Trusted Platform Module

07/06/2018

The Trusted Platform Module is a hardware chip that can be leveraged, with software, to create a strong, robust root of trust for many things ranging from PCs, networking endpoints and embedded systems.

This hardware-based root of trust can be far more secure than software alone and can be modified to achieve the desired level of security over a range of applications fairly easily and quickly.

1.0 Purpose

This document describes Trusted Platform Module (TPM) technology, why use it, when to use it and how to interact with it using Windows 10 APIs.
The general concept of Trusted Computing refers to a set of technologies that allows for a safe run-time on trusted hardware and software. Support for Trusted Computing requires hardware support. In the context of Trusted Computing, there is a term named trusted platform. It's a set of trusted components in hardware and software that provide trusted security functions. It creates a trusted basis for running applications and provides hardware protection for sensitive data.

2.0 Hardware-based security vs. software-based security?

To quote from NIST SP800-164:

Roots of Trust (RoTs). RoTs are security primitives composed of hardware, firmware and/or software that provide a set of trusted, security-critical functions. They must always behave in an expected manner because their misbehavior cannot be detected. As such, RoTs need to be secured by their design. **Hardware RoTs are preferred over software RoTs due to their immutability, smaller attack surface, and more reliable behavior**. To support device integrity, isolation, and protected storage, devices should implement the following RoTs:

- Root of Trust for Storage (RTS)- provides a protected repository and a protected interface to store and manage keying material
- Root of Trust for Verification (RTV)- provides a protected engine and interface to verify digital signatures associated with software/firmware and create assertions based on the results
- Root of Trust for Integrity (RTI)- provides protected storage, integrity protection, and a protected interface to store and manage assertions
- Root of Trust for Reporting (RTR)- provides a protected environment and interface to manage identities and sign assertions
- Root of Trust for Measurement (RTM)- provides measurement used by assertions protected via the RTI and attested to with the RTR

3.0 Things that can be done with TPM

- Set password
- Store digital credentials such as passwords in a hardware-based vault
- Manage keys with the TPM
- Augment smart cards, fingerprint readers and fobs for multi-factor authentication
- Encrypt files and folders to control access
- Establish state information to enable endpoint integrity
- Hash state information prior to hard drive shutdown for endpoint integrity
- Enable more secure VPN, remote and wireless access
- Use in conjunction with Full Disk Encryption to restrict access to sensitive data

4.0 What is the Trusted Platform Module

The Trusted Computing Group (TCG) designed the TPM as a low-cost, mass-market security solution that provides hardware-based, security-related functions. The TPM is a secure cryptographic integrated circuit (IC) which provides a hardware-based approach to manage user authentication, network access, data protection and more that takes security to higher level than software-based security. It can also be used for cryptographic operations such as generating, storing and limiting the use of cryptographic keys and performing device authentication using the TPM’s unique RSA key. It includes multiple physical security mechanisms that make it tamper resistant and malicious software is unable to tamper with the functions of the TPM.

In many systems, the TPM can be used for validating basic boot properties before allowing network access, or for storing platform measurements, or for providing self-measurement to provide anchors of trust to hypervisors.

Computers that incorporate a TPM can create cryptographic keys and encrypt them so that they can only be decrypted by the TPM. This can help protect the key from disclosure. Each TPM has a storage root key, which is stored within the TPM itself. The private portion of a storage root key or endorsement key that is created in a TPM is never exposed to any other component, software, process, or user.

Computers that incorporate a TPM can also create a key that has not only been wrapped but is also tied to certain platform measurements. This type of key can be unwrapped only when those platform measurements have the same values that they had when the key was created. This process is referred to as “sealing the key to the TPM.” The TPM can also seal and unseal data that is generated outside the TPM. With this sealed key and software, such as BitLocker Drive Encryption, you can lock data until specific hardware or software conditions are met.

With a TPM, private portions of key pairs are kept separate from the memory that is controlled by the operating system. Keys can be sealed to the TPM, and certain assurances about the state of a system can be made before the keys are unsealed and released for use. Because the TPM uses its own internal firmware and logic circuits to process instructions, it does not rely on the operating system, and it is not exposed to vulnerabilities that might exist in the operating system or application software.

TPMs also have anti-hammering protection which is designed to prevent brute force attacks or more complex dictionary attacks, that attempt to determine authorization values for using a key. The basic approach is for the TPM to allow only a limited number of authorization failures before it prevents more attempts to use keys and locks. Providing a failure count for individual keys is not technically practical, so TPMs have a global lockout when too many authorization failures occur.

Because many entities can use the TPM, a single authorization success cannot reset the TPM's anti-hammering protection. This prevents an attacker from creating a key with a known authorization value and then using it to reset the TPM's protection. Generally, TPMs are designed to forget about authorization failures after a period of time so the TPM does not enter a lockout state unnecessarily. A TPM owner password can be used to reset the TPM's lockout logic.

