TCG Physical Presence Interface Specification

Specification Version 1.2
Revision 1.00
February 10th, 2011

Contact: admin@trustedcomputinggroup.org

TCG Published

TCG

Disclaimers, Notices, and License Terms

Table of Contents

Change History

Version / Revision	Date	Description
1.00 / 1.00	5 April, 2007	• Initial release of Version 1.00.
1.10 / 1.00	10 June, 2009	• Added new action 12: Deferred Physical Presence • Added "Submit TPM Operation Request to Pre-OS Environment 2"
1.20 / 1.00	10 February 2011	• Added BIOS TPM Management Flags and corresponding operations • Added return code 3 to "Submit TPM Operation Request to Pre-OS Environment 2" • Added "Get User Confirmation Status for Operation" • Added Operations 21 and 22 that are analogous to 5 and 14 but perform enable and activate first.

List of Tables

Corrections and Comments

Please send comments and corrections to techquestions@trustedcomputinggroup.org.

TPM Dependency and Requirements

A PC Client platform in adherence to either:

1. The TCG PC Client Specific Implementation Specification for Conventional BIOS Version 1.20

   or

2. The TCG EFI Platform Specification Version 1.20 **and** the TCG EFI Protocol Specification Version 1.20

1. TPM Management Overview

Start of informative comment:

Physical Presence

Physical Presence is a form of authorization to perform certain TPM functions. This authorization must come from the platform operator. The functions requiring physical presence are those which perform provisioning and un-provisioning operations such as allowing ownership and clearing ownership when the current owner authorization value is unknown.

There are two methods defined in the TPM Main Specifications for providing an indication of physical presence. The first and most obvious method is the hardware method. An example of an implementation of this method is a button on the front of the platform wired to a pin on the TPM. Pressing the button causes the pin to change to the polarity determined by the TPM to set the TPM's internal physical presence flag. Using this hardware method, commands requiring the indication of physical presence could be executed at any time (in the pre-OS environment or during from the OS environment). Implementation of the hardware method is outside the scope of this specification.

Providing a wire along with a button or switch to the outside of the platform is not feasible in some cases due to cost, form factor, usage or any number of reasons. For this reason, the TPM provides the second method of asserting physical presence called the "command method". One of the properties required of the indication of physical presence is it must be done by the operator, typically the user, when physically at the platform. This requires the command method be restricted to be available only while the platform state can provide this assurance. On a PC Client, this state is during the boot strapping of the platform prior to the availability of a network stack or untrusted software – specifically during the early Static Root of Trust for Measurement (SRTM). These commands are therefore available only after TPM_Init and must be "closed" once the SRTM determines the user has not provided an indication of physical presence. This specification is targeted at platforms implementing this "command method" of providing an indication of physical presence.

Enhancements added in version 1.2 of this specification permit the operator to set persistent BIOS TPM Management flags that allow certain actions previously requiring an operator to be physical present to occur without an operator being physical present. The intent is to allow a Platform Manufacturer or an IT department to configure a platform once, so the operating system is able to perform TPM management actions without operator involvement in the future.

TPM Management Authorization

The TPM may be used in a variety of scenarios on a PC Client Platform to provide trusted computing. Depending on the scenario, different authorization schemes are appropriate to manage TPM actions like enabling, disabling, clearing the TPM or updating the TPM firmware.

This specification defines the two authorization mechanisms below to control the ability to perform TPM actions:

- Physical Presence Control – A platform operator physically present at the platform must press a key to authorize an action. A benefit is software cannot manage the TPM without operator involvement and unevaluated software may be permitted to run without risk that the TPM state will change without operator approval. A drawback is an operator must be physically present at the computer to manage the TPM, inhibiting IT administration of the TPM state where the IT administrator is not in the same location

as the system. This mechanism may be appropriate for a high security scenario where the platform operator has enough knowledge to participate in TPM management.

- Software Control – An OS or other software program controlling the system manages the TPM without the need for a physically present operator to approve changes. A benefit is an operator does not need to be physically present to manage the TPM. A drawback is software may change the TPM state without operator involvement, so unevaluated software should be prevented from controlling the system. This mechanism may be appropriate for a managed desktop scenario where OS enforced access controls strictly regulate access to TPM management APIs and TPM management is performed remotely by an IT department.

Different TPM management actions have different consequences. For example, disabling the TPM can be reversed by enabling the TPM. However, clearing the TPM owner cannot be reversed. For this reason, this specification allows different aspects of TPM management to be configured to independently use one of the authorization mechanisms described above. The different aspects defined in this specification are Provisioning (and un-provisioning), Clearing and Maintenance of the TPM.

Other TPM Management Authorization

Additional Platform Manufacturer specific authorization mechanisms not defined in this specification may be provided on platforms. An example is BIOS configuration utilities or remote system management tools which enable or disable the TPM with authorization controlled by a BIOS password. These implementations are vendor specific.

Terminology

This specification uses the term “BIOS” to generically refer to conventional BIOS firmware (e.g. POST) or UEFI firmware.

End of informative comment.

2. Physical Presence Interface

Start of informative comment:

The Physical Presence Interface utilizes the industry-standard Advanced Configuration and Power Interface (ACPI) to provide a communication mechanism between the OS and the BIOS, enabling the OS and the BIOS to cooperate to provide a simple and straightforward platform user experience for administering the TPM without sacrificing security.

This Interface was designed under the assumption that TPM commands requiring physical presence should only be executable in the pre-OS environment. Given this constraint, the Physical Presence Interface eases administration by minimizing the need to understand and configure a platform's BIOS -- a significant deterrent to the mass deployment of trusted platforms, especially across heterogeneous platforms.

The implementation of the Physical Presence Interface is highly recommended for platforms that support the software method of asserting physical presence (as indicated by the physicalPresenceCMDEnable permanent flag of the TPM).

In order to minimize the platform storage required to implement this Interface, the OS is restricted to requesting the execution of at most one operation at a time. An operation is defined as one or more TPM commands that require physical presence authorization. By enumerating the most likely sequences of TPM commands and mapping them to unique operations, all standard functionality involving physical presence can be carried out within one restart of the OS.

For definitions of the TPM commands that are associated with each operation identified in the Interface, see Part 3 of the v1.2 TCG Main specification.

Some operations permitted through this Physical Presence Interface do not change the state of the TPM, but instead change BIOS TPM Management flags which control whether a physically present user must confirm future physical presence operations.

For more information on ACPI, see the materials available for download at http://acpi.info/. Refer to Section 9.15.1 of the ACPI 3.0 spec for information on the _DSM control method object. Refer to section 17.2.5 of the ACPI 3.0 spec for information about data types. For example, the "Integer" data type is a 64-bit little-endian unsigned integer and the "String" type is a null-terminated ASCII string with up to 200 characters.

Defined Uses:

The primary use case of the Physical Presence Interface is as follows:

1. Within the OS environment, the user requests an operation that requires physical presence.
2. The OS informs the user that a platform-specific procedure (e.g. reboot or shutdown then user confirmation or not) must take place to successfully execute the operation.
3. The OS communicates the requested operation to the BIOS through ACPI. The platform's ACPI handler stores the requested operation in a location accessible to the pre-OS environment (e.g. CMOS, Flash ROM, TPM NVRAM, etc.).
4. The OS reboots or shuts down the platform; this is a user-visible event and transitions the platform to the CRTM-initiated pre-OS environment.
5. The BIOS reads the OS's operation request from its stored location.

6. Depending on the operation and the state of persistent BIOS TPM Management flags, the BIOS automatically approves the operation or asks a physically present user for confirmation to perform the operation.

