-existing) applications. Tivoli Access Manager authorization allows an organization to securely control user access to protected information and resources. By providing a centralized, flexible, and scalable access control solution, Tivoli Access Manager allows you to build highly secure and well-managed network-based applications and e-business infrastructure.

In addition to its state-of-the-art security policy management feature, Tivoli Access Manager supports authentication, authorization, data security, secure communication and resource management capabilities. You use Tivoli Access Manager in conjunction with standard Internet-based applications to build highly secure and well-managed intranets.

At its core, Tivoli Access Manager provides:

- Authentication Service

  with Tivoli Access Manager range of built-in authenticators.

- Authorization Service

  The Tivoli Access Manager authorization service, accessed through a standard authorization API, provides permit and deny decisions on access requests for native Tivoli Access Manager servers and third-party applications.

IBM Tivoli Access Manager WebSEAL is the Resource manager / Authorization evaluator responsible for managing and protecting Web-based information and resources.

WebSEAL acts as a reverse Web proxy by receiving HTTP/HTTPS requests from a Web browser and delivering content from its own Web server or from junctioned back-end Web application servers. Requests passing through WebSEAL are evaluated by the Tivoli Access Manager authorization service to determine whether the user is authorized to access the requested resource.

WebSEAL provides the following features:

- Supports multiple authentication methods

  Built-in authentication methods allow flexibility in supporting a variety of authentication mechanisms.

- Accepts HTTP and HTTPS requests
- Integrates and protects back-end server web resources
- Provides fine-grained access control for the back-end server Web space

  Supported resources include URLs, URL-based regular expressions, CGI programs, HTML files, Java servlets, and Java class files.
- Performs as a reverse Web proxy

  WebSEAL appears as a Web server to clients and appears as a Web browser to the back-end servers it is protecting.

1.3. CC Conformance Claim

The evaluation is based upon

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; Version 2.1, August 1999,
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Requirements; Version 2.1, August 1999,
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; Version 2.1, August 1999

referenced as [CC].

For the evaluation the following methodology will be used:

- Common Methodology for Information Technology Security Evaluation CEM-99/045 Part 2: Evaluation Methodology, Version 1.0, August 1999, [CEM]

The assurance level selected is EAL 3 augmented by ALC_FLR.1.

This Security Target claims the following CC conformance:

- part 2 conformant
- part 3 conformant
- evaluation assurance level (EAL) 3 augmented by ALC_FLR.1

1.4. Strength of Function

The claimed strength of function for this TOE is: **SOF-medium**

1.5. Rationale for the Selection of the Assurance Level and Strength of Function

The evaluation assurance level (EAL) 3 was chosen as a medium level of assurance reflecting the expected assurance requirements of commercial customers using the TOE for the protection of data with a low or medium level of sensitivity. The TOE is intended to provide a reasonable level of protection for this data comparable to the protection provided by most commercial-off-the-shelf operating system products.

In line with this medium level of assurance the functions provided by the TOE that are subject to probabilistic or permutational analysis are claimed to have a medium strength (SOF-medium).

1.6. Definition of Terms

Unauthenticated users	Individuals who are not known to the system but are part of the user community allowed to access resources available to unauthenticated users
Authorized users	Individuals who have successfully authenticated themselves to the TOE and may access resources as defined by the access control policy of the TOE
Authorized administrators	Individuals who have successfully authenticated themselves to the TOE as administrators and are allowed to perform administrative tasks via the PD Admin interface within their administrative responsibilities

2. TOE Description

The TOE consists of Tivoli Access Manager for e-Business Version 4.1 with Fixpack 5. The following sections provide a description of the structure of the TOE and an overview on the security functions provided by the TOE.

2.1. Access Control Framework according to ISO 10181-3 and the Open Group Authorization API

The TOE is a specific implementation of the access control framework defined by the ISO 10181-3 [ISO 10181-3] standard and the Authorization API (aznAPI) [AZNAPI]. The TOE uses the overall access control model and the interface described in those two standards. To explain those ideas we provide a short summary of them.

ISO 10181-3 contains the following figure to explain the general access control model:

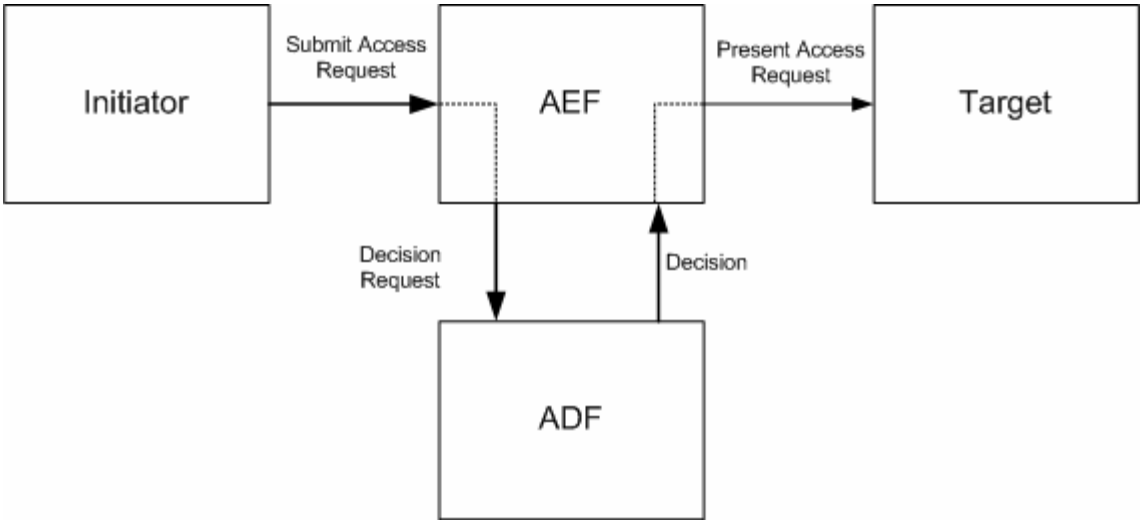


Figure 1: ISO 10181-3 fundamental access control functions

In this model an initiator submits an access request to the "Access Enforcement Function" (AEF) of a system. This function passes the request to an "Access Decision Function" (ADF), which makes the decision based on the rules of the access control system which may be based on:

- The identity and attributes of the initiator
- The identity and attributes of the resource being requested
- Contextual information (e. g. time and date, number of request from the same initiator, information from other systems)

Separating the access enforcement from the access decision function as well as separating the access enforcement function from the actual target allows to implement highly flexible access control and management systems in distributed environments. ISO 10181-3 actually is a general framework for such kind of access control and management system.

The Open Group now defined a standard for an application programming interface (API) for the interface between the Access Enforcement Function (AEF) and the Access Decision Function (ADF)

which allows AEF and ADF components from different vendors to co-operate. The following figure shows the aznAPI system structure as defined in [AZNAPI]

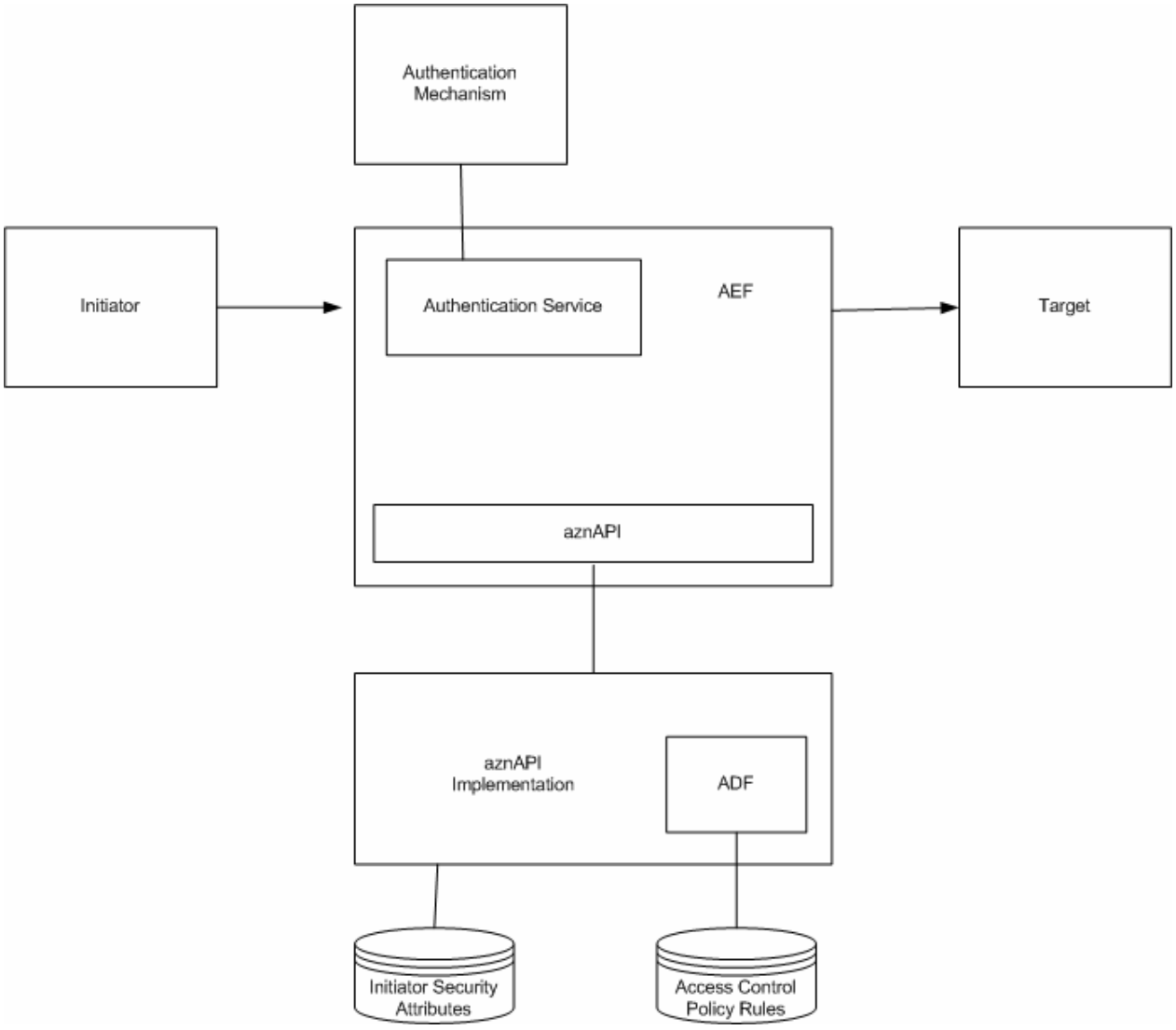


Figure 2: aznAPI System Structure as defined in the Open Group Standard

In this model the initiator submits his access request to the AEF, which then (if required by the policy) authenticates the identity of the initiator using the authentication service within the AEF. This authentication service may use an external authentication mechanism (e. g. a directory server storing user attributes and credentials).

The request together with the initiator attributes is then passed via the aznAPI interface to the implementation of the aznAPI, which in turn may itself store or request from an external entity additional security attributes of the initiator of the request. Those together with the information passed via the aznAPI about the request (including the target of the request) as well as the information about the initiator of the request is passed to the ADF component, which uses the Access Control Policy Rules stored in some kind of database.

2.2. Mapping the TOE to the aznAPI System Structure

The TOE is a specific implementation of the access control model defined in [ISO 10181-3] and [AZNAPI]. The overall TOE architecture is illustrated in figure 3 where the dotted line indicates the TOE boundary. With relation to the model defined in figure 2 the TOE includes the Access Enforcement Function (AEF) and the Access Decision Function (ADF) together with the Access Control Policy rules. The Authentication Mechanism as well as the "Initiator Security Attributes" Database are implemented using a Directory Server, which itself is not part of the TOE. Also the Target system which has the actual resource to be protected is not part of the TOE.

In this model a user on a client submits a request for a resource (e. g. accessing a URL on a network protected by the TOE). This request is intercepted by the TOE (much in the same way as an application gateway firewall system intercepts network requests). The TOE performs the following actions:

- Checking if the requested resource is protected but accessible to unauthenticated users. If this is true, the request is passed through.
- Checking if the user has already been authenticated (i. e. there is a protected session where the user has been authenticated). If not, the user is required to authenticate (this is the case for password based authentication. Certificate based authentication will always take place when the session is established. Please read the section on authentication for details). This authentication makes use of an external Directory Server which stores user attributes and user credentials.
- Checking if the user has the right to access the requested resource in the requested mode. If not, the request is rejected. If yes, the request is passed through to the server holding the resource (the TOE works like a reverse proxy here).

To explain how the access rights are checked a overview on the Tivoli Access Manager components is provided first (please see figure 3 for an architectural overview of the TOE).

The "Resource Manager" is implemented within the TOE by the WebSEAL component. This component includes also the "Authorization Evaluator" as a subsystem. The Resource Manager communicates with the "Authorization Evaluator" via the aznAPI.

The "Policy Server" is responsible to define and maintain the access control policy. It uses the "Matster Authorization Policy" database to store the access control policy rules.

To speed up the time required to make an access decision, the "Authorization Evaluator" manages a replica of the "Master Authorization Policy". The Policy Server informs all Authorization Evaluators about modifications to the "Master Authorization Policy" (actually what it does is to use a standard compression utility to compress the whole database and the transfer the whole new database). An Authorization Evaluator can also request the Policy Server to submit a new copy of the Master Authorization Policy (which it does upon startup, since there may be updates it did not get during a down-time). Also the Policy Server can demand an Authorization Evaluator to update the replica of the Master Authorization Policy in cases it is not sure that it has the latest version.

Administration of the TOE is done via a workstation or terminal directly connected to the Policy Manager component. Only the command line interface and C language API for administration are part of the evaluated configuration. The C language API may be used by an organization to define its own tools to automate some of the administration tasks. But of course such tools are not part of the evaluated configuration and it is up to the organization to ensure that those tools perform their task correctly.

Administration includes the management of the Master Authorization Policy (defining access rules

for protected objects) as well as management of the TOE. It should be noted that access rights of administrators to administrative objects of the TOE are also stored and maintained in the Master Authorization Policy.

To perform authentication the TOE uses an external directory server supporting the LDAP protocol. The directory server is used as a repository for user and administrator attributes and credentials. Authentication of users is done by the Resource Manager, authentication of administrators is performed by the Policy Server (in the sense of the authentication service in figure 2) and both use the external Directory Server as the authentication mechanism.

The communication links between the TOE and the LDAP server as well as between the TOE and the client systems and the TOE and the target systems is protected using the SSL v3 or TLS v1 protocol. The TOE uses the GSKit library for the implementation of the those protocols and their underlying cryptographic functions. The GSKit library is therefore part of the Policy Server and part of the Resource Manager. Also the communication link between the Policy Server and the different Resource Manager is secured by SSL v3 resp. TLS v1 using the GSKit library.

The Master Authorization Policy as well as the Replica Authorization Policy are databases. The Master Authroization Policy is a database held by the Policy Server and the Replica Authorization Policy is a database held by each Authorization Evaluator.

This TOE architecture (showing also the directory server and the servers holding the resources, although they are not part of the TOE) is shown in the following figure:

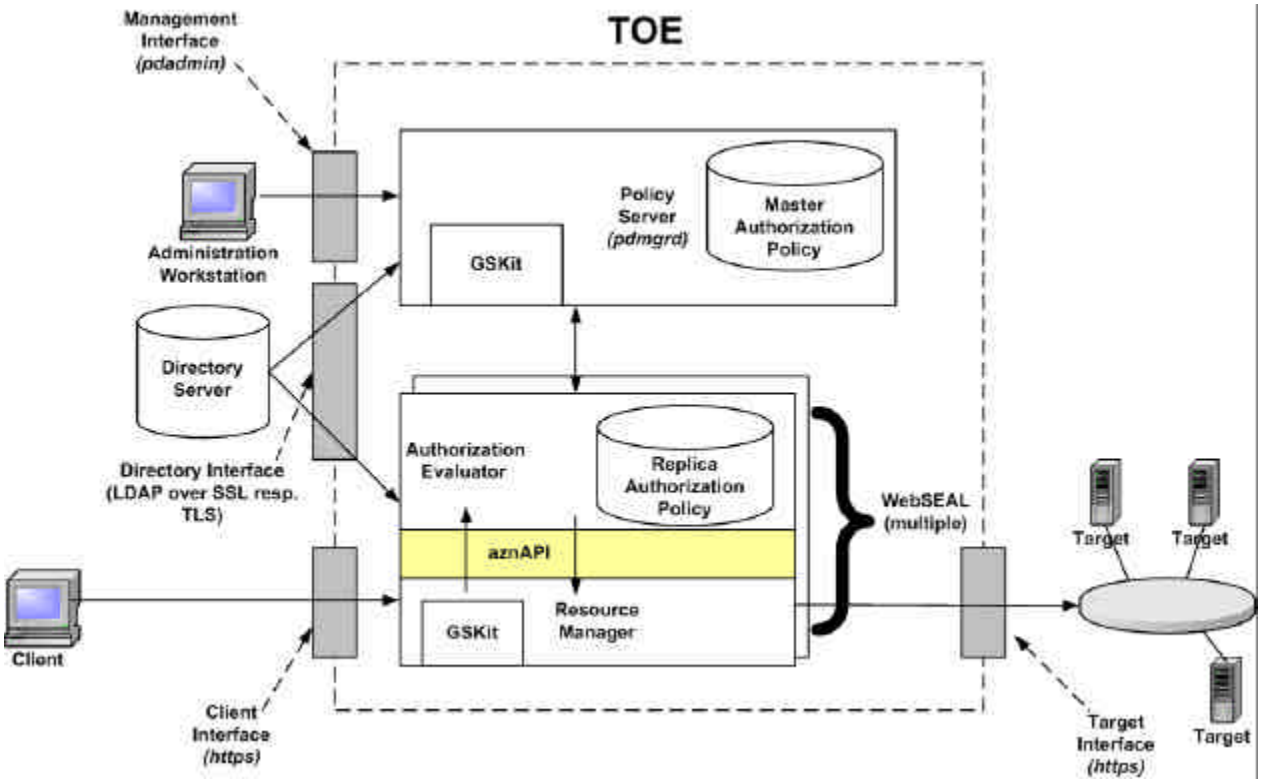


Figure 3: Components and TOE Boundary

The TOE now maps in the following way to the system structure shown in figure 2 as defined in

[AZNAPI]:

- The “Initiator” maps to the client
- The “Access Enforcement Function” (AEF) **and** the “Authentication Service” map to the Resource Manager (part of WebSEAL)
- The “Access Decision Function” (ADF) maps to the “Authorization Evaluator” (part of WebSEAL)
- The “aznAPI” maps to the “aznAPI”
- The “aznAPI” implementation maps to the “Authorization Evaluator” **and** the “Policy Server”.
- The “Authentication Mechanism” **and** the “Initiator Security Attributes” map to the “Directory Server” (which is not part of the TOE but part of the TOE environment)
- The “Access Control Policy Rules” map to the “Master Authorization Policy” **and** the “Replica Authorization Policy”
- The “Target” maps to the “Target”

2.3. Resource Manager

The Resource Manager shown in the previous figure as part of WebSEAL is responsible to protect the external resources on the target system (Web servers).

