REFERENCE

TCG Storage
Opal Integration Guidelines

Version 1.00
Revision 1.14
August 17, 2015
DRAFT

Contact: admin@trustedcomputinggroup.org

Work in Progress:
This document is an intermediate draft for comment only and is subject to change without notice. Readers should not design products based on this document

TCG

TCG Public Review

Disclaimers, Notices, and License Terms

Revision History

Version 1.00	Date	Description
Rev 1.01	29 September, 2011	Initial Draft
Rev 1.03	February 14, 2012	Integrated UEFI Secure Boot Section
Rev 1.04	March 20, 2012	Integrated Out of Band SID Delivery Section
Rev 1.05	March 22, 2012	Updated UEFI section to accommodate handling Admin1 credentials as well. Added a section on Out of Band SID Delivery Added an introductory section on the overview of ownership Many typo and edits Updated references.
Rev 1.06	May 24, 2012	Many editorial updates
Rev 1.07	August 14, 2012	Editorial Updates based on TC Feedback
Rev 1.08	November 13, 2014	Added PSID
Rev 1.09	February 20, 2015	• Added additional references • Added Block SID • Reorganized sections • Cleaned up existing text
Rev 1.10	March 22, 2015	• Added para in 4.0 about Admin rights requirement • Added section 5.1 – Command Execution default • Added Section 5.2.3 TPM PPI
Rev 1.11	April 17, 2015	• Removed sections related to using UEFI to transfer PINs. • Added additional context to Block SID section • Removed Enterprise SSC guidance
Rev 1.12	May 15, 2015	• Added flow diagram for capsule update • Added hardware resets/block sid interactions • Added section about PSID and block sid
Rev 1.13	June 4, 2015	• Miscelleneous editorial updates • Re-inserted sections related to using UEFI to transfer PINs • Added section on leveraging the TPM • Added additional details to TPM PPI section
Rev 1.14	July 29, 2015	• Transitioned to Reference Document template • Editorial corrections based on TC feedback

Table of Contents

1 Introduction

This section summarizes the purpose, scope, and intended audience for this document. The contents of this document are informative.

1.1 Purpose and Scope

This document provides guidelines on integrating SDs implemented according to the Opal Family of specifications. This includes a description of the ownership model utilized in the TCG Storage specifications; the SID authority and its role in managing the storage device; and the processes and guidelines for taking ownership of the TPer. This document also provides insight on integration of other Opal features into systems.

1.2 Intended Audience

The intended audience for this document is implementers of systems using devices implementing the Opal Family of specifications.

1.3 Document References

[1] Trusted Computing Group (TCG), “TCG Storage Architecture Core Specification”, Version 2.01
[2] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Opal”, Version 1.00
[3] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Opal”, Version 2.00
[4] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Opal”, Version 2.01
[5] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Enterprise”, Version 1.00
[6] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Opalite”, Version 1.00
[7] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Pyrite”, Version 1.00
[8] Trusted Computing Group (TCG), “TCG Storage Interface Interactions Specification“, Version 1.04
[9] Trusted Computing Group (TCG), “TCG Storage Opal SSC Feature Set: PSID”, Version 1.00
[10] Trusted Computing Group (TCG), “TCG Storage Feature Set: Block SID Authentication”, Version 1.00
[11] Trusted Computing Group (TCG), “TCG PC Client Platform Physical Presence Interface Specification”, Version 1.30
[12] Unified Extensible Firmware Interface Specification Version 2.3.1
[13] INCITS 482-2012, "Information technology - ATA/ATAPI Command Set - 2 (ACS-2)". Available from http://webstore.ansi.org/.ansi.org/

1.4 Terminology

This section provides definitions specific to this document.

Table 1 Terminology

Term	Definition
IBV	Independent BIOS Vendor
Opal Family	Security Subsystem Class (SSC) specs based on Opal SSC, including all published versions of Opal ([2], [3], and [4]); Opalite ([6]); and Pyrite ([7]).
MSID Credential	The value of the PIN column of the MSID C_PIN object. May also be referred to as just “MSID”.

2 Overview

A TCG Storage solution is a combination of:

1. A Storage Device that implements TCG Storage specifications
2. Host Tools/Software that enable easy & secure provisioning, configuration, and ongoing management

In the following figure, only a single example of user ownership is defined. Given the diversity of TCG Storage Device options, tools vendors, participating authorities, and customer configuration options, a permutation rich ownership path exists.

