DriveLock Agent 2019.2 (Device and Application Control)

Document type:	Security Target
Version:	1.32
Last changed:	2021-03-17 15:36

Abstract: This document contains the Security Target for the CC EAL3+ evaluation of the DriveLock Agent 2019.2 (Device and Application Control) software.

DriveLock SE 2021

DON'T GAMBLE WITH YOUR DATA

Contents

1 Document Information

1.1 Ownership and Classification

This Security Target was written by

Joachim Schneider
LinkedIn: https://www.linkedin.com/in/joachim-schneider-a663756
Email: joachim.schneider[at]gmx.at

for DriveLock SE.

The document is public in PDF format and may be redistributed in unmodified form.

2 ST Introduction

2.1 ST Reference

ST Title	DriveLock Agent 2019.2 (Device and Application Control) Security Target
ST Version	1.32
ST Author	Joachim Schneider
ST Date	2021-03-17

2.2 TOE Reference

TOE Identification	DriveLock Agent 2019.2 (Device and Application Control) SP 1
TOE Developer	DriveLock SE
TOE Type	Access Control Software

2.3 TOE Overview

2.3.1 Definitions

The following terms are used throughout the remainder of this document:

Operator	Organization or entity using the TOE
Administrator	privileged administrative user of the Operator's infrastructure, maintaining systems and implementing policies on behalf of the Operator
Workstation	a regular user's personal computer where the TOE is installed
Administrative Backend	a set of systems and applications that supply services for deployment and administration of the TOE instances

2.3.2 Usage and Major Security Features

DriveLock Agent 2019.2 (Device and Application Control) is an application and device control software-only TOE for use on workstation PCs running a Windows operating system as defined in section 2.3.3. Its main functions are:

- Blocking unwanted devices from use, therefore preventing unwanted data import or export and potential system compromise by malicious devices.
- Blocking unwanted applications from executing, preventing system degradation and other undesirable effects that could be caused by these applications.
- Auditing events that trigger the security functions mentioned above.

The DriveLock software is designed to efficiently enforce rules and policies required by an organization on a large number of workstations. A rich set of options for defining rules guarantees that virtually any company policy can be implemented as a DriveLock policy. Machine learning and inventory functions allow an administrator to rapidly set up policies without painstaking manual work. Using all the available tools, DriveLock can be up and protecting your systems in as little as two hours.

A typical DriveLock installation requires four components, as shown in Figure 1:

DriveLock Enterprise Service (DES)

The DriveLock Enterprise Service is the central component of an installation and will usually be installed on a server. It serves as the contact point for the workstation agents, delivering policy and software updates to them and accepting audit data for later analysis.

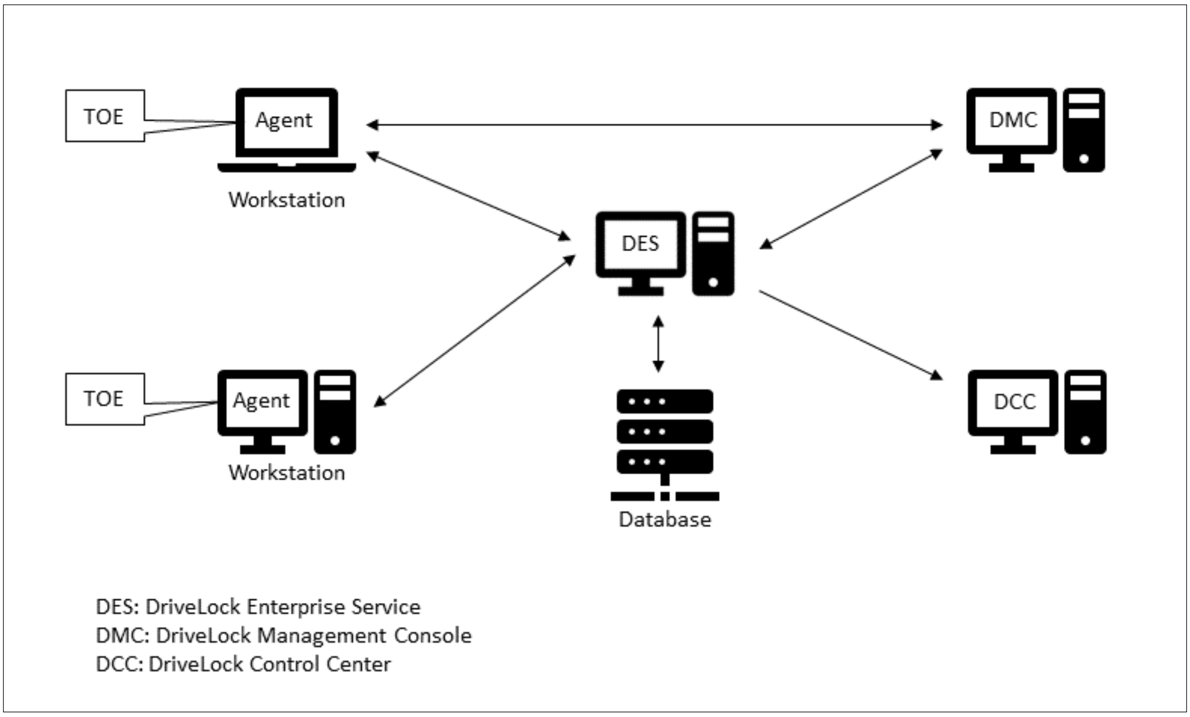
TOE
Agent
Workstation
DMC
DES
TOE
Agent
Workstation
DCC
Database
DES: DriveLock Enterprise Service
DMC: DriveLock Management Console
DCC: DriveLock Control Center

Figure 1 – Overview of TOE installation and environment

DriveLock Management Console (DMC)

The DriveLock Management Console is used to configure the DES and for policy management. Actual deployment of updates is handled by the Enterprise Service (DES).

DriveLock Control Center (DCC)

The DriveLock Control Center serves as the dashboard primarily for help desk personnel. The control center can observe client agents and interact with them, as well as analyze the audit data by generating reports.

DriveLock Agent

The DriveLock Agent comprising the TOE needs to be installed on every managed workstation. It consists of both user mode components and kernel mode drivers and enforces the policies defined in the Management Console on the workstation. The device and application control functionality of the

DriveLock Agent, together with its associated management and auditing support, constitutes the TOE for this evaluation.

Device Control

Device Control restricts the external devices that can be connected to a managed system. The set of acceptable devices is defined by rules in a policy. A flexible set of options allows control over who is allowed to use which device(s) at which time. An inventory function (not part of the TOE functionality) enables the administrator to effortlessly generate a policy that allows use of currently known devices. As a consequence, any newly detected devices can be blocked from use until they are cleared by an administrator.

Application Control

Application Control restricts the set of executable files that can be run on a system. Besides limiting what users can do on a workstation, Application Control will also block the execution of unknown software. This is a valuable second line of defense against malware or ransomware that the antivirus software missed. In addition, Application Control can also enforce application permissions, controlling the access of permitted processes to files and settings on the workstation.

Again, a flexible set of options allows detailed rules to control what is executed on the workstation by which user. Several criteria for executable files are available; however, the evaluated configuration requires the use of a whitelist of hash values calculated over the permissible files. A powerful machine learning feature (not part of the TOE functionality) allows creation and maintenance (e.g. required after a software update) of the hash database without much effort by administrators.

Audit

The TOE includes a configurable audit trail which can be centrally monitored. The DriveLock Agent captures events such as blocking or allowing an application or device, including machine and user information. An audit policy allows detailed control over

- which events should be audited
- where the events should be sent to (multiple destinations possible per event, e.g. Windows event log, DES, SMTP, …).