Note that the BIOS may measure and execute a Platform Manufacturer (PM) pre-boot environment utility to prompt the user for confirmation.

If the operation is approved, depending on the type of operation, the BIOS executes TPM commands that carry out the requested operation or changes BIOS TPM Management flags which control the approval mechanism of future physical presence actions.

The BIOS clears the storage location containing the operation request.

The BIOS remembers the response of the last requested operation. When the OS is running again, it can query the response of the last requested operation by using an ACPI method. The response can be a success code if the operation was confirmed and executed, a failure code if the user failed to issue a confirmation, or a TPM command error.

The OS loads, determines the response of the requested operation, and takes action as necessary.

Table 2 identifies the set of operations available to the OS.

Security Implications:

When BIOS TPM management settings require a physically present user to approve an operation, it is imperative that the pre-OS environment verifies the physical presence of the user and confirms that the physically present user in fact approves the execution of an operation. This confirmation is achieved via a pre-OS dialog. The dialog should be implemented such that the user can understand at a high level the security implications of the operation and must actively choose to execute the operation (e.g. the default should not be to confirm the operation). Table 4 has recommended user confirmation text for each operation. If the request is rejected by the physically-present user, the pre-OS environment must clear the request so that the user is not prompted to confirm again on the next reboot.

There also exists the possibility that malicious software can attempt to launch a denial of service attack on the platform by repeatedly invoking the ACPI call to submit operation requests. It is the responsibility of the PM to ensure that such abuse will not cause irreparable damage to the platform (e.g. by burning out flash memory used to store the requests).

The location(s) used by the BIOS to store the pending operation (including any necessary platform reset) is not required to be trusted or secure. This is because if the operation is not performed, or performed incorrectly, the TPM's operations that are intended to be affected will not perform as the user expects. This does not introduce a vulnerability to the user.

For usability reasons it is desirable to not prompt the user to provide more than one assertion of physical presence for a single operation. However, as stated in the PC Client specification, the method for checking the indication of physical presence does need to be a trusted operation. Therefore, if between Platform Resets, the BIOS stores the assertion of the indication of physical presence, that location must be trusted.

Storage Implications for passing information between the OS and the BIOS:

The minimum platform storage required to implement the information exchange between the OS and the BIOS (and vice-versa) for this Interface is as follows:

5 bits – Stores the pending operation request submitted by the OS

5 bits – Stores the most recent operation request acted upon by the BIOS

7 bits – Stores the most recent operation response acted upon by the BIOS

The data above assumes that no vendor-specific operations or error codes are implemented and there exist less than 128 possible operation responses. As of the time of this writing, there exist 105 possible operation responses consisting of 99 TPM fatal errors, 4 TPM non-fatal errors, and 2 additional Physical Presence Interface-specific responses (User Abort and BIOS Failure).

Conditional Physically Present User Confirmation:

Several persistent BIOS TPM Management flags control whether a physically present user must confirm each physical presence operation.

The flags may be set independently and their values may be changed through the use of physical presence operations or other vendor specific configuration utilities. Table 1 is a list of the flags.

Flag Description	Operations permitted without user confirmation when set	Recommended Default
NoPPIProvision	Enable Disable Activate Deactivate SetOwnerInstall_True SetOwnerInstall_False SetOperatorAuth	True
NoPPIClear	TPM_ForceClear	False
NoPPIMaintenance	Deferred Physical Presence-unownedFieldUpgrade	False

Table 1: Persistent BIOS TPM Management Flags

Persistent BIOS TPM Management Flag Storage

The BIOS TPM Management flags control whether physical presence is required for TPM management actions. By design physical presence is required to change the flags to not ask for physical presence confirmation in the future. Care should be taken by Platform Manufacturers to prevent unintended modification of the flags circumventing the intended physical presence required to modify the flags. For example, the flags should not be stored in NV RAM locations able to be manipulated by malicious software.

One mechanism to protect the BIOS TPM Management flags is to store the flags in the TPM's NV RAM. To prevent entities other than the BIOS from modifying the flags, write permission for the index may be configured to require physical presence for which the BIOS controls assertion. The "D" bit attribute should be set so the values persist even if the TPM is cleared. This specification has reserved index 0x50010000 for Platform Manufacturers if the TPM NV RAM used. The Platform Manufacturer should carefully provision appropriate default values for the BIOS TPM Management flags.

Method for performing field upgrade without using physical presence:

A field upgrade of an unowned TPM could be accomplished without physical presence by taking ownership and performing the field upgrade. Therefore the default for the NoPPIMaintenance flag is False.

Transitioning to the pre-OS environment from the OS – Restart versus Shutdown

Platform manufacturers may require the OS to restart, shutdown or perform some other vendor specification action for the BIOS to fulfill a requested operation during the next boot of the platform. However, to permit operations without any physical presence when the BIOS TPM Managements are set to TRUE, using restart instead of shutdown is necessary to permit remote management of a platform without a user being physically present to turn the platform back on.

Additional Notes:

This Interface assumes that the BIOS is able to execute the TSC_PhysicalPresence command to configure physical presence assertion flags as well as the TPM commands that require physical presence authorization (e.g. TPM_PhysicalEnable).

For some TPM implementations, execution of these commands prior to issuing the TPM_ContinueSelfTest command causes an error of TPM_NEEDS_SELFTEST or TPM_DOING_SELFTEST to be returned. In these cases, the BIOS should ensure that TPM_ContinueSelfTest is issued appropriately so that the self-test errors are not propagated to the OS.

After the user confirms the physical presence operation, the BIOS may need one or more additional reboot cycles to carry out the user's request, as well as 1 bit of additional platform storage to track the extra boot cycle. The additional reboot is needed when executing the command TPM_PhysicalSetDeactivated because the TPM requires an additional boot cycle for an activation or deactivation to take effect (see main spec related to TPM_PERMANENT_FLAGS.deactivated and TPM_STCLEAR_FLAGS.deactivated for more information). Additional reboots may need to take place in the middle of some multi-command physical presence operations.

Affected operations are:

Operation 3:	Activate
Operation 4:	Deactivate
Operation 5:	Clear
Operation 6:	Enable + Activate
Operation 7:	Deactivate + Disable
Operation 10:	Enable + Activate + SetOwnerInstall_True
Operation 11:	SetOwnerInstall_False + Deactivate + Disable
Operation 12:	Deferred Physical Presence unownedFieldUpgrade
Operation 14:	Clear + Enable + Activate
Operation 21:	Enable + Activate + Clear
Operation 22:	Enable + Activate + Clear + Enable + Activate

When the BIOS carries out operations that require an additional reboot cycle, the user should not be prompted again for confirmation. When this occurs, the only indication to the user of the additional reboot cycle is the reappearance of the BIOS splash screen. For the OS user, there is still only one restart.

Platform Implementation

This Interface is designed for platforms implementing the command method for indication of physical presence. Platforms are allowed to implement both methods in which case the user may choose the one most convenient. If a Platform Manufacturer provides a utility, the utility prompting the user to provide the indication of physical presence should be aware of the method(s) implemented on the platform and prompt the user accordingly. This information can be found by querying the TPM's capabilities (e.g. calling TPM_GetCapability).

Though this specification describes the hardware method of asserting physical presence, it does not specify how an OS would interact with the hardware method. For example, when using the hardware method, how TPM management actions like changing from deactivated to activated (which requires a platform reset to occur) with an OS present are not specified. In the absence of additional specifications, implementations using a hardware method may want to emulate the "command method" of asserting physical presence to accommodate the same OS interface for TPM management.

ACPI Command Status:

The command status field within each command indicates whether the command is mandatory, optional or deprecated.