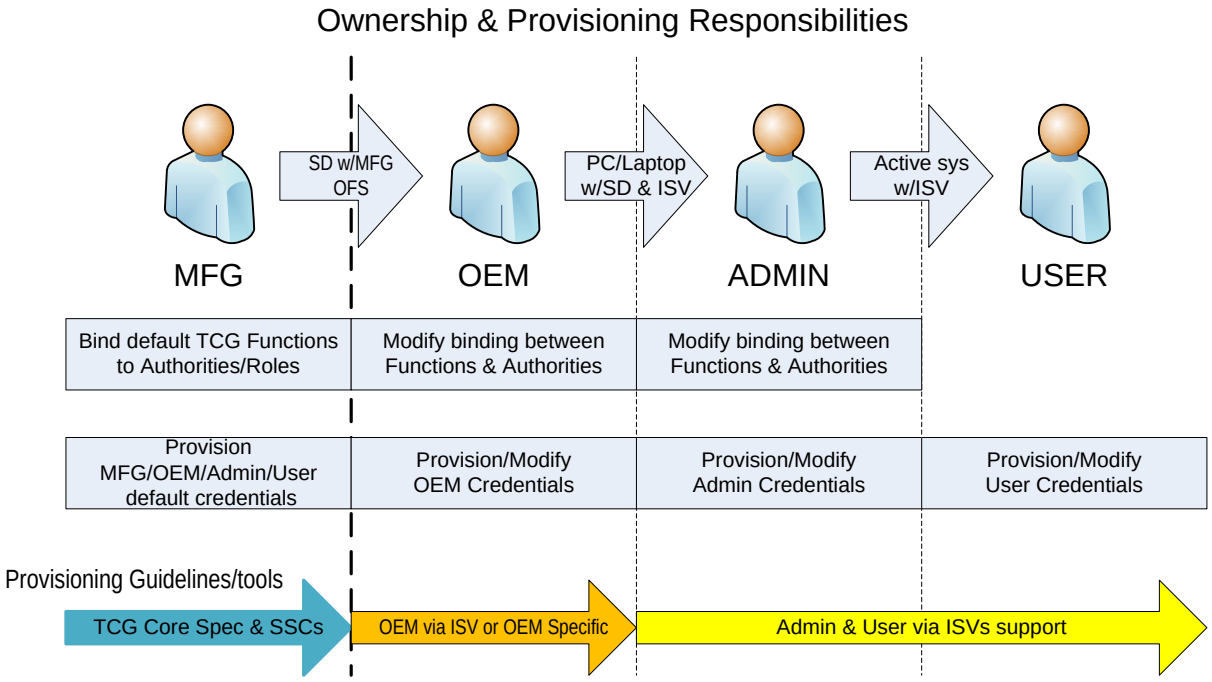


Figure 1 – Ownership & Provisioning Responsibilities

This document provides guidelines on:

1. How to Take Ownership models may be implemented.
2. Best practices for deploying TCG Storage capabilities.

3 Take Ownership Model

The TCG Storage Architecture Core Specification [1] describes the underlying architecture upon which other TCG Storage specifications are based.

The SID authority, which is defined in [1], is used by the TPer owner (i.e., “storage device owner”) to authenticate to the Admin SP and perform management activities on the security subsystem or on individual Security Providers (SPs).

However, [1] does not discuss the process by which the TPer owner obtains the initial SID credential value.

The Security Subsystem Classes (SSCs) define the initial state for the SID credential, and by extension the process of obtaining and setting the SID credential, known as “Taking Ownership”.

3.1 MSID Credential

In order to facilitate the process of taking ownership and in particular of integrating Self-Encrypting Drives (SEDs) in an automated way during the manufacturing of systems, the Enterprise SSC (see [5]) introduced the concept of the MSID Credential. The MSID Credential is a PIN credential that does not have an associated authority. Rather, the MSID is used to store a fixed PIN value that can be read by anybody.

The MSID Credential is set at manufacturing time by the storage device vendor. It can be used as the device’s initial SID PIN value (the default storage device owner’s password). The MSID Credential is electronically readable over the interface by the host without authentication.

During enrollment and take ownership, the host reads the MSID PIN value from the device and uses that value to authenticate to the SID authority in the Admin SP, and then change the SID from its default value.

3.2 Opal, Opalite, and Pyrite SSCs

Opal (see [3] and [4]), Opalite (see [6]), and Pyrite (see [7]) SSCs borrow the concept of MSID from the Enterprise SSC. The MSID Credential value is set at manufacturing time by the storage device vendor, and is readable by anybody. MSID represents the device’s initial (at the Original Factory State – OFS) SID PIN value.

During enrollment the host reads the MSID Credential value from the device and uses that to authenticate and change the SID PIN value.

3.3 Opal SSC 2.00

Opal SSC 2.00 (see [3]) and subsequent versions (see [4]) introduced an additional option, to set the default SID credential to a Vendor Unique value other than MSID. As noted in the referenced specifications, if a vendor elects to set SID to a Vendor Unique value, an alternative mechanism for Taking Ownership must be provided.

3.4 Take Ownership Model – Implications

The mechanism whereby the default SID credential is set to an MSID value that can be read across the interface provides a “first comer” ownership model. In other words, the first to gain access to the MSID Credential value, which is readable by anybody, can take ownership of the SED.

The rationale for choosing this ownership model is to facilitate the automated integration of SEDs into systems by OEMs during the system manufacturing process. This is essential to avoid costly manufacturing overhead on the integration of SEDs. This ownership model also simplifies deployment of management software by IT. It is important, however, for system integrators and end users to understand the security implications of this ownership model.