The WebSEAL Resource Manager acts as a reverse Web proxy by receiving HTTP/HTTPS requests from a Web browser and delivering content from junctioned back-end Web application servers. Requests passing through the WebSEAL Resource Manager are evaluated by the Authorization Evaluator (see figure 3) to determine whether the user is authorized to access the requested resource.

The WebSEAL Resource Manager provides the following features:

- Supports multiple built-in authentication services (only password based and client certificate based authentication are supported in the evaluated configuration). It uses an external Directory Server storing user attributes and credentials as supporting mechanism for the authentication services.
- Accepts HTTPS requests (The TOE is configured to accepts only SSL v3 or TLS v1 secured connections whenever a client establishes a connection to a WebSEAL component. Attempts to establish a connection not secured by SSL or TLS will be rejected.)
- Integrates and protects back-end server resources through WebSEAL junction technology

  Supported resources include (but is not restricted to) URLs, URL-based regular expressions, CGI programs, HTML files, Java servlets, Java class files, Active Server Pages and other scripts.
- Performs as a reverse Web proxy

  The WebSEAL Resource Manager appears as a Web server to clients and appears as a Web browser to the junctioned back-end servers it is protecting.

2.4. Authorization Evaluator

This component is running as a part of a WebSEAL system. When the Resource Manager component calls functions of the aznAPI to check if the client has the right to access the resource in the intended way, the Authorization Evaluator will check the local replica of the Master Authorization Policy to

decide if the request can be granted or not.

The Authorization Evaluator is also responsible to synchronize the local replica of the Master Authorization Policy with the Policy Server. The Authorization Evaluator may ask the Policy Server if his version is still up-to-date (which it always does on start-up and continuously during operation) and will store a new version of the replica database on demand of the Policy Server. In addition the Authorization Evaluator will serve additional system management commands coming from the Policy Server.

The Authorization Evaluator sets up a secured communication channel with the Policy Server using the SSL v3 or TLS v1 protocol with client and server authentication. The Policy Server will take the role of the server while the Authorization Evaluator will take the role of a client. Before an Authorization Evaluator can set up such a communication channel, it needs to have its public key certified by the Policy Server (which acts as a certification authority for SSL client certificates within the TOE).

2.5. Policy Server

This component is responsible for the management of the Master Authorization Policy. For this management it provides a separate interface for administrators (pdadmin interface) as a command line interface as well as a C API on the Policy Server. To perform administrative actions an administrator has to identify and authenticate himself via these interfaces. Only password based authentication is possible at this interface.

An administrator using a remote terminal or remote workstation to connect to the Policy Server for administration needs to ensure that the remote terminal or workstation are in a secured environment and managed securely. Management is performed by setting up a secured connection to communicate with a shell of the operating system on the Policy Server. There it calls the pdadmin command line interface and authenticates himself. Note that the security measures to protect the remote terminal or remote workstation as well as the security measures used to protect the communication link between the remote terminal or workstation are not part of the TOE but have to be assured in the TOE environment.

Administrators can now define and / or modify rules in the Master Authorization Policy as well as perform administrative actions for the Authorization Evaluator or Resource Manager components. Whenever they modify the Master Authorization Policy a request for synchronization of their replica database is sent to all Authorization Evaluator components.

The Policy Server uses the Master Authorization Policy to store access control rules to system management objects and uses the Directory Server to store attributes and credentials of administrators. Management of the TOE can only be performed via the pdadmin interface of the Policy Server, since the “management objects” are not known to the Resource Manager or the Authorization Evaluator and therefore not accessible via a client interface to the TOE.

2.6. GSKit

GSKit is a library that implements the SSL Version 3 and TLS Version 1 protocol to secure the communication between the TOE and other trusted systems (including the LDAP server and the back-end Web Servers) as well as the communication between distributed parts of the TOE itself. SSL resp. TLS is also used to secure the communication between the TOE and client systems. GSKit is part of each Resource Manager / Authorization Evaluator system as well as the Policy Server. Since it provides the functions to secure the communication between the machines that are part of the TOE as well as the Clients, the Target systems and the Directory Server it is part of the TSF. On the Policy Server this component also provides the functions to generate and sign the certificates for the

public keys of the WebSEAL systems.

The functions provided by GSKit include those required for the generation of public/private RSA key pairs, generation of X.509 V3 certificate and certificate management (signing, distribution and revocation). RSA key pairs are required for the SSL resp. TLS ciphersuites supported by the TOE. Please read the SSL resp. TLS specification documents how RSA is used as within those protocols with the ciphersuites listed in chapter 5 of this Security Target.

Within the TOE the Policy Server component is used as a Certification Authority for the certificates of the other TOE components. When a new WebSEAL system is added to the TOE, during the installation of those components an RSA key pair is generated and the key pair and the PDCA certificate is imported via the pdadmin interface. After this, the new WebSEAL system and the Policy Server component can set up a trusted communication path with mutual authentication using the SSL v3 or TLS v1 protocol with client and server authentication. Signing those certificates is performed using the functions of GSKit.

2.7. TOE configuration

The following describes the specifics of the configuration of Tivoli Access Manager for eBusiness version 4.1 that conforms to the description in this Security Target and is henceforth called the evaluated configuration:

- The Policy Server component of the TOE is installed and operated on a dedicated system that communicates via a network connection to the Resource Manager / Authorization Evaluator.
- The Resource Manager and Authorization Evaluator are installed and operated on the same system. They communicate with each other via a library interface (the aznAPI). They communicate with the Policy Server via a network connection with a dedicated application layer protocol running over SSL v3 or TLS v1.
  Note that the evaluated configuration does not include Authorization Evaluator components running on a machine separate from the Resource Manager that uses them.
- The evaluated configuration has one Policy Server and one or more Resource Manager / Authorization Evaluator systems. All Resource Manager / Authorization Evaluator systems operate independent from each other and are only connected to the central Policy Server. Load balancing and failover configurations of Resource Manager / Authorization Evaluator systems are therefore not supported in the evaluated configuration.
- The Policy Server and all the Resource Manager / Authorization Evaluator use the same operating system as a basis. Configurations using different operating system platforms for different components of the TOE are not part of the evaluated configuration.
- Communication between client systems and the TOE, the web server systems and the TOE, the LDAP server and the TOE as well as the communication between the Policy Server and the Resource Manager / Authorization Evaluator systems is protected using the SSL v3 or TLS v1 protocol with one of the ciphersuites defined in this Security Target. The use of unencrypted communication is disabled in the TOE. Also the use of version 2 of the SSL protocol is disabled for communication to client systems and target systems. Within the TOE all components are configured to use SSL v3 or TLS v1. The external LDAP server also needs to support SSL v3 or TLS v1 and configured to use either of those as its preferred protocol.
- No hardware encryption device is used. The cryptographic services are fully provided by the

software implementation of the GSKit component.

- The TOE is configured to use password based authentication and SSL client certificate based authentication for the authentication of users. Other authentication mechanisms for user authentication are disabled.
- The TOE is configured to use password based authentication for administrators that request access to the TOE via the pdadmin interface.
- The use of the Web Portal Manager component for the administration of the TOE is **not** supported. Instead only the command line interface of pdadmin and the C API are supported in the evaluated configuration.
- No Application Development Kit is installed in the evaluated configuration.
- Only LDAP is supported for the access to the directory server. Active Directory or other protocols are not supported. LDAP Replica are also not supported.

To set up the evaluated configuration compliant with the description above the user needs to follow the guidance provided in the Tivoli Access Manager for eBusiness Base Installation Guide [AMBADM].

The system components to be installed are:

1. Policy Server:
   - IBM Global Security Toolkit (GSKit)
   - IBM Directory Client
   - Tivoli Access Manager runtime
   - Tivoli Access Manager policy server
   - Fixpack 5
2. Resource Manager / Authorization Server (WebSEAL)
   - IBM Global Security Toolkit (GSKit)
   - IBM Directory Client
   - Tivoli Access Manager runtime (including the authorization evaluator)
   - Tivoli Access Manager WebSEAL server
   - Fixpack 5

Policy Server and all Resource Manager / Authorization Evaluator within an evaluated configuration use the same operating system platform (but run on different machines). Those platforms will be one of the following:

- AIX 5.2
- Solaris 8
- Windows 2000 Advanced Server SP3
- SuSE Linux Enterprise Server 8

2.8. TOE Boundary

Figure 3 shows the boundary of the TOE. It shows that Policy Server, the Authorization Server, as the Resource Manager / Authorization Evaluator, the Master Authorization Policy database as well as the Replica Authorization Policy database are part of the TOE. It also shows that the Directory Server, the Client system as well as the Web Application Server are all part of the TOE environment.

2.9. TOE Security Model

The TOE Security Model has the following components:

1. A User Registry (LDAP). **Note: The LDAP Server itself is not part of the TOE!**
   The user registry contains all users and groups allowed to participate in the Tivoli Access Manager secure domain.
2. A Master Authorization Policy Database
   This database contains a representation of all resources in the domain (= protected object space). The security administrator can define Access Control Lists (ACL) and Protected Object Policies (POP) for those resources that require protection
3. An Authentication Service
   This service verifies the claimed identity of a user. All users that are going to be authenticated must have an entry in the User Registry. Users that are not authenticated can only access resources where the resource and the type of access are allowed for unauthenticated users.

   When a user is successfully authenticated a set of identification information (credentials, which include user identity, group membership and security attributes) is extracted from the information stored in the User Registry and maintained for the user within the TOE.
4. An Authorization Service
   For each attempted access this service verifies if the user attempting access has the right to access the resource in the intended way. This is done by comparing the user's credentials with the rules defined for the resource in the Authorization Policy Database. The Authorization Service is called by the Resource Manager and return "yes" or "no" depending on the evaluation of the rules from the database.
5. An Audit Service
   A configurable number of events will generate an audit record that allows to trace when the event has happened and which user caused the event.
6. An Administration Interface
   This interface is used to administer the TOE (including the WebSEAL systems).
7. Configuration Files
   A number of configuration files are used by the components of the TOE. The settings of those files define the behavior of the security functions of the TOE. Configuration files need to be defined correctly at the installation time of the TOE to ensure a secure initial configuration. The administrator will maintain configuration files either by using the administration commands of the pdadmin interface or directly by editing the files.

The TOE has the following security functionality:

1. Authentication of administrators
   Administrators allowed to administer the TOE via the pdadmin interface are identified and authenticated.
2. Authentication of users

Users are identified and authenticated. The TOE itself performs authentication of users either by a userid/password combination or by authenticating the client systems using client certificates **Note: This implies some trust into the client system to protect the user's authentication data. This is expressed in the assumptions and requirements for client systems.**

3. Assigning user credentials to authenticated users
   The credentials of authenticated users are created from the information stored in the user's record in the user registry within the Directory Server.
4. Access Control to protected objects of the Web Application Server
   Those objects are protected as defined in the policy defined by an administrator on the Policy Manager.
5. Access Control to TOE management objects
   A flexible management model allows limit the administration capabilities of administrative users to defined sections of the protected object space
6. Auditing of activities
   The TOE is capable of auditing defined events.

3. TOE Security Environment

3.1. Assumptions

The description of assumptions describes the security aspects of the environment in which the TOE will be used or is intended to be used. This includes the following:

- information about the intended usage of the TOE, including such aspects as the intended application, potential asset value, and possible limitations of use; and
- information about the environment of use of the TOE, including physical, personnel, and connectivity aspects.

A.NOBYPASS	It has to be ensured that protected resources can not be accessed in a way that bypasses the TOE. All internal and external access attempts to protected resources have to be channeled through the TOE.
A.CLIENT_KEYMAN	Users have to administer and protect private keys of their client system used for authentication and key exchange with the TOE in a secure way. This includes the secure generation of strong keys as well as the protection of private keys against any kind of unauthorized access and use.
A.CLIENT_PWMAN	Users have to protect their passwords used for authentication to the TOE such that no unauthorized access to them is possible.
A.ADM_PWMAN	Administrators have to protect their passwords used for authentication to the TOE such that no unauthorized access to them is possible.
A.PHYS_PROT	The machines running the TOE software need to be protected against unauthorized physical access and modification. All machines running parts of the TOE software require this protection.
A.SINGLE_APP	Any machine used to run all or a part of the TOE software are assumed to be used solely for this purpose and are not used to run other application software except those required for the management and maintenance of the underlying operating system and hardware.
A.OS_CONF_MGMT	The operating system of the machines running the TOE are assumed to be configured and maintained by trained and trustworthy personnel such that the operating system provides a reliable basis for the operation of the TOE software. Especially it is assumed that the operating system is configured such that no unauthorized access to functions provided by the operating system software (including network daemons) is possible either locally or via any network connection.
A.ADMIN	The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the administrator documentation. They

	will perform administration activities from a secure environment using terminals and / or workstations they trust via secured connections to the Policy Server. All administrative commands themselves will be executed on the Policy Server.
A.USER	Users of the TOE are not hostile and trying to deliberately attack the TSF. They also carefully protect their authentication information within their operating environment.
A.DIR_PROT	The directory server used by the TOE provides protection mechanism against unauthorized access to TSF data stored in the directory. This includes the requirement for authentication when accessing user entries and the configuration to use SSL v3 or TLS v1 as the preferred protocol to protect the communication links.

3.2. Threats

The assumed security threats in the TOE environment are listed below.

The **assets** to be protected by the TOE comprise the information stored, processed or transmitted by the TOE. The term “information” is used here to refer to all data held within the TOE or parts of the TOE, including data in transit between parts of the TOE, if appropriate. This does not include resources managed in the IT environment of the TOE representing the target of access requests, since the TOE is only responsible for the access decision making but not the enforcement of the access control in the IT environment (the access control decision is in turn achieved by relying on the access control policy rules, which again is information held within the TOE and therefore has to be protected by the TOE). The TOE counters the general threat of unauthorized access to information, where “access” includes disclosure, modification and destruction.

The assets to be protected are therefore:

- The Authorization Policy database (including replica of this database) which stores the data upon which the access decision is taken
- The content of protected resources when passed through the TOE

The **threat agents** can be categorized as either

- unauthenticated users of the TOE, i.e. individuals who are not known to the system but may access resources available to unauthenticated users
- authorized users of the TOE, i.e. individuals who have successfully authenticated themselves to the TOE and may access resources as defined by the access control policy via the Resource Manager component
- authorized administrators of the TOE, i. e. individuals who have successfully authenticated themselves to the TOE and may perform administrative tasks via the PD Admin interface within their administrative responsibilities

The threat agents are assumed to originate from a well managed user community in a non-hostile working environment, and hence the product protects against threats of inadvertent or casual attempts to breach the system security. The TOE is not intended to be applicable to circumstances in which protection is required against determined attempts by hostile and well funded attackers to breach system security. An example of such an environment is a company intranet well protected from

external attacks and with an overall user community (including unauthenticated users) that can be assumed to be non-hostile. System administrators of the TOE as well as those for the underlying operating system and the Directory Server used are assumed to be trustworthy and follow the instructions provided to them with respect to the secure configuration and operation of the systems under their responsibility.

The threats listed below are grouped according to whether or not they are countered by the TOE. Those that are not countered by the TOE are to be countered by environmental or external mechanisms.

3.2.1 Threats to be countered by the TOE

T.BYPASS	An attacker accesses protected resources of the TOE in a way that bypasses the TSF, exploiting non-TSF portions of the TOE.
T.UAACTION	An undetected violation of the TSP may be caused as a result of an attacker (possibly, but not necessarily, a person allowed to use the TOE) attempting to perform actions that the individual is not authorized to do.
T.UAUSER	An attacker (possibly, but not necessarily, a person allowed to use the TOE) may impersonate an authorized user of the TOE. This includes the threat of an authorized user that tries to impersonate as another authorized user without knowing the authentication credentials.
T.COM_ATT	An attacker intercepts the communication between the TOE and an external entity or between different parts of the TOE in order to get access to confidential information, to impersonate as an authorized user or part of the TOE or to manipulate the data transmitted between the TOE and an external or internal entity.

3.2.2 Threat to be countered by the TOE environment

TE.GET_CRED	An attacker may obtain credentials within the TOE environment that allow him to impersonate an authorized TOE user, or get unauthorized access to the directory information.

3.3. Organizational Security Policies

The following organizational security policies are deemed appropriate in a security environment for the TOE:

P.AUTHORIZED_USERS	Only those users who have been authorized to access web resources protected by the TOE may access those resources after they have been successfully authenticated (unless a protected web resource is defined to be accessible by

	unauthenticated users, in which case no prior authentication is required).
P.AUTHORIZED_ADMIN	Only administrators authorized for access to defined management resources of the TOE may access those resources after they have been successfully authenticated.
P.NEED_TO_KNOW	The system must allow to limit the access to, modification of, and destruction of the information in protected web resources to those authorized users which have a “need to know” for that information.
P.ACCOUNTABILITY	The administrators of the system shall be held accountable for their actions within the system.
P.ADM_DELEGATION	Specific administration tasks as well as management operations to defined subsets of the web resources protected by the TOE may be delegated to administrators that are only allowed to perform the management tasks within their defined area of responsibility and are not able to extend this area themselves.

4. Security Objectives

This section defines the security objectives of the TSF and its supporting environment. Security objectives are categorized as IT security objectives for the TOE or the IT environment as well as non-IT security objectives to be met by organizational means in the TOE environment.