Mandatory: If a platform implements this version of this specification it must implement the indicated command.

Optional: If a platform implements this version of this specification it may implement the indicated command.

Deprecated: This command may be removed from future versions of this specification. Software should not use it.

End of informative comment.

All PC Client Platforms MUST implement this interface when the ACPI namespace contains an ACPI device object for the TPM.

2.1 ACPI Functions

1. If the PM implements an interface between the OS and the pre-OS environment for supporting the execution of TPM operations requiring physical presence, the BIOS MUST support the Physical Presence Interface as defined below.
2. A Platform Manufacturer that implements the Physical Presence Interface SHOULD continue to expose TPM management within the BIOS setup, to allow users to configure the TPM independent of an available OS.
3. These ACPI functions reside in the _DSM control method object. The following UUID function identifier MUST be used exclusively for the Physical Presence Interface:

   `3DDDFAA6-361B-4eb4-A424-8D10089D1653.`

4. Functions start at index 1, since function 0 is the standard _DSM query function.

Below is the list of ACPI Physical Presence functions which MUST be implemented:

2.1.1 Get Physical Presence Interface Version

Command Status:

Mandatory

Input Arguments:

Arg0 (Buffer): UUID = 3DDDFAA6-361B-4eb4-A424-8D10089D1653

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 1

Arg3 (Package): Arguments = Empty Package

Returns:

Type: String

Purpose: Supported Physical Presence Interface revision

Functional Behavior:

This function returns the version of the Physical Presence Interface supported by the platform.

For this version of this specification, the return value MUST be “1.2” as a NULL terminated ASCII string

Note that the Physical Presence Interface revision number does not correspond with the Revision ID parameter (Arg1).

Examples:

A return value of “1.0” indicates that the Interface is compatible with Physical Presence Interface specification v1.0.

A return value of “1.1” indicates that the Interface is compatible with Physical Presence Interface specification v1.1.

A return value of “1.2” indicates that the Interface is compatible with Physical Presence Interface specification v1.2.

2.1.2 Submit TPM Operation Request to Pre-OS Environment

Command Status:

Deprecated and Mandatory

Input Arguments:

Arg0 (Buffer): UUID = `3DDDFAA6-361B-4eb4-A424-8D10089D1653`

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 2

Arg3 (Package): Arguments = Package --

Type: Integer

Purpose: Operation Value of the Request

Description: [see Table 2]

Returns:

Type: Integer

Purpose: Function Return Code

Description:

0: Success

1: Operation Value of the Request Not Supported

2: General Failure

Functional Behavior:

This function allows the OS to submit a request for an operation to be executed in the pre-OS environment (see Table 2 for a list of defined operations). This request is the only input from the OS to the pre-OS environment.

If 0 is returned, the requested operation can be read and acted upon by the BIOS once the transition to the pre-OS environment takes place (e.g. after the platform has restarted). The OS expects that the pre-OS environment verifies physical presence and confirms that the physically-present user in fact requested the execution of the operation.

If 1 is returned, the BIOS does not support the operation request. For example, the implementation of the operation may be optional or vendor-specific.

If 2 is returned, the BIOS is otherwise unable to read and act upon the request. For example, platform-specific security protections may exist to prevent burnout of the storage location for the operation.

The OS may call this function or “Submit TPM Operation Request to Pre-OS Environment 2” multiple times before transitioning to the pre-OS environment. However, only the last submitted request is valid. The OS may submit an operation request of value 0 to clear any previous requests.

Start of informative comment:

Many as-built operating systems and hardware implementations have implemented the Submit TPM Operation Request to Pre-OS Environment function differently than this specification for the Arg3 argument. Due to the large number of deployed systems in the marketplace it is recommended Platform Manufacturers and operating system vendors consider compatibility with the following scenarios:

Some operating systems incorrectly pass a buffer of an Integer instead of a Package containing an Integer for Arg3. The ACPI Machine Language Interpreter on those operating systems treats the Arg3 argument type as a buffer when executing ACPI Machine Language instructions.

Some existing hardware platforms in the marketplace are dependent on this operating system mistake and explicitly convert the buffer of an Integer to an Integer using the ACPI Machine Language operation ToInteger. Using ToInteger would fail if the operating system actually passed a Package containing an Integer.

Other existing hardware platforms use the ACPI Machine Language Index operation. The original intent of these hardware platforms using Index was probably to extract the Integer which should be the first element of the Package. The result on operating systems which pass a buffer of an Integer is to extract the first byte of the Integer. Because ACPI Integers are defined as little-endian unsigned values and the number of defined operations is less than 256, this implementation works as expected.

End of informative comment.

Example:

A submitted operation value of 6 and a return value of 0 indicates that the platform has successfully received a request to enable and activate the TPM.

2.1.3 Get Pending TPM Operation Requested By the OS

Command Status:

Mandatory

Input Arguments:

Arg0 (Buffer): UUID = `3DDDFAA6-361B-4eb4-A424-8D10089D1653`

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 3

Arg3 (Package): Arguments = Empty Package

Returns:

Type: Package

Integer 1:

Purpose: Function Return code

Description:

0: Success

1: General Failure

Integer 2:

Purpose: Pending operation requested by the OS

Description:

0: None

>0: Operation Value of the Pending Request [see Table 2]

Functional Behavior:

This function returns the pending operation that was previously requested since the last host platform boot, if any. This function is necessary to allow the OS to accurately determine platform state. One use case is to ensure that a previously submitted operation request is not overwritten.

If 1 is returned as the first integer, the BIOS is unable to return meaningful information due to an internal failure. In this case, the second integer is undefined.

Assume in the following paragraphs that 0 is returned as the first integer.

If 0 is returned as the second integer, no pending operation exists. No previous request has been submitted.

If a value greater than 0 is returned as the second integer, a previously requested operation is pending (see Table 2 below for a list of defined operations). This request will be read and acted upon by the BIOS once the transition to the pre-OS environment takes place.

Examples:

A return value of {0, 0} indicates that no operation is pending.

A return value of {1, 0} indicates that the function failed to retrieve the pending operation. The meaning of the second Integer is undefined.

2.1.4 Get Platform-Specific Action to Transition to Pre-OS Environment

Command Status:

Mandatory

Input Arguments:

Arg0 (Buffer): UUID = `3DDDFAA6-361B-4eb4-A424-8D10089D1653`

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 4

Arg3 (Package): Arguments = Empty Package

Returns:

Type: Integer

Purpose: Action that the OS should take to transition to the pre-OS environment for execution of a requested operation.

Description:

0: None

1: Shutdown

2: Reboot

3: OS Vendor-specific

Functional Behavior:

This function allows the OS to determine the platform-specific action that should take place in order to transition to the BIOS for execution of a requested operation. This function provides Platform Manufacturers the flexibility to vary how their platforms meet TCG's physical presence requirements, while minimizing the impact of these platform changes on OS applications.

If 0 is returned, no action is required. This return value MUST NOT be used for platforms implementing the command method of asserting physical presence.

If 1 is returned, the OS must shut down the machine to execute a pending operation requested by the OS. A physically-present user restarts the machine.

If 2 is returned, the OS must cause a warm reboot of the machine to execute a pending operation requested by the OS. This SHOULD be used for platforms implementing the command method of physical presence.

If 3 is returned, an OS-specific action can take place. For example, instructions can be displayed for the physically-present user to consult platform documentation. The OS may specify additional requirements outside of this specification to determine the exact behavior for this return value.