In particular, having the MSID Credential value electronically available to the host constitutes a potential risk to the overall security of the system if the take ownership procedure is not executed promptly.

For example:

> An SED is manufactured and shipped to a PC/Notebook OEM. The OEM integrates the SED into the PC/Notebook but does not execute any procedure to take ownership. The OEM includes trial security software for managing the SED, into which the user can opt.
>
> A user purchases this PC/Notebook and does not opt into the security software offering. The user may not be aware of, or may not care about, the security features offered. This user will use the PC/Notebook as if it contained a non-SED.

In this scenario, since neither the OEM nor the end user took steps to take ownership by changing the SID PIN, the value of the SID PIN will still be the same as the MSID Credential value. Since the MSID Credential is readable by anybody, it is easy for an adversary to write malicious code that will read the MSID Credential value, authenticate as the SID Authority, and take ownership of the device. The malicious code can then permanently lock the device; perform a cryptographic erase, etc.

It is also important to note that the ability to execute the commands necessary to take ownership of SID typically require Admin or system level rights in the OS. Given this, malicious code that can permanently lock the device, perform cryptographic erase, etc., will need to be able to execute with the appropriate level of privilege, and will by extension be capable of causing harm to the system in a variety of ways. This will potentially impact policies relevant to meeting organizational security objectives related to SID management.

4 Alternative Take Ownership Mechanisms

4.1 Transferring SID PIN via an Out of Band Mechanism

The approach discussed in this section allows system builders (i.e. OEMs) to protect Opal-based SEDs from being taken over by malware, while allowing software from trusted ISVs to take over the ownership of SEDs during PC deployment.

In this scenario, system builders take ownership of the SED by reading the MSID PIN value, authenticating to the SID authority using the MSID Credential, which is equal to the SID PIN value at manufacturing time, then changing the value of the SID PIN.

The storage device is then integrated into the system and deployed as usual. The MSID Credential no longer matches the SID PIN value and the system upon deployment will be protected from malware that attempts to take over the SED.

The new SID PIN value can be provided to the owner of the system together with the system documentation; as a sticker on the box; together with other materials packaged with the system, etc. This process can be automated and performed during the time system builders gather other machine unique information, such as serial numbers and MAC addresses, which are typically printed on the box of the system and/or provided together with documentation materials. Additional care can be taken to protect the SID PIN value with a tamper evident mechanism to avoid supply-chain attacks.

The owner of the system would make use of the actual SID PIN value received out of band to activate the SED. This can be done upon ISV software installation, which must contain a procedure for taking ownership where the owner provides the SID credential.

Alternatively, the system builder can include a simple BIOS or OS present application that allows the user to authenticate using the SID PIN value obtained out of band and set the SID PIN value back to to the MSID PIN value. Subsequently, the user can take ownership using any piece of software that assumes the SID PIN value to be equal to the MSID PIN value during the process of activation / taking ownership.

The system builder should use best practices when setting the SID PIN value. It should satisfy the usual cryptographic requirements for entropy and should not be derivable from other machine readable information.

One drawback of this mechanism is that it reduces the automatability of the take ownership process by the end customer (e.g. the IT department), since it requires end user intervention to complete the process.

4.2 Transferring PINs via UEFI

The approach discussed in this section allows system builders (i.e. OEMs) to protect Opal-based SEDs from being taken over by malware, while allowing software from trusted ISVs to take over the ownership of SEDs during PC deployment. It also allows ownership transfer from one piece of trusted software to another without exposing SEDs to malware attacks.

This approach makes use of Unified Extensible Firmware Interface (UEFI) available in modern PCs and replacing the 30+ year old Basic Input/Output System (BIOS). The UEFI firmware is responsible for performing basic initialization of the hardware, loading and executing operating system (OS) images, and providing a set of services that the OS may use in booting.

Specifically, the approach described in this section makes use of the Secure Boot feature of UEFI [11], allowing system builders/integrators to restrict UEFI boot environment to digitally signed executables certified by certain Certification Authorities (CAs).

System builders provision SEDs in such a way that:

- SID PIN values are not equal to MSID PIN
- SID PIN values have sufficient entropy to make it impractical for an attacker familiar with system design to pick the actual SID PIN value based on programmatically readable data, such as system serial number, SED serial number, asset tag, etc.

The SID value is then stored in a UEFI environment variable that is made accessible only until the ExitBootServices() UEFI call is made, which signifies completion of system boot and transferring control from a boot loader to an operating system. This is achieved by not setting the EFI_VARIABLE_RUNTIME_ACCESS attribute for that variable during its creation.

Software from a trusted ISV contains a UEFI application with a valid Secure Boot signature. Upon installation or activation, that application is added to UEFI boot order in such a way that it's booted before OS boot loader and the system is rebooted. Upon the next boot, that application performs the following:

- Gets the SID PIN value from the UEFI environment variable
- Using that value, authenticates with the SID authority of the Admin SP, activates the Locking SP and performs its initial configuration (e.g. changes the PIN of the Admin1 authority).
- Optionally, changes SID PIN to a different value, tied to a user token or/and protected from malware using cryptography or another technique making brute-force attack impractical.