4.1. Security Objectives for the TOE

O.AUTHORIZATION	The TSF must ensure that only authorized administrators and users gain access to the TOE and the resources it protects. **Note:** The access control rules may also allow unauthenticated users to access resources explicitly defined to be accessible to unauthenticated users.
O.AUTHENT_ADMIN	The TSF must authenticate administrators which request access to the TOE and its resources. **Note:** The access policy rules may define some resources that can be accessed by everybody including unauthenticated users. Those resources are not seen as part of the resources protected by the TOE.
O.AUTHENT_USER	The TSF must authenticate users which request access to resources protected by the TOE unless the resource is allowed to be accessed by unauthenticated users.
O.ACCESS_DECISION	The TSF must base its access decision on defined access control rules and policies defined by administrators.
O.ACC_ADM	The TSF must control the definition and management of access control rules and policies and restrict those activities to authorized administrators. The TSF must allow to restrict the rights of some administrators to define access control rules for a subset of the protected object space only.
O.AUDITING	The TSF must record the security relevant actions of users of the TOE. The TSF must present this information to authorized administrators.
O.MANAGE	The TSF must provide all the functions and facilities necessary to support the authorized administrators that are responsible for the management of TOE security.
O.ENFORCEMENT	The TOE must ensure that all access requests passed to him are evaluated using the access control rules and policies defined in the Master Authorization Policy. If this requires proper authentication of the user submitting the request the TOE will perform this authentication.[1]

[1] The originally stated text was an objective taken from the evaluated Controlled Access Protection Profile.

O.SEC_COM	Communication of TOE external entities with the TOE as well as communication between physically distributed parts of the TOE must be secured to ensure the integrity and confidentiality of the communication.
O.RSA_KEYGEN	The TOE must be able to generate RSA key pairs with 1024 bit key length.

4.2. IT Security Objectives for the Environment

4.2.1 IT Security Objectives for the underlying operating system

OE.OS_TIME	The operating system within IT environment must provide a reliable time source.
OE.OS_CONFFILE_PROT	The operating system within the IT environment must provide protection of the configuration files against unauthorized access.

4.2.2 IT Security Objectives for the Directory Server

OE.DS_ACCESS_CNTRL	The Directory Server must provide access control mechanisms to prohibit unauthorized access to directory entries. This access control must also be enforced when importing and exporting data.
OE.DS_AUTHENT	The Directory Server must identify and authenticate users that request access to directory entries.

4.3. Non-IT Security Objectives for the Environment

OE.INSTALL	Those responsible for the TOE must ensure that the TOE is delivered, installed, managed, and operated in a manner which maintains IT security objectives.
OE.PHYSICAL	Those responsible for the TOE must ensure that those parts of the TOE critical to security policy are protected from physical attack, which might compromise IT security objectives.
OE.CREDEN	Those responsible for the TOE must ensure that all access credentials, such as passwords or other authentication information, are protected by the users in a manner which maintains IT security objectives
OE.OS_OPERATE	The operating system of the machines running the TOE is assumed to be configured and maintained such that it provides a reliable basis for the operation of the TOE software. The operating system is configured such that no unauthorized

	access to functions provided by the operating system software (including network daemons) is possible either locally or via any network connection. Any machine used to run all or a part of the TOE software is used solely for this purpose and is not used to run other application software except those required for the management and maintenance of the underlying operating system and hardware.
OE.SEC_INTEGRATE	Those responsible for the TOE must ensure that the TOE is integrated into the overall system in a way that prohibits direct access to resources to be protected by the TOE in a way that bypass the TOE and its security functions.
OE.USER	Those responsible for the TOE shall control the user community that can request access to resources protected by the TOE. This includes a configuration where the client systems allowed to submit requests to the TOE is controlled (e. g. a company internal network with a known and controlled user community protected against unauthorized access from external networks).

5. IT Security Requirements

This chapter defines the security requirements for the TOE as well as the TOE environment.

Chapter 5.1 defines the security requirements for the TOE itself, separated into security functional requirements and security assurance requirements. Those requirements use the appropriate Common Criteria functional and assurance components with all the required operations performed. Operations are marked in bold and italics. In addition some refinements of SFRs as defined in the Common Criteria had to be made. Those are marked in bold, italics and underlined.

Chapter 5.2 defines the security requirements for the IT environment, separate for each component within the environment. Here only security functional requirements are defined using the Common Criteria functional components where appropriate. Not all operations have been performed for those components to allow for the necessary flexibility in the selection of the products used in the environment. The security functional requirements defined in this section try to identify a minimum set of requirements needed to get a secure Tivoli Access Manager Environment.

Chapter 5.3 then defines the security requirements for the non-IT environment. They are expressed without using the Common Criteria functional components because they are not suitable to describe non-technical security requirements

5.1. TOE Security Requirements

5.1.1 TOE Security Functional Requirements

FAU_GEN.1 Audit data generation

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 defined events:***

WebSEAL:

- ***successful and unsuccessful authentication attempts with a userid / password combination***
- ***failed authorization for access to a protected resource***
- ***user change of password***
- ***user locking***

Policy Server

- ***creation of user by administrator***
- ***user locked by administrator***
- ***user unlocked by administrator***
- ***all commands of administrators that result in a modification of the policy database***

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, ***for auditing administrator commands: parameters passed to the command***

FAU_GEN.2 User identity association

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

FAU_SAR.1 Audit review

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

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

Application Note: The TOE does not provide a direct interface to read the audit trail. Instead the administrator has to use a tool outside of the TOE to read the audit records. The responsibility of the TOE with respect to the requirements of FAU_SAR.1 are related to set the access rights to the audit files and to store the information in the audit files in a way suitable for interpretation even when read with a program like an editor.

FAU_SEL.1 Selective audit

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

a) object identity, host identity

b) audit event category (authn, azn, mgmt).

Application Note: The audit categories can be defined on a per-server basis, which satisfies the selection of "host identity" in a). POP can be used to define auditing on a per object basis.

FAU_STG.1 Protected audit trail storage

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

FAU_STG.1.2 The TSF shall be able to ***prevent*** modifications to the audit records.

Application Note: The protection from unauthorized deletion is achieved with the TOE setting the access permissions to the audit files appropriately. Prevention of modifications also is based on the access rights to the audit files and by the fact that the TOE itself does not provide any function that could be used to modify the audit records once stored in the audit file. This security functional requirement of course also relies on the appropriate protection of the TOE itself against unauthorized access in the TOE environment.

FAU_STG.4 Prevention of audit data loss

FAU_STG.4.1 The TSF shall ***rollover to a new audit file*** and ***continue operation*** if the audit trail is full.

Application Note: By specifying a positive value for the "rollover_size" parameter in the audit configuration the administrator can define that the TOE saves the current log file under a defined name that includes a timestamp and rolls over to a new audit file when the defined limit size for the log file is reached. When a log file with the standard name already exists when the TOE is started, the TOE will append audit records to the end of the existing file until the defined limit size is reached.

FCS_CKM.1(1) Cryptographic key generation (Symmetric algorithms)

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm ***as defined in the SSL v3 and TLS v1 standard*** and specified cryptographic key sizes ***128 bit (RC4), 168 bit (TDES)*** that meet the following: ***generation and exchange of session keys as defined in the SSL v3 and TLS v1 standard with the cipher suites defined in FCS_COP.1(2).***

Application Note: Generation of symmetric keys is defined in section 6.3 of the TLS v1 standard and section 6.2 in the SSL v3 standard. Tivoli Access Manager for e-Business (in this the GSKIT component) also supports SSL v2, but this is seen as being not part of the evaluated configuration. The TOE in the evaluated configuration rejects any attempts of a client system to set up an unencrypted connection, a connection using the SSL v2 protocol or using the SSL v3 or TLS v1 protocol with another cipher suite than the ones listed in FCS_CKM.1. Clients are therefore requested to support SSL v3 or TLS v1 with one of those cipher suites. FMT_MSA.2 is associated with this requirement to ensure that the values generated as keys are "secure" in the sense that they can not be easily guessed. Weakness in keys caused by the cryptographic algorithm or deficiencies of the key generation process are not considered.

FCS_CKM.1(2) Cryptographic key generation (RSA)

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm ***product specific*** and specified cryptographic key sizes ***1024 bit*** that meet the following: ***not specified***

Application Note: The TLS v1 and SSL v3 specification do not define how the RSA key pair is generated. This is up to the implementation. Almost all implementations of the TLS v1 and SSL v3 standard have their own algorithm for RSA key pair generation (if they support cipher suites that use RSA). Therefore the key generation and algorithm and the standard to follow are not defined here. Only the required key size is specified.

FCS_CKM.2(1) Cryptographic key distribution (RSA public keys)

FCS_CKM.2.1 The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method ***digital certificates*** that meets the following: ***certificate format as defined in X.509 Version 3***.

Application Note: This requirement addresses the exchange of public RSA keys as part of the SSL client and server authentication. For a definition of the certificate format see [X.509].

FCS_CKM.2(2) Cryptographic key distribution (Symmetric keys)

FCS_CKM.2.1 The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method ***Secure Socket Layer handshake using RSA encrypted exchange of session keys*** that meets the following: ***SSL Version 3 (Internet Draft dated November 1996, Netscape Communication) resp. TLS v1 (RFC 2246)***.

Application Note: This requirement addresses the exchange of SSL session keys as part of the SSL handshake protocol.

FCS_COP.1(1) Cryptographic operation (RSA)

FCS_COP.1.1 The TSF shall perform ***digital signature generation and digital signature verification*** in accordance with a specified cryptographic algorithm ***RSA*** and cryptographic key sizes ***1024 bit*** that meet the following: ***SSL Version 3 (Internet Draft dated November 1996, Netscape Communication) resp. TLS v1 (RFC 2246)***.

Application Note: This requirement addresses the RSA digital signature generation and verification operations using the RSA algorithm as required by the SSL session establishment protocol (provided a cipher suite including RSA is used). Note that the details of the signature format like the use of the PKCS#1 block type 1 and block type 2 are defined in the SSL Version 3 standard.

FCS_COP.1(2) Cryptographic operation (Symmetric operations)

FCS_COP.1.1 The TSF shall perform ***encryption and decryption*** in accordance with a specified cryptographic algorithm ***RC4 and TDES*** and cryptographic key sizes ***128 bit (RC4) or 168 bit (TDES)*** that meet the following: ***SSL Version 3 (Internet Draft dated November 1996, Netscape Communication) resp. TLS v1 (RFC 2246) and the following cipher suites: SSL_RSA_WITH_RC4_128_SHA, SSL_RSA_WITH_RC4_128_MD5, SSL_RSA_WITH_3DES_EDE_CBC_SHA as defined in the SSL v3 standard resp. TLS_RSA_WITH_RC4_128_SHA, TLS_RSA_WITH_RC4_128_MD5, TLS_RSA_WITH_3DES_EDE_CBC_SHA as defined in the TLS v1 standard.***

Application Note: GSKIT supports also the other the other ciphersuites that use RSA based key exchange listed in the SSLv3 or TLS v1 standard. The ciphersuites listed above are the first ones in the list of ciphersuites defined in GSKIT. If a client decides to use another – potentially less secure – ciphersuite, the TOE will not establish a session.

FDP_ACC.2(1) Complete access control

FDP_ACC.2.1 The TSF shall enforce the ***Web-Space access control policy*** on ***users as subjects and objects in the WebSEAL protected object space*** and all operations among subjects and objects covered by the SFP.

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

FDP_ACC.2(2) Complete access control

FDP_ACC.2.1 The TSF shall enforce the ***management access control policy*** on ***administrators as***

subjects and objects in the management protected object space and all operations among subjects and objects covered by the SFP.

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

FDP_ACF.1(1) Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the ***security attribute based Web-Space access control policy*** to objects based on ***access control lists (ACL) and protected object policies (POP)***.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: ***users have the requested type of access to a protected object in the Web-Space under the following conditions:***

a) the user has been successfully authenticated and

the user has the "traverse" right for all objects from the root object down the path to the requested object and

- ***the user has an entry in the ACL associated with the object that contains the requested type of access, or***
- ***the user is member of a group that has an entry in the ACL associated with the object that contains the requested type of access, or***
- ***the ACL associated with the object has an entry of type "any-other" that contains the requested type of access***

b) the user has not been authenticated and

a traverse right exist for all objects from the root object down the path to the requested object for unauthenticated users and

- ***the ACL associated with the object has both an entry of type "any-other" and an entry of type "unauthenticated" where the requested access right is contained in both entries***

The "ACL associated with the object" is the ACL of the object if the object has an explicit ACL or the ACL inherited from the next object up on the path to the root that has an explicit ACL.

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: ***none***

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the ***rules defined by the Protected Object Policy, if such a Protected Object Policy has been defined for the requested object. Protected Object Policies can deny access based on***

- ***the time-of-day***
- ***the strength of the authentication mechanism***
- ***the IP address***
- ***the Quality of Protection***.

FDP_ACF.1(2) Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the ***security attribute based management access control policy*** to objects based on ***access control lists (ACL) and protected object policies (POP)***.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: ***users have the requested type of access to a protected object in the Web-Space under the following conditions:***

c) the user has been successfully authenticated and

the user has the "traverse" right for all objects from the root object down the path to the requested object and

- ***the user has an entry in the ACL associated with the object that contains the requested type of access, or***
- ***the user is member of a group that has an entry in the ACL associated with the object that contains the requested type of access, or***
- ***the ACL associated with the object has an entry of type "any-other" that contains the requested type of access***

d) the user has not been authenticated and

a traverse right exist for all objects from the root object down the path to the requested object for unauthenticated users and

- ***the ACL associated with the object has both an entry of type "any-other" and an entry of type "unauthenticated" where the requested access right is contained in both entries***

The "ACL associated with the object" is the ACL of the object if the object has an explicit ACL or the ACL inherited from the next object up on the path to the root that has an explicit ACL.

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: ***none***

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the ***rules defined by the Protected Object Policy, if such a Protected Object Policy has been defined for the requested object. Protected Object Policies can deny access based on***

- ***the time-of-day***
- ***the strength of the authentication mechanism***
- ***the IP address***
- ***the Quality of Protection***.

FIA_AFL.1 Authentication failure handling

FIA_AFL.1.1 The TSF shall detect when ***three*** unsuccessful authentication attempts occur related to ***password based authentication attempts of users***.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall ***disable further login attempts of that user for the time defined by the administrator in the disable-time-interval configuration parameter***.

Application Note: The number of unsuccessful authentication attempts allowed before the action defined in FIA_AFL.1.2 is taken, can be configured by the administrator in the set-max-login-failures configuration parameter. The administrator guidance defines the value of 3 for the evaluated configuration.

FIA_ATD.1 User attribute definition

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: ***user name, registry identifier (distinguished name), password, list of groups the user belongs to.***

Application Note: The user's common name and surname are not seen as security attributes.

FIA_SOS.1 Verification of secrets

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

Minimum password length is 8 character

Minimum number of alphabetic character is 4

Minimum number of non-alphabetic character is 1

Maximum number of repeated characters is 2

No space character is allowed within the password.

Application Note: The conditions defined are the default settings of the parameter in the WebSEAL configuration. Those parameters are configurable by the administrator.

FIA_UAU.1 Timing of authentication

FIA_UAU.1.1 The TSF shall allow ***access as defined in the 'any-other' entry bitwise 'and' masked with the 'unauthenticated' entry in an object's ACL*** 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.

Application Note: An ACL may contain an entry that defines the access modes allowed for anonymous users, i. e. users that are not identified and authenticated. See page 99 of the WebSEAL Admin Guide for the description of the policy. As defined above access is granted only if both the 'any-unauthenticated' and the 'unauthenticated' entry allow the mode of access.

FIA_UAU.2 ***Administrator*** authentication before any action

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

Application Note: The term "user" in the CC SFR FIA_UAU.2 has been refined by "administrator" to differentiate the authentication policy of users and administrators. While there is a possibility of unauthenticated users, all administrators (which use a different interface for authentication) are required to authenticate successfully before performing any administrative action on the TOE.

FIA_UAU.5 Multiple authentication mechanisms

FIA_UAU.5.1 The TSF shall provide ***the following authentication methods for users:***

- ***Combination of user ID and password***
- ***SSL client certificate***

and the combination of user ID and password for administrators

to support user authentication.

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

administrators are authenticated by a user ID / password combination only

users are authenticated according to the following rules:

- ***the TOE first checks if the client can present a SSL client certificate. If yes, this is used for authentication of the client.***
- ***If no client certificate is presented, the client can either request user ID password authentication or a form to enter the user ID and the password is presented to the client the first time it tries to access an object in an access mode not allowed for unauthenticated users***.

Application Note: "Client-side Certificate", "Forms Authentication" and "Basic Authentication" are the only authentication methods for users used in the TOE configuration of Tivoli Access Manager.

FIA_UAU.6 Re-authenticating

FIA_UAU.6.1 The TSF shall re-authenticate the user under the conditions ***the user has connected to WebSEAL and the client browser terminates the SSL session or the values for the session cache entry lifetime timeout or the inactive-timeout for the session are exceeded***.

Application Note: This conditions for re-authentication applies for users only, not for administrators that have connected via the pdadmin interface.

FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow ***access as defined in the bitwise 'and' of the 'any-other' and 'unauthenticated' entry in an object's ACL*** on behalf of the user to be performed before the user is identified.

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

Application Note: An ACL may contain an entry that defines the access modes allowed for anonymous users, i. e. users that are not identified and authenticated.

FIA_UID.2 *Administrator* identification before any action

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

Application Note: The term "user" in the CC SFR FIA_UID.2 has been refined by "administrator" to

differentiate the authentication policy of users and administrators. While there is a possibility of unauthenticated users, all administrators (which use a different interface for authentication) are required to authenticate successfully before performing any administrative action on the TOE.

FIA_USB.1 User-subject binding

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

FMT_MOF.1 Management of security functions behaviour

FMT_MOF.1.1 The TSF shall restrict the ability to ***modify the behaviour of*** the functions ***the authentication function, the audit function, the ACL access control policy*** to ***authorized administrators***.

FMT_MSA.1 (1) Management of security attributes

FMT_MSA.1.1 The TSF shall enforce the ***management access control policy*** to restrict the ability to ***modify, delete*** the security attributes ***ACL entries*** to ***users or groups having 'control' access for the ACL*** .

FMT_MSA.1 (2) Management of security attributes

FMT_MSA.1.1 The TSF shall enforce the ***management access control policy*** to restrict the ability to ***change_default, modify, delete*** the security attributes ***ACL and POP*** to ***authorized administrators***.

FMT_MSA.2 Secure security attributes

FMT_MSA.2.1 The TSF shall ensure that only secure values are accepted for security attributes.

FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1 The TSF shall enforce the ***management access control policy and Web-Space access control policy*** to provide ***inherited*** default values for security attributes that are used to enforce the *SFP*.

FMT_MSA.3.2 The TSF shall allow the ***administrator authorized to modify the ACL of the container object*** to specify alternative initial values to override the default values when an object or information is created.

Application Note: If no ACL is attached to an object, this object inherits the ACL attached to container object that contains the object. This inheritance rule goes 'upward' in the protect object space tree until a container object with an ACL is reached. This rule is expressed with this requirement.

FMT_MTD.1 Management of TSF data

FMT_MTD.1.1 The TSF shall restrict the ability to ***modify and delete*** the ***user attribute data*** to

authorized administrators.