Additional functionality

The DriveLock Agent supports further functionality that is not part of the TOE, such as:

- Drive encryption
- File and folder encryption
- Automated encryption of data transferred to permitted removable drives
- Creation and management of encrypted container files
- Shadow copying of data transferred to removable devices
- Security awareness module to train users
- BitLocker management
- Temporary unlocking of e.g. device usage restrictions

2.3.3 Required Non-TOE Hardware, Software, and Firmware

The TOE requires the following for proper operation:

- An industry-standard PC running Microsoft Windows 10 (64-bit version) with all security updates installed.
- 1 GB of free disk space
- An operational TCP/IP network connection to the administrative backend
- The administrative backend itself:
  - DriveLock Enterprise Service (DES)
  - DriveLock Management Console (DMC)
  - DriveLock Control Center (DCC)

2.4 TOE Description

2.4.1 Physical Scope

The TOE consists of the following component inside the downloadable .ISO file:

- DriveLock Agent X64.msi (for 64-bit operating system)

The .ISO image contains further installation files, notably those for the server and administration components. These are not part of the TOE. The image can be downloaded at https://drivelock.support/hc.

The following guidance documentation is available on the DriveLock web site at https://drivelock.help/:

- Installation Guide 2019.2 (PDF)
- Admin Guide 2019.2 (PDF)
- Manual Supplement for Certification Compliant Operation 2019.2 (PDF, English only)
- Control Center User Guide 2019.2 (PDF)
- Events 2019.2 SP1 (PDF, English only)
- User Guide 2019.2 (PDF)
- Release Notes 2019.2 (PDF)

in both English and German languages.

Note: Only the English language manuals are part of this evaluation.

2.4.2 Logical Scope

The TOE security functions are contained in the DriveLock Agent installed on the workstation. The agent consists of both user-mode and kernel-mode components (drivers).

The drivers implement most of the TOE security function interaction with the operating system. They are implemented as filter drivers and inserted in various operating system driver stacks. From there, the TOE drivers can influence the outcome of various operating system calls or events, e.g. prevent activation of a device or prevent the creation of a process.

The user-mode components reside in a user-mode service. They implement the interaction with the user and the backend, such as the administration interface over which the TOE receives its policy and configuration data, or the mechanism forwarding the audit trail entries to the destination(s) configured for these entries. The user-mode service also passes policy and configuration data to the drivers. For specific functions the policy enforcement by the driver(s) is supported by a user mode component as well, e.g. where a policy decision would be very difficult to make in kernel-mode code.

Figure 2 shows an overview of the TOE and its environment on the workstation.

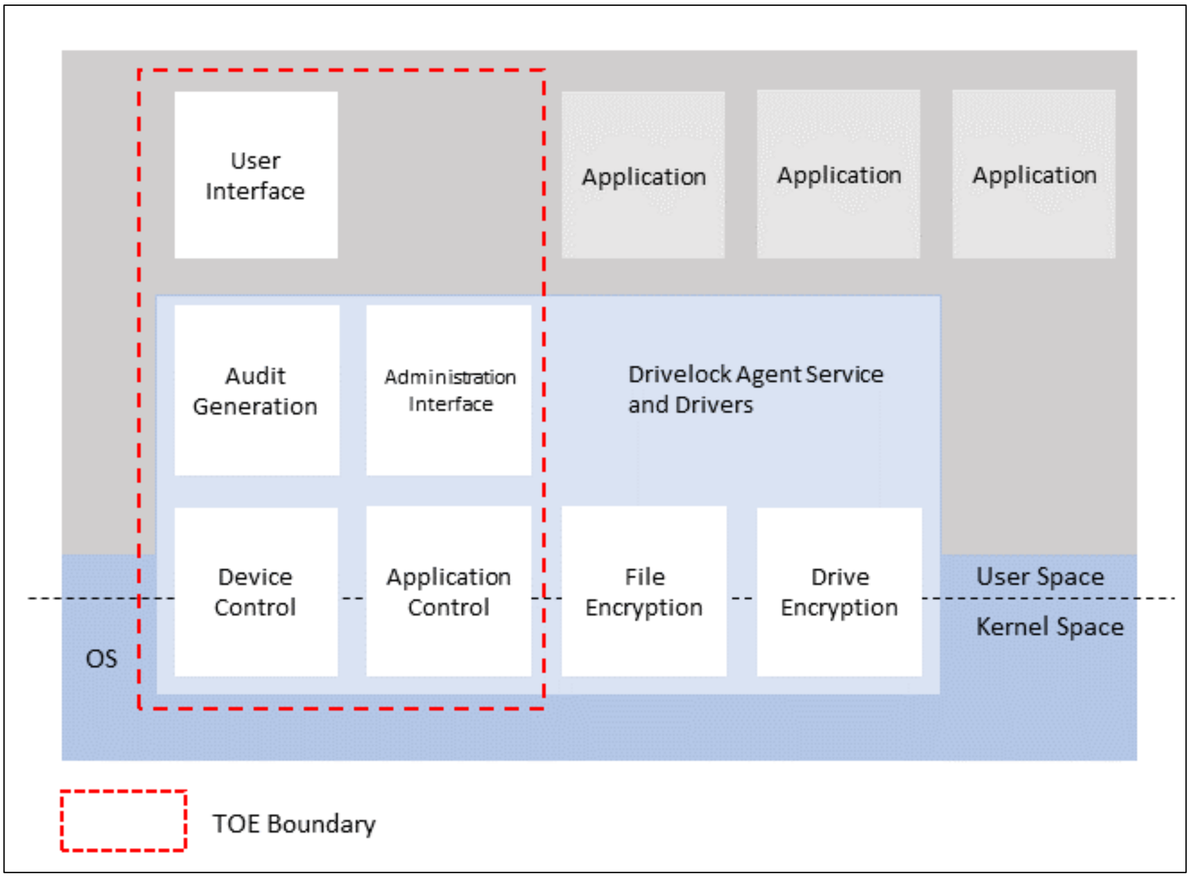

Figure 2 – Overview of the TOE and its environment

As shown above, the primary security functions of the DriveLock Client are implemented as combinations of a kernel mode driver and a matching user mode component:

- Device Control: This component enforces device filtering, i.e. control over access to devices and ports.
- Application Control: This component controls execution of executable files, and access to system resources by running processes.
- File Encryption: This component enforces transparent file and folder encryption with strong algorithms and keys. This component is not part of the evaluated TOE functionality.
- Drive Encryption: This component encrypts entire drives and provides encrypted container files. This component is not part of the evaluated TOE functionality.

The DriveLock Agent service also contains additional components that run in user mode only; those relevant for the TOE are:

- Audit Generation: This component collects and routes all audit events logged by the other components.
- Administration Interface: This component implements a bidirectional interface with a DriveLock Enterprise Service. It retrieves configuration updates from and sends audit data to the server.
- User Interface: This component runs as a separate application and mainly provides status information to the current user. Its other functionality, e.g. management of encrypted drives and containers, is not part of this evaluation.

Within this architecture, the TOE implements the following security functions:

Device Control

The TOE prevents unwanted devices from becoming accessible to the operating system and therefore to the user. This is achieved by inserting the Device Filter driver in a driver stack provided by the operating system. As a result, the device filter driver is notified whenever a device is connected to the workstation, as the I/O requests required to activate a device are passed through the driver stack. Whenever the Device Filter intercepts such a request it evaluates if the device permitted and continues processing the request accordingly. If a device is not permitted, the TOE forces certain I/O requests to fail, preventing device activation. A policy received from the administrative backend defines which devices are permissible.

Application Control

The TOE prevents unwanted applications from executing. As with Device Control, a filter component intercepts and monitors operating system functions that are required to complete successfully to execute an application. If an application is not permitted the driver forces the operating system function to fail, which in turn causes the application launch to fail. A policy received from the administrative backend defines which executable files are permissible. The policy is effectively a list of permitted executables and associated hash values. The TOE enforces this policy by calculating a hash value over the executable file about to be launched and comparing this value against the list of hash values for the permitted executables.