If the original SID PIN is retained, SED ownership remains with the system and another piece of trusted software can repeat this process at a later time. If SID PIN is changed, then the trusted software that changed it becomes responsible for managing SED ownership. If there is a need to transfer it to a another piece of trusted software, that's done by transferring it to the system first (by putting the current SID PIN value to a UEFI environment variable) and then having the second piece of trusted software take the ownership in the same manner as described above.

The same method can be used to transfer the PIN of the Admin1 authority of the Locking SP from one ISV to another. A key difference between the SID and Admin1 authorities in Opal SSC is that the latter allows trusted software possessing it to reset most of the SED's state without destroying user data under some conditions, while the SID authority can only be used to perform a destructive reset.

This approach may introduce challenges in OEM manufacturing. For instance, if a failure occurs such that either the SD or the motherboard requires replacement after the SID value has been stored in the UEFI environment variable, then physical intervention, via execution of the Revert command using the PSID credential as authentication, may be required in order to return the SID to its default value. If the SD fails, some mechanism to clear the UEFI environment variable must be executed.

4.2.1 UEFI Environment Variable Names

In order to foster interoperability between a wide range of OEMs and ISVs, the following naming convention and format is recommended for the UEFI environment variables that are used for transferring SID and Admin1 PIN values:

TcgPin<*SED unique ID*>**=**[<*SID PIN value*>][*:*<*Admin1 PIN value*>]

Where:

- <*SED unique ID*> is a string with the hexadecimal representation of the unique identifier of the SED retrieved from the GUDID column of the TPerInfo table of Admin SP. Note that this column is optional in Opal SSC v1.0, so this approach is guaranteed to work only with Opal SSC v2.0 compliant SEDs.
- <*SID PIN value*> is an optional string with the hexadecimal representation of the current SID PIN value for the SED with the <*SED unique ID*> identifier.
- <*Admin1 PIN value*> is an optional string with the hexadecimal representation of the current Admin1 PIN value for the SED with the <*SED unique ID*> identifier.

If present, <*Admin1 PIN value*> is separated from <*SID PIN value*> with a single ':' (colon) character. If the Locking SP is not activated or the Admin1 PIN is not shared, <*Admin1 PIN value*> and the preceding ':' character are omitted. If Admin1 PIN is equal to SID PIN, a single '*' (asterisk) character can be used instead of the actual <*Admin1 PIN value*> to save the memory used for the UEFI environment variables.

If SID PIN is not shared but Admin1 PIN is, <*SID PIN value*> is omitted, but the ':' separator and <*Admin1 PIN value*> are present.

For example:

Consider a system with four SEDs having the following properties:

Property	**Value**
SED #1	
GUDID	02 23 00 08 50 11 22 33 44 55 66 77
SID PIN	*SED1-SIDPIN*
Admin1 PIN	*same as SID PIN*
SED #2	
GUDID	02 23 00 08 51 23 45 67 89 AB CD EF
SID PIN	*SED2-SIDPIN*
Admin1 PIN	*SED2-Admin1PIN*
SED #3	
GUDID	02 23 00 08 50 88 99 AA BB CC DD EE
SID PIN	*not shared*
Admin1 PIN	*SED3-ADMIN1PIN*
SED #4	
GUDID	02 23 00 08 50 89 9A AB BC CD DE EF
SID PIN	*SED4-SIDPIN*
Admin1 PIN	*not shared*

On such a system, the UEFI environment variables containing SID and Admin1 PIN information would be as follows:

TcgPin02230008501122334455667 7= *SED1-SIDPIN*:*
TcgPin0223000851234567 89ABCDEF= *SED2-SIDPIN*: *SED2-Admin1PIN*
TcgPin02230008508899AABBCCDDEE=: *SED3-ADMIN1PIN*
TcgPin0223000850899AABBCCDDEEF= *SED4-SIDPIN:*

Where *SED1-SIDPIN, SED2-SIDPIN, SED2-Admin1PIN, SED3-ADMIN1PIN, SED4-SIDPIN* should have sufficient entropy to make it impractical for an attacker familiar with system design to pick the actual PIN values based on programmatically readable data, such as system serial number, SED serial number, asset tag, etc.

4.2.2 Transferring PINs between UEFI and OS environments

After a new SID/Admin1 PIN value is set, some ISVs may want to continue managing PINs in the portion of their software that runs under an operating system (i.e. outside of the UEFI boot environment). To do that, they need to transfer the new PIN values from UEFI boot environment to OS environment. A number of techniques can be used for that and some of those techniques can be exploited by malware.

ISVs are encouraged to perform threat analysis of their designs to avoid or mitigate the following type of attack:

- Extracting their signed UEFI boot applications, bundling them with malware to force boot applications to set a new SID PIN and reveal it to malware by spoofing the portion of their software running in the OS.