Examples:

None

2.1.5 Return TPM Operation Response to OS Environment

Command Status:

Mandatory

Input Arguments:

Arg0 (Buffer): UUID = 3DDDFAA6-361B-4eb4-A424-8D10089D1653

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 5

Arg3 (Package): Arguments = Empty Package

Returns:

Type: Package

Integer 1:

Purpose: Function Return Code

Description:

0: Success

1: General Failure

Integer 2:

Purpose: Most recent operation request

Description:

0: None

>0: Operation value of the most recent request

[See Table 2]

Integer 3:

Purpose: Response to the most recent operation request

Description:

0: Success

0x00000001..0x00000FFF: Corresponding TPM error code

0xFFFFFFF0: User Abort

0xFFFFFFF1: BIOS Failure

Functional Behavior:

This function allows the BIOS to communicate the response to the most recent operation request it acted upon. The function returns both the operation number and the outcome of the BIOS attempt to perform the operation.

The OS is advised to query the TPM directly to determine whether a request was indeed fulfilled and to use this function's return values only for troubleshooting failures and auditing purposes. The reason is that for some platforms, the return values may not be reliable. For example, if multiple OS's exist on the platform, the request-response values cannot be correlated to any particular OS. Furthermore, there is no guarantee that the response is available to the OS that originated the request; another OS may submit a new request, overwriting the response to the previous request.

If 1 is returned as the first integer, the BIOS is unable to return meaningful information due to an internal failure during this function call. In this case, the second and third integers are undefined.

Assume in the following paragraphs that 0 is returned as the first integer.

If 0 is returned as the second integer, no previously-requested operations exist and hence no response exists. In this case, the third integer is undefined.

If a value greater than 0 is returned as the second integer, that value is the most recent operation request seen by the BIOS (see Table 2 below for a list of defined operations).

Assume in the following paragraphs that 0 is returned as the first integer and a value greater than 0 is returned as the second integer. The definitions below refer to the response (i.e. the third integer) to the operation request (i.e. the second integer).

If 0 is returned as the response, then the requested operation was confirmed and successfully executed in the pre-OS environment.

A response value from 0x00000001 to 0x00000FFF inclusive corresponds to the TPM error codes defined in Chapter 16 of the TPM Main Specification – Part 2 TPM Structures document.

If 0xFFFFFFF0 is returned as the response, the user failed to confirm the operation request.

If 0xFFFFFFF1 is returned as the response, then a BIOS failure prevented the successful execution of the requested operation (e.g. invalid operation request, BIOS communication error with the TPM).

Examples

A return value of {1, 0, 0} indicates that a BIOS internal failure prevented the function from retrieving meaningful information. (This corresponds to a BIOS error during this function call.)

A return value of {0, 0, 0} indicates that no previously requested operation exists and hence no response exists.

A return value of {0, 6, 0} indicates that the last operation request to enable and activate the TPM succeeded.

A return value of {0, 6, 0xFFFFFFF0} indicates that the last operation request to enable and activate the TPM was rejected by the physically present user in the pre-OS environment.

A return value of {0, 6, 0xFFFFFFF1} indicates that the last operation request to enable and activate the TPM failed to execute successfully due to an internal BIOS error. (This corresponds to a previous BIOS error performing the operation during the boot process.)

2.1.6 Submit preferred user language

Command Status:

Deprecated and Mandatory

Input Arguments:

Arg0 (Buffer): UUID = `3DDDFAA6-361B-4eb4-A424-8D10089D1653`

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 6

Arg3 (Package): Arguments = Package --

Type: String

Purpose: Preferred language code

Description:

Language code that begins with the 2-character ISO-639-1 format

See http://www.loc.gov/standards/iso639-2/php/English_list.php

Returns:

Type: Integer

Purpose: Function Return Code

Description:

3: Not implemented

Functional Behavior:

It is mandatory this function be implemented as part of the ACPI interface, but this function is deprecated and MUST NOT perform any action. The function MUST return 3. (Refer to prior versions of this specification for more information about past usage of this function.)

2.1.7 Submit TPM Operation Request to Pre-OS Environment 2

Command Status:

Mandatory

Input Arguments:

Arg0 (Buffer): UUID = `3DDDFAA6-361B-4eb4-A424-8D10089D1653`

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 7

Arg3 (Package): Arguments = Package --

Type: Integer

Purpose: Operation Value of the Request

Description: [see Table 2]

Returns:

Type: Integer

Purpose: Function Return Code

Description:

0: Success

1: Not Implemented

2: General Failure

3: Operation blocked by current BIOS settings

Functional Behavior:

This function allows the OS to submit a request for an operation to be executed in the pre-OS environment (see Table 2 for a list of defined operations). This request is the only input from the OS to the pre-OS environment.

If 0 is returned, the requested operation can be read and acted upon by the BIOS once the transition to the pre-OS environment takes place (e.g. after the platform has restarted). The OS expects that the pre-OS environment verifies physical presence and confirms that the physically present user in fact requested the execution of the operation.

If 1 is returned, the BIOS does not support the OS requesting the operation because it is not implemented and is never able to be requested by the OS regardless of BIOS settings. For example, the implementation of the operation may be optional or vendor-specific (e.g. implemented as an action that may only be performed using a Platform Manufacturer utility).

If 2 is returned, the BIOS is otherwise unable to read and act upon the request. For example, platform specific security protections may exist to prevent burnout of the storage location for the operation.

If 3 is returned, the current BIOS settings prohibit this action, but the platform may permit the operation if the platform owner configures the BIOS settings differently.

The OS may call this function or “Submit TPM Operation Request to Pre-OS Environment” multiple times before transitioning to the pre-OS environment. However, only the last submitted request is valid. The OS may submit an operation request of value 0 to clear any previous requests.

Examples:

A submitted operation value of 6 (Enable + Activate) and a return value of 0 indicate the platform has successfully received a request to enable and activate the TPM.

A submitted operation value of 18 (SetNoPPIClear_True) and a return value of 1 indicate the Platform Manufacturer did not implement the Persistent BIOS TPM Management Flag named NoPPIClear and did not implement the corresponding operation to set the value.

A submitted operation value of 5 (Clear) and a return value 3 indicates a current BIOS setting currently blocks the OS from requesting to clear the TPM. However, a BIOS administrator may configure the BIOS settings to permit the OS to request to clear the TPM in the future.

2.1.8 Get User Confirmation Status for Operation

Command Status:

Mandatory

Input Arguments:

Arg0 (Buffer): UUID = 3DDDFAA6-361B-4eb4-A424-8D10089D1653

Arg1 (Integer): Revision ID = 1

Arg2 (Integer): Function Index = 8

Arg3 (Package): Arguments = Package --

Type: Integer

Purpose: Operation Value which may need user confirmation

Description: [see Table 2]

Returns:

Type: Integer

Purpose: Function Return Code

Description:

0: Not implemented

1: BIOS only

2: Blocked for OS by BIOS configuration

3: Allowed and physically present user required

4: Allowed and physically present user not required

Functional Behavior:

This function allows the OS to determine whether it is allowed to request an operation to be executed in the pre-OS environment (see Table 2 for a list of defined operations) by the BIOS. If the OS is not allowed by the BIOS to perform the operation, the function returns whether the BIOS administrator may reconfigure the BIOS to allow the OS to request the operation.

If the BIOS permits the OS to request an operation, this function allows the OS to determine whether a request would require confirmation from a physically present user or would be performed automatically because of the current settings for the BIOS TPM Management flags.

This function does not cause the operation to be requested.

Note: It is preferable for an OS to use this function to determine if an operation is supported instead of using “Submit TPM Operation Request to Pre-OS Environment 2” because this function does not write to NVRAM.