Cryptographic Support

The TOE calculates cryptographic SHA-256 hash values over executable files to verify that an executable is permitted by the Application Control policy. Any other cryptographic primitives are provided by the Operating Environment, i.e. the Windows cryptographic API (Cryptography API: Next Generation). The Windows 10 Cryptographic API is trusted because it has been evaluated recently (refer to [WIN10CERT] and [WIN10ST]).

Audit generation

The TOE generates and stores audit records of its activities. The review of the audit data is performed in the administrative backend, i.e. outside the TOE. An audit policy defines which events need to be recorded. For these events, at least the following data is stored:

- Event type / Action by the TOE
- Timestamp
- Object(s) involved
- User currently logged on
- Computer on which the event occurred

Security Management

The policies required for proper operation of the TOE are defined in the administrative backend outside the TOE. The TOE receives these policies over a secure connection and verifies their integrity, using digital signature verification. The secure connection and signature verification algorithm are supplied by the TOE environment. The certificates used for policy verification are maintained by the TOE and are set up as part of the initial configuration of the TOE configuration data. If the policies are authentic, they are forwarded by the user-mode component of the TOE to the respective drivers.

3 Conformance Claims

3.1 CC Conformance Claims

This Security Target is CC Part 2 conformant and CC Part 3 conformant. This ST claims conformance to CC version 3.1 Revision 5, April 2017.

This ST claims no conformance to any Protection Profile. This ST claims conformance to the EAL3 package of security assurance requirements, augmented with ALC_FLR.3.

4 Security Problem Definition

4.1 Assets and Agents

To simplify the description of the threats averted by the TOE definitions of assets and agents precede the actual security problem definition.

4.1.1 Assets

Assets, i.e. the things the TOE claims to protect, are:

AS.LOCALDATA
This is the data accessible to the current user of the workstation, either stored locally or on a network share. Some of this data will be of a proprietary nature, and its disclosure may be harmful to the TOE user.

AS.RESOURCES
These are the computing resources of the workstation. It is in the interest of the TOE operator and users that these resources are used for legitimate and intended purposes only.

AS.LEGITPROCESS
This a process that is intended by the TOE user to run on the workstation.

AS.INTEGRITY
This is the integrity of legitimate processing on the workstation or system. This is an important asset on systems used e.g. for accounting or manufacturing.

AS.TSFDATA
This is the configuration and policy data required for operation of the TOE as intended by the TOE user. This data is treated as an asset here since it is defined outside the system running the TOE and must be imported from an administration backend.

4.1.2 Agents

AG.USER
This is a regular user of the system running the TOE, authenticated by the operating system. This user may or may not have Windows administrative privileges on the workstation, but has no privileges to modify the TOE configuration. An AG.USER may be malicious in the sense that he tries to circumvent some limitations imposed by the TOE.

AG.ADMIN
This is an administrator of the TOE. An administrator acts on behalf of the TOE operator and defines the TOE configuration and policies on the administrative backend.

AG.LEGITPROCESS
This a process that is intended by the TOE user to run on the workstation. Since access to system resources by an AS.LEGITPROCESS is also controlled by the TOE, the process also acts as an Agent in that context.

AG.OUTSIDER
This is an unauthorized person with no legitimate access to the system protected by the TOE.

AG.ILLICITPROCESS This is a process created for an executable file that is not expected to run on the workstation. This could be malware (virus, trojan, ransomware, etc.) that found its way onto the workstation, or simply an executable that is undesirable for other reasons (e.g. adware).

4.2 Threats

The threats the TOE claims to avert, in conjunction with a properly configured TOE environment, are:

T.DATAEXPORT A user AG.USER exports local data AS.LOCALDATA from the workstation to a removable device which is unknown and/or not allowed. Alternatively, a permitted process AG.LEGITPROCESS accesses local data it is not allowed to access.

T.DATAIMPORT A user AG.USER imports data from a removable device which is unknown or not allowed to connect to the workstation, modifying the data AS.LOCALDATA stored on the workstation.

T.DEGRADESYS An illicit process AG.ILLICITPROCESS degrades the AS.RESOURCES, such as performance (e.g. real-time capability) or resources (e.g. storage capacity) of the workstation by using these for its own purposes.

T.CORRUPTSYS An illicit process AG.ILLICITPROCESS manipulates or sabotages the execution of the legitimate processes AS.LEGITPROCESS on the workstation, compromising system integrity AS.INTEGRITY. Alternatively, an illicit process AG.ILLICITPROCESS denies legitimate users AG.USER access to the workstation (AS.RESOURCES) or its data (AS.LOCALDATA). Finally, a permitted process AG.LEGITPROCESS modifies files or settings it is not allowed to access.

T.CORRUPTTSFD A malicious user AG.USER or process AG.ILLICITPROCESS modifies the TSF data (AS.TSFDATA) to manipulate or degrade the security functions of the TOE. Alternatively, a malicious user AG.USER impersonates the server and supplies unauthorized updates to the TSF data.

T.HOSTILEDEVICE An attacker AG.OUTSIDER connects (or induces an unknowing AG.USER to do so) a manipulated device to the system to gain control (e.g. a USB cable secretly posing as a keyboard), compromising system integrity (AS.INTEGRITY) or accessing workstation data (AS.LOCALDATA).

4.3 Organizational Security Policies

OSP.AUDIT Events relevant to the management and enforcement of the TOE security policies shall be recorded as specified by the operator.

4.4 Assumptions

A.EVENTLOG Operating system event log:
It is assumed that the operating system event log is properly configured to receive and retain the TOE-generated audit records until they can be analyzed by the TOE administrator(s)

A.OSLogon Operating system logon:
It is assumed that user identification and authentication is performed by the operating system and that the TOE can query the current Windows user to determine access rights and associate user identities with its audit records where applicable.

A.Policy Policy definition and maintenance:
It is assumed that the TOE administrator defines and deploys suitable policies for Device Control, Application Control, and Audit, carefully following the available guidance for the TOE. It is also assumed that the administrator keeps the policies current and that policy rules are configured to apply to the intended users and computers.

A.ReliableTime Reliable time source:
It is assumed that the TOE and its environment have access to the correct time by using the operating system functions intended for this purpose.

A.SecureConn Secure connection to the administrative backend:
It is assumed that a secure network connection is available to the TOE to connect to its server.

A.TrustedAdmin Trustworthy administrators:
It is assumed that the administrators of the TOE are trustworthy and sufficiently familiar with the TOE to minimize the risk inadvertent misconfiguration, and do not intentionally subvert the TOE’s operation.

5 Security Objectives

5.1 Security Objectives for the TOE

The TOE security objectives are defined as follows:

OT.BLOCKDEVICE The TOE shall selectively block user access to devices according to the operator-defined device control policy.

OT.BLOCKEXECUTE The TOE shall selectively block the execution of unwanted executable files according to the operator-defined application control policy.

OT.BLOCKACCESS The TOE shall selectively block unwanted access by running processes to data, executable, and script files, as well as registry keys according to the operator-defined application permissions policy.

OT.AUDIT The TOE shall generate audit events according to the operator-defined audit policy.

OT.PROTECTTSFD The TOE shall ensure that only authorized administrators can change its configuration.

5.2 Security Objectives for the Operational Environment

The security objectives for the environment are defined below. Note that the term *operator* here denotes the organization or entity using the TOE.

OE.EVENTLOG The operator shall ensure that the operating system event log is properly configured to receive and retain the TOE-generated audit records until they can be analyzed by the TOE administrator(s).