Similarly, caution should be used if a custom mechanism is used to transfer a new PIN value from OS environment to UEFI boot environment. Here are a few examples of possible threats posed by such mechanisms:

- Malware causes a signed UEFI boot application to overwrite TcgPin variables created by legitimate key management software with random strings, thus causing a loss of the PINs stored in UEFI.
- Malware causes a signed UEFI boot application to create a large number of TcgPin variables, thus exhausting UEFI resources and obstructing the work of legitimate key management software.

4.2.3 Interactions with Revert and RevertSP Methods

The Revert and RevertSP methods return a given SP to the Original Manufacturing State. Consequently, if either method is executed on the Admin SP, the method resets the SID PIN value. In order to keep TcgPin variables in sync with SED state, the software invoking Revert or RevertSP for the Admin SP should modify the TcgPin variable for the corresponding SED accordingly: either by storing the new SID PIN value in it; or by deleting the variable if the new SID PIN is not known or not to be shared.

4.2.4 Keeping UEFI Environment Clean

UEFI resources tend to be quite limited, so the number of environment variables defined at any given time should be minimized whenever it is possible. Key management software dealing with SID/Admin1 PINs should delete outdated TcgPin variables when those PINs change and new values are not to be shared.

4.2.5 Moving SED to Different Host

If a SED is moved from one host to another, the value of the corresponding TcgPin variable is not automatically transferred to the new host. This may later lead to inability of key management software

to take ownership of that SED on the new host. To prevent that, the SID/Admin1 PINs of the SED should be reset to MSID prior to moving the SED from one host to another, unless they are migrated in another way (e.g. by management software installed on the host).

4.2.6 Role of UEFI Firmware Vendors

UEFI firmware vendors can facilitate SID management by performing initial SED provisioning for or on behalf of OEMs. Specifically, during storage device enumeration, the UEFI firmware can check if a storage device is a TCG Opal compliant SED and, if yes, whether its SID PIN is equal to MSID. If it is, then the UEFI firmware can generate/assign a random SID PIN and store it in a TcgPin variable. This would simplify system provisioning process at OEM manufacturing floor and prevent malware attacks on drives that are added to the system later, even in the absence of SED-aware key management software.

UEFI firmware can also provide a user-friendly interface for basic SED management, e.g. resetting SID/Admin1 PINs to MSID for the purposes of transferring an SED from one host to another.

4.3 Protecting PINs in the TPM

Platform firmware can also use the TPM for storage of a customized SID PIN. This is similar in some respects to mechanisms described in 4.2, but uses the TPM instead of UEFI variables.

At power-on, platform firmware can perform a detection routine to determine whether or not the SID PIN is still set to the MSID credential value. If platform firmware detects that the SID PIN is still set to the MSID Credential, then platform firmware can request a 32 byte random value from the TPM, or in the open session to the Admin SP (if the Random method is supported).

Once platform firmware has a random value, the value can be set as the SID PIN. The random value will also be set in TPM NV index, which is locked to the platform authorization.

When the user is ready to take ownership, the user can select the appropriate platform firmware option. This enables the user to have platform firmware either reset the SID PIN to the MSID credential value, or change the SID PIN to a value of the user's choice.

ISV software can use the TPM PPI to request the BIOS to clear SID authentication.

5 Block SID Authentication

The Block SID Authentication Feature Set (see [10]) defines a command that can be used to prevent authentication of the SID authority if the SID PIN is set to the MSID Credential. This is useful to control authentication of SID in systems where ownership has not been taken, and where UEFI-based or other mechanisms have not been implemented (see 4), but requires special handling so that the Take Ownership process can still be accomplished.

This functionality is similar to that provided by the ATA Security Feature Set's FreezeLock command (see [13]). Because of certain properties of the ATA Security Feature Set, the FreezeLock command is necessary to control access to execution of ATA Security Feature Set commands that modify the security state of the device (e.g., set passwords, disable security).

From an implementation, integration, and security standpoint, Block SID is analogous to the FreezeLock command, in that if Block SID Authentication is enabled (i.e., platform firmware executes the command) then the command would need to be executed in situations similar to those where FreezeLock is executed – for example, when the storage device is powered on during a system resume from S3.

It is important to note that the Block SID command only controls the ability to authenticate the SID authority. Unlike the FreezeLock command, Block SID is not specified to prevent any other security operations. As such, once the SD's Locking SP has been activated, it's possible for an OS-resident application to perform management activities without the Block SID command introducing any impediment.

The Block SID command should be executed as late in the platform firmware process (e.g., as close to ExitBootServices) as possible.

5.1 Bypassing Block SID

It is expected that platform firmware will execute the Block SID Authentication command immediately prior to, or as close as possible to, ExitBootServices, in order to prevent OS-present malware from maliciously performing Take Ownership of SID in scenarios where SID is still set to MSID.