If 0 is returned, the BIOS does not support this operation because it is not implemented. An example is if the Platform Manufacturer did not support the NoPPIClear flag and the OS queried for the operation SetNoPPIClear_True. Note: For an implementation of version 1.2 or later of this interface, a return value of 0 means the operation value passed in Arg3 is not implemented. If an earlier version of this specification is implemented, a return value of 0 means this method is not implemented.

If 1 is returned, the BIOS unconditionally does not permit the OS to request this action, but a BIOS administrator may perform the equivalent action through a vendor specific mechanism like a BIOS configuration utility.

If 2 is returned, the current BIOS settings do not permit the OS to request this operation, but a BIOS administrator may change the setting through a vendor specific mechanism like a BIOS configuration utility to allow the OS to request the operation.

If 3 is returned, the BIOS currently permits the OS to request this action and the operation would require physically present user approval

If 4 is returned, the BIOS currently permits the OS to request this action and the current BIOS settings permit execution of the operation without a physically present user approving the operation

Start of informative comment:

To implement this function, the BIOS may need to copy the persistent settings for the BIOS TPM Management flags when accessible during boot as the settings influence the return values for this function. The logic of this function will also need to implement the logic in the “When Physical Presence Confirmation is Required” column of Table 2.

End of informative comment.

Examples:

A submitted operation value of 18 (NoPPIClear_True) and a return value of 0 indicate the Platform Manufacturer did not implement the Persistent BIOS TPM Management Flag named NoPPIClear and did not implement the corresponding operation to set the value.

A submitted operation value of 12 and a return value of 0 indicate the platform does not support Deferred Physical Presence for unowned field upgrade.

A submitted operation value of 5 and a return value of 2 indicate the BIOS is currently configured to not allow an OS to request to clear the TPM using physical presence, but a BIOS administrator may modify the BIOS configuration using a vendor specific BIOS configuration utility to permit the OS to request the operation.

A submitted operation value of 5 and a return value of 3 indicate the BIOS is currently configured to allow an OS to request to clear the TPM, but the NoPPIClear flag is not implemented or has a value of FALSE so a physically present user would need to approve the operation.

A submitted operation value of 14 and a return value of 4 indicate the BIOS is currently configured to allow an OS to request to clear, enable and activate the TPM, the NoPPIClear flag is Set to TRUE, and the NoPPIProvision flag is Set to TRUE so a physically present user is not needed to approve the operation.

2.2 Parameter Passing

Start of informative comment:

With the exception of operation #13 defined in Table 2, all parameters between the OS and the BIOS are passed using the standard ACPI defined parameter passing method. The values passed for operations, except #13, are small and are easily accommodated by the 4 arguments. However, operation #13 requires the passing of an additional value of a relatively large size: 20 bytes. As it is not possible to add another parameter, the method for passing this value from the OS to the BIOS is not defined in this version of the specification. A later version of this specification may address, define and standardize the method for passing this value. Until then, any suitable method that is in agreement between the OS and BIOS is permitted.

End of informative comment.

The method for passing the operator authentication value from the OS to the BIOS is to be determined by convention between the OS and the BIOS. Standardization and definition of the method will be addressed in a later revision of this specification.

3. Operation Definitions

Start of informative comment:

Table 3 holds the definition of what ordered actions are performed by the BIOS to complete each of the operations defined in this specification. The BIOS sends commands to the TPM, performs platform resets and/or changes BIOS TPM Management Flags.

If a platform implements the hardware method of asserting physical presence, the utility which prompts the user to indicate physical presence should display the same confirmation dialogs and perform the same actions listed in Table 3 for each operation for consistency with this specification.

End of informative comment.

1. In order to persist the OS requested operation to the next boot cycle, the BIOS must allocate a location for storing the operation's corresponding value.
2. The operation value of 0 is reserved to indicate that no Physical Presence Interface operation has been requested.
3. The BIOS MUST implement operations 0 through 11, operations 14, 15, 16, 21 and 22. The BIOS MAY implement all other operations.
4. Operations with values at or above 128 (0x80) are designated for vendor-specific usage.
5. For each operation implemented by the BIOS, the BIOS MUST perform the TPM commands listed in the third column of Table 3 in the order listed. (Note: A reboot is required for some changes to take effect. This is noted with the keyword: <PLATFORM RESET>.)
6. For operations 5 and 14 the platform manufacturer MAY perform the TPM commands Enable and Activate before performing the TPM_ForceClear operation. (Note: This would make operation 5 identical to operation 21 and/or operation 14 identical to operation 22.)
7. The BIOS is responsible for tracking and performing operations that require a PLATFORM RESET.
8. The location for tracking the pending operation, including the tracking of necessary PLATFORM RESET operations, does not need to be a secure or trusted location.
9. If during an operation that requires a Platform Reset, the BIOS stores the assertion of an indication of physical presence across the Platform Reset, that location must be trusted.
10. The BIOS MUST provide persistent storage for the NoPPIProvision flag.
11. If the BIOS implements operations 17 and 18, the BIOS MUST provide persistent storage for the NoPPIClear flag.
12. If the BIOS implements operations 19 and 20, it MUST provide persistent storage for the NoPPIMaintenance flag.

Operation Value	Operation Name	What the operation may change		Mandatory versus Optional	When Physical Presence Confirmation is Required	May need additional boot cycle
		TPM State	BIOS TPM Management Flags			
0	No Operation			M		
1	Enable	X		M	NoPPIProvision is FALSE	
2	Disable	X		M	NoPPIProvision is FALSE	
3	Activate	X		M	NoPPIProvision is FALSE	X
4	Deactivate	X		M	NoPPIProvision is FALSE	X
5	Clear	X		M	NoPPIClear is FALSE	X
6	Enable + Activate	X		M	NoPPIProvision is FALSE	X
7	Deactivate + Disable	X		M	NoPPIProvision is FALSE	X
8	SetOwnerInstall_True	X		M	NoPPIProvision is FALSE	
9	SetOwnerInstall_False	X		M	NoPPIProvision is FALSE	
10	Enable + Activate + SetOwnerInstall_True	X		M	NoPPIProvision is FALSE	X
11	SetOwnerInstall_False + Deactivate + Disable	X		M	NoPPIProvision is FALSE	X
12	Deferred Physical Presence-unownedFieldUpgrade	X		O	NoPPIMaintenance is FALSE	X
13	SetOperatorAuth	X		O	NoPPIProvision is FALSE	
14	Clear + Enable + Activate	X		M	NoPPIClear is FALSE OR NoPPIProvision is FALSE	X
15	SetNoPPIProvision_False		X	M		
16	SetNoPPIProvision_True		X	M	Always	
17	SetNoPPIClear_False		X	O1[1]		
18	SetNoPPIClear_True		X	O1	Always	
19	SetNoPPIMaintenance_False		X	O2[2]		
20	SetNoPPIMaintenance_True		X	O2	Always	
21	Enable + Activate + Clear	X		M	NoPPIClear is FALSE	X
22	Enable + Activate + Clear + Enable + Activate	X		M	NoPPIClear is FALSE OR NoPPIProvision is FALSE	X
23 – 127	Reserved					
>=128	Vendor Specific	X	X	O		X

Table 2: Physical Presence Interface Operation Summary

[1] If the BIOS implements the items marked “O1” or “O2” it MUST implement them as a set.

[2] The BIOS MUST NOT implement operations 19 and 20 if it does not implement operation 12.