OE.OSLOGON The operator shall ensure that the operating system is configured so that all users are required to authenticate themselves before they can use the system protected by the TOE. If an interactive login to the system is not feasible due to operational concerns the operator shall ensure by other (technical) means that no unauthorized users can interact with the system.

OE.POLICY The operator shall ensure that the TOE administrator defines and deploys suitable policies for Device Control, Application Control, and Audit, carefully following the available guidance for the TOE. He shall also ensure that the administrator keeps the policies current and that policy rules are configured to apply to the intended users and computers.

OE.RELIABLETIME The operator shall ensure that the TOE and its environment have access to the correct time by using the operating system functions intended for this purpose.

OE.CRYPTO The operator shall ensure that the TOE has access to the cryptographic algorithms and infrastructure required for digital certificate and signature verification using RSA certificates and SHA-512 provided by the Windows cryptographic subsystem.

OE.SECURECONN The operator shall ensure that that a secure network connection is available to the TOE to connect to its server.

OE.TrustedAdmin The TOE operator shall ensure that the administrators of the TOE are trustworthy and sufficiently familiar with the TOE to minimize the risk of inadvertent misconfiguration, and do not intentionally subvert the TOE’s operation.

OE.StrongCreds The operator shall ensure that the TOE administrator(s) and users use strong passwords and do not disclose them to any other parties.

OE.Install The operator shall ensure that the TOE and its supporting infrastructure (server and administration application) are properly installed and configured as described in [INSTGUIDE] and [CCGUIDE]. This specifically includes using the Agent Hardening features to prevent users with Windows administrative privileges from disabling TOE functions.

OE.Backend The operator shall ensure that the administrative backend for the TOE is protected from unauthorized access or manipulation.

5.3 Security Objectives Rationale

5.3.1 Tracing of Security Objectives to Threats, OSPs, and Assumptions

The tracing of the security objectives for the TOE and its environment is described in the table below.

	OT.BlockDevice	OT.BlockExecute	OT.BlockAccess	OT.Audit	OT.ProtectTSFD	OE.TrustedAdmin	OE.StrongCreds	OE.Install	OE.Backend	OE.Policy	OE.ReliableTime	OE.EventLog	OE.SecureConn	OE.OSLogon	OE.Crypto
T.DataImport	X			X				X		X		X		X	
T.DataExport	X		X	X				X		X		X		X	
T.DegradeSys		X		X				X		X		X		X	
T.CorruptSys		X	X	X				X		X		X		X	
T.CorruptTSFD				X	X	X	X	X	X			X	X		X
T.HostileDevice	X			X				X		X		X			
OSP.Audit				X							X	X		X	
A.Policy										X					
A.ReliableTime											X				
A.EventLog												X			
A.SecureConn													X		
A.OSLogon														X	
A.TrustedAdmin						X									

Table 1: Mapping of threats, OSPs, and assumptions to objectives

The table shows that each threat or assumption is addressed by at least one security objective for the TOE or the IT environment.

5.3.2 Justification

The threat T.DataImport is addressed as follows:

The TOE administrator deploys a policy that defines which devices are permissible for which users on which computers (OE.Policy). OE.OSLogon requires that the current user is identified before interacting with the system to allow user-specific evaluation of policy rules. The TOE then blocks access to all devices that are not permitted by this policy (OT.BlockDevice), so that data cannot be transferred to or from the device. OE.Install ensures that non-administrative users cannot simply override the policy definition locally. OT.Audit, in combination with OE.EventLog, ensures that the audit records required for review of the TOE activities by the TOE administrator are available.

The threat T.DATAEXPORT is addressed by the same mechanisms as T.DATAIMPORT; in addition, the TOE also blocks unwanted accesses (OT.BLOCKACCESS) to policy-defined files and registry locations by processes running on the workstation.

The threat T.DEGRADESYS is countered as follows:

The administrator deploys a policy that defines which executable images are permitted to run for which users on which computers (OE.POLICY). OE.OSLOGON requires that the current user is identified before interacting with the system to allow user-specific evaluation of policy rules. The TOE then prevents execution of all executable images that are not permitted by this policy (OT.BLOCKEXECUTE). OE.INSTALL ensures that non-administrative users cannot simply override the policy definition locally. OT.AUDIT, in combination with OE.EVENTLOG, ensures that the audit records required for review of the TOE activities by the TOE administrator are available.

The threat T.CORRUPTSYS is countered by the same mechanisms as T.DEGRADESYS; in addition, the TOE also prevents unwanted modifications of data or registry settings (OT.BLOCKACCESS) by processes executing on the workstation.

The threat T.CORRUPTTSFD is countered by the following objectives:

OT.PROTECTTSFD ensures that only authorized administrators can change configuration and policy data using the administrative backend. OE.TRUSTEDADMIN furthermore defines that these administrators must be trustworthy and familiar with the TOE to minimize the risk of both intentional and inadvertent misconfiguration. OE.STRONGCREDS requires that the trusted administrators do not knowingly compromise their credentials. OE.INSTALL requires that the TOE and its administration backend be properly installed to protect the TOE, the backend, and the connection between them. OE.INSTALL also ensures that non-administrative users cannot simply modify the TSF data locally. OE.BACKEND additionally mandates the server and administration components be protected from uncontrolled access to ensure their integrity. OE.SECURECONN ensures that the policies defined by TOE administrators are transferred to the workstation using a trusted and protected channel. OE.CRYPTO ensures that the TOE can verify integrity and authenticity of updates received over this channel, using the Windows cryptographic subsystem.

In short, the combination of the objectives above ensures that only legitimate policy changes by trusted administrators using the intended administrative tools are deployed to the system protected by the TOE. Finally, OT.AUDIT, in combination with OE.EVENTLOG, ensures that attempts to supply manipulated update packages to the TOE can be detected by the TOE administrators.

The threat T.HOSTILEDEVICE is countered by these objectives:

OE.POLICY mandates that a policy defining the permissible devices is deployed to the system protected by the TOE. OT.BLOCKDEVICE then blocks the unknown device from becoming accessible to the system and therefore prevents its harmful interaction with the system. OE.INSTALL ensures that non-administrative users cannot simply override the policy definition locally. OT.AUDIT, in combination with OE.EVENTLOG, ensures that attempts to connect an unknown device can be detected by the TOE administrators.

The assumptions A.RELIABLETIME, A.EVENTLOG, A.SECURECONN, A.OSLOGON, and A.POLICY are covered by respective objectives for the environment.

The organizational security policy OSP.AUDIT introduces the requirement for auditing, which is addressed by OT.AUDIT. For auditing to be effective, correct information on the acting user and a correct timestamp is necessary, which is mandated by OE.OSLOGON and OE.RELIABLETIME. Audit record storage is provided by the underlying operating system, as mandated by OE.EVENTLOG.

6 Extended Components Definitions

There are no extended components defined or used in this ST.

7 Security Requirements

7.1 TOE Security Functional Requirements

7.1.1 Notational Conventions for Operations on SFRs

The *refinement* operation is used to add detail to a requirement and thus further restricts a requirement. Refinements of security requirements are denoted in such a way that added words are in **bold text** and removed words are ~~crossed out~~. If a refinement is added as a separate paragraph to an SFR instead of modifying its wording, this paragraph starts with the word "**Refinement:**" in bold text.

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

The *assignment* operation is used to assign a specific value to an unspecified parameter such as the length of a password. Assignments having been made denoted as underlined text.

The *iteration* operation is used when a component is repeated with varying operations. The fact, that an iteration operation was used is obvious from the fact that a component is contained (at least) twice in the ST. To distinguish the individual instances of a component the component title is amended by adding an (individual name) in parentheses after the component identifier.

7.1.2 Class FAU: Security Audit

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 events specified in the audit policy defined by the TOE administrator.