However, use of the Block SID Authentication command by platform firmware to prevent malicious usage of SID also has the effect of blocking legitimate software from being able to Take Ownership of SID, and subsequently Activate the Locking SP.

Since this is the case, it is a requirement that mechanisms are supported to bypass execution of the Block SID Authentication command.

5.1.1 Command Execution Default

As mentioned in 3.4, the ability to perform Take Ownership typically requires Administrator rights on the system. Since this is the case, an IBV could reasonably elect to not execute the Block SID Authentication command by default, instead providing an opt-in model.

In such a case, where the Block SID Authentication command execution is disabled by default, a platform firmware option would be provided to allow the user to enable execution of the command.

Leaving the Block SID Authentication command as an opt-in feature, particularly in initial deployments, allows for development of the necessary ecosystem components to enable a smooth ecosystem transition. ISV management software and platform firmware would be more likely to be prepared for scenarios that require bypass of Block SID in order to allow Take Ownership to occur.

5.1.2 User Present Bypass

The primary means by which the Block SID Authentication command execution can be bypassed is via an option provided by platform firmware during boot. At a minimum, this should allow the user to bypass execution of the command until the next system boot. An option could also be provided to allow the user to bypass execution of the command permanently (or until the user selects otherwise later).

5.1.2.1Revert

The Revert method is defined by [10] as an event that is able to clear the effects of the Block SID Authentication command.

The most common means of authenticating to invoke Revert is via the SID authority, but if the Block SID Authentication command has been successfully executed, it is obviously not possible to authenticate the SID authority in order to successfully invoke Revert.

Successful authentication of the PSID authority (see [9]) will permit invocation of the Revert method. Successful invocation of the Revert method on the Admin SP when the Locking SP is in the Manufactured-Inactive state has limited side effects. In particular, Revert will set SID back to MSID, and will clear the effects of the Block SID Authentication command.

Since this is the case, this provides another means of clearing the effects of the Block SID Authentication command and install ISV software to activate and configure the Locking SP, if the user has the PSID credential and a tool to authenticate with that credential and invoke the Revert method.

Note that if for some reason SID is still set to MSID even though the Locking SP has been activated, then successful execution of the Revert method will cryptographically erase all user data and return the SD to its original factory state.

5.1.3 No Touch Bypass

Certain ISV software installation scenarios rely on the ability for software installation, and Take Ownership and provisioning activities, to occur without user intervention.

An option to meet this use case is for the Block SID command execution to be disabled by default (see 5.1.1).

In order to facilitate this use case in the presence of platform firmware that executes the Block SID command, it is necessary for Opal management software to be able to securely notify the platform firmware to bypass command execution.

5.1.3.1TPM Physical Presence Interface

The PC Client Platform Physical Presence Interface specification (see [11]) defines mechanisms for asserting physical presence as authorization to perform certain TPM functions. [11] also includes enhancements that allow management of the Block SID feature via the specified ACPI communication mechanism.

Note that a TPM is not necessarily required to support the capabilities related to management of the Block SID Feature. Support for the storage management flags and components can be achieved by implementing an appropriate subset of the functionality defined in [11].

The following list summarizes the available operations and their meanings (Operation Value in parentheses):

1. Enable_BlockSIDFunc (96) – this operation is used to enable execution of the Block SID Authentication command by the platform firmware.
    a. Note: Physical presence requirement for execution of this command is controlled by the value of PPRequiredForEnable_BlockSIDFunc flag.
2. Disable_BlockSIDFunc (97) – this operation is used to disable execution of the Block SID Authentication command by the platform firmware.
    a. Note: Physical presence requirement for execution of this command is controlled by the value of PPRequiredForDisable_BlockSIDFunc flag.
3. SetPPRequiredForEnable_BlockSIDFunc_True (98) – this operation is used to set the PPRequiredForEnable_BlockSIDFunc flag to TRUE, resulting in physical presence confirmation being required to enable execution of the Block SID Authentication command via the Enable_BlockSIDFunc operation.
    a. Note: Per [11], successful execution of this operation requires physical presence.
4. SetPPRequiredForEnable_BlockSIDFunc_False (99) – this operation is used to set the PPRequiredForEnable_BlockSIDFunc flag to FALSE, resulting in physical presence confirmation not being required to enable execution of the Block SID Authentication command via the Enable_BlockSIDFunc operation.
5. SetPPRequiredForDisable_BlockSIDFunc_True (100) – this operation is used to set the PPRequiredForDisable_BlockSIDFunc flag to TRUE, resulting in physical presence confirmation being required to disable execution of the Block SID Authentication command via the Disable_BlockSIDFunc operation.
    a. Note: Per [11], successful execution of this operation requires physical presence.
6. SetPPRequiredForDisable_BlockSIDFunc_False (101) – this operation is used to set the PPRequiredForDisable_BlockSIDFunc flag to FALSE, resulting in physical presence confirmation not being required to disable execution of the Block SID Authentication command via the Disable_BlockSIDFunc operation.