FMT_SMF.1 Specification of Management Functions

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

- ***user and group management***
- ***ACL and POP management***
- ***Audit management***.

Application Note: This requirement is introduced to satisfy AIS 32, Final Interpretation 065.

FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles ***users and administrators***.

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

Application Note: Administration tasks can be delegated by the initially defined administrator (sec-master) to other administrators that he has created. The tasks a specific administrator is allowed to perform can be defined on a fine-grained basis as described in chapter 6. The term 'administrator' is used in this Security Target for any administrator that has been defined to perform administrative actions via the pdadmin interface.

FPT_ITT.1 Basic internal TSF data transfer protection

FPT_ITT.1.1 The TSF shall protect TSF data from ***disclosure and modification*** when it is transmitted between separate parts of the TOE.

FPT_RVM.1 Non-bypassability of the TSP

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

FPT_TRC.1 Internal TSF consistency

FPT_TRC.1.1 The TSF shall ensure that TSF data is consistent when replicated between parts of the TOE.

FPT_TRC.1.2 When parts of the TOE containing replicated TSF data are disconnected, the TSF shall ensure the consistency of the replicated TSF data upon reconnection before processing any requests for ***access permission***.

Application Note: The documentation describes process of notification that can be send out by the policy server to all authorization servers with replica of the master database. It is not described how the policy server checks that the authorization server have received this notification (e. g. if they are not reachable at the time the notification has been sent out). As an alternative authorization servers can check for database updates by polling the policy server.

FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and a remote 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.

FTP_ITC.1.2 The TSF shall permit ***the TSF (for connection to client systems) or the remote trusted IT product (for connection to the LDAP server or the back-end system)*** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for ***the communication with client systems, the communication with the LDAP server and the communication to the back-end systems***.

Application Note: Within FTP_ITC.1.2 a refinement of the CC has been introduced. This refinement was necessary since – depending on the connection – the TOE as well as the remote trusted IT product is allowed to initiate communication via the trusted channel.

The trusted channel is established using the SSL v3 or TLS v1 protocol with client authentication. In the case of client systems the TOE acts as a server (as seen by the SSL protocol). In the case of the LDAP server and the back-end systems the TOE acts as a client (as seen by the SSL protocol). The SSL v3 and TLS v1 protocol define the client as the communication partner that initiates the communication over the trusted channel. But since the TOE plays the role of a client (for the LDAP server and the back-end systems) as well as a server (for the client systems), both the TOE as well as the remote trusted IT product may initiate the communication.

5.1.2 TOE Security Assurance Requirements

The target evaluation assurance level for the product is EAL3 [CC] augmented by ALC_FLR.1.

5.2. Security Requirements for the IT Environment

There are several components used in the IT environment of the TOE that need to satisfy a number of security requirements to ensure a secure operation of the TOE in its environment. Those security functional requirements are defined in this section, separate for each component within the TOE environment. The functional components of the Common Criteria (part 2) have been taken to describe those security functional requirements and the operations within those components have been performed whenever possible and useful. It should be noted that the security functional requirements defined here are a minimum set that need to be satisfied by those components to ensure the secure operation of Tivoli Access Manager within its intended environment. In most cases the components will have more security functions than defined here, allowing for a more flexible type of operation or even provider a higher level of security.

It should be noted that the security requirements for the components in the IT environment are not intended to be complete specifications of all security requirements of such a component. Only those requirements required by the TOE are defined. **As a result there may of course be unresolved dependencies within the requirements for each component, which need to be resolved in a Security Target for such a component in accordance with the security policy of the component.**

5.2.1 Directory Server

Summary and Justification

The TOE uses the external Directory Server to store and maintain data related to the users and

administrators of the TOE. The Directory Server is required to protect this data against unauthorized access and disclosure. This requires functions for identification and authentication as well as access control for those directory entries belonging to the TOE.

It is not required to have a Directory Server dedicated to one specific instantiation of Tivoli Access Manager for e-Business. But it is required that the subset of the directory "belonging" to a specific instantiation of Tivoli Access Manager for e-Business is protected against unauthorized access of any kind by anybody else than a server belonging to the specific instantiation of Tivoli Access Manager. If one directory server is used for several instantiations of Tivoli Access Manager or other applications, the separation or sharing of directory entries has to be defined by the organization (which also includes a clarification of the question how to manage the directory in the case of shared directory entries).

The following section defines the minimum security functional requirements for the Directory Server using Common Criteria functional requirements components. Not all operations on the components have been performed. Some of them can not be performed without placing unnecessary restrictions on the Directory Server.

Security Functional Requirements

FDP_ACC.1 Subset access control

FDP_ACC.1.1 The TSF shall enforce the ***Directory Access Control Policy*** on ***servers belonging to a Tivoli Access Manager instantiation as subject, directory entries belonging to the Tivoli Access Manager instantiation as objects and all operations that create, read, modify or delete those objects***.

FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the ***Directory Access Control Policy*** to objects based on ***successful authentication of a server that is part of the Tivoli Access Manager instantiation***.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: ***If the server is successfully authenticated to belong to the Tivoli Access Manager instantiation all access to directory records belonging to this instantiation of Tivoli Access Manager is allowed***.

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: ***None***.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*].

Application Note: The assignment in FDP_ACF.1.4 has not been performed, since it is up to the local security policy of the Directory Server to define such additional restrictions. The security of Tivoli Access Manager for eBusiness does not rely on such additional restriction but also does not demand that they should not exist.

FDP_ETC.1 Export of user data without security attributes

FDP_ETC.1.1 The TSF shall enforce the ***Directory Access Control Policy*** when exporting user data, controlled under the SFP(s), outside of the TSC.

FDP_ETC.1.2 The TSF shall export the user data without the user data's associated security attributes.

FDP_ITC.1 Import of user data without security attributes

FDP_ITC.1.1 The TSF shall enforce the ***Directory Access Control Policy*** when importing user data, controlled under the SFP, from outside of the TSC.

FDP_ITC.1.2 The TSF shall ignore any security attributes associated with the user data when imported from outside the TSC.

FDP_ITC.1.3 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: [assignment: *additional importation control rules*].

Application Note: The assignment in FDP_ITC.1.3 has not been performed, since it is up to the local security policy of the Directory Server to define those rules. The security of Tivoli Access Manager for eBusiness does not rely on those rules.

FIA_UAU.2 User authentication before any action

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

FIA_UID.2 User identification before any action

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

FMT_MSA.1 Management of security attributes

FMT_MSA.1.1 The TSF shall enforce the ***Directory Access Control Policy*** to restrict the ability to [selection: change_default, query, modify, delete, [assignment: other operations]] the security attributes [assignment: list of security attributes] to [assignment: the authorised identified roles].

Application Note: The Directory Server shall enforce an access control policy on directory entries, but the details of this policy including the rights of the individual roles have to be defined in a Security Target for the Directory Server. This TOE does not define specific requirements for the capability of this policy and the role model of the Directory Server.

FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1 The TSF shall enforce the ***Directory Access Control Policy*** to provide [selection: restrictive, permissive, other property] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the [assignment: the authorised identified roles] to specify

alternative initial values to override the default values when an object or information is created.

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following security management functions: [assignment: list of security management functions to be provided by the TSF].

FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles [assignment: the authorised identified roles].

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

5.2.2 Underlying Operating System of the TOE components

The underlying operating system is required to provide a reliable time stamp used for the generation of the date and time in the audit records generated by the TOE. In addition the TOE makes use of a configuration file containing the list of events to be audited. This file needs to be edited by an OS administrator and should be protected from unauthorized access and modifications by any other user (if such a user is installed in the underlying OS). The TOE makes no use of any other security function provided by the underlying operating system.

FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow [assignment: list of TSF-mediated actions] on behalf of the user to be performed before the user is identified.

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

FPT_STM.1 Reliable time stamps

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

FMT_MTD.1 Management of TSF data

FMT_MTD.1.1 The TSF shall restrict the ability to modify the list of events to be audited to ***OS administrators***.

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following security management functions: [assignment: list of security management functions to be provided by the TSF].

FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles ***OS administrators and*** [assignment: other authorised identified roles].

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

FPT_SEP.1 TSF domain separation

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

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

5.3. Security Requirements for the Non-IT Environment

The following requirements exists for the non-IT environment of the TOE:

RE.1 The operating system for the TOE components needs to be configured such that no unauthorized access to functions of the operating system is possible and that no unnecessary network services are supported.

RE.2 The system(s) that are part of the TOE need to be protected from unauthorized physical access and modification.

RE.3 Administrators need to be trained in the administrative functions they are supposed to perform. The training need to include clear advise how to avoid TOE configurations, ACL policies and Protected Object Policies that may give users or other administrators more rights than required.

RE.4 Persons responsible for network configuration need to ensure that they don't define a network configuration that allows to bypass the TOE when accessing objects within the protected Web-Space.

RE.5 Persons responsible for the administration of the operating systems of TOE components need to ensure that no unnecessary applications are running on top of those operating systems when the TOE software is running. For all applications other than the TOE software that are necessary to execute on the systems of the TOE they need to ensure that those applications do not interfere with the TOE software in a way that undermines the security functions of the TOE software.

RE.6 Persons responsible for client systems are required to protect the client's private key associated with the client's certificate against unauthorized access, disclosure and misuse. They are also required to "bind" the client certificate to a defined user, i. e. the certificate is used to authenticate as the user mentioned in the certificate. The client is responsible to ensure that the user is actually "bound" to the SSL v3 or TLS v1 session when this session is established.

RE.7 Users at client systems needs to protect their password from unauthorized access, disclosure and misuse.

5.4. Strength of Function Claim

This Security Target does not make different strength of function claims for different security functional requirements or security functions. For all security functions and security functional requirements based on probabilistic or permutational algorithms (except for the cryptographic algorithms (including the cryptographic hash functions) and the key generation process for those algorithms) a claim of SOF-medium is made ensuring that at least this level is met. No specific metric is defined and used for any permutational or probabilistic algorithm.

The following security functional requirements associated with functions where a strength of function claim is made contribute to requirements to be taken into account for the strength of function analysis:

- FIA_AFL.1
- FIA_SOS.1
- FIA_UAU.1
- FIA_UAU.2
- FIA_UAU.5

are the security functional requirements to be taken into account for the strength of the password based part of the authentication function.

Please note that the cryptographic algorithms including the cryptographic hash algorithms as well as the key generation process for the keys used in those algorithms are excluded from the strength of function analysis in this evaluation in accordance with the statements of the Common Criteria and the evaluation scheme.

6. TOE Summary Specification

6.1. Statement of TOE Security Functions

6.1.1 F.Audit

The TOE subsystems can be individually configured with respect to the audit functions they perform. This is done using a defined configuration file, which defines the type of events to be collected.

The TOE provides the capability to generate audit records for the following events:

1. WebSEAL (pdweb)
   a) Authentication attempts (successful and unsuccessful).
      Note: SSL handshake failures are not audited, but presenting a valid certificate that does not match a valid user in the directory is audited.
   b) Authorization failures
   c) locking of User ID (after three consecutive unsuccessful authentication attempts)
   d) User changing his password
2. Policy Server (pdmgrd):
   a) New user created
   b) User locked by administrator
   c) User unlocked by administrator
   d) All administrator actions that result in modifications to the policy database
3. GSKit

   None

Each audit event is recorded with the date and time, the identity of the user that caused the event, the type of event and the success or failure. In the case of the Policy Server also all parameters of commands issued by an administrator are audited together with the command.

6.1.2 F.Authentication

The TOE is capable to authenticate users as well as administrators. In the case of administrators successful authentication is required before they can perform any administrative action. In the case of users, defined access to defined resources may be possible for users that are not authenticated. The administrator defines which type of access to which resources is allowed for unauthenticated users.

6.1.2.1 User Authentication

“Users” are those entities that attempt to access resources via the HTTPS interface to WebSEAL.

User authentication is performed by the WebSEAL systems.

Users operate on client systems. Tivoli Access Manager supports a set of authentication mechanisms, not all of which are part of the TOE. The authentication mechanisms that are part of the TOE are:

Password based authentication

In this case the user has to provide a user ID and password to the TOE for authentication. The TOE supports Basic Authentication (as defined in the definition of the HTTP protocol) as well as forms based authentication where the TOE defines a form used by the client system, where the user can enter his user ID and password. Both methods require that a secured communication channel between the client system and the TOE has been set up using the SSL v3 or TLS v1 protocol. This ensures that passwords are protected during their transfer from the client system to the TOE. It is the responsibility of the user at the client system to check the certificate provided by the server site to be a valid certificate of the intended access before accepting the connection and transferring the password.

After having received the user ID and password from the client the TOE will contact the directory server to check for a correct user ID – password combination. If the user has been successfully authenticated the TOE will retrieve the user's credentials from the directory server and store them for the time of the session in a cache (since those credentials might be used in access decisions for the user).

The TOE allows to define a limit of successive failed login attempts when authenticating with passwords. This value is not maintained in the directory but locally on the component that performs the authentication process. The recommended value for this parameter is 3.

The TOE also has parameters an administrator can use to define the password policy. Those parameters are:

- The minimum length of a password (default: 8)
- The minimum number of alphabetic characters (default: 4)
- The minimum number of non-alphabetic characters (default: 1)
- The maximum number of repeated characters (default: 2)

Certificate-based Authentication

The TOE may also authenticate users using client certificate when setting up the SSL v3 or TLS v1 connection. It is the responsibility of the user to ensure that this certificate is bound to a dedicated user. How this is done is left to the user.

With certificate-based authentication the TOE validates the client's certificate during the establishment of the SSL session. The TOE authenticates the client by using the built-in shared library to exactly match the Distinguished name (DN) in the Subject field of the client-side certificate with an existing DN entry in the directory.

The result of successful authentication is a user identity that is then used to build a credential for that user. It is the credential that is required for the client to participate in the TOE secure domain.

Re-Authentication

When the client browser terminates the SSL session, when the session cache life time is exceeded or when the user has been inactive for an administrator definable amount of time the user is required to re-authenticate himself to the TOE when it attempts to access resources not accessible to unauthenticated users.

6.1.2.2 Authentication of Administrators

Administrators are authenticated with user ID and password. The TOE provides the pdadmin command line interface for administration tasks. To execute a single administration command the

administrator uses the following pdadmin command structure:

pdadmin [-a *admin_user*] [-p *password*] *command*

where *admin_user* is replaced by the administrator's userid and *password* is replaced by the password of the administrator. As an alternative the administrator can specify the command without the password, the system will prompt the administrator for the correct password.

To execute a set of commands the administrator can create a file containing all the commands and then issue the command

pdadmin [-a *admin_user*] [-p *password*] *filename*

where *filename* is replaced by the name of the file containing the set of commands the administrator wants to be executed. As above he may omit the password from the command line in which case he is prompted by the system for the correct password.

A third alternative is to start an interactive administrative session and using the **login** command in the form

login –a *admin_user* –p <*password*>

Again the administrator may omit the password in which case he is prompted to enter the correct password. The interactive session is terminated with the **logout** command.

The password policy defined in the section on user authentication also applies for the administrators.

6.1.3 F.Authorization

The authorization model of the TOE is based on Access Control Lists (ACL) and "Protected Object Policies" (POP). The objects that are protected (the protected object space) are defined in a tree structure that maintains three types of objects:

Web objects, which represent anything that can be addressed by an HTTP URL. This includes static Web pages as well as dynamic URLs.

Tivoli Access Manager Management Objects, which represent the management objects that can be managed through the pdadmin interface.

User-defined Objects, which can be any resource a customer defines that he wants to be access control protected by the TOE. This requires that access to those objects is guarded by applications using the authorization service through the authorization API. In the case of the TOE only those user-defined objects are considered that are defined by the WebSEAL subsystem of the TOE.

Administrators can define Access Control List policies and "Protected Object Policies" that together build the set of rules the authorization evaluator subsystem checks to decide if a user can be given the requested type of access to an object within the protected object space.

6.1.3.1 Access Control Lists (ACL)

As mentioned before the protected object space is organized as a tree with a single root, addressed by a forward slash. The next level of hierarchy consists of the Web Objects (/WebSEAL), the Tivoli Access Manager Management Objects (/Management) and (eventually) the user-defined objects.

The leaves within the tree that defines the protected object space are actually the individual objects. All branches within the tree are called "container objects" since they represent the container for all the leaves within subtree defined by the "container object".

Within the Tivoli Access Manager Management Object space, the following categories exist in the

next level of the tree:

- User management objects (/Management/Users)
- Group management objects (/Management/Groups)
- *Global Sign On (GSO) management objects (/Management/GSO)*
- Server management objects (/Management/Server)
- ACL policy objects (/Management/ACL)
- POP objects (/Management/POP)
- Configuration authorization control objects (/Management/Config)
- *Third-party authorization control objects (/Management/Action)*
- Authorization database replication control objects (/Management/Replica)

(Categories marked in italics are not used in the TOE configuration)

An administrator can create a new object with the **pdadmin object** command by defining the fully qualified location within the protected object space (provided he has the required permission).

An administrator can define and modify Access Control Lists (ACL) for objects within the protected object space. An ACL consists of:

1. A Type, which can be either “user”, “group”, “any-other” or “unauthenticated”.
   The type identifies, if the ACL defines permissions for specific user(s), group(s), any authenticated user or unauthenticated users.
2. An ID, which defines the unique identifier for the user (if of type “user”) or group (if of type “group”). ACLs of the types “any-other” or “unauthenticated” do not have such an ID.
3. A set of permissions, that define the type of access (action) allowed with the ACL. The possible permissions are:

   a – Attach

   A – Add

   b – Browse

   B – Bypass POP

   c – Control

   d – Delete

   g – Delegation

   l – List Directory

   m – Modify

   N – Create

   r – Read

   s – Server Administration

   t – Trace

   T – Traverse

   v – View

W – Password

x – Execute

For user objects, the semantics of those permissions is defined by the resource manager that uses the authorization evaluator and the access manager database for access decision. The semantics of those permissions within the WebSEAL TOE subsystem are defined later in this chapter.

The TOE also uses the ACLs to determine access to management objects within the protected object space. The semantics of those access modes for the different types of management objects are described in the next section. Please note that other access modes than the ones described with the individual types of management objects have no effect.

Additional permissions for user objects may be defined for third party applications, but this option is not used in the configuration of the TOE. (Note: the permissions l r x are used by third party applications only).

ACLs may be either explicit or inherited. Any object without an explicit ACL inherits the ACL of the container object above in the object space tree. Note that this container object may also just have an inherited ACL. The root object must always have an ACL. A default ACL for this object is set at the TOE installation and initial configuration.

