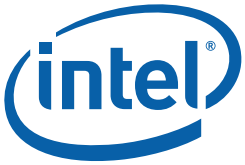

Intel® Trusted Platform Module Vendor Specific Ordinals

Document Type

July 2008

Revision 1.00

V 0.3

Contents

Tables

Revision History

Table 1. Revision history

Document Number	Description	Revision Date
0.1	Initial draft	5 sep 2007
0.2	First content	26 sep 2007
0.26	Changes based on feedback	18 oct 2007
0.3	Changes based on feedback	22 oct 2007
0.31	Name change	11 Nov 2007
0.9	Update based on feedback	28 Apr 2008
0.92	Remove depreciated ordinals	26 June 2008
1.0		29 July 2008

§

1 Introduction

This document is intended to provide all information necessary to allow one to run the Intel® Trusted Platform Module (Intel® TPM) ordinals.

1.1 Terminology

Table 2. Terminology

Term	Description
Intel® AMT	Intel® Active Management Technology (Intel® AMT)
FIT	Flash Image Tool
FPT	Flash Programming Tool
FW	Firmware
Intel® TPM	Intel® Trusted Platform Module (Intel® TPM)
LMS	Local Manageability Service
Intel® ME	Intel® Manageability Engine (Intel® ME)
NVM	Non Volatile Memory
OS	Operating System
SKU	Stock Keeping Unit
SOL	Serial Over LAN
SPI Flash	Serial Peripheral Interface Flash
Sx	Sleep State (where x is the specific state)
IDE-R	IDE Redirection

1.2 Reference Documents

The documents listed in the table below provide supplementary and background information.

Table 3. Reference documents

Document	Location
Standard Intel® TPM documents	*https://www.trustedcomputinggroup.org/specs/TPM* Design principles, structures and command documents within the above link are of use.
Intel® TPM Tools User Guide	Found as part of the OEM kit downloadable from the ARMS Web site.
Intel® ME System Tools User Guide	Found as part of the OEM kit downloadable from the ARMS Web site.
OEM Bringup Guide	Found as part of the OEM kit downloadable from the ARMS Web site.
Intel® TPM Compliance & Test Guide	WIP—will be part of the OEM kit downloadable from the ARMS Web site.
AMT Tools User Guide	Found as part of the OEM kit downloadable from the ARMS Web site.

§

2 Intel® Trusted Platform Module (Intel® TPM) Specific Ordinals

2.1 TPM_GetCapability

2.1.1 TPM_CAP_MFR

Used to retrieve the firmware (FW) version, in the following format:

Table 4. TPM_GetCapability TPM_CAP_MFR

Type	Description
UINT16	Major version
UINT16	Minor version
UINT16	Hot Fix
UINT16	Build

2.1.2 TPM_CAP_DA_LOGIC

Uses **vendorData** field of struct **TPM_DA_INFO** to return the Dictionary Attack parameters, in the following format:

Table 5. TPM_GetCapability TPM_CAP_DA_LOGIC

Type	Description
UINT16	Auth Failure Threshold
UINT16	Initial Lockout Time
UINT16	Lockout Increase Factor
UINT16	Fade Out Time

2.1.3 TPM_CAP_VERSION_VAL

Uses the **vendorSpecific** field of struct **TPM_CAP_VERSION_INFO** to return the FW version, in the same format as TPM_CAP_MFR.

2.2 TPM_FieldUpgrade

A FW update is performed through the **TPM_FieldUpgrade** ordinal.

This ordinal can be validated either by an owner authorization (if owner is present), or by deferred physical presence (only if owner is not present).

Since the FW image is larger than the input ordinal buffer, getting the FW image involves calling this ordinal multiple times.

After the update is complete, Intel® TPM is deactivated until the next **TPM_Init**.

If the ordinal is sent with owner authorization, the authorization will be verified on each call to the ordinal. Failing authorization on one call will invalidate the entire FW update.

TPM_AUTHDATA (Owner Authorization) is a 160-bit (20 Byte) shared-secret plus high-entropy random number. The usual algorithm used to create the AuthData is by taking the shared-secret and random number and mix using SHA-1 digesting. No specific function for generating AuthData is specified by TCG Specification.

Please note that Intel® ME FW Update tool is generating the Intel® TPM_AuthData using the above algorithm from the TPM Owner password parameter (-key), or using the vista generated AuthData file if using –msf parameter.

TPM_FieldUpgrade input and output parameters are not defined by the Intel® TPM spec. Table 6 and Table 7 detail the parameters of **TPM_FieldUpgrade** as implemented by Intel® TPM.