Default physical presence requirements, represented by the PPRequiredForEnable_BlockSIDFunc flag and the PPRequiredForDisable_BlockSIDFunc flag, are not specified in [11].

1. To ensure the ability to meet remote deployment use cases, it is recommended that at a minimum the default value of the PPRequiredForDisable_BlockSIDFunc flag be FALSE.
2. To meet more restrictive security policies on control over the take ownership process, it is recommended that at a minimum the default value of the PPRequiredForDisable_BlockSIDFunc flag be TRUE, and that the Block SID Authentication command be executed by platform firmware by default.

5.1.3.2UEFI Capsule Update

This mechanism leverages a special driver (herein called the "*BlockSidDriver*"), which is signed by the UEFI CA or the OEM. This assumes that the system would ordinarily have UEFI Secure Boot enabled and any UEFI secure bootable entity is "Trusted".

A new "Boot-service only" variable called "*BlockSidVariable*" is defined, and has a Default value of "Off". "Boot-service only" and "Secure Boot Enabled" combine to indicate that only trusted pre-OS entities are able to manipulate the variable value. This effectively provides an ACL to pre-OS entities only, and avoids some of the complexity of developing a solution based on, for instance, an Authenticated Variable.

On a reboot, the SD interface bus driver and other UEFI firmware would check *BlockSidVariable*. If the value of *BlockSidVariable* is "True", then UEFI firmware would not issue the Block SID Authentication command.

When the IT administrator or ISV wants to reclaim the machine, the non-access controlled, UEFI-specification defined variable *DriverLoadList* ("*DriverOrder*", in the UEFI specification) can be written with the device path pointer to the *BlockSid.efi* driver (the *BlockSidDriver)* on disk, or passed across reset via a Capsule using the PE/COFF encapsulation type of capsule defined in UEFI 2.4.

If the *BlockSidDriver* is correctly signed, then this driver would run and set the *BlockSidVariable* appropriately. This code would run prior to the storage bus drivers in case of capsule.

The *BlockSidDriver* would register for an exit boot services event so that it can clear the *BlockSidVariable* upon end of UEFI execution.

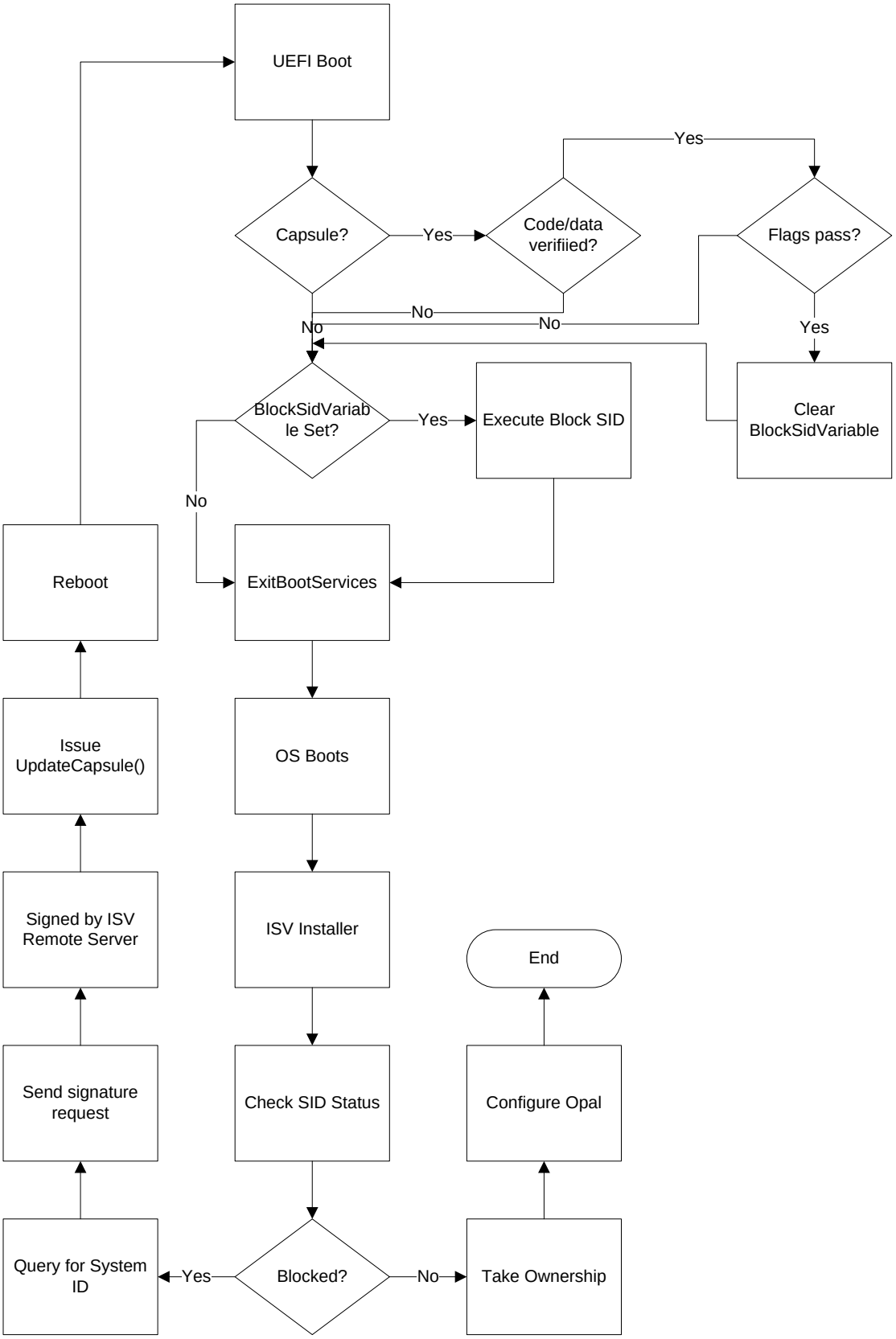
The IT administrator can remove the *BlockSidDriver* from the *DriverLoadList* on the next boot. The capsule flow is one-shot, so for that delivery mechanism there is no clean-up.