The TOE uses the following rule to determine if an authenticated user has the permission for the action requested for a defined object within the protected object space

(Note: When checking for the existence of an ACL for an object, it always means checking for an explicit or inherited ACL):

1. Check that the user has the traverse permission for all container objects on the path from the root container object down to the actual object. To check this, use the steps 2 to 4 of this algorithm for all container objects on the path and the “Traverse” (T) permission.
2. Check if an ACL entry of type “user” exists for the user and the object. If this is the case, permission is granted if the requested action is defined in the ACL entry. The ACL evaluation algorithm stops if the permission is granted.
3. Check if ACL entries of type “group” exist for the groups the user belongs to and the object. If they exist, check if the requested permission is contained in at least one of those entries. If yes, the permission is granted and the evaluation algorithm stops.
4. Check if an ACL entry of type “any-other” exist for the object. If yes, check if the permission is granted within this ACL entry. If yes, permission is granted. Permission is denied, if it is not granted by the “any-other” ACL entry or if the ACL entry of type “any-other” does not exist for the object.

The TOE uses the following rule to determine if an unauthenticated user has the permission for the action requested for a defined object within the protected object space.

1. Check that unauthenticated users have the traverse permission for all container objects on the path from the root container object down to the actual object. To check this, use the steps 2 to 4 of this algorithm for all container objects on the path and the “Traverse” (T) permission.
2. Check if an ACL entry of type “unauthenticated” exists for the object. If no such ACL entry exists, access is denied and the evaluation algorithm stops.
3. Check if an ACL entry of type “any-other” exists for the object. . If no such ACL entry exists, access is denied and the evaluation algorithm stops.
4. Check if the requested access is granted in both the ACL entries of type “unauthenticated” and in

the ACL entries of type “any-other” for the object. Access is granted if the requested type of access is granted in both ACL entries. Otherwise access is denied.

As a result, a user has the requested access to an object if the two following conditions are satisfied:

1. The user has traverse permission for all container objects on the path from the root down to the object
2. The user has the requested permission being explicitly granted by the object’s ACL, which may be an explicit ACL or an inherited ACL.

6.1.3.2 Administration of the Object Space

As mentioned all objects (i. e. representation of objects) in the overall protected object space build a tree structure with a single root. The tree itself is structured into different “object spaces”.

Objects within an object space can be created by an administrator that has the “m” (modify) permission for the object container where the object is created. Objects can be deleted by an administrator that has the “d” (delete) permission for the object container of the object..

The following access rights to objects are managed by pdmgrd, since they relate to object management activities that are not controlled by the resource manager (WebSEAL):

- b (browse)
  Permission to browse objects and object spaces using the following administration commands: *objectspace list, object list, object listandshow*. Note: The command *object listandshow* requires the permission “v” in addition to “b”.

- d (delete)
  Permission to delete objects and object spaces using the following administration commands: *objectspace delete, object delete, object modify set name*. Note: The command *object modify set name* requires the permission “m” in addition to “d”.

- m (modify)
  Permission to create and modify objects and object spaces using the following administration commands: *objectspace create, object create, object modify*. Note: The command *object modify set name* requires the permission “d” in addition to “m”.

- v (view)
  Permission to show object values and attributes using the following administration commands: *object listandshow, object show*. Note: The command *object listandshow* requires the permission “b” in addition to “v”.

6.1.3.3 ACL Semantics for Management Objects

As mentioned above the TOE uses ACLs also to control access to its own management objects. The “container objects” (object spaces) that exist for TOE management objects have been identified in the previous section.

ACLs for management objects can be used to define the commands an administrator is allowed to use with a defined management object. This allows for flexible delegation of specific administrative tasks to specific administrators or administrator groups.

The following semantics for permissions exist for TOE management objects:

Management/ACL Permissions :

- d (delete)
  Permission to delete the ACL policy with the *acl delete* command. Requires “c” permission also be become effective.
- m (modify)
  Permission to create a new ACL policy using the *acl create* command
- v (view)
  Permission to find, list and show ACLs using the *acl find, acl list* and *acl show* command

Management/Action Permissions

This object defines the administrators allowed to manage custom actions. Permissions are:

- d (delete):
  Permission to delete an existing action or action group using the *action delete* and *action group delete* commands
- m (modify):
  Permission to create a new action or action group using the *action create* and *action group create* commands

Management/POP Permissions

The object defines the permissions of administrators to manage protected object policies (POP). Permissions are:

- d (delete)
  Permission to delete a POP using the *pop delete* command
- m (modify)
  Permission to create POPs and modify POP attributes using the *pop create* and *pop modify* commands.
- v (view)
  Permission to find and list POPs and show POP details using the *pop find*, *pop list* and *pop show* commands.
- B (Bypass TOD)
  Permission to overwrite the time-of-day POP attribute on an object using the *acl modify set attribute B* command.

Management/Server Permissions

The object defines the permissions of administrators to perform server management tasks. Permissions are:

- s (server):
  Permission to replicate the authorization database using the *server replicate* command.
- v (view):
  Permission to list registered servers and display server properties using the *server list* and *server show* commands.
- t (trace):
  Permission to enable dynamic trace or statistics administration using the *server task server_name trace* and *server task server_name stats* command.

Management/Config Permissions

The object defines the permissions of administrators to perform configuration management tasks. Permissions are:

- m (modify)
  Permission to define and modify the configuration using the *svrsslcfg –config* and *svrslcfg –modify* commands.
- d (delete)
  Permission to delete (unconfigure) the configuration using the *svrsslcfg –unconfig* command.

Management/Policy Permissions

The object defines the permissions of administrators to perform *policy get* and *policy set* commands to define or retrieve the overall user related policy attributes (like password restrictions, etc). Permissions are:

- v (view)
  Permission to perform the *policy get* command.
- m (modify)
  Permission to perform the *policy set* command.

Management/Replica Permissions

The object defines the permission of administrators to control the replication of the master authorization database. Permissions are:

- v (view)
  Permission to read the master authorization database

Management/Users Permissions

The object defines the permissions of administrators to manage user accounts. Permissions are:

- d (delete)
  Permission to delete a user account using the *user delete* command.
- m (modify)
  Permission to modify a user account using the *user modify* command.
- N (create)
  Permission to create a user account using the *user create* and *user import* commands.
- v (view)
  Permission to view a user account and user account details using the *user list, user list-dn, user list-gsouser, user show, user show-dn* and *user show-groups* command.
- W (password)
  Permission to reset and validate a user password using the *user modify password* and *user modify password-valid* command.

Management/Groups Permissions

The object defines the permissions of administrators to manage groups. Permissions are:

- d (delete)
  Permission to delete a group using the *group delete* command.
- m (modify)
  Permission to modify a group using the *group modify description* and *group modify remove* commands.

- N (create)
  Permission to create a group using the *group create* and *group import* commands.
- v (view)
  Permission to view a group definition using the *group list, group list-dn, user, group show, group show-dn* and *group show-members* command.
- A (add)
  Permission to a member to a group using the *group modify add* command.

Management/GSO Permissions

The object defines the permission to perform Global-Sign-On management tasks. Permissions are:

- m (modify)
  Permission to modify single-sign-on credentials and to add or remove a single-sign-on resource from a single-sign-on resource group using the *rsrcgroup modify* and *rsrccred modify* commands.
- v (view)
  Permission to view single-sign-on resources using the *rsrc list, rsrcgroup list, rsrccred list, rsrc show, rsrcgroup show* and *rsrccred show* commands.
- N (create)
  Permission to create single-sign-on resources using the *rscr create, rscrgroup create* and *rscrcred create* commands. Note: to perform the create operation, the permission “m” is also required in addition to “N”.
- d (delete)
  Permission to delete single-sign-on resources using the *rscr delete, rscrgroup delete* and *rscrcred delete* commands. Note: to perform the create operation, the permission “m” is also required in addition to “d”.

Further details on the management of the TOE are defined in the description of the function F.Management.

6.1.3.4 Protected Object Policies (POP)

Protected Object Policies contain additional conditions on the request that are passed back to the resource manager (in the case of the TOE: WebSEAL) in the case the evaluation of the ACLs for the request was positive (i. e. according to the ACL policy request is granted). For those conditions of a POP it is the responsibility of the resource manager to enforce the conditions defined by the “Protected Object Policy”.

The following attributes can be set in a “Protected Object Policy”:

- **Warning Mode**. This attribute is used for debugging purpose mainly. Possible values are: “yes” and “no”. If set to “yes”, audit records are generated that capture the result of all ACL authorization decisions that would have been made if the warning mode would have been set to “no”.
- **Audit Level**. This attribute defines the level of audit for the object. Possible values are: “permit”, “deny” and “error”. In the case of “permit”, all requests on a protected object that result in successful access are audited. In the case of “deny”, all requests on a protected object that result in denial of access are audited. In the case of “error”, all internally generated error messages resulting from the denial of access to the protected object are audited.
- **Time-of-Day**. This attribute defines the day and time conditions on the access to a protected object. This attribute is overwritten by the “B” (Bypass POP) ACL policy permission.

- **Authentication Strength**. This attribute can be used to define restrictions on the authentication method required to gain access to the protected object. This is useful, if access to the object requires a high grade of confidence in the correct authentication of the user. It is the task of the resource manager (in the case of the TOE: WebSEAL) to ensure that the user has authenticated with required authentication method before granting access to the object.
- **Network-based Authentication**. This attribute allows to control access based on the IP address of the user. This can be used to prevent access to protected objects from specific IP addresses or range of IP addresses.
- **Quality of Protection**. This attribute allows to define the required level of protection for an object. Possible values are: "Privacy" and "Integrity". In the case of "Privacy" the resource manager has to ensure that the object is transferred over an SSL encrypted communication link. In the case of "Integrity" the resource manager has to ensure that a mechanism for the protection of the integrity of the object is used when transferred.

  Note: WebSEAL as the Resource Manager in the TOE does not support the "Integrity" attribute of a POP. Therefore setting this attribute in the TOE has no effect.

6.1.4 F.Management

6.1.4.1 Tivoli Access Manager Administrators

At installation the TOE the group "iv-admin" is created with an initial administrator "sec_master". In addition a default ACL is defined for the "root" object in the protected object space. This default ACL for this object is:

Group iv-admin	TcmdbvaB
Any-other	T
Unauthenticated	T

which allows members of the group iv-admin (after installation only the user *sec_master* exists, which is a member of the *iv-admin* group) to create I, modify (m), delete (d), browse (b), view (v), attach (a) and define the bypass POP (B) attribute.

There are also default values for the different management object spaces, which are defined in the Base Administrator's Guide.

The mechanisms described in F.Authorization allow the initial administrator to define other administrators and/or administration groups and assign them the right to perform only specific administration tasks. This is achieved by assigning them the appropriate permissions for the individual management object spaces and objects within those object spaces as well as the appropriate permissions to individual objects or object spaces within the overall user object space.

6.1.4.2 User and Group Management

Users are managed using an external LDAP server. An administrator with the appropriate permission in the /Management/User object space can perform user management operations like creating users, deleting users or reset a user's password. The commands how to create and manage user accounts are defined in the Command Reference. The required access rights to perform the commands are defined in section 6.1.3.3 under "Management/Users".

Groups are managed using the *pdadmin group* set of commands. The required access rights to perform the commands are defined in section 6.1.3.3 under "Management/Groups".

Users can be assigned to more than one group. Section 6.1.3.1 describes, how access rights to objects are evaluated which includes the evaluation of access rights in the case a user belongs to more than one group.

6.1.4.3 ACL and POP Management

ACLs are managed using the *pdadmin a*cl set of commands defined in the Command Reference. The required access rights to perform the commands are defined in section 6.1.3.3 under "Management/ACL".

Protected Object Policies are managed using the *pdadmin pop* set of commands. The required access rights to perform the commands are defined in section 6.1.3.3 under "Management/POP".

6.1.4.4 TOE Certificate Management

The TOE maintains its own Certification Authority and certificate management functions for the certificates it needs for authentication and key exchange between different servers that are part of the TOE (i. e. TOE internal communication). The single pdmgrd instance within the TOE also acts as the Certification Authority for the TOE internal public key infrastructure.

During installation of the TOE management subsystem (pdmgrd instance) two RSA key pairs (key size 1024 bit) need to be generated. One of those key pairs is used as the CA key to sign certificates (PDCA key), the other one is used for authentication when setting up a SSL v3 or TLS v1 connection to another server within the TOE (pdmgrd-SSL key). The TOE management subsystem will then create a self-signed certificate for the PDCA public key and also sign the pdmgrd-SSL key using the PDCA private key.

When a new server (WebSEAL) is installed and configured, a new key pair for the server needs to be generated and the public key is signed by the TOE management subsystem using the PDCA private key. The PDCA public key certificate will then be transferred to the new server together with the server's certificate.

Certificates and private keys of TOE components are held in special files ("keyring files"). Protection of those files against unauthorized access and modification is essential for the security of the TOE.

If a server's private key is compromised (or needs to be revoked for other reasons) the administrator can do this using the chgcert command. This will require to create a new key pair, generate a certificate for the public key and invalidate the old certificate (by removing it from the keyring files of all servers that are part of the TOE).

6.1.5 F.Communication

Communication between the TOE and the LDAP server, the client system and the back-end systems are protected by SSL (in the case of the client system the use of SSL v3 resp TLS v1 is optional but mandatory for the evaluated configuration). The TOE acts as the server for client systems and acts as a client system for the communication with the LDAP server and the back-end systems.

The TOE also uses SSL v3 resp. TLS v1 to protect the communication between different servers that are part of the TOE. This communication always requires client and server authentication using digital certificates. The management of those certificates is part of the security function F.Management.

The TOE uses the GSKit subsystem to implement the SSL Version 3 and TLS version 1 protocol and the certificate management functions required. The certificate management functions are addressed under F.Management.

The SSL resp. TLS protocol itself is defined in [SSLv3] and [RFC2246]. The TOE supports the following cipher suites defined in those standards:

- CipherSuite SSL_RSA_WITH_NUL_MD5
- CipherSuite SSL_RSA_WITH_NUL_SHA
- CipherSuite SSL_RSA_EXPORT_WITH_RC4_40_MD5
- CipherSuite SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
- CipherSuite SSL_RSA_WITH_DES_CBC_SHA
- CipherSuite SSL_RSA_WITH_RC4_128_MD5
- CipherSuite SSL_RSA_WITH_RC4_128_SHA
- CipherSuite SSL_RSA_WITH_3DES_EDE_CBC_SHA
- CipherSuite TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA
- CipherSuite TLS_RSA_EXPORT1024_WITH_RC4_56_SHA
- CipherSuite TLS_RSA_WITH_RC4_128_SHA
- CipherSuite TLS_RSA_WITH_RC4_128_MD5
- CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA

The last two cipher suites are only supported with the TLS protocol.

Only SSL_RSA_WITH_RC4_128_SHA, SSL_RSA_WITH_RC4_128_MD5 and SSL_RSA_WITH_3DES_EDE_CBC_SHA for SSL v3 and TLS_RSA_WITH_RC4_128_SHA, TLS_RSA_WITH_RC4_128_MD5, TLS_RSA_WITH_3DES_EDE_CBC_SHA for TLS v1 are considered part of the evaluation. Those cipher suites are configured as the only cipher suites taken, i.e. the TOE will choose them if they are supported by the client and not establish a connection if the client does not use one of those cipher suites.

6.1.5.1 Communication with Client Systems

In the evaluated configuration the TOE will always establish a SSL connection to the client system using either the SSL v3 or TLS v1 protocol.

Depending on the configuration for authentication the TOE may require client systems to authenticate themselves with a client certificate when establishing a SSL connection. The client certificate is required to contain the user's distinguished name in the subject field. After validating the certificate, this name is used to search for a match with an existing DN entry in the LDAP registry. If such an entry is found the user is authenticated and the credentials stored in the LDAP registry for this entry will be used in access decisions.

If no client-side certificate authentication is used as the authentication process for a user, the TOE will authenticate the user with a user ID / password combination. The SSL v3 resp. TLS v1 protocol will protect the transfer of the passwords from the clients to the TOE.

A session is identified by the SSL session ID. WebSEAL will maintain a "session cache" of such sessions Ids. The maximum number of entries in this cache (which is the maximum number of concurrent sessions) is configurable (*ssl-max-entries* parameter in the *ssl* stanza of the *webseald.conf* configuration file). When the maximum number of concurrent sessions is reached, entries will be removed from the session cache using the "least recently used" algorithm.

The TOE also allows to define the maximum lifetime for an entry in the session cache (*ssl-v3-timeout* parameter in the *ssl* stanza of the *webseald.conf* configuration file).

A cache entry also has maximum lifetime timeout value

6.2. TSF that are subject to a Strength of Function Analysis

All TSF that are based on probabilistic or permutational algorithms are subject to a strength of function analysis except those that use cryptographic algorithms. Those functions are:

- F.Authentication (SOF-medium)

Within F.Authentication the mechanisms are the password mechanism and the certificate based authentication mechanism. Only the password based authentication is rated for a strength of function, since the certificate based authentication relies on cryptographic functions, which are not within the scope of the strength of function analysis.