Operation Value	Operation Name	TPM commands sent by BIOS and other BIOS actions to perform the operation
0	No Operation	*No operation*
1	Enable	TPM_PhysicalEnable
2	Disable	TPM_PhysicalDisable
3	Activate	TPM_PhysicalSetDeactivated = FALSE <PLATFORM RESET>
4	Deactivate	TPM_PhysicalSetDeactivated = TRUE <PLATFORM RESET>
5	Clear	TPM_ForceClear <PLATFORM RESET>
6	Enable + Activate	TPM_PhysicalEnable TPM_PhysicalSetDeactivated = FALSE <PLATFORM RESET>
7	Deactivate + Disable	TPM_PhysicalSetDeactivated = TRUE TPM_PhysicalDisable <PLATFORM RESET>
8	SetOwnerInstall_True	TPM_SetOwnerInstall = TRUE
9	SetOwnerInstall_False	TPM_SetOwnerInstall = FALSE
10	Enable + Activate + SetOwnerInstall_True	TPM_PhysicalEnable TPM_PhysicalSetDeactivated = FALSE <PLATFORM RESET> TPM_SetOwnerInstall = TRUE
11	SetOwnerInstall_False + Deactivate + Disable	TPM_SetOwnerInstall = FALSE TPM_PhysicalSetDeactivated = TRUE TPM_PhysicalDisable <PLATFORM RESET> Note: The BIOS MAY perform the PLATFORM RESET immediately after the TPM_PhysicalSetDeactivated = TRUE but that requires tracking the next command.
12	Deferred Physical Presence-unownedFieldUpgrade	TPM_SetCapability -> setValue (Cap= TPM_SET_STCLEAR_FLAGS; SubCap= TPM_SD_DEFERREDPHYSICALPRESENCE ; value= unownedFieldUpgrade Implementer's note: The BIOS calls the above SetCapability while the operator is asserting Physical Presence. Rather than a command to perform a specific function, this function sets an attribute within the TPM indicting that the Operator has asserted Physical Presence to authorize the TPM_FieldUpgrade function.
13	SetOperatorAuth	TPM_SetOperatorAuth, with operatorAuth prompted by the BIOS
14	Clear + Enable + Activate	TPM_ForceClear TPM_PhysicalEnable TPM_PhysicalSetDeactivated = FALSE <PLATFORM RESET>
15	SetNoPPIProvision_False (Require physical presence confirmation for provisioning.)	This operation does not change TPM state. Clears the BIOS TPM Management Flag NoPPIProvision to FALSE.
16	SetNoPPIProvision_True (Do not require physical presence confirmation for provisioning.)	This operation does not change TPM state. Sets the BIOS TPM Management Flag NoPPIProvision to TRUE.
17	SetNoPPIClear_False (Require physical presence for clear.)	This operation does not change TPM state. Clears the BIOS TPM Management Flag NoPPIClear to FALSE.
18	SetNoPPIClear_True (Do not require physical presence confirmation for clear.)	This operation does not change TPM state. Sets the BIOS TPM Management Flag NoPPIClear to TRUE.
19	SetNoPPIMaintenance_False (Require physical presence confirmation for maintenance.)	This operation does not change TPM state. Clears the BIOS TPM Management Flag NoPPIMaintenance to FALSE.
20	SetNoPPIMaintenance_True (Do not require physical presence confirmation for maintenance.)	This operation does not change TPM state. Sets the BIOS TPM Management Flag NoPPIMaintenance to TRUE.

Operation Value	Operation Name	TPM commands sent by BIOS and other BIOS actions to perform the operation
21	Enable + Activate + Clear	If the TPM is disabled then perform TPM_PhysicalEnable OR unconditionally perform TPM_PhysicalEnable If the TPM is deactivated, then perform TPM_PhysicalSetDeactivated = FALSE & <PLATFORM RESET> OR unconditionally perform TPM_PhysicalSetDeactivated = FALSE & <PLATFORM RESET> TPM_ForceClear <PLATFORM RESET>
22	Enable + Activate + Clear + Enable + Activate	If the TPM is disabled then perform TPM_PhysicalEnable OR unconditionally perform TPM_PhysicalEnable If the TPM is deactivated, then perform TPM_PhysicalSetDeactivated = FALSE & <PLATFORM RESET> OR unconditionally perform TPM_PhysicalSetDeactivated = FALSE & <PLATFORM RESET> TPM_ForceClear TPM_PhysicalEnable TPM_PhysicalSetDeactivated = FALSE <PLATFORM RESET>
23 – 127	Reserved	Reserved, do not implement or use
>=128	Vendor Specific	TPM commands mapping to vendor specific operation

Table 3: Definition of Operation Actions

4. Confirmation Dialogs and Keys

Start of informative comment:

Confirmation Dialogs

This section is intended to provide design guidance for the BIOS dialogs used to confirm that the physically present user in fact requested execution of the operation. Each operation request is mapped to the dialog presented to the user to accept or reject the operation.

In consideration of the limited space in the BIOS to store text, the confirmations presented in this section attempt to maximize the potential for text reuse while minimizing impact on security and user understanding.

For consistency in user experience, it is recommended that Platform Manufacturers implement confirmation dialogs in a similar manner.

Platform Manufacturers should also take into consideration localization requirements for the dialog text. The BIOS should provide a means of ensuring that the dialog text displayed to the user can be understood by that user (e.g. allow the user to change the language of the confirmation request, default to the user's preferred language, etc.).

While the text and confirmation keys in Table 4 have been carefully considered to meet a broad range of language and cultural environments, these may not convey the meaning of the action to all persons depending on their language, culture, education, and expertise with computing platforms. For this reason, the text provided is preferred but is not mandatory for all situations.

Confirmation Keys

For security purposes, note that the key used to confirm the Clear and the Deferred Physical Presence unownedFieldUpgrade operations differs from the confirmation key used for other operations.

The confirmation keys are selected for most keyboards and expected user experiences. While the keys selected do not conflict with a majority of BIOS implementation, some conflicts may occur. The BIOS provider may have used the confirmation keys in Table 5 for other purposes already familiar to the platform user. In this case, changing the meaning of the key would be confusing to the user. For this reason, the BIOS is allowed to use other keys when such conflicts occur.

End of informative comment.

1. For each operation to be performed through the Physical Presence Interface the BIOS MUST ask for confirmation from a physically present operator per the "When Physical Presence Confirmation is Required" column in Table 2. (For example, for operation 14, the BIOS MUST ask for a physically present user to confirm the operation if either of the flags NoPPIClear or NoPPIProvision is FALSE. However, if both flags are TRUE, then the BIOS MUST NOT ask for confirmation.)
2. The text in Table 4 is the preferred confirmation text wording to be displayed to the user. The BIOS MAY provide alternate confirmation text wording that differs from Table 4 based upon the platform's expected user experience and language; however, the meaning of each message MUST be retained. The text in Table 4 MAY be translated into languages other than the English text indicated in Table 4 provided the meaning of the text in Table 4 is conveyed in the translated text.

3. Table 5 contains labels for the following user confirmation keys: reject key (labeled as <RK>), accept key (labeled as <AK>), and cautionary accept key (labeled as <CAK>). The key associated with each label represents a user action indicated by the associated text. The actual key used for each labeled key is to be determined by the platform's expected user experience. Typical and example keys that SHOULD be used for most environments are provided in Table 5. All occurrences of confirmation keys must be consistent for each operation for the same platform, i.e. the BIOS for each platform MUST use the same key associated to each label. The text representing the key SHOULD exclude the angle bracket (i.e. the '<' and '>' characters) and SHOULD use the string representing the actual key indicated in Table 5 when presenting the text in Table 4.
4. The BIOS may provide alternate confirmation keys that differ from Table 5 based upon the platform's keyboard configuration or expected user experience.