The ISV software that is performing the capsule update to bypass execution of the Block SID Authentication command should implement a mechanism to prevent replay usage of the signed update. This could include the installer querying the system for some unique value and reporting this value via trusted channel to the backend server that signs the *BlockSidDriver*, to be included as part of the *BlockSidDriver* update. Note that the *BlockSidDriver* signing may require a key signed by the UEFI CA or by the OEM.

The signed *BlockSidDriver* is passed to the UEFI firmware via the UpdateCapsule() command. After a reset, the ISV's UEFI driver can verify the signature of the *BlockSidDriver*, along with any other information included (such as the system identifier such as type 1 SMBIOS table UUID, and a nonce). If the *BlockSidDriver* signature is successfully verified, it will run and clear the *BlockSidVariable*, or, alternatively, set the SID credential (possibly with a value included in the Capsule), which would render the Block SID Authentication command ineffectual.

At this point, after UEFI firmware completes execution and the OS boots, the ISV's OS-present installer can complete the Take Ownership process.

Figure 2 – Capsule Update Flow



5.1.3.3Custom Notification Solutions

Custom solutions from OEMs, IBVs, and ISVs could be used to provide a secure notification scheme to request bypass of the Block SID command.

A custom solution may, for instance, consist of a mailboxing mechanism to allow an OS-present or network entity to “leave a message” for the platform firmware, which could be accessible after reboot. Such a method is similar to the capabilities provided by the TPM PPI or the Capsule Update.

Security of this mechanism could depend on either controlling access to the ability to leave the message; or on verification steps on the message itself.

6 PSID

The PSID (Physical SID) authority (see [9]) provides a means of performing recovery of the SED when the access control credentials are lost.

6.1 Implications

The “classic” PSID delivery mechanism, where the PSID PIN value is simply printed on the storage device label or on the box for the PC/Notebook, or otherwise supplied “in the clear”, adds the possibility of a supply-chain attack. A determined attacker can intercept the PSID PIN value before it arrives at the end user and can subsequently perform a denial of service software attack that uses the PSID to erase the data on the SED.

To avoid the possibility of supply-chain attacks, the PSID PIN value should be delivered to the end user in such a manner that prevents intermediaries from reading it.

6.1.1 Out of Band Delivery

The system OEM can record the PSID PIN value and store it in a database and only provide it through a secure out of band channel. For example, the user sets up a web account with the system OEM and if they need to obtain the PSID PIN value they can log into their account and retrieve the PSID PIN value(s) by providing the system serial number.

However, collecting PSID values in a database introduces another potential avenue of exposure.

6.1.2 Tamper Evident Packaging

The PSID PIN value can also be delivered with the system itself. In this case, it could be packaged in opaque tamper evident packaging. With this delivery mechanism the PSID PIN value is readily available for anyone to read, but the tamper evident packaging allows the end user to be sure that the value has not been read by anyone in the supply-chain.

6.2 PSID and Pyrite

Party of the security associated with using PSID to perform recovery on a SED where the Locking SP has been Activated is that the Revert method is data destructive. When the Revert method is executed and the Locking SP is in the Manufactured (i.e., “Activated”) state, the SD performs a cryptographic erase on all Media Encryption Keys as part of the Revert process.

Pyrite (see [7]) does not specify the use of encryption for data at rest protection. As such, execution of the Revert method on an “Activated” Locking SP is not specified to cryptographically erase all user data. As such, the Revert method as implemented in [7] is not data destructive. Pyrite SSC does not specify cryptographic protection of data at rest; there are no Media Encryption Keys to destroy; and thus there is no specified capability to perform cryptographic erase.

Since there is no data destruction associated with the Revert method in SDs based on [7], it is highly recommended that SDs based on the Pyrite SSC NOT support the PSID authority and credential for use with the Revert method. Doing so would provide an authentication credential that permits easy bypass of the access control on the storage interface provided by [7].