6.3. Statement of Assurance Measures

The following table provides an overview how the assurance measures of EAL3 and ALC_FLR.1 are satisfied:

Assurance Component	**Description how the requirements are met**
ACM_CAP.3	IBM uses CMVC as the configuration management tool for source code, user supplied documentation, test plans and test cases. For design documentation, Lotus Team Room is used as the configuration management tool. Both tools are capable to authenticate users and restrict the access of individual users to configuration items. A CM plan describes the use of those tools within the development environment.
ACM_SCP.1	As mentioned above, source code, design documentation, user and administrator documentation as well as test documentation are maintained within the CM system.
ADO_DEL.1	Delivery procedures are described as part of the developer documentation. This includes also the measures taken to ensure the integrity and authenticity of the TOE during the delivery process.
ADO_IGS.1	The guidance documentation provided to the customer includes a detailed description how to install and configure the individual components that define the TOE. Additional guidance for the installation and configuration of exactly the evaluated configuration is provided as part of the guidance documentation..
ADV_FSP.1	The TSFI are identified in a separate document which points to the documents describing the different interfaces.
ADV_HLD.2	A high level design document exists that describes the internal structure of the TOE into subsystems, how the security functions of the TOE are implemented and how the subsystems contribute to the security functions.
ADV_RCR.1	Correspondence between the TSF as defined in the TOE summary specification and the functional specification as well as correspondence between the functional specification and the high level design is provided in form of commented tables that show the correspondence

Assurance Component	Description how the requirements are met
AGD_ADM.1	Administrator guidance documents exist for the Policy Server as well as for WebSEAL as the Resource Manager / Authorization Evaluator used in the TOE. They describe the administrative tasks, the commands to be used and the different management aspects.
AGD_USR.1	There is no user guidance required for the TOE, since the user can be anybody that tries to access protected resources and users will not need to know anything about the security functions of the TOE. Organizations using the TOE within their environment have to educate their users to satisfy the requirements for protecting their passwords or private keys.
ALC_DVS.1	The security measures for the IBM development environment are derived from the IBM Global documents that define the minimum requirements for the physical and organizational security.
ALC_FLR.1	Problems that are reported either from the development process or by a customer will result in a “defect” that is managed by CMVC. Defects are classified with respect to their impact and one of the possible classifications is “security”. Since all defects are tracked and managed by CMVC, it is easily possible to extract all security relevant defects, their status and what has been done to fix them.
ATE_COV.1	Testing is performed as functional verification testing using a defined test suite in accordance with defined test procedures as described in the test plan. Coverage of security functions is provided in form of a table showing which test cases test which security functions at which interface. The table shows that all security functions and their parameter are tested at the interfaces defined in the functional specification.
ATE_DPT.1	A mapping is produced that shows the mapping of test cases to details defined in the high level design. The mapping shows that those details are covered by test cases and the test cases themselves show that the TOE operates in accordance with its high level design.
ATE_FUN.1	A test plan is provided that describes the test procedures, test cases, purpose of each test and expected results. Records of actual tests performed and their results are maintained under CM.
ATE_IND.2	Independent testing is performed as part of the evaluation by the evaluation facility. The test plan and test cases as well as the TOE suitable for testing will be provided to the evaluation facility such that all the test cases can be performed by the independent evaluator.
AVA_MSU.1	An analysis of the user provided documentation describing the installation and configuration, the administrator interface and commands and the configuration files is performed to ensure that those documents are consistent and provide all the required guidance for an administrator to install, configure and administer the TOE in a secure manner.
AVA_SOF.1	A strength of function analysis is provided for the mechanisms based on permutational or probabilistic properties (except for cryptographic mechanism) to demonstrate that those mechanisms have a strength of SOF-medium or better.

Assurance Component	Description how the requirements are met
AVA_VLA.1	A process is in place and documented to search for vulnerabilities of the TOE using open sources of vulnerabilities on the Internet like CVE or CERT advisories. The results of this process are documented and provide the developer vulnerability analysis as required.

7. PP claims

This Security Target does not claim compliance with any existing Protection Profile.

8. Rationale

This chapter provides the rationale for the selection of security objectives and requirements within APP.

8.1. Security Objectives Rationale

8.1.1 Security Objectives Coverage

The mapping in Table 1 indicates how each security objective for the TOE is traced back to at least one threat or organizational security policy.

Objective	**threat / OSP**
O.AUTHORIZATION	P.AUTHORIZED_USERS, P.AUTHORIZED_ADMIN
O.AUTHENT_ADMIN	P.AUTHORIZED_ADMIN, T.UAUSER
O.AUTHENT_USER	T.UAUSER P.AUTHORIZED_USERS
O.ACCESS_DECISION	P.AUTHORIZED_ADMIN, P.AUTHORIZED_USERS
O.ACC_ADM	P.ADM_DELEGATION, P.NEED_TO_KNOW
O.AUDITING	T.UAUSER, T.UAACTION P.ACCOUNTABILITY
O.MANAGE	P.ADM_DELEGATION
O.ENFORCEMENT	T.BYPASS
O.SEC_COM	T.UAUSER, T.COM_ATT
O.RSA_KEYGEN	T.COM_ATT

Table 1: security objectives traced back to threats and organizational security policies

The mappings in Table 2 and Table 3 indicate how each security objective for the environment is traced back to at least one assumption, threat or organizational security policy.

Objective (IT Environment)	**threat / OSP / assumption**
OE.OS_TIME	P.ACCOUNTABILITY
OE.OS_CONFFILE_PROT	P.ACCOUNTABILITY
OE.DS_ACCESS_CNTRL	P.AUTHORIZED_USERS P.AUTHORIZED_ADMIN TE.GET_CRED A.DIR_PROT
OE.DS_AUTHENT	P.AUTHORIZED_USERS P.AUTHORIZED_ADMIN TE.GET_CRED A.DIR_PROT

Table 2: security objectives for the IT environment traced back to threats, organizational security policies and assumptions

Objective (non-IT Environment)	**threat / OSP / assumption**
OE.INSTALL	A.NOBYPASS A.ADMIN, A.SINGLE_APP, A.OS_CONF_MGMT
OE.CREDEN	TE.GET_CRED A.ADM_PWMAN, A.CLIENT_PWMAN, A.CLIENT_KEYMAN, A.USER
OE.PHYSICAL	A.NOBYPASS A.PHYS_PROT
OE.OS_OPERATE	A.NOBYPASS A.SINGLE_APP, A.ADMIN, A.OS_CONF_MGMT
OE.SEC_INTEGRATE	A.NOBYPASS
OE.USER	A.USER

Table 3: security objectives for the non-IT environment traced back to threats, organizational security policies and assumptions

8.1.2 Security Objectives Sufficiency

The following arguments provide justification that the security objectives are suitable to counter each single threat and that each security objective tracing back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat:

The threat **T.BYPASS**, imposing that an attacker uses non-TSF portions of the TOE to bypass the TSF, is removed by *O.ENFORCEMENT* requiring an implementation of the TOE that prevents such bypassing. Note that bypassing the TOE completely is addressed by the assumption A.NOBYPASS discussed later.

T.UAACTION imposes the threat of an attacker to perform unauthorized, TSP-violating actions without detection of those actions. This threat is removed by *O.AUDITING* requiring the TSF to log security relevant actions.

The threat of an attacker impersonating an authorized user, **T.UAUSER**, is efficiently diminished by *O.AUTHENT_ADMIN* and *O.AUTHENT_USER* requiring authentication for the TOE's users (i.e. clients and administrators) and further mitigated by *O.AUDITING* implementing audit records for security relevant actions. In addition *O.SEC_COM* prohibits that authentication credentials can be intercepted while transferred via the network connections.
Note: Impersonating as an unauthenticated user is not regarded as a problem.

The threat of an attacker intercepting and/or modifying the communication traffic between the TOE and an external entity or between physically distributed parts of the TOE, **T.COM_ATT**, is efficiently diminished by *O.SEC_COM* requiring that all communication between the TOE and external entities as well as between physically separated parts of the TOE is protected to maintain its confidentiality and integrity and *O.RSA_KEYGEN* providing for RSA key pairs to enable such communication security.

The threat of an attacker getting (within the TOE environment) credentials he can use to access information protected by the TOE, **TE.GET_CRED** is efficiently diminished by the objective

OE_CREDEN which requires that access to credentials is prohibited in the TOE environment. In addition the threat is diminished by the objectives *OE.DS_ACCESS_CNTRL* and *OE.DS_AUTHENT* which require authentication and access control for the Directory Server used by the TOE.

The following arguments provide justification that the security objectives are suitable to cover each single organizational security policy, that each security objective that traces back to an OSP, when achieved, actually contributes to the implementation of the OSP, and that if all security objectives that trace back to an OSP are achieved, the OSP is implemented.

P.ACCOUNTABILITY requires accounting for actions of administrators and access decision requests made by any user. This is covered by *O.AUDITING* containing the requirement of audit records for those very actions. The creation of audit records is supported by the environment as required in *OE.OS_TIME* by providing an accurate time source to be included in those records. In addition *OE.OS_CONFFILE_PROT* supports this policy and requires that the audit configuration file is protected against unauthorized access.

The TOE shall allow the delegation of administrative tasks to manage only access control policy rules related to a dedicated subset of objects (targets) as in **P.ADM_DELEGATION**. This is implemented by *O.ACC_ADM* providing means to control the (management) access to certain access control policy rules by, again, access control policy rules (in this case, the targets of an access control decision request initiated by an administrator are access control policy rules related to a certain object space in the TOE environment) as well as *O.MANAGE* which requires the existence of functions to perform those management activities.

The organizational policy **P.AUTHORIZED_ADMIN** requires that administrators can only access the management resources they are authorized to access. This requires on the one hand the authorization and an access control decision of the TOE, as provided by *O.AUTHORIZATION* and *O.ACCESS_DECISION* – on the other hand, this decision is based on the authentication of administrators as required in *O.AUTHENT_ADMIN*. In addition also the Directory Server used by the TOE requires a policy for authorized administrators as expressed by *OE.DS_ACCESS_CNTRL* and *OE.DS_AUTHENT*.

Similarly, **P.AUTHORIZED_USERS** requires that protected resources in the IT environment, which are not defined to be accessible by unauthenticated users, can only be accessed by users that are authorized to do so. While the access control decision is again performed by the TOE as implemented by *O.AUTHORIZATION* and *O.ACCESS_DECISION*, the TOE is responsible for the authentication of users as provided by *O.AUTHENT_USERS*. In addition also the Directory Server used by the TOE requires a policy for authorized users as expressed by *OE.DS_ACCESS_CNTRL* and *OE.DS_AUTHENT*.

Similar to the possibility of delegation for the administration of dedicated parts of the access control policy rules in P.ADM_DELEGATION, **P.NEED_TO_KNOW** requires that access of authorized users to protected objects can be limited to only dedicated objects. This is achieved by the same means, i.e. *O.ACC_ADM* allowing specifying which subjects may access which objects using access control policy rules.

The following arguments provide justification that the security objectives for the environment are suitable to cover each single assumption, that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when achieved, actually contributes to the environment achieving consistency with the assumption, and that if all security objectives for the environment that trace back to an assumption are achieved, the intended usage is supported.

The assumption **A.NOBYPASS** is covered by the *OE.SEC_INTEGRATE* which requires that the TOE is integrated into the overall system in a way that prohibits any bypass of the TOE access control functions, *OE.INSTALL* which requires the correct installation and secure operation of the

TOE, *OE.PHYSICAL* which requires physical protection for all parts of the TOE and *OE.OS_OPERATE* which requires proper configuration of the underlying operating system for the machines the TOE is running on.

The assumption on physical protection for the underlying machine of the TOE, **A.PHYS_PROT**, is covered by *OE.PHYSICAL* requiring protection of those machine(s) from unauthorized physical access.

The assumption **A.ADM_PWMAN** on the protection of authentication credentials by administrators of the TOE is achieved by *OE.CREDEN* requiring that appropriate measures for the protection of access credentials are ensured by the responsible personnel.

A.ADMIN assumes that administrators of the TOE and the underlying systems are trained, trustworthy and follow the guidance. This is covered by *OE.INSTALL* requiring competent and trustworthy administrators that deliver, install, manage and operate the TOE in a manner which maintains the IT security objectives and by *OE.OS_OPERATE* which makes dedicated requirements on the operation and configuration of the underlying machines hosting the TOE application.

The assumption **A.CLIENT_PWMAN** on the protection of authentication credentials by users being the subjects of access control policy rules is achieved by *OE.CREDEN* requiring that appropriate measures for the protection of access credentials are ensured by the responsible personnel.

The assumption **A.CLIENT_KEYMAN** on the protection of client keys used for authentication and exchange of a SSL session key is achieved by *OE.CREDEN* requiring that appropriate measures for the protection of access credentials are ensured by the responsible personnel

A.OS_CONF_MGMT assumes the reliable configuration and maintenance of the underlying operating system, emphasizing on the prevention of unauthorized (local or remote) access to operating system functions including network daemons. This is achieved by *OE.INSTALL* and *OE.OS_OPERATE* which includes the demand for an appropriate installation, configuration and maintenance of the underlying operating system.

The assumption **A.SINGLE_APP** assumes that the machines used to run the TOE are not used for other applications except those required for the management and maintenance of the TOE. This is achieved by the objectives *OE.INSTALL* which requires the installation and maintenance of the TOE in accordance with its IT security objectives and *OE.OS_OPERATE* which demands that all machines running TOE software are solely used for this purpose.

The assumption **A.DIR_PROT** assumes that protection features for the directory server used by the TOE exist, which prohibit unauthorized access to directory entries. This is achieved by the objectives *OE.DS_AUTHENT* requiring the directory server to perform user identification and authentication and *OE.DS_ACCESS_CNTRL* requiring the directory server to control the access to directory entries.

The assumption **A.USER** is addressed by the objectives *OE.USER* which requires that the persons responsible for the TOE control the user community that can request access to resources protected by the TOE and by the objective *OE.CREDEN* requiring users to protect their access credentials such that no other person can access and use them.

8.2. Security Requirements Rationale

This chapter provides the rationale for the selection of security requirements within APP. In addition to this rationale, chapter 5 includes application notes for several security functional requirements to further improve the interpretation of those requirements with respect to an APP-conformant implementation of the TOE.

8.2.1 Security Requirements Coverage

The following tables illustrate which security objectives are implemented by which security functional requirements. Table 4 indicates how each TOE security functional requirement can be traced back to at least one security objective for the TOE, Table 5 indicates how each functional security requirement for the IT environment can be traced back to at least one security objective for the environment.

SFR	**Objective**
FAU_GEN.1	O.AUDITING
FAU_GEN.2	O.AUDITING
FAU_SAR.1	O.AUDITING O.MANAGE
FAU_SEL.1	O.AUDITING O.MANAGE
FAU_STG.1	O.AUDITING
FAU_STG.4	O.AUDITING
FCS_CKM.1(1)	O.SEC_COM
FCS_CKM.1(2)	O.SEC_COM
FCS_CKM.2(1)	O.SEC_COM
FCS_CKM.2(2)	O.SEC_COM
FCS_COP.1(1)	O.SEC_COM
FCS_COP.1(2)	O.SEC_COM
FDP_ACC.2(1)	O.ACCESS_DECISION
FDP_ACC.2(2)	O.ACC_ADM O.ACCESS_DECISION
FDP_ACF.1 (1)	O.ACCESS_DECISION
FDP_AFC.1 (2)	O.ACC_ADM O.ACCESS_DECISION
FIA_AFL.1	O.AUTHENT_ADMIN
FIA_ATD.1	O.ACCESS_DECISION O.AUTHENT_ADMIN O.AUTHORIZATION
FIA_SOS.1	O.AUTHENT_ADMIN O.AUTHENT_USER
FIA_UAU.1	O.AUTHENT_USER O.AUTHORIZATION
FIA_UAU.2	O.AUTHENT_ADMIN O.AUTHORIZATION
FIA_UAU.5	O.AUTHENT_USER

SFR	Objective
FIA_UAU.6	O.AUTHENT_USER
FIA_UID.1	O.AUTHENT_USER O.AUTHORIZATION
FIA_UID.2	O.AUTHENT_ADMIN O.AUTHORIZATION
FIA_USB.1	O.ACCESS_DECISION O.AUTHORIZATION
FMT_MOF.1	O.ACCESS_DECISION O.AUDITING O.AUTHENT_ADMIN O.ACC_ADM O.MANAGE
FMT_MSA.1(1)	O.ACCESS_DECISION O.ACC_ADM O.MANAGE
FMT_MSA.1(2)	O.ACCESS_DECISION O.ACC_ADM O.MANAGE
FMT_MSA.2	O.SEC_COM
FMT_MSA.3	O.ACCESS_DECISION O.ACC_ADM O.MANAGE
FMT_MTD.1	O.MANAGE
FMT_SMF.1	O.ACCESS_DECISION O.AUDITING O.AUTHENT_ADMIN O.ACC_ADM O.MANAGE
FMT_SMR.1	O.ACCESS_DECISION
FPT_ITT.1	O.SEC_COM
FPT_RVM.1	O.ENFORCEMENT
FPT_TRC.1	O.ACCESS_DECISION
FTP_ITC.1	O.SEC_COM

Table 4: SFRs for the TOE traced back to objectives for the TOE

SFR (environment)	Objective (environment)
Operating System	
FIA_UID.1	OE.OS_CONFFILE_PROT
FPT_STM.1	OE.OS_TIME
FMT_MTD.1	OE.OS_CONFFILE_PROT
FMT_SMF.1	OE.OS_CONFFILE_PROT
FMT_SMR.1	OE.OS_CONFFILE_PROT
FPT_SEP.1	OE.OS_CONFFILE_PROT
Directory Server	
FDP_ACC.1	OE.DS_ACCESS_CNTRL
FDP_ACF.1	OE.DS_ACCESS_CNTRL
FDP_ETC.1	OE.DS_ACCESS_CNTRL
FDP_ITC.1	OE.DS_ACCESS_CNTRL
FIA_UAU.2	OE.DS_AUTHENT
FIA_UID.2	OE.DS_AUTHENT
FMT_MSA.1	OE.DS_ACCESS_CNTRL
FMT_MSA.3	OE.DS_ACCESS_CNTRL
FMT_SMF.1	OE.DS_ACCESS_CNTRL
FMT_SMR.1	OE.DS_ACCESS_CNTRL

Table 5: SFRs for the environment traced back to objectives for the environment

8.2.2 Security Requirements Sufficiency

The following arguments provide justification for each security objective for the TOE that the TOE security requirements are suitable to meet and achieve that security objective.

O.AUTHORIZATION requires that only authorized administrators and users gain access to the TOE and the resources it protects. This is achieved by the authentication of administrators as defined in *FIA_UAU.2* and *FIA_UID.2* and users as defined in *FIA_UAU.1* and *FIA_UID.1* supported by *FIA_ATD.1* which defines user attributes used in the access control decisions and *FIA_USB.1* which binds subjects to users.

O.ACCESS_DECISION requires that access control decisions regarding access of authenticated administrators and users to objects are based on the identity of those administrators and users and made in accordance with the access control policy rules held by the TOE. *FMT_SMR.1* establishes the roles administrator and user. Access decisions are made according to two access control policies: the Web-Space access control policy as defined by *FDP_ACC.2 (1)* and the management access control policy as defined by *FDP_ACC.2 (2)*. The requirements *FDP_ACF.1 (1)* and *FDP_ACF.1 (2)* define the access control SFPs that lay out the base rules for access control decisions related to administrators who want to access objects managed by the TOE and users who want to access objects managed in the IT environment. Those rules relate to a data base of initiator security attributes, which is reflected by the requirement *FIA_ATD.1* demanding the maintenance of security attributes for

users. *FIA_USB.1* guarantees that those security attributes can be associated with the subjects acting on behalf of those users. *FMT_SMF.1* introduces a security function to manage the access control policy rules, which is restricted to be accessible only by administrators by *FMT_MOF.1*. *FMT_MSA.1(1), FMT_MSA.1(2)* and *FMT_MSA.3* refine the management of those rules. In addition *FPT_TRC.1* ensures the consistency of the replicated database for the access control policies and is therefore also required to achieve the objective.