Application Note: The auditable events for the respective SFRs are summarized in the following table. Except for *device removal*, the events set in this table must be enabled in the audit policy to allow review of the TSF activities.

SFR	Event(s)
FDP_ACC.1 (Device), FDP_ACF.1 (Device)	Device arrival and reaction of the TSF (blocked or allowed) device removal device access reconfiguration due to Windows user change
FDP_ACC.1 (Execute), FDP_ACF.1 (Execute)	Attempts to execute and reaction of the TSF (blocked or allowed) Problems accessing TSF data (hash database)
FDP_ACC.1 (Permissions), FDP_ACF.1 (Permissions)	Attempts to access controlled resources and reaction of the TSF (blocked or allowed) Problems accessing TSF data (application permissions rules)
FMT_SMF.1	Problems accessing the Enterprise Service Missing TSF data (policies) Problems with connection security (TLS, certificates, etc.) Problems with received TSF data update packages

Table 2: Auditable events for relevant SFRs.

FAU_GEN.1.2

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

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

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, no other audit relevant information.

FAU_GEN.2 User identity association

FAU_GEN.2.1

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

7.1.3 Class FCS: Cryptographic Support

FCS_CKM.1 (Certificate)

FCS_CKM.1.1

The ~~TSF~~ **Windows Cryptographic Subsystem** shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm as defined in [FIPS186-4] and specified cryptographic key sizes 4096 bit that meet the following: [FIPS186-4].

Application Note: The cryptographic key generation is performed within the Operational Environment. CSEC policy requires the relevant FCS SFRs to be claimed, even though the corresponding functions are not part of the TOE.

FCS_CKM.2 (Certificate)

FCS_CKM.2.1

The ~~TSF~~ **Windows Cryptographic Subsystem** shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method proprietary distribution method that meets the following: None.

Application Note: The distribution of digital certificates is performed within the Operational Environment. CSEC policy requires the relevant FCS SFRs to be claimed, even though the corresponding functions are not part of the TOE.

FCS_COP.1 (SHA-2 (Ident))

FCS_COP.1.1

The TSF shall perform executable file identification in accordance with a specified cryptographic algorithm SHA-256 or SHA-512 ~~and cryptographic key sizes~~ that meet the following: [FIPS180-4].

Application Note: The algorithm used is configurable. The default setting is SHA-256.

FCS_COP.1 (SHA-512 (Policy))

FCS_COP.1.1

The TSF shall perform policy file integrity verification in accordance with a specified cryptographic algorithm SHA-512 ~~and cryptographic key sizes~~ that meet the following: [FIPS180-4].

Application Note: The algorithm is provided by the Operational Environment (described further in [WIN10ST]. CSEC policy requires the relevant FCS SFRs to be claimed, even though the corresponding functions are not part of the TOE.

FCS_COP.1 (RSA(Policy))

FCS_COP.1.1

The TSF shall perform digital signature verification in accordance with a specified cryptographic algorithm RSA with SHA-256 and cryptographic key sizes 4096 bit that meet the following: [FIPS186-4], section 4.

Application Note: RSA digital signature verification is used by the TOE management to ensure authenticity of policy updates. The algorithm is provided by the Operational Environment (described further in [WIN10ST]. CSEC policy requires the relevant FCS SFRs to be claimed, even though the corresponding functions are not part of the TOE.

7.1.4 Class FDP: User Data Protection

7.1.4.1 Device Control

SFP_DEV		
Type	Short Name	Definition
Subjects	S_PNP	The PnP (Plug-and-Play) enumerator invoked by the operating system when a device arrival is detected
Objects	O_DEV	PnP devices (e.g. disks, cameras) Ports (e.g. serial or parallel) Controllers (e.g. IEEE-1394 bus controllers)
Operations	Block_Access	The O_DEV is prevented from becoming active and available
	Allow_Access	The O_DEV is allowed to complete activation and becomes available to the User
Subject security attributes	User_ID	The currently logged-on Windows user, on whose behalf S_PNP acts
Object security attributes	Device class Device bus Device manufacturer Device hardware ID	Properties of the device the user tries to connect.
Rules	R_Device	Any device connected to the system is blocked unless the Device_Policy allows it. The decision is derived from the security attributes of Subject and Object.
	R_NoDevPolicy	If no Device_Policy exists (i.e. before the initial configuration has been received by the TOE), PnP devices are not blocked. Legacy serial and parallel ports are blocked.
TSF Data	Device_Policy	The definition of allowed devices set by the TOE administrator

Table 3: Device Control SFP

FDP_ACC.1 (Device) Subset access control

FDP_ACC.1.1

The TSF shall enforce the SFP_DEV on all subjects, objects defined by the SFP_DEV, and all operations among subjects and objects covered by the SFP.

Application Note: The device configuration process controlled by this function is initiated by the operating system, i.e. outside the TOE.

FDP_ACF.1 (Device) Security attribute based access control

FDP_ACF.1.1

The TSF shall enforce the SFP_DEV to objects based on the following: All subjects and objects together with their respective security attributes as defined in SFP_DEV.

FDP_ACF.1.2

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: Rules for all access methods and access rules defined in SFP_DEV.

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 following additional rules: None.

7.1.4.2 Application Execution Control

SFP_APP.Execution		
Type	Short Name	Definition
Subjects	S_SHELL	The process attempting to launch an application on behalf of the current user.
Objects	O_EXE	Executable files (.EXE, .DLL, etc.)
Operations	Block_Execute	The O_EXE is prevented from loading and executing
	Allow_Execute	The O_EXE is allowed to run
Subject security attributes	User_ID	The currently logged-on Windows user, on whose behalf S_SHELL acts
Object security attributes	Hash_Value	The cryptographic hash value calculated over the contents of O_EXE
Rules	R_Application	Any executable is blocked from executing unless the Application_Policy allows it. The decision is derived from the security attributes of Subject and Object.
	R_NoAppPolicy	If no Application_Policy exists (i.e. before the initial configuration has been received by the TOE), no applications or resource accesses are blocked.
TSF Data	Application_Policy	The definition of permitted executables set by the TOE administrator

Table 4: Application Control SFP

FDP_ACC.1 (Execute) Subset access control

FDP_ACC.1.1

The TSF shall enforce the SFP_APP.Execution on all subjects, objects defined by the SFP_APP.Execution, and all operations among subjects and objects covered by the SFP.

Application Note: The application launch process controlled by this function is initiated by the operating system, i.e. outside the TOE.

FDP_ACF.1 (Execute) Security attribute based access control

FDP_ACF.1.1

The TSF shall enforce the SFP_APP.Execution to objects based on the following: All subjects and objects together with their respective security attributes as defined in SFP_APP.Execution.

FDP_ACF.1.2

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: Rules for all access methods and access rules defined in SFP_APP.Execution.

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 following additional rules: None.

7.1.4.3 Application Permissions

Note that this SFP distinguishes between rules of the SFP, termed *SFP_Rules*, and TOE configuration defined *rules*, which are used to enforce the more abstract SFP_Rules.

SFP_APP.Permissions		
Type	Short Name	Definition
Subjects	S_PROCESS	The process attempting to access a resource
Objects	O_FILE	Any file (or a set of files if wildcards are used)
	O_REGKEY	A registry key
	O_DLL	A dynamically loadable module or a set thereof (if wildcards are used). The file doesn’t necessarily have a .DLL file name extension
	O_SCRIPT	A file (or a set thereof) with one of the file name extensions defined in script types configuration.