Op Value	Operation Name	Confirmation Text
0	No Operation	None
1	Enable	A configuration change was requested to enable this computer's TPM (Trusted Platform Module) Press <AK> to enable the TPM Press <RK> to reject this change request and continue
2	Disable	A configuration change was requested to disable this computer's TPM (Trusted Platform Module) WARNING: Doing so might prevent security applications that rely on the TPM from functioning as expected Press <AK> to disable the TPM Press <RK> to reject this change request and continue
3	Activate	A configuration change was requested to activate this computer's TPM (Trusted Platform Module) Press <AK> to activate the TPM Press <RK> to reject this change request and continue
4	Deactivate	A configuration change was requested to deactivate this computer's TPM (Trusted Platform Module) WARNING: Doing so might prevent security applications that rely on the TPM from functioning as expected Press <AK> to deactivate the TPM Press <RK> to reject this change request and continue
5	Clear	A configuration change was requested to clear this computer's TPM (Trusted Platform Module) WARNING: Clearing erases information stored on the TPM. You will lose all created keys and access to data encrypted by these keys. Press <CAK> to clear the TPM Press <RK> to reject this change request and continue
6	Enable + Activate	A configuration change was requested to enable and activate this computer's TPM (Trusted Platform Module) NOTE: This action will turn on the TPM Press <AK> to enable and activate the TPM Press <RK> to reject this change request and continue

Op Value	Operation Name	Confirmation Text
7	Deactivate + Disable	A configuration change was requested to deactivate and disable this computer's TPM (Trusted Platform Module) NOTE: This action will turn off the TPM WARNING: Doing so might prevent security applications that rely on the TPM from functioning as expected Press <AK> to deactivate and disable the TPM Press <RK> to reject this change request and continue
8	SetOwnerInstall_True	A configuration change was requested to allow a user to take ownership of this computer's TPM (Trusted Platform Module) Press <AK> to allow a user to take ownership of the TPM Press <RK> to reject this change request and continue
9	SetOwnerInstall_False	A configuration change was requested to disallow a user to take ownership of this computer's TPM (Trusted Platform Module) Press <AK> to disallow a user to take ownership of the TPM Press <RK> to reject this change request and continue
10	Enable + Activate + SetOwnerInstall_True	A configuration change was requested to enable, activate, and allow a user to take ownership of this computer's TPM (Trusted Platform Module) NOTE: This action will turn on the TPM Press <AK> to enable, activate, and allow a user to take ownership of the TPM Press <RK> to reject this change request and continue
11	SetOwnerInstall_False + Deactivate + Disable	A configuration change was requested to deactivate, disable, and disallow a user to take ownership of this computer's TPM (Trusted Platform Module) NOTE: This action will turn off the TPM WARNING: Doing so might prevent security applications that rely on the TPM from functioning as expected Press <AK> to deactivate, disable, and disallow a user to take ownership of the TPM Press <RK> to reject this change request and continue
12	Deferred Physical Presence-unownedFieldUpgrade	A configuration change was requested to allow changes to the TPM's (Trusted Platform Module's) firmware WARNING: Allowing changes to the TPM's firmware may affect the operation of the TPM and may erase information stored on the TPM. You may lose all created keys and access to data encrypted by these keys Press <CAK> to Allow field upgrade of the TPM Press <RK> to reject this change request and continue
13	SetOperatorAuth	A configuration change was requested to allow the creation of an operator authentication value that permits the temporary deactivation of this computer's TPM (Trusted Platform Module) Press <AK> to allow the creation of the operator authentication value of the TPM Press <RK> to reject this change request and continue

Op Value	Operation Name	Confirmation Text
14	Clear + Enable + Activate	A configuration change was requested to clear, enable, and activate this computer's TPM (Trusted Platform Module) NOTE: This action will clear and turn on the TPM WARNING: Clearing erases information stored on the TPM. You will lose all created keys and access to data encrypted by these keys. Take ownership as soon as possible after this step. Press <CAK> to clear, enable, and activate the TPM Press <RK> to reject this change request and continue
15	SetNoPPIProvision_False	Physical presence confirmation is not necessary. The BIOS MAY display a message during boot to the user but MUST not ask the user for confirmation.
16	SetNoPPIProvision_True	A configuration change was requested to allow the Operating System to provision the computer's TPM (Trusted Platform Module) without asking for user confirmation in the future. Press <AK> to approve future Operating System requests to provision the TPM Press <RK> to reject this change request and continue
17	SetNoPPIClear_False	Physical presence confirmation is not necessary. The BIOS MAY display a message during boot to the user but MUST not ask the user for confirmation.
18	SetNoPPIClear_True	A configuration change was requested to allow the Operating System to clear the computer's TPM (Trusted Platform Module) without asking for user confirmation in the future. NOTE: This action does not clear the TPM, but by approving this configuration change, future actions to clear the TPM will not require user confirmation. WARNING: Clearing erases information stored on the TPM. You will lose all created keys and access to data encrypted by these keys. Press <CAK> to approve future Operating System requests to clear the TPM Press <RK> to reject this change request and continue
19	SetNoPPIMaintenance_False	Physical presence confirmation is not necessary. The BIOS MAY display a message during boot to the user but MUST not ask the user for confirmation.
20	SetNoPPIMaintenance_True	A configuration change was requested to allow the Operating System to maintain the computer's TPM (Trusted Platform Module) without asking for user confirmation in the future. WARNING: Allowing future changes to the TPM's firmware may affect the operation of the TPM and may erase information stored on the TPM. You may lose all created keys and access to data encrypted by these keys Press <CAK> to approve future Operating System requests to maintain the TPM Press <RK> to reject this change request and continue
21	Enable + Activate + Clear	A configuration change was requested to enable, activate and clear this computer's TPM (Trusted Platform Module) WARNING: Clearing erases information stored on the TPM. You will lose all created keys and access to data encrypted by these keys. Press <CAK> to clear the TPM Press <RK> to reject this change request and continue

Op Value	Operation Name	Confirmation Text
22	Enable + Activate + Clear + Enable + Activate	A configuration change was requested to enable, activate, clear, enable, and activate this computer's TPM (Trusted Platform Module) NOTE: This action will clear and turn on the TPM WARNING: Clearing erases information stored on the TPM. You will lose all created keys and access to data encrypted by these keys. Take ownership as soon as possible after this step. Press <CAK> to clear, enable, and activate the TPM Press <RK> to reject this change request and continue
23 – 127	Reserved	
>=128	Vendor Specific	

Table 4: TPM functions for and Confirmations of Physical Presence Interface Operations

User Confirmation Keys	Key Meaning	Example Actual Key	Text to Represent the Actual Key
<RK>	Reject Key – user rejects the action	ESC key	[ESC]
<AK>	Accept Key – user accepts the action where the action does not cause a potential loss of data	Function F10 key	[F10]
<CAK>	Caution Accept Key – user accepts the action using a different key than the Accept Key to mitigate the potential of accepting an action that may cause the loss of data.	Function F12 key	[F12]

Table 5: User Confirmation Key Mappings

5. Physical Presence Interface Pseudo Code

Start of informative comment:

The pseudo code below is intended to provide design guidance for the implementation of the Physical Presence Interface. As long as normative requirements are adhered to, the PM is free to vary implementation details.