O.AUDITING requires audit records for all actions performed by administrators as well as for all access control decisions made by the TOE related to administrators or users as initiators and the ability of administrators to inspect those records. *FAU_GEN.1* specifies the kind of audit events that is to be recorded, *FAU_GEN.2* ensures that those records are associated with the originating user identity (as far as possible). *FAU_SEL.1* allows selective auditing to the discretion of the administrator of the TOE. *FAU_SAR.1* implement the requirements for the later analysis of audit records by authorized administrators. *FAU_STG.1* protects the audit records against modification, and *FAU_STG.4* regulates the behavior of the TSF in case of exhausted audit space. *FMT_SMF.1* enables the management of the audit functionality and *FMT_MOF.1* restricts it to authorized administrators, *FMT_MTD.1* for the operating system in the IT environment enables the administrator to select which audit events are to be audited.

O.MANAGE requires the TSF to provide all required functions to support administrators to manage the TOE. The areas of management are defined in *FMT_SMF*.1. This includes management of the audit functions as defined by *FMT_MOF.1, FAU_SEL.1* and *FAU_SAR.1* (supported by *FMT_MTD.1* in the operating system as part of the environment), user and group management as defined by *FMT_MTD.1*, ACL and POP management as defined by *FMT_MOF.1, FMT_MSA.1 (1), FMT_MSA.1 (2)* and *FMT_MSA.3*.

O.AUTHENT_ADMIN requires that administrators, being initiators of access control decision requests, must be authenticated by the TOE. This is implemented by *FIA_UID.2* requiring identification and *FIA_UAU.2* requiring authentication before any action other than authentication can be performed on behalf of an administrator. Password are stored in the initiator security attribute data base for each administrator (*FIA_ATD.1*). *FIA_SOS.1* imposes a “password policy” for secrets used to prove authenticity, while *FIA_AFL.1* limits the attempts of unsuccessful authentication attempts to prevent password guessing. *FMT_SMF.1* enables the management of the authentication function and *FMT_MOF.1* restricts it to authorized administrators.

O.ACC_ADM requires that administrators must be able to specify which objects may be accessed by which administrators or users (i.e. to manage according access control policy rules). This is implemented by requiring appropriate access control policy rules in *FDP_ACF.1 (2)* and *FDP_ACC.2(2)*, which in turn allow administrators to access information – including access control policy rules – that is maintained by the TOE. *FMT_SMF.1* introduces a security function to manage the access control policy rules, which is restricted to be accessible only by administrators by *FMT_MOF.1*. *FMT_MSA.1 (1), FMT_MSA.1 (2)* and *FMT_MSA.3* refine the management of those rules. Note: while the selection of functional requirements to achieve this objective may be (validly) considered as a sub-set of the management of access control policy rules already covered by the requirements implementing O.ACCESS_DECISION, O.ACC_ADM emphasizes on the explicit possibility to delegate the management of parts of the access control policy (i.e. only rules related to a dedicated object space) to certain administrators and, therefore, has been included as a separate aspect in this ST.

O.ENFORCEMENT requires a design and implementation of the TSF that ensures TSP enforcement, preventing the bypassing of the TSF. This is achieved by the requirement *FPT_RVM.1* requiring non-bypassability of the TSP. Since it is assumed by A.SINGLE_APP that no other application than the TOE is running on the machines used, there is no need for the TOE to enforce its

own separation policy. This is left to the underlying operating system.

O.AUTHENT_USER requires the ability of the TOE to authenticate users trying to access protected objects. This is implemented by *FIA_UID.1* requiring identification. *FIA_SOS.1* defines the minimum requirements for the strength of password based authentication while *FIA_UAU.1* defines what unauthenticated users can do. *FIA_UAU.5* defines the two different authentication mechanism users may use to authenticate themselves. *FIA_UAU.6* defines the situations where a re-authentication of users is required.

O.SEC_COM requires the protection of communication links between the TOE and external entities as well as between separated parts of the TOE. *FTP_ITC.1* addresses this by demanding such a protected link between the TOE and external entities while *FPT_ITT.1* requires protected communication between different parts of the TOE when transferring TSF data. In both cases this requires the use of cryptographic functions, which is expressed by *FCS_CKM.1(1)* (addressing the generation of session keys), *FCS_CKM.1(2)* (addressing the generation of RSA key pairs), *FCS_CKM.2(1)* for exchange of RSA public keys, *FCS_CKM.2(2)* for the exchange of session keys, *FCS_COP.1(1)* for the digital signature generation and verification and *FCS_COP.1(2)* for the encryption of the data using a symmetric algorithm. In addition *FMT_MSA.2* ensures the selection of secure values for keys.

The following section provides a mapping of security objectives for the TOE environment to security functional requirements for the IT environment.

OE.OS_TIME requires the provision of a reliable time source by the operating system in the IT environment. This is reflected in *FPT_STM.1* for the operating system requiring the operating system in the IT environment to provide such a time source.

OE.OS_CONFFILE_PROT requires the protection of configuration files within the operating system. This is reflected in *FMT_MTD.1* for the operating system requiring the operating system to restrict the ability to modify one of those files (the configuration of audit events) to be accessible to authorized administrators only. Protection also requires the definition of roles in the operating system to define those allowed to access and modify configuration files, which is expressed in *FMT_SMR.1* (the assignment of roles requires user identification, which is expressed in FIA_UID.1). In addition *FMT_SMF.1* is included to address the requirement for user and access management. *FPT_SEP.1* for the operating system supports the access protection.

OE.DS_ACCESS_CNTRL requires an access control mechanism on the directory server allowing to prohibit unauthorized access to directory entries. This is reflected in the security functional requirements *FDP_ACC.1* and *FDP_ACF.1* for the directory server requiring an access control system as well as the SFRs *FDP_ETC.1* and *FDP_ITC.1* addressing the import and export of user data (i. e. directory information). Management of users and the access control function is expressed using the SFRs *FMT_MSA.1, FMT_MSA.3* and *FMT_SMF.1*. To define a useful access control framework the directory server should have a concept to define user roles, which is expressed with *FMT_SMR.1*.

OE.DS_AUTHENT requires identification and authentication on the directory server as a basis to implement an access control mechanism. This is reflected in the security functional requirements *FIA_UAU.2* and *FIA_UID.2* for the directory server requiring identification and authentication of users.

8.2.3 Security Requirements Dependencies

The following table, Table 6, shows the fulfillment of dependencies imposed on security functional

requirements by Part 2 of the Common Criteria (the left column identifies the CC Part 2 component, the middle column identifies the dependencies on that component drawn from CC Part 2, and the right column illustrates how the dependency is fulfilled). Deviations between requirements and fulfillment are explained subsequent to those tables. No additional dependencies exist for the security functional requirements.

Dependencies within the EAL3 "package" selected for the security assurance requirements have been considered by the authors of CC Part 3 and are not analyzed again in this Security Target. The included component on flaw remediation, ALC_FLR.1, has no dependencies on other requirements.

The security functional requirements in this Security Target do not introduce dependencies on any security assurance requirement; neither do the security assurance requirements in this Security Target introduce dependencies on any security functional requirement.

SFR	**Dependencies**	**Fulfillment of dependencies**
FAU_GEN.1	FPT_STM.1	FPT_STM.1 (environment)
FAU_GEN.2	FAU_GEN.1 FIA_UID.1	FAU_GEN.1 FIA_UID.1
FAU_SAR.1	FAU_GEN.1	FAU_GEN.1
FAU_SEL.1	FAU_GEN.1 FMT_MTD.1	FAU_GEN.1 FMT_MTD.1 (environment)
FAU_STG.1	FAU_GEN.1	FAU_GEN.1
FAU_STG.4	FAU_STG.1	FAU_STG.1
FCS_CKM.1(1)	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	FCS_CKM.2 FMT_MSA.2 (see note 1 below)
FCS_CKM.1(2)	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	FCS_CKM.2 FMT_MSA.2 (see note 1 below)
FCS_CKM.2(1)	[FDP_ITC.1 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	FDP_ITC.1 (environment) FMT_MSA.2 (see note 1 below)
FCS_CKM.2(2)	[FDP_ITC.1 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	FCS_CKM.1 FMT_MSA.2 (see note 1 below)
FCS_COP.1(1)	[FDP_ITC.1 or FCS_CKM.1] FCS_CKM.4	FDP_ITC.1 FMT_MSA.2 (see note 1 below)

SFR	Dependencies	Fulfillment of dependencies
	FMT_MSA.2	
FCS_COP.1(2)	[FDP_ITC.1 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	FCS_CKM.1 FMT_MSA.2 (see note 1 below)
FDP_ACC.2(1)	FDP_ACF.1	FDP_ACF.1(1)
FDP_ACC.2 (2)	FDP_ACF.1	FDP_ACF.1 (2)
FDP_ACF.1 (1)	FDP_ACC.1 FMT_MSA.3	FDP_ACC.2 (1) FMT_MSA.3
FDP_ACF.1 (2)	FDP_ACC.1 FMT_MSA.3	FDP_ACC.2 (2) FMT_MSA.3
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_ATD.1	no dependency	no dependency
FIA_SOS.1	no dependency	no dependency
FIA_UAU.1	FIA_UID.1	FIA_UID.1
FIA_UAU.2	FIA_UID.1	FIA_UID.1
FIA_UAU.5	no dependency	no dependency
FIA_UAU.6	no dependecy	no dependency
FIA_UID.1	no dependency	no dependency
FIA_USB.1	FIA_ATD.1	FIA_ATD.1
FMT_MOF.1	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.1
FMT_MSA.1(1)	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACC.2 (2) FMT_SMF.1 FMT_SMR.1
FMT_MSA.1(2)	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACC.2 (2) FMT_SMF.1 FMT_SMR.1
FMT_MSA.2	ADV_SPM.1 [FDP_ACC.1 or FDP_IFC.1] FMT_MSA.1 FMT_SMR.1	FDP_ACC.1 FMT_MSA.1 FMT_SMR.1 (see note 2 below)
FMT_MSA.3	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1 FMT_SMR.1
FMT_MTD.1	FMT_SMF.1	FMT_SMF.1

SFR	Dependencies	Fulfillment of dependencies
	FMT_SMR.1	FMT_SMR.1
FMT_SMF.1	no dependency	no dependency
FMT_SMR.1	FIA_UID.1	FIA_UID.1
FPT_ITT.1	no dependency	no dependency
FPT_RVM.1	no dependency	no dependency
FPT_TRC.1	FPT_ITT.1	FPT_ITT.1
FTP_ITC.1	no dependency	no dependency

Table 6: Dependency Analysis for TOE SFRs

Note 1: The dependency FCS_CKM.4 on the secure destruction of cryptographic keys is applicable for the session keys used as well as the private RSA keys. Since those keys are within the TOE and with the physical security of the TOE and the requirement not to have any other application running on a machine of the TOE unless full separation between this application and the TOE can be provided, it does not seem to be suitable to require a secure destruction of session keys.

Note 2: FMT_MSA.2 has been included to address the dependencies from FCS_CKM.1, FCS_CKM.2, FCS_COP.1 on this requirement. This addresses the use of “secure values” for keys used for cryptographic operations. The symmetric keys used for secure communication between the TOE and external entities as well as for communication between different parts of the TOE are generated automatically as defined by the SSL resp. TLS standard. The requirement for an informal security policy model to satisfy the dependency is not resolved, since the generation and distribution of the symmetric keys would anyhow not been addressed by such a model. The other dependencies are formally resolved, but since the management of those session keys does not involve any user or administrator, none of the security functional requirements listed actually addresses the key generation process. It is therefore argued that the dependencies listed in FMT_MSA.2 are not applicable in this TOE.

SFR	Dependencies	Fulfillment of dependencies
Directory Server		
FDP_ACC.1	FDP_ACF.1	satisfied
FDP_ACF.1	FDP_ACC.1 FMT_MSA.3	satisfied
FDP_ETC.1	[FDP_ACC.1 or FDP_IFC.1]	satisfied
FDP_ITC.1	[FDP_ACC.1 or FDP_IFC.1] FMT_MSA.3	satisfied
FIA_UAU.2	FIA_UID.1	Satisfied by FIA_UID2
FIA_UID.2	No dependencies	Satisfied
FMT_MSA.1	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	Satisfied
FMT_MSA.3	FMT_MSA.1 FMT_SMR.1	Satisfied
FMT_SMF.1	No dependency	
FMT_SMR.1	FIA_UID.1	Satisfied by FIA_UID.2
Underlying Operating System		
FIA_UID.1	No dependencies	Satisfied
FPT_STM.1	No dependencies	Satisfied
FMT_MTD.1	FMT_SMR.1 FMT_SMF.1	Satisfied
FMT_SMF.1	No dependency	
FMT_SMR.1	FIA_UID.1	Satisfied
FPT_SEP.1	no dependency	no dependency

This table shows that all dependencies on security functional requirements are satisfied for all the identified components in the IT environment.

8.2.4 Internal Consistency and Mutual Support

Chapter 8.2.2 has already shown how the security functional requirements work together to implement the single objectives for the TOE and the IT environment. This chapter will elaborate on the internal consistency and mutual support of the security functional requirements. Further information can as well be found in the application notes to the security functional requirements in chapter 5.

Internal Consistency and Mutual Support of security requirements for the TOE

The main goal of the TOE is to perform access control decisions for resources stored in the TOE environment and allows a sound management of the information that is necessary to do so. The most vital information needed for access control decisions are the access control policy rules, containing basically information on which target is allowed to be accessed by whom. An appropriate access control security function policy, the **Web-space access control policy** is implementedd by *FDP_ACF.1 (1)*. The enforcement of the access control decisions made by the TOE is expressed by *FDP_ACC.2 (1)*. Another input to this access control decision is the identity of the access request initiators. The TOE performs this identification and authentication of users (*FIA_UID.1* and *FIA_UAU.1* together with *FIA_UAU.5* for the support of multiple authentication mechanisms and *FIA_UAU.6* for the definition of cases where re-authentication is required), there is the need to match the authenticated initiators to the descriptors employed by the access control policy rules within the TOE to identify the subjects of access control decisions. This is accomplished by maintaining a data base of initiator security attributes, which is reflected by *FIA_ATD.1*, and requiring the appropriate user-subject binding in *FIA_USB.1*. Two different authentication mechanism are supported for the authentication of users: password based authentication and client certificate based authentication. This is expressed in *FIA_UAU.5*. For password based authentication a minimum strength of the passwords is required and defines by *FIA_SOS.1*. In addition the number of consecutive unsuccessful authentication attempts is restricted as expressed by *FIA_AFL.1*. There are also situations were a re-authentication of a user is required as expressed by *FIA_UAU.6*. Since the policy database may be replicated, *FPT_TRC.1* ensures the consistency between the master policy database and the replica.

To allow for management of the access control policy rules used by the TOE for its access control decisions, administrators of the TOE need to be able to execute appropriate management functionality. To clearly separate those duties (and the originators of appropriate actions), a separation between user and administrator roles is introduced by *FMT_SMR.1*. To protect the integrity of the TSF, including the TSF data, access control is required for the administrative access to the TOE. The same mechanisms are employed to implement this access control as they are used to fulfill the access control decisions for the access requests: access control policy rules define which administrator is allowed to access which administrative functionality of the TOE. To prevent confusion, this is expressed in a separate access control policy, the **management access control policy**, defined by *FDP_ACF.1 (2)*. Since it is not the intention of the TOE to leave the enforcement of access control decisions with respect to the management of the TSF up to its environment, the authentication of the administrators (*FIA_UAU.2* and *FIA_UID.2*) as well as the enforcement of the access control decisions (*FDP_ACC.2 (2)*) must be implemented by the TOE. For this reason, the initiator security attribute data base contains as well the authentication secrets for administrative users of the TOE (again, *FIA_ATD.1*). With respect to the authentication of administrators, authentication failure handling (*FIA_AFL.1*) and a definition of minimum constraints to the choice of authentication secrets (*FIA_SOS.1*) are also required.

To detect possible attacks and to allow accounting for administrative actions, requirements for the **generation of audit data** are included. Those comprise the definition of auditable events and the outline of audit records (*FAU_GEN.1*) as well as the association of auditable events with the identity of their originator, if the originator was required to authenticate himself (*FAU_GEN.2*). In order to comprehend the audited events, administrators must be able to inspect the audit trails (*FAU_SAR.1*) and the audit information has to be protected against modifications (*FAU_STG.*1) and loss (*FAU_STG.4*). To include the correct time of an event in the audit records, appropriate information has to be delivered by the IT environment (*FPT_STM.1* for the OS). The requirements *FAU_SEL.1* and the requirement *FMT_MTD.1* for the underlying OS allow an administrator to specify which events are to be audited by the TOE.

Management functions are established by *FMT_SMF.1* to allow the management of authentication, audit and access control functionality. Such management is restricted by *FMT_MOF.1* to authorized

administrators, while *FMT_MSA.1* and *FMT_MSA.3* impose the **management access control policy** on the management actions.

Secure communication between different distributed parts of the TOE is required by *FPT_ITT.1* and also between the TOE and other trusted IT products (LDAP server, client systems, Web-Servers) as required by *FTP_ITC.1*. In both cases this requires the use of cryptographic functions as defined by the SSL resp. TLS protocol. While the generation and import of the RSA public key pair is addressed in FCS_CKM.1(2), the generation of the symmetric session keys as expressed by *FCS_CKM.1(1)*, the distribution of RSA public key certificates as expressed by *FCS_CKM.2(2)*, the distribution of the symmetric session keys as expressed by *FCS_CKM.2(1)* and the associated cryptographic operations for signing / signature verification (*FCS_COP.1(1)*) and symmetric encryption (*FCS_COP.1(2)*) are all addressed as requirements for the TOE as well. The requirement for secure session keys is expressed with *FMT_MSA.2*.

The bypassing of any of the IT security requirements for the TOE is prevented by introducing a requirement on TSP enforcement in *FPT_RVM.1*, and tampering with the functionality introduced by any IT security requirement for the TOE is not possible since according to the assumption A.SINGLE_APP no other application is allowed to run on the machines comprising the TOE.

Internal Consistency and Mutual Support of security requirements for the IT environment

The TOE makes use of the underlying **operating system** to provide a reliable time stamp (as required by *FPT_STM.1*) and the ability to manage the audit events of the TOE by editing the configuration file (as required by *FMT_MTD.1* and called out by *FMT_SMF.1*). Authorized roles need to be defined that are allowed to access the audit records as required by *FMT_SMR.1* which depend on the correct identification of users as required by *FIA_UID.1*. This is supported by *FPT_SEP.1* requiring that the security functions of the underlying operating system maintain a domain for their own execution protected from inference or tampering by untrusted subjects.