Operations	Block_Access	The S_PROCESS is prevented from accessing the Object
	Allow_Access	The S_PROCESS is allowed to access the Object
Subject security attributes	Executable file specification	The executable file that was loaded when the S_PROCESS was created
Object security attributes	Object path	A file name specification or a registry path of the Object
SFP_Rules	R_Permissions	Application_Permissions rules are evaluated in order of priority, highest to lowest. The highest priority rule that matches the combination of (Subject, Object, Access) is applied to determine if the Access is allowed or denied.
	R_ImpliedAccess	Allowing Write Access to an Object will implicitly also allow Read Access to that Object.
	R_NoAppPerm	If no Application_Permissions are defined (e.g. before the initial configuration has been received by the TOE), no resource accesses are blocked. Similarly, if no rule is found for a given (Subject, Object, Access) combination, the Access is permitted.
TSF Data	Application_Permissions	The defined application permission rules set by the TOE administrator. A rule is a quintuple of (Subject, Object, Access, Priority, Permission).
	Script type definitions	A set of configuration items describing which file name extensions are to be treated as scripts and which S_PROCESS would run scripts of this type

Table 5: Application Permissions SFP

FDP_ACC.1 (Permissions) Subset access control

FDP_ACC.1.1

The TSF shall enforce the SFP_APP.Permissions on all subjects, objects defined by the SFP_APP.Permissions, and all operations among subjects and objects covered by the SFP.

FDP_ACF.1 (Permissions) Security attribute based access control

FDP_ACF.1.1

The TSF shall enforce the SFP_APP.Permissions to objects based on the following: All subjects and objects together with their respective security attributes as defined in SFP_APP.Permissions.

FDP_ACF.1.2

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: Rules for all access methods and access rules defined in SFP_APP.Permissions.

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 following additional rules: None.

7.1.5 Class FMT: Security Management

FMT_SMR.1 Security roles

FMT_SMR.1.1

The TSF shall maintain the roles AG.USER, AG.ADMIN.

FMT_SMR.1.2

The TSF shall be able to associate users with roles.

Application Note: For the sake of clarity, the following description distinguishes between these three user classes although the TOE does not maintain three roles:

User Type	Description
Workstation User	This is a regular, non-privileged user of the workstation (and the TOE). This user type corresponds to the role AG.USER.
Workstation Administrator	This is a privileged user with respect to the operating system on the workstation. For the TOE, this user type assumes the role AG.USER.
TOE Administrator	This is a privileged user with respect to the TOE and corresponds to the role AG.ADMIN. However, the role AG.ADMIN is primarily known in the administrative environment (i.e. outside the TOE boundary). In the TOE it is only represented by the digital signatures of configuration or policy changes, and the certificate(s) installed to verify them. Consequently, this role is solely maintained by the TOE administrative interface. For all other parts of the TOE, this user type assumes the role AG.USER.

Table 6: User classes vs. TOE security roles

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

The TSF shall be capable of performing the following management functions: Receive policy and configuration updates from the administrative backend.

FMT_MTD.1 Management of TSF data

FMT_MTD.1.1

The TSF shall restrict the ability to modify, create the

(i) Application Control Policy
(ii) Application Permissions Policy
(iii) Device Control Policy
(iv) Audit Policy
(v) Configuration Signing Certificate

to the role AG.ADMIN.

Application Note: The role AG.ADMIN is primarily maintained outside the TOE in the Administration Console. The restriction is enforced by the TOE only accepting configuration changes that include a valid digital signature created with a key defined and exclusively owned by the role AG.ADMIN, i.e. the private key corresponding to the currently installed Configuration Signing Certificate. Changes with invalid signatures or with valid signatures by other keys are ignored.

FMT_MOF.1 Management of security functions behaviour

FMT_MOF.1.1

The TSF shall restrict the ability to determine the behaviour of, disable the functions all security functions to the role AG.ADMIN.

Application Note: Disabling all security functions here means uninstalling the TOE.

Application Note: The role AG.ADMIN is only maintained outside the TOE in the Administration Console. The restriction is enforced by the TOE accepting only digitally signed configuration changes defined in the Administration Console, which limits access to the role AG.ADMIN.

7.1.6 Class FPT: Protection of the TSF

FPT_ITI.1: Inter-TSF detection of modification

FPT_ITI.1.1

The TSF shall provide the capability to detect modification of all TSF data during transmission between the TSF and another trusted IT product within the following metric: The likelihood of an undetected change is less or equal to that of a successful collision attack on SHA-256.

Application Note: The trusted IT product in this context is the administrative backend. The data transmitted from the backend to the TOE is digitally signed (see FCS_COP.1 (RSA(Policy))). An undetectable modification would therefore require a successful collision attack on the cryptographic hash function used in the signature or a successful attack on 4096-bit RSA, which is deemed even harder than the collision attack.

FPT_ITI.1.2

The TSF shall provide the capability to verify the integrity of all TSF data transmitted between the TSF and another trusted IT product and perform rejection of the data if modifications are detected.

Application Note: Since the TOE periodically checks for updates a verification failure is audited but no further corrective action is taken.

FPT_TDC.1: Inter-TSF basic TSF data consistency

FPT_TDC.1.1

The TSF shall provide the capability to consistently interpret

(i) Configuration update packages
(ii) Configuration certificate packages

when shared between the TSF and another trusted IT product.

Application Note: The trusted IT product in this context is the administrative backend.

FPT_TDC.1.2

The TSF shall use the rules listed below when interpreting the TSF data from another trusted IT product.

(i) Any configuration update package must include a valid digital signature created by the administrative backend using the private key corresponding to the configuration certificate currently installed on the workstation.

(ii) The configuration data in the update package must be newer than the corresponding TSF data, otherwise the package is rejected.

(iii) The currently installed configuration certificate must be valid at the time a configuration update package or a configuration certificate package is verified, otherwise the package is rejected.

(iv) A configuration certificate package must be a valid [PKCS#7] file created by the administrative backend and signed by the currently installed configuration certificate.

7.2 Security Functional Requirements Rationale

7.2.1 Tracing of SFRs to Security Objectives

	OT.AUDIT	OT.BLOCKACCESS	OT.BLOCKDEVICE	OT.BLOCKEXECUTE	OT.PROTECTTSFD	OE.SECURECONN	OE.CRYPTO
FAU_GEN.1	X						
FAU_GEN.2	X						
FCS_CKM.1 (Certificate)						X	
FCS_CKM.2 (Certificate)						X	
FCS_COP.1 (RSA (Policy))							X
FCS_COP.1 (SHA-2 (Ident))				X			
FCS_COP.1 (SHA-512 (Policy))							X
FDP_ACC.1 (Device)			X				
FDP_ACC.1 (Execute)				X			
FDP_ACC.1 (Permissions)		X					
FDP_ACF.1 (Device)			X				
FDP_ACF.1 (Execute)				X			
FDP_ACF.1 (Permissions)		X					
FMT_MOF.1					X		
FMT_MTD.1					X		
FMT_SMF.1	X	X	X	X	X		
FMT_SMR.1					X		
FPT_ITI.1					X		
FPT_TDC.1					X		

Table 7: Tracing of SFRs to security objectives

7.2.2 Justification

To meet the objective OT.AUDIT the SFRs FAU_GEN.1 and FAU_GEN.2 require that the TOE creates audit records of its activities, especially on events relating to the access control functions of the TOE. FMT_SMF.1 ensures that an audit policy can be deployed on the client.

The objective OT.BLOCKACCESS is implemented by the SFRs FDP_ACC.1 (Permissions) and FDP_ACF.1 (Permissions), which establish and enforce an access control function that allows or prevents access by running processes to files or registry keys according to the defined policy. FMT_SMF.1 ensures that the required application permissions policy can be deployed on the client.

To implement the objective OT.BLOCKDEVICE the SFRs FDP_ACC.1 (Device) and FDP_ACF.1 (Device) establish and enforce an access control function which prevents any devices not permitted by the device policy from becoming accessible to the user. FMT_SMF.1 ensures that the required device control policy can be deployed on the client.