5.0 TPM Specification

5.1 TPM 1.2 vs. 2.0

	TPM 1.2	TPM 2.0
Cryptography	Only allows for the use of RSA and the SHA-1 hash algorithm	Supports newer cryptographic algorithms, which can improve drive signing and key generation performance
Lockout	Implementations can vary in policy settings, which can result in support issues as lockout policies vary	Lockout policy is configured by Windows, making for a more consistent dictionary attack protection guarantee
Hardware	Parts are discrete silicon components that are typically soldered on the motherboard	Available as a discrete silicon component in a single semiconductor package, an integrated component incorporated in one or more semiconductor and as a firmware-based component running in a trusted execution environment on a general-purpose System on a Chip (SoC)
Anti-hammering	Anti-hammering protection is implemented by the manufacturer and the logic can vary widely	Windows configured the anti-hammering protection

5.2 What's new in 2.0

The TCG created TPM 2.0 with a library specification, which allows the users to choose which aspects of the TPM functionality for different implementation levels and levels of security. The specification describes all the commands/features that could be implemented and might be needed in platforms from servers to laptops to embedded systems. Each platform can implement the features needed and the level of security or assurance required. This flexibility allows the TPMs to be applied to many different embedded applications and for developers to select the appropriate capabilities for their targeted use case.

Many IoT systems include sensors and cloud processing, or virtualization. In a cloud environment, one way to implement a TPM is with a virtual TPM. The virtual TPM is part of the cloud-based environment and provides the same commands that a physical TPM would but it provides those commands separately to each virtual machine.

6.0 TPM Variations

There are three implementation options for TPMs, offering different trade-offs between cost, features, and security:

- A Discrete TPM provides the highest level of security. The intent of this level is to ensure that the device it's protecting does not get hacked via even sophisticated methods. To accomplish this, a discrete chip is designed, built and evaluated for the highest level of security that can resist tampering with the chip, including probing it and freezing it with all sorts of sophisticated attacks.

- An Integrated TPM is the next level down in terms of security. This level still has a hardware TPM but it is integrated into a chip that provides functions other than security. The hardware implementation makes it resistant to software bugs, however, this level is not designed to be tamper-resistant.

- Firmware TPM is implemented in protected software. The code runs on the main CPU, so a separate chip is not required. While running like any other program, the code is in a protected execution environment called a trusted execution environment (TEE) that is separated from the rest of the programs that are running on the CPU. By doing this, secrets like private keys that might be needed by the TPM but should not be accessed by others can be kept in the TEE creating a more difficult path for hackers.

In addition to the lack of tamper resistance, the downside to the TEE or firmware TPM is that now the TPM is dependent on many additional aspects to keep it secure, including the TEE operating system, bugs in the application code running in the TEE, etc.

7.0 Using the TPM in Windows 10

Starting with Windows 10, the operating system automatically initializes and takes ownership of the TPM, if it is enabled in the BIOS. In certain specific enterprise scenarios limited to Windows 10, Group Policy may be used to back up the TPM owner authorization value in Active Directory. Because the TPM state persists across operating system installations, this TPM information is stored in a location in Active Directory that is separate from computer objects.

After a user takes ownership of the TPM, the TPM owner can limit which TPM commands can be run by creating a list of blocked TPM commands. The list can be created and applied to all computers in a domain by using Group Policy, or a list can be created for individual computers by using the TPM Management Console (TPM MMC). Because some hardware vendors might provide additional commands or the Trusted Computing Group may decide to add commands in the future, the TPM MMC also supports the ability to block new commands.

Domain administrators can configure a list of blocked TPM commands by using Group Policy. Local administrators cannot allow any TPM commands that have been blocked through Group Policy.

Local administrators can also block commands by using the TPM MMC, and commands on the default block list are also blocked unless the Group Policy settings are changed from the default settings.

Commands can be blocked through the Local Group Policy Editor (gpedit.msc at the command line) or with the TPM MMC (tpm.msc at the command line). Blocking new commands, turning the TPM on/off, clearing keys from the TPM and changing the TPM owner password must be done through the TPM MMC.

7.1 Interacting with the TPM With Your Application

Using the Trusted Platform Module Provider through the unified management framework of Windows Management Instrumentation (WMI), developers can use the Win32_Tpm class to manage a TPM.

The Win32_Tpm class exposes 8 read-only properties and 38 methods for interacting with and managing the TPM.

The following are the exposed properties:

Name	Type	Description
IsActivated_InitialValue	boolean	Indicates whether the TPM is activated (true), or not (false). This value is stored when the class is instantiated. It is possible for the TPM to change state between the instantiation and when you check this value. To check whether the TPM is activated in real time, use the IsActivated method.

IsEnabled_InitialValue	boolean	Indicates whether the TPM is enabled (true), or not (false). This value is stored when the class is instantiated. It is possible for the TPM to change state between the instantiation and when you check this value. To check whether the TPM is enabled in real time, use the IsEnabled method.
IsOwned_InitialValue	boolean	Indicates whether the TPM has an owner (true), or not (false). This value is stored when the class is instantiated. It is possible for the TPM to change state between the instantiation and when you check this value. To check whether the TPM is owned in real time, use the IsOwned method.
ManufacturerId	uint32	The identifying information that uniquely names the TPM manufacturer. When the data is unavailable, zero is returned. This integer value can be translated to a string value by interpreting each byte as an ASCII character.
ManufacturerVersion	string	The version of the TPM, as specified by the manufacturer. When the data is unavailable, "Not Supported" is returned.
ManufacturerVersionInfo	string	Other manufacturer-specific version information for the TPM. When the data is unavailable, "Not Supported" is returned.
PhysicalPresenceVersionInfo	string	The version of the Physical Presence Interface, a communication mechanism used to run device operations that require physical presence, that the computer supports. When the data is unavailable, "Not Supported" is returned. This interface must be available to run TPM physical presence operations (the SetPhysicalPresence… and