End of informative comment.

```
/*****************************************************************************
This section of the pseudo code illustrates the behavior of the operating system user
mode components that initiate the Physical Presence Interface call into the BIOS. Note
that this section is for understanding only and does not dictate actual OS behavior.
*****************************************************************************/
```

OS Admin Tool()

```
If (User_Desires_Functionality_That_Requires_Physical_Presence()) {

       Display_Platform_Specific_Dialog_Documentation()

       If (Software_Physical_Presence_Supported()) {

              Write_Physical_Presence_Interface_Request(OperationValue)

              // see pseudocode in the method below for more details
              Optional_Register_OS_Component_To_Run_On_Next_Boot(OS_Component_Post_Rebo
              ot())

              Execute_Platform_Specific_Transition()
       }
  }
```

OS Component Post Reboot()

```
If (Get_Physical_Presence_Interface_Response()!= SUCCESS) {

       Optional_Provide_Failure_Notice()
}
```

```
/*****************************************************************************
This section of the pseudo code illustrates the behavior of the pre-OS environment
including the CRTM and portion of the BIOS that acts upon the Physical Presence
Interface requests. Implementation details may differ as the CRTM can be either the
boot block or the entire BIOS.
*****************************************************************************/
```

Main()

```
CRTM()
Post_ BIOS()
Boot_Into_OS()
```

CRTM()

```
Reset_Vector()

If (Physical_Presence_Asserted()) {
       Turn_On_Physical_Presence_Flag()
} Else {
       Lock_Physical_Presence()
}
```

Post BIOS()

```
If (OperationValue =Read_Physical_Presence_Interface_Request()) {

       RequestConfirmed = False

       If (OperationValue = 1, 2, 3, 4, 6, 7, 8, 9, 10, 11 Or 13) {
              If (NoPPIProvision = True) {
                     RequestConfirmed = True
              }
       } Else If (OperationValue = 5 Or 21) {
              If (NoPPIClear = True) {
                     RequestConfirmed = True
              }
       } Else If (OperationValue = 12) {
              If (NoPPIMaintenance = True) {
                     RequestConfirmed = True
              }
       } Else If (OperationValue = 14 Or 22) {
              If (NoPPIProvision = True And NoPPIClear = True) {
                     RequestConfirmed = True
              }
       } Else If (OperationValue = 15, 17, Or 19) {
              RequestConfirmed = True
       }

       If (RequestConfirmed = False) {
              If (Physical_Presence_Asserted_In_CRTM()) {
                     If (Optional_BIOS_Administrative_Password_Authenticated()){
                            RequestConfirmed = Prompt_Confirmation_Of_Request()
                     }
              }
       }
       Response = UserAbort

       If (RequestConfirmed) {

              // see pseudocode in the method below for more details
```

```
                Response =
                Execute_Physical_Presence_Interface_Request(OperationValue)
        }

        Clear_Physical_Presence_Interface_Request()
        Set_Physical_Presence_Interface_Response(Request, Response)

        If (RequestsConfirmed) {
                // a reboot is necessary to activate the operation
                ResetVector()
        }



} Else If (Request_To_Enter_Startup_Setup() within Timeout Period) {

        Standard_Setup()
}

Lock_Physical_Presence()
```

Execute Physical Presence Interface Request(OperationValue)

```
For Each TPM_Command in OperationCommands.Get(OperationValue)  {
        ReturnValue = Execute(TPM_Command)

        If ReturnValue != SUCCESS
                Return ReturnValue

}
```

```
/*****************************************************************************
This section of the pseudo code describes functions used in previous sections.
*****************************************************************************/
```

1. User_Desires_Functionality_That_Requires_Physical_Presence()

   If true, the user logged in to the OS has chosen to perform an action that requires physical presence authorization. For example, the user has chosen to force clear the TPM.

2. Display_Platform_Specific_Dialog_Documentation()

   The OS displays platform-specific documentation to inform the user of the procedure that must take place to successfully execute the desired functionality. The platform-specific documentation is multi-lingual for users of different locales. The text may be customized by the PM during OS pre-install time.

   The dialog that displays the platform-specific documentation also contains a button to carry out the platform's transition action as specified through ACPI (see Table 4).

   If the specified ACPI functions are not implemented, the default dialog alerts the user to consult the documentation that shipped with their platform to execute the desired functionality.

3. Software_Physical_Presence_Supported()

   If true, the PM has implemented the software/command method of asserting physical presence. This method reads the TPM permanent flag structure's physicalPresenceCMDEnable flag.

4. Write_Physical_Presence_Interface_Request(OperationValue)

   The OS calls the "Submit TPM Operation Request to Pre-OS Environment 2" Function of the Physical Presence Interface to submit the operation request to the BIOS (see Table 2).

5. Optional_Register_OS_Component_To_Run_On_Next_Boot(OS_Component_Post_Reboot())

   The OS optionally registers a component to run post-reboot in order to perform clean up based on the return value. See the OS_Component_Post_Reboot() method in the pseudocode for more details

6. Execute_Platform_Specific_Transition()

   The OS executes the action requested by the user in the platform-specific dialog (see Display_Platform_Specific_Dialog_Documentation()). For example, the action may be to shut down the platform.

7. Get_Physical_Presence_Interface_Response()

   The OS calls the "Return TPM Operation Response to OS Environment" function of the Physical Presence Interface to read the return value of the request (see Table 2).

8. Optional_Provide_Failure_Notice()

   Provides notification of the failure (e.g. write failure to event log).

9. Reset_Vector()

   The vector within the CRTM that is first executed upon platform reboot.

10. Physical_Presence_Asserted()

    If true, the CRTM has sensed the physical presence assertion of the user. For example, the user has pressed the startup button or inserted a USB dongle. The details of the implementation are vendor-specific.

11. Turn_On_Physical_Presence_Flag()

    Sets the PhysicalPresence permanent flag to true by executing TSC_PhysicalPresence(TPM_PHYSICAL_PRESENCE = TPM_PHYSICAL_PRESENCE_PRESENT). At this time, the physicalPresenceLock has already been reset to false due to execution of TPM_Startup(STCLEAR).

12. Lock_Physical_Presence()

    Sets the PhysicalPresenceLock = True and PhysicalPresence – False using the command TSC_PhysicalPresence(TPM_PHYSICAL_PRESENCE = TPM_PHYSICAL_LOCK)

13. Read_Physical_Presence_Interface_Request()

    Reads from the storage location modified by Write_Physical_Presence_Interface_Request(OperationValue) and returns this operation value.

14. Physical_Presence_Asserted_In_CRTM()

    If true, physical presence was asserted in the CRTM. This method reads the TPM permanent flag structure's physicalPresence flag or optionally, a manufacturer-specific flag that was set when physical presence was sensed via Physical_Presence_Asserted().

15. Optional_BIOS_Administrative_Password_Authenticated()

    The PM may optionally decide to implement the ability to authenticate with a BIOS administrative password before a TPM command that requires physical presence can be executed. If true is returned, the BIOS administrative password has been authenticated.

16. Prompt_Confirmation_Of_Request()

    This is the dialog displayed to the user that prompts for confirmation of the Physical Presence Interface request. The return value is True or False depending on whether the user confirms or rejects the request, respectively. The confirmation should be implemented such that the user can understand at a high level the security implications of the operation and must actively choose to execute the operation (e.g. the default should not be to confirm the operation).

    See Table 4 for guidance on the dialog text.

17. Clear_Physical_Presence_Interface_Request()

    Clears the storage area modified by Write_Physical_Presence_Interface_Request(OperationValue), for example by setting it to a value of 0.

18. Set_Physical_Presence_Interface_Response(Request, Response)

    Writes the response so that it can be retrieved by calling the “Return TPM Operation Response to OS Environment” function of the Physical Presence Interface.

19. Request_To_Enter_Startup_Setup()

    There must be a way to enter standard setup of the BIOS when no Physical Presence Interface requests are present. In this pseudo code, standard setup cannot be entered if a request exists on the Physical Presence Interface. The BIOS implementation may choose a different method.