Similarly, to implement the objective OT.BLOCKEXECUTE the SFRs FDP_ACC.1 (Execute) and FDP_ACF.1 (Execute) establish and enforce an access control function which prevents any executable files not permitted by the application control policy from running. To match a given executable file against the whitelist a cryptographic hash is calculated over the file contents (FCS_COP.1 (SHA-2 Ident)). FMT_SMF.1 ensures that the required application control policy can be deployed on the client.

The objective OT.PROTECTTSFD is met by restricting the modification of TOE configuration data (FMT_MTD.1, FMT_SMF.1, FPT_ITI.1, FPT_TDC.1) and execution parameters (FMT_MOF.1) to TOE administrators (FMT_SMR.1). FMT_MTD.1 also ensures that administrator-defined configuration data updates are unchanged and authentic, with the help of FCS_COP.1 (SHA-512 (Policy)) and FCS_COP.1 (RSA (Policy)), which are implemented by the operational environment.

All other FCS_* SFRs that map against the objective for the operational environment OE.SECURECONN are required by CSEC policy to be claimed, even though the corresponding functions are not part of the TOE. These are implemented by the operational environment.

7.2.3 Dependency Rationale

SFR ID	Dependencies	Dependency Met by TOE	Details, Comments
FAU_GEN.1	FPT_STM.1	No	Satisfied by an objective for the environment, OE.RELIABLETIME, which ensures the TOE receives accurate time from the operating system.
FAU_GEN.2	FAU_GEN.1	Yes	Satisfied by FAU_GEN.1
	FIA_UID.1	No	Satisfied by an objective for the environment, OE.OSLOGON, as the TOE uses Windows session attributes instead of a proprietary logon.
FCS_CKM.1 (Certificate)	FCS_CKM.2	Yes	Satisfied by FCS_CKM.2 (Certificate)
	FCS_CKM.4	No	Dependency not applicable as key pairs remain in existence until replaced
FCS_CKM.2 (Certificate)	FCS_CKM.1	Yes	Satisfied by FCS_CKM.1 (Certificate)
	FCS_CKM.4	No	Dependency not applicable as key pairs remain in existence until replaced
FCS_COP.1 (SHA-2 (Ident))	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	No	Dependencies not applicable as SHA hash functions do not use a key
	FCS_CKM.4	No	Dependency not applicable as SHA hash functions do not use a key
FCS_COP.1 (SHA-512 (Policy))	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	No	Dependencies not applicable as SHA hash functions do not use a key
	FCS_CKM.4	No	Dependency not applicable as SHA hash functions do not use a key
FCS_COP.1 (RSA(Policy))	FCS_CKM.1	Yes	Satisfied by FCS_CKM.1 (Certificate)
	FCS_CKM.4	No	Dependency not applicable as key pairs remain in existence until replaced
FDP_ACC.1 (Device)	FDP_ACF.1	Yes	Satisfied by FDP_ACF.1 (Device)
FDP_ACC.1 (Execute)	FDP_ACF.1	Yes	Satisfied by FDP_ACF.1 (Execute)
FDP_ACC.1 (Permissions)	FDP_ACF.1	Yes	Satisfied by FDP_ACF.1 (Permissions)
FDP_ACF.1 (Device)	FDP_ACC.1	Yes	Satisfied by FDP_ACC.1 (Device)
	FMT_MSA.3	No	Dependency not applicable as the identified security attributes are provided by the operating system, i.e. not initialized by the TSF
FDP_ACF.1 (Execute)	FDP_ACC.1	Yes	Satisfied by FDP_ACC.1 (Execute)
	FMT_MSA.3	No	Dependency not applicable as the identified security attributes are either

SFR ID	Dependencies	Dependency Met by TOE	Details, Comments
			provided by the operating system or are part of the TSF data, i.e. not initialized by the TSF
FDP_ACF.1 (Permissions)	FDP_ACC.1	Yes	Satisfied by FDP_ACC.1 (Permissions)
	FMT_MSA.3	No	Dependency not applicable as the identified security attributes are either provided by the operating system or are part of the TSF data, i.e. not initialized by the TSF
FMT_MOF.1	FMT_SMR.1	Yes	Satisfied by FMT_SMR.1
	FMT_SMF.1	Yes	Satisfied by FMT_SMF.1
FMT_MTD.1	FMT_SMR.1	Yes	Satisfied by FMT_SMR.1
	FMT_SMF.1	Yes	Satisfied by FMT_SMF.1
FMT_SMF.1	None	No	n/a
FMT_SMR.1	FIA_UID.1	No	Dependency is not included in the TOE but satisfied by an objective for the environment, OE.OSLOGON, as the TOE uses Windows session attributes instead of a proprietary logon.
FPT_ITI.1	None	No	n/a
FPT_TDC.1	None	No	n/a

Table 8: SFR Dependencies

The table shows that all applicable dependencies are either met by the requisite SFRs or an objective for the TOE environment.

7.3 Security Assurance Requirements

The assurance requirements quoted in the table below are taken from [CC3] for the claimed assurance package of *EAL 3 augmented with ALC_FLR.3*.

Assurance Requirements	
Class ASE: Security Target Evaluation	ASE_CCL.1: Conformance claims
	ASE_ECD.1: Extended components definition
	ASE_INT.1: ST introduction
	ASE_OBJ.2: Security objectives
	ASE_REQ.2: Derived security requirements
	ASE_SPD.1: Security problem definition
	ASE_TSS.1: TOE summary specification
Class ADV: Development	ADV_ARC.1: Security architecture description
	ADV_FSP.3: Functional specification with complete summary
	ADV_TDS.2: Architectural design
Class ALC: Life Cycle Support	ALC_CMC.3: Authorisation controls
	ALC_CMS.3: Implementation representation CM coverage
	ALC_DEL.1: Delivery procedures
	ALC_DVS.1: Identification of security measures
	ALC_FLR.3: Systematic flaw remediation
	ALC_LCD.1: Developer defined life-cycle model
Class AGD: Guidance Documents	AGD_OPE.1: Operational user guidance
	AGD_PRE.1: Preparative procedures
Class ATE: Tests	ATE_COV.2: Analysis of coverage
	ATE_DPT.1: Testing: basic design
	ATE_FUN.1: Functional testing
	ATE_IND.2: Independent testing - sample
Class AVA: Vulnerability Assessment	AVA_VAN.2: Vulnerability analysis

Table 9: Assurance requirements package

7.4 Assurance Requirements Rationale

The EAL 3 package is chosen as a reasonable trade-off between evaluation depth and time-to-market for a first evaluation. The augmentation of this package with ALC_FLR.3 is seen as useful for the intended market as the TOE includes automated update capabilities and the requisite procedures for reporting and remediation of flaws are already in place.

8 TOE Summary Specification

This section describes how the TOE meets the functional requirements described in previous sections of this ST.

8.1 TOE Security Functions

8.1.1 SF1 – Device Control

Device Control enables the TOE administrator to control which devices can be used on a workstation by which user. The rules are defined by deploying a policy that essentially uses a white-list approach. To enforce these rules the TOE contains a kernel-mode component (driver) that is aware of a device being connected before that device becomes operational. Whenever a device is connected to the system, the operating system processes a sequence of PnP (Plug-and-Play) events to install and activate the device. Device Control intercepts some of these events and evaluates if the device should become accessible, based on the policy data and the current Windows user.

If the result of this evaluation is that the device shall not be permitted, Device Control prevents completion of the required PnP activities, forcing the device activation to fail. As a result, the device remains unable to interact with the system and thus inaccessible to system and users. If the device is permitted, the installation and activation of the device by the Windows PnP manager proceeds normally and the device becomes available to the system and thus to the user.