2.2.1 Incoming Parameters and Sizes

Table 6. TPM_FieldUpgrade incoming parameters and sizes

PARAM		HMAC		Type	Name	Description
#	SZ	#	SZ			
1	2			TPM_TAG	tag	TPM_TAG_RQU_AUTH1_COMMAND
2	4			UINT32	paramSize	Total number of input bytes including paramSize and tag
3	4	1S	4	TPM_COMMAND_CODE	ordinal	Command ordinal: TPM_ORD_FieldUpgrade
4	4	2S	4	UINT32	totalLength	Total length of the update image
5	1	3S	1	BOOL	lastSegment	TRUE if this is the last data segment
6	4	4S	4	UINT32	Offset	Offset in the buffer of the current data segment
7	4	5S	4	UINT32	dataLength	Length in bytes of data

PARAM		HMAC		Type	Name	Description
#	SZ	#	SZ			
8	<>	6S	<>	BYTE	Data	Data segment
9	4			TPM_AUTHHANDLE	authHandle	The authorization session handle used for owner authentication.
		2H1	20	TPM_NONCE	authLastNonceEven	Even nonce previously generated by TPM to cover inputs
10	20	3H1	20	TPM_NONCE	nonceOdd	Nonce generated by system associated with authHandle
11	1	4H1	1	BOOL	continueAuthSession	The continue use flag for the authorization session handle
12	20			TPM_AUTHDATA	ownerAuth	HMAC key: ownerAuth.

2.2.2 Outgoing Parameters and Sizes

Table 7. TPM_FieldUpgrade outgoing parameters and sizes

PARAM		HMAC		Type	Name	Description
#	SZ	#	SZ			
1	2			TPM_TAG	tag	TPM_TAG_RSP_AUTH1_COMMAND
2	4			UINT32	paramSize	Total number of output bytes including paramSize and tag
3	4	1S	4	TPM_RESULT	returnCode	The return code of the operation. NOTE: The upgrade operation was successful only if upgradeStatus is STATUS_SUCCESS.
		2S	4	TPM_COMMAND_CODE	ordinal	Command ordinal: TPM_ORD_FieldUpgrade
4	4	3S	4	UINT32	upgradeStatus	The status code returned from the kernel
4	20	2H1	20	TPM_NONCE	nonceEven	Even nonce newly generated by TPM to cover outputs
		3H1	20	TPM_NONCE	nonceOdd	Nonce generated by system associated with authHandle
5	1	4H1	1	BOOL	continueAuthSession	Continue use flag, TRUE if handle is still active
6	20			TPM_AUTHDATA	resAuth	The authorization session digest for the returned parameters. HMAC key: ownerAuth.

2.2.3 Action

1. If **TPM Owner** is installed.
   a. Validate the command and parameters using TPM owner authentication. On error, abort the FW Update and return **TPM_AUTHFAIL**.
2. Else
   a. If **TPM_STCLEAR_DATA** -> **deferredPhysicalPresence** -> **unownedFieldUpgrade** is **FALSE** return **TPM_BAD_PRESENCE**.
3. Copy the given data segment of the update image.
4. If **lastSegment** = **TRUE**
   a. Verify the RSA signature on the image and perform the update process.
   b. Set the **TPM_STCLEAR_FLAGS** deactivated to **TRUE.**
5. Return the status code received from the kernel FW Update protocol as **upgradeStatus**. If the status was not **STATUS_SUCCESS**, abort the FW Update.

2.2.4 Field Upgrade Error codes

1. upgradeStatus value can be interpreted from the table below:

Table 8. TPM_FieldUpgrade UpgradeStatus return codes

Error Code	Error Name
0	NO_UPDATE
1	STATUS_UPDATE_SUCCESS
2	STATUS_UPDATE_IMAGE_INVALID
3	STATUS_UPDATE_INTEGRITY_FAILURE
4	STATUS_UPDATE_SKU_MISMATCH
5	STATUS_UPDATE_FW_VERSION_MISMATCH
6	STATUS_UPDATE_GENERAL_FAILURE
7	STATUS_UPDATE_OUT_OF_RESOURCES
8	STATUS_UPDATE_AUDIT_POLICY_FAILURE
9	STATUS_UPDATE_ERROR_CREATING_FT
10	STATUS_UPDATE_SAL_NOTIFICATION_ERROR
11	STATUS_UPDATE_IMG_LOADING
12	STATUS_UPDATE_IMG_AUTHENTICATING

Error Code	Error Name
13	STATUS_UPDATE_IMG_PROCESSING
14	STATUS_UPDATE_CREATING_FT
15	STATUS_UPDATE_UPDATING_CODE
16	STATUS_UPDATE_UPDATING_NFT
17	STATUS_UPDATE_FLASH_CODE_PARTITION_INVALID
18	STATUS_UPDATE_FLASH_NFT_PARTITION_INVALID
19	STATUS_UPDATE_ILLEGAL_IMAGE_LENGTH
20	STATUS_UPDATE_NOT_READY
0x98	STATUS_SECURITY_VIOLATION
0xFFFFFFFF	STATUS_UPDATE_UNKNOWN