The TOE uses a **Directory Server** to store and maintain the user registry. To rely on the security of this Directory Server it has to be ensured that this server provides functions for user authentication (as required by *FIA_UAU.2* and *FIA_UID.2* for the Directory Server) and access control (as required by *FDP_ACC.1* and *FDP_ACF.1*). In addition export and import of user data has to be performed in accordance with the access control policy (as required by *FDP_ETC.1* and *FDP_ITC.1*). Management aspects are addressed by the security functional requirements *FMT_MSA.1*, *FMT_MSA.3*, *FMT_SMF.1* and *FMT_SMR.1*

8.2.5 Evaluation Assurance Level and Strength of Function

The evaluation assurance level (EAL) 3 was chosen as a medium level of assurance reflecting the expected assurance requirements of commercial customers using the target of evaluation (TOE) for the protection of data with a low or medium level of sensitivity. The TOE is intended to provide a reasonable level of protection for this data comparable to the protection provided by most commercial-off-the-shelf operating system products. This is reflected as well in the definition of the TOE environment in chpt. 2 and the security objectives for the TOE in chpt. 4 of this Security Target.

The assurance level EAL3 was augmented with ALC_FLR.1 to address the flaw remediation process used within Tivoli. Since the evaluation methodology for ALC_FLR has been harmonized and is also covered by the Mutual Recognition Arrangement, this was considered to be a useful augmentation for the assurance level chosen.

In line with this medium level of assurance the functions provided by the TOE that are subject to probabilistic or permutational analysis (except for cryptographic algorithms and algorithms related to the cryptographic functions) are claimed to have at least a medium strength (SOF-medium). The functions that are subject to a strength of function analysis are F.Authentication, which uses

passwords. The certificate based authentication function and the cryptographic algorithms used within F.Communication as well as the related key generation process are not subject to a strength of function analysis in accordance with the CEM, remarks on ASE_REQ.1.15, para 424.

8.3. TOE Summary Specification Rationale

8.3.1 Security Functions Justification

The following table shows that the IT security functions as specified in the TOE summary specification, meet all the security functional requirements for the TOE and work together to satisfy the TOE security functional requirements.

SFR	**Security Functions from the TOE Summary Specification**
FAU_GEN.1	This requirement is addressed by the security function **F.Audit**, which defines the events that the different parts of the TOE are able to generate and what is audited.
FAU_GEN.2	This requirement is addressed by the functions **F.Authentication**, which requires users and administrators to be authenticated and **F.Audit** which audits the identity of the user that caused the event together with other relevant information.
FAU_SAR.1	This requirement is addressed by the function **F.Audit** which restricts the access to the audit trail by setting appropriate rights in the underlying operating system.
FAU_SEL.1	This requirement is addressed by **F.Audit** which allows to define the events to be audit in a configuration file. Note that in theory the access control function of the TOE could be used to control access to the audit configuration file, but in most cases this would be left to the access control functions of the underlying operating system.
FAU_STG.1	This requirement is addressed by **F.Authorization** which allows only administrators to access the audit trail as part of the TOE protected management objects.
FAU_STG.4	This requirement is addressed by **F.Audit** which states that an audit trail are rolled over to a new file when the audit trail has reached a defined size.
FCS_CKM.1(1)	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between different parts of the TOE as well as between the TOE and external entities. The cryptographic operations themselves are described in the SSL resp. TLS standard.
FCS_CKM.1(2)	This requirement is addressed by **F.Communication** which defines that the administrator can generate RSA key pairs for the use with the SSL v3 resp. TLS v1 protocol. The TOE will only generate key pairs for its own internal use. External entities authenticating to the TOE using a digital certificate are assumed to have this authentication credential generated and protected

SFR	Security Functions from the TOE Summary Specification
	securely.
FCS_CKM.2(1)	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between different parts of the TOE as well as between the TOE and external entities. The cryptographic operations themselves are described in the SSL resp. TLS standard.
FCS_CKM.2(2)	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between different parts of the TOE as well as between the TOE and external entities. The cryptographic operations themselves are described in the SSL resp. TLS standard.
FCS_COP.1(1)	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between different parts of the TOE as well as between the TOE and external entities. The cryptographic operations themselves are described in the SSL resp. TLS standard.
FCS_COP.1(2)	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between different parts of the TOE as well as between the TOE and external entities. The cryptographic operations themselves are described in the SSL resp. TLS standard.
FDP_ACC.2(1)	This requirement is addressed by **F.Authorization** where the ACL and POP policies for web objects are described.
FDP_ACC.2(2)	This requirement is addressed by **F.Authorization** where the access control policy for management objects is described.
FDP_ACF.1 (1)	This requirement is addressed by **F.Authorization** where the ACL and POP policies for web objects are described.
FDP_AFC.1 (2)	This requirement is addressed by **F.Authorization** where the access control policy for management objects is described.
FIA_AFL.1	This requirement is described in **F.Authentication** which describes the limits on the number of successive authentication failures allowed.
FIA_ATD.1	This requirement is described in **F.Authentication** where the different user attributes are defined.
FIA_SOS.1	This requirement is described in **F.Authentication** where the different possibilities for the password policy are defined.
FIA_UAU.1	This requirement is described in **F.Authentication** where the different authentication mechanism to authenticate users are described.

SFR	Security Functions from the TOE Summary Specification
FIA_UAU.2	This requirement is described in **F.Authentication** where the authentication process for administrators is described.
FIA_UAU.5	This requirement is described in **F.Authentication** where the different authentication mechanism are described.
FIA_UAU.6	This requirement is described in **F.Authentication** where the different conditions for re-authentication are described.
FIA_UID.1	This requirement is described in **F.Authentication** where the identification of users is described.
FIA_UID.2	This requirement is described in **F.Authentication** where the identification of administrators is described.
FIA_USB.1	This requirement is described in **F.Authentication** where the binding of users / administrators to subjects is described.
FMT_MOF.1	This requirement is described in **F.Authorization,** where the different management objects and their management functions are described as well as in **F.Audit** with respect to the management of the audit function, **F.Authentication** with respect to the management of users, user attributes and password policies, and **F.Management** where the management ACLs and POPs are described.
FMT_MSA.1(1)	This requirement is described in **F.Authorization** where the different management objects and their management functions are described and in **F.Management** where ACL and POP management is described.
FMT_MSA.1(2)	This requirement is described in **F.Authorization** where the different management objects and their management functions are described and in **F.Management** where ACL and POP management is described.
FMT_MSA.2	This requirement is described in **F.Authentication** where the security attributes of users are described.
FMT_MSA.3	This requirement is described in **F.Authorization** and **F.Management** where the inheritance policy for ACLs is described.
FMT_MTD.1	This requirement is described in **F.Authorization** where the different management objects and their management functions are described, and in **F.Management** where the management functions of administrators are described.
FMT_SMF.1	This requirement is described in **F.Authorization** where the different management objects and their management functions are described as well as in **F.Audit** with respect to the management of audit configuration and **F.Management** with respect to user and group management as well as ACL and POP management.

SFR	Security Functions from the TOE Summary Specification
FMT_SMR.1	This requirement is described in **F.Authorization** where the different roles are described.
FPT_ITT.1	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between different parts of the TOE. The cryptographic operations themselves are described in the SSL resp. TLS standard.
FPT_RVM.1	This requirement is addressed by the overall architecture of the TOE. Every request from a client system is passed through the authorization function **F.Authorization**.
FPT_TRC.1	This requirement is described in **F.Authorization** where the replication mechanism for the policy database is described
FTP_ITC.1	This requirement is addressed by **F.Communication** which defines that the SSL resp. TLS protocol is used to secure the communication between the TOE and external entities. The cryptographic operations themselves are described in the SSL resp. TLS standard.

Table 7: Mapping Security Functional Requirements to Security Functions

8.3.2 Justification that the Security Functions are Mutually Supportive

The main objective of the TOE is to provide access control for external web objects and act as a system that authenticates users that want to access those protected resources, allows to define access rules between those users and objects and enforces those access rules.

As a result the TOE requires an authentication mechanism for users and administrators of the TOE, which is defined in **F.Authentication**. Access control is divided into access control for user objects and access control for management objects as defined in **F.Authorization**. The rules for administrators how to define and manage access control lists and protected object policies – the two mechanisms the TOE uses for access control – as well as the rules for user and group management are defined in **F.Management**.

To allow for individual accountability the TOE also includes an audit function as described by **F.Audit**. This function can be used to trace the activities of users and administrators and make them accountable for the actions they have performed.

To be able to enforce this policy the TOE needs to establish secure communication links between itself and all external entities as well as between distributed parts of the TOE itself. This is accomplished by **F.Communication** that defines that the SSL resp. TLS protocol is used for all those communication links. This protects the TSF data (including the protected objects) from unauthorized disclosure and modification when transmitted over communication links. It also ensures that the different parts of the TOE itself authenticate themselves to each other when establishing a communication link. Management of the certificates used for protected communication is defined in **F.Management**.

As a result one can state:

- Protected resources can only be accessed by users authorized for this access

- Management of the TOE functions is restricted to defined administrators
- Bypassing the TOE security functions or attacking the communication between the TOE and external entities or between distributed parts of the TOE is actively prohibited by the TOE
- Accountability is enforced by an audit trail capable to audit all administrator actions as well as all access of users to protected resources

This shows that the TOE security functions are mutually supportive.

8.3.3 Rationale for Strength of Function Claim

The claim for the strength of function is SOF-medium. This claim is consistent with the evaluation assurance level chosen as well as with the characterization of the attack potential.

The functions that are subject to a strength of function analysis are the authentication mechanism using passwords.

The cryptographic algorithms (including the cryptographic hash functions) as well as the key generation process for the keys used within those algorithms are not subject to the strength of function analysis in compliance with Common Criteria (part 1, page 2) and the CEM (part 2, para 424) and the German CC evaluation scheme.

8.4. PP Claims Rationale

No compliance with any existing Protection Profile is claimed.

9. Abbreviations

Abbreviations are jointly derived from [1] and [10181-3].

ACI	Access Control Information
ACL	Access Control List
ADF	Access Control Decision Function
ADI	Access Control Decision Information
AEF	Access Control Enforcement Function
API	Application Programming Interface
APP	Authorization Protection Profile
CC	Common Criteria, the name used historically for this multipart standard ISO/IEC 15408 in lieu of its official ISO name of “Evaluation criteria for information technology security”
EAL	Evaluation Assurance Level
PP	Protection Profile
SF	Security Function
SFP	Security Function Policy
SOF	Strength of Function
ST	Security Target
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSFI	TSF Interface
TSP	TOE Security Policy

10. Glossary

The glossary is jointly derived from [1] and [10181-3].

Access Control Decision Function (ADF)	A specialized function that makes access control decisions by applying access control policy rules to an access request, ADI (of initiators, targets, access requests, or that retained from prior decision), and the context in which the access request is made.
Access Control Decision Information (ADI)	The portion (possibly all) of the ACI made available to the ADF in making a particular access control decision.
Access Control Enforcement Function (AEF)	A specialized function that is part of the access path between an initiator and a target on each access request and enforces the decision made by the ADF.
Access Control Information (ACI)	Any information used for access control purposes, including contextual information.
Access Control Policy	The set of rules that define the conditions under which an access may take place.
Access Control Policy Rules	Security policy rules concerning the provision of the access control service.
Access Request	The operations and operands that form part of an attempted access.
Assets	Information or resources to be protected by the countermeasures of a TOE.
Assignment	The specification of an identified parameter in a component.
Assurance	Grounds for confidence that an entity meets its security objectives.
Attack potential	The perceived potential for success of an attack, should an attack be launched, expressed in terms of an attacker's expertise, resources and motivation.
Augmentation	The addition of one or more assurance component(s) from Part 3 to an EAL or assurance package.
Authentication data	Information used to verify the claimed identity of a user.
Authorised user	A user who may, in accordance with the TSP, perform an operation.
Class	A grouping of families that share a common focus.
Component	The smallest selectable set of elements that may be included in a PP, an ST, or a package.
Connectivity	The property of the TOE which allows interaction with IT entities external to the TOE. This includes exchange of data by wire or by wireless means, over any distance in any environment or configuration.
Contextual Information	Information about or derived from the context in which an access request is made (e.g. time of day).
Dependency	A relationship between requirements such that the

	requirement that is depended upon must normally be satisfied for the other requirements to be able to meet their objectives.
Element	An indivisible security requirement.
Evaluation	Assessment of a PP, an ST or a TOE, against defined criteria.
Evaluation Assurance Level (EAL)	A package consisting of assurance components from Part 3 that represents a point on the CC predefined assurance scale.
Evaluation authority	A body that implements the CC for a specific community by means of an evaluation scheme and thereby sets the standards and monitors the quality of evaluations conducted by bodies within that community.
Evaluation scheme	The administrative and regulatory framework under which the CC is applied by an evaluation authority within a specific community.
Extension	The addition to an ST or PP of functional requirements not contained in Part 2 and/ or assurance requirements not contained in Part 3 of the CC.
External IT entity	Any IT product or system, untrusted or trusted, outside of the TOE that interacts with the TOE.
Family	A grouping of components that share security objectives but may differ in emphasis or rigour.
Formal	Expressed in a restricted syntax language with defined semantics based on well established mathematical concepts.
Human user	Any person who interacts with the TOE.
Identity	A representation (e.g. a string) uniquely identifying an authorised user, which can either be the full or abbreviated name of that user or a pseudonym.
Informal	Expressed in natural language.
Initiator	An entity (e.g. human user or computer-based entity) that attempts to access other entities.
Internal communication channel	A communication channel between separated parts of TOE.
Internal TOE transfer	Communicating data between separated parts of the TOE.
Inter-TSF transfers	Communicating data between the TOE and the security functions of other trusted IT products.
Iteration	The use of a component more than once with varying operations.
Object	An entity within the TSC that contains or receives information and upon which subjects perform operations.
Organisational security policies	One or more security rules, procedures, ractices, or guidelines imposed by an organisation upon its operations.
Package	A reusable set of either functional or assurance

	components (e.g. an EAL), combined together to satisfy a set of identified security objectives.
Product	A package of IT software, firmware and/or hardware, providing functionality designed for use or incorporation within a multiplicity of systems.
Protection Profile (PP)	An implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs.
Reference monitor	The concept of an abstract machine that enforces TOE access control policies.
Reference validation mechanism	An implementation of the reference monitor concept that possesses the following properties: it is tamperproof, always invoked, and simple enough to be subjected to thorough analysis and testing.
Refinement	The addition of details to a component.
Role	A predefined set of rules establishing the allowed interactions between a user and the TOE.
Secret	Information that must be known only to authorised users and/or the TSF in order to enforce a specific SFP.
Security attribute	Information associated with subjects, users and/or objects that is used for the enforcement of the TSP.
Security Function (SF)	A part or parts of the TOE that have to be relied upon for enforcing a closely related subset of the rules from the TSP.
Security Function Policy (SFP)	The security policy enforced by an SF.
Security objective	A statement of intent to counter identified threats and/or satisfy identified organisation security policies and assumptions.
Security Target (ST)	A set of security requirements and specifications to be used as the basis for evaluation of an identified TOE.
Selection	The specification of one or more items from a list in a component.
Semiformal	Expressed in a restricted syntax language with defined semantics.
SOF-basic	A level of the TOE strength of function where analysis shows that the function provides adequate protection against casual breach of TOE security by attackers possessing a low attack potential.
SOF-high	A level of the TOE strength of function where analysis shows that the function provides adequate protection against deliberately planned or organised breach of TOE security by attackers possessing a high attack potential.
SOF-medium	A level of the TOE strength of function where analysis shows that the function provides adequate protection against straightforward or intentional breach of TOE security by attackers possessing a moderate attack potential.
Strength of Function (SOF)	A qualification of a TOE security function expressing the minimum efforts assumed necessary to defeat its

	expected security behaviour by directly attacking its underlying security mechanisms.
Subject	An entity within the TSC that causes operations to be performed.
System	A specific IT installation, with a particular purpose and operational environment.
Target	An entity to which access may be attempted.
Target of Evaluation (TOE)	An IT product or system and its associated administrator and user guidance documentation that is the subject of an evaluation.
TOE resource	Anything useable or consumable in the TOE.
TOE Security Functions (TSF)	A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the TSP.
TOE Security Functions Interface (TSFI)	A set of interfaces, whether interactive (man-machine interface) or programmatic (application programming interface), through which TOE resources are accessed, mediated by the TSF, or information is obtained from the TSF.
TOE Security Policy (TSP)	A set of rules that regulate how assets are managed, protected and distributed within a TOE.
TOE security policy model	A structured representation of the security policy to be enforced by the TOE.
Transfers outside TSF control	Communicating data to entities not under control of the TSF.
Trusted channel	A means by which a TSF and a remote trusted IT product can communicate with necessary confidence to support the TSP.
Trusted path	A means by which a user and a TSF can communicate with necessary confidence to support the TSP.
TSF data	Data created by and for the TOE, that might affect the operation of the TOE.
TSF Scope of Control (TSC)	The set of interactions that can occur with or within a TOE and are subject to the rules of the TSP.
User	Any entity (human user or external IT entity) outside the TOE that interacts with the TOE.
User data	Data created by and for the user, that does not affect the operation of the TSF.

11. References

[CAPP]	Controlled Access Protection Profile, Version 1.d, National Security Agency
[CC]	Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; Version 2.1, August 1999
	Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Requirements; Version 2.1, August 1999
	Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; Version 2.1, August 1999
[CEM]	Common Methodology for Information Technology Security Evaluation CEM-99/045 Part 2: Evaluation Methodology, Version 1.0, August 1999
[SSLv3]	Alain O. Freier, Philip Karlton, Paul C. Kocher: The SSL Protocol, Version 3; IETF Memo, Internet Draft, November 1996
[RFC2246]	T.Dierks, C. Allen: The TLS Protocol, Version 1.0; IETF, RFC 2246
[AZNAPI]	Open Group Technical Standard: Authorization (AZN) API, The Open Group, January 2000
[ISO10181-3]	ISO/IEC 10181-3: Information Technology – Open Systems Interconnection – Security frameworks for open systems: Access control framework, 1996
[AMBADM]	Tivoli Access Manager for eBusiness Base Installation Guide, Version 4.1
[X.509]	ITU-T RECOMMENDATION X.509 | ISO/IEC 9594-8: INFORMATION TECHNOLOGY - OPEN SYSTEMS INTERCONNECTION - THE DIRECTORY: PUBLIC-KEY AND ATTRIBUTE CERTIFICATE FRAMEWORKS