Relevant SFRs: FDP_ACC.1 (Device), FDP_ACF.1 (Device)

8.1.2 SF2 – Application Control

Application Control enables the operator to control which executables are permitted to run on a workstation and what permitted executables are allowed to access. A white-list based policy defines the permitted executables. The criterion to permit an executable image is a match of the cryptographic hash calculated over the executable contents to an entry in a hash database. Multiple whitelist rules can be defined; additional parameters control the applicability of a specific whitelist rule, e.g. the currently logged-on Windows user.

To enforce the operator-defined policy the TOE contains a kernel-mode component that monitors any requests to create a new process or load an executable file (e.g. a DLL). Whenever such an event occurs the TOE evaluates if the executable image about to be loaded is permitted by its policy or not. If the executable is not permitted, the request is aborted by the TOE, preventing the executable from running.

If an executable is permitted to run, the TOE administrator can additionally control the access of the resulting process to files or registry keys. Processes can be explicitly allowed or denied access to such resources. The TOE kernel mode components intercept access attempts and block them if they are not permissible according to the defined policy. Note that access is permitted unless a rule exists that forbids it. Explicitly allowing access is only required if access to a resource is generally not allowed by a rule, and an exception is required for a specific process, overriding the general rule. Selectable rule priorities are used to unambiguously define such situations.

Relevant SFRs: FDP_ACC.1 (Execute), FDP_ACF.1 (Execute), FDP_ACC.1 (Permissions), FDP_ACF.1 (Permissions), FCS_COP.1 (SHA-2 (Ident))

8.1.3 SF3 – Security Audit

The TOE generates audit records of various events related to its security functions (see 7.1.2 for details). An operator-defined audit policy controls which events are recorded and where they are sent. This audit trail is buffered locally using the event log facilities provided by the operating system. In addition, events can be sent to other destinations, like the DriveLock Enterprise Service, SMTP, or SNMP addresses. The table below details the most relevant audit events in relation to the security functions of this summary.

Function	Event(s)
SF1 – Device Control	Device arrival and reaction of the TSF (blocked or allowed) device removal device access reconfiguration due to Windows user change
SF2 – Application Control	Attempts to execute and reaction of the TSF (blocked or allowed) Attempts to access files or registry keys and reaction of the TSF (blocked or allowed) Problems accessing TSF data (hash database) or policy data
SF3 – Security Audit	Agent service start-up and shutdown
SF4 – TOE Management	Problems accessing the Enterprise Service Missing TSF data (policies) Problems with connection security (TLS, certificates, etc.) Problems with received TSF data update packages

Table 10: Relevant events audited for security functions of the TOE

Review of the audit data recorded is possible in the DriveLock Control Center (not part of the TOE), where a variety of filtering options is available.

Relevant SFRs: FAU_GEN.1, FAU_GEN.2

8.1.4 SF4 – TOE Management

The administration of the TOE takes place on a central server (DriveLock Enterprise Service), using the Management Console. All management data is maintained by the server in a database. All these components are not part of the TOE.

The TOE receives its configuration and policy data over a network connection to an intranet web service running on the server. This connection is a secure channel that ensures confidentiality and authenticity. The secure channel is provided by the environment. To protect the integrity of the local TSF data all policy data received via this channel must be digitally signed with a specific key, otherwise the data is rejected. This key is deployed as part of the initial configuration of the TOE by the TOE administrator. The cryptographic functions for signature verification are provided by the TOE environment. The TOE also verifies that the received policy data is newer than the data it is intended to replace. This ensures that an attacker cannot reuse outdated but validly signed policy updates to undo later policy changes.

Relevant SFRs: FMT_MOF.1, FMT_MTD.1, FMT_SMF.1, FMT_SMR.1, FPT_ITI.1, FPT_TDC.1

8.2 Security Functions Rationale

8.2.1 Tracing of SFRs to Security Functions

	SF1 – Device Control	SF2 – Application Control	SF3 – Security Audit	SF4 – TOE Management	Operational Environment
FDP_ACC.1 (Device)	X				
FDP_ACF.1 (Device)	X				
FDP_ACC.1 (Execute)		X			
FDP_ACF.1 (Execute)		X			
FDP_ACC.1 (Permissions)		X			
FDP_ACF.1 (Permissions)		X			
FCS_COP.1 (SHA-2 (Ident))		X			
FCS_COP.1 (SHA-512 (Policy))					X
FCS_COP.1 (RSA(Policy)					X
FCS_CKM.1 (Certificate)					X
FCS_CKM.2 (Certificate)					X
FAU_GEN.1			X		
FAU_GEN.2			X		
FMT_MOF.1				X	
FMT_MTD.1				X	
FMT_SMF.1				X	
FMT_SMR.1				X	
FPT_ITI.1				X	
FPT_TDC.1				X	

Table 11: Tracing of SFRs to Security Functions

8.2.2 Justification / Explanation

SF1 *Device Control* implements the SFRs FDP_ACC.1 (Device) and FDP_ACF.1 (Device) by filtering all device activation attempts according to the operator-defined policy.

SF2 *Application Control* implements the SFRs FDP_ACC.1 (Execute) and FDP_ACF.1 (Execute) by intercepting all attempts to load an executable and preventing execution if the executable is not permissible according to the operator-defined policy. To decide if an executable is permissible the TOE calculates a cryptographic hash over the executable contents (FCS_COP.1 (SHA-2 (Ident)) and compares the result to the whitelist in the operator-provided policy. It also implements the SFRs FDP_ACC.1 (Permissions) and FDP_ACF.1 (Permissions) by intercepting all attempts to access files or registry keys and blocking them if the operator-defined policy requires this.

SF3 *Audit* implements the SFRs FAU_GEN.1 and FAU_GEN.2 by generating audit records on the activities of SF1 Device Control and SF2 Application Control and transmitting them to the destination(s) defined by the Audit policy.

SF4 *TOE Management* implements the interface to the administrative backend. Configuration changes can only be introduced with properly signed updates (FMT_SMF.1, FPT_ITI.1, FPT_TDC.1) created by TOE administrators (FMT_SMR.1, FMT_MTD.1).

While the cryptographic functions for signature verification of policy updates (FCS_COP.1 (SHA-512 (Policy)), FCS_COP.1 (RSA (Policy))) are not part of the TOE the results of the verification are used to enforce the update rules described in FPT_TDC.1.

Suspending TOE functions or uninstalling the TOE is also restricted to TOE administrators (FMT_MOF.1).

9 References

[CC1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; CCMB-2017-04-001, Version 3.1, Revision 5, April 2017

[CC2] Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components; CCMB-2017-04-002, Version 3.1, Revision 5, April 2017

[CC3] Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Components CCMB-2017-04-003, Version 3.1, Revision 5, April 2017

[INSTGUIDE] DriveLock Installation Guide 19.2; DriveLock SE, March 2020

[CCGUIDE] Manual Supplement for Certification Compliant Operation 2019.2; DriveLock SE, March 2020

[FIPS180-4] FIPS PUB 180-4: Secure Hash Standard (SHS); NIST, 2015

[FIPS186-4] FIPS PUB 186-4: Digital Signature Standard (DSS); NIST, 2013

[PKCS#7] RFC2315: PKCS #7: Cryptographic Message Syntax, Version 1.5; B. Kaliski, RSA Laboratories, 1998

[TLS] RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2; T. Dierks, E. Rescorla; August 2008

[WIN10ST] Microsoft Windows 10 and Server version 1903 (May 2019 Update) Security Target; Microsoft, July 2019; Location: https://commoncriteriaportal.org/files/epfiles/2019-22-ST_lite.pdf

[WIN10CERT] Microsoft Windows 10 and Server version 1903 (May 2019 Update) Certification Report; Centro Criptológico Nacional (CCN) del Centro Nacional de Inteligencia (CNI), August 2019; Location: https://commoncriteriaportal.org/files/epfiles/2019-22-INF-2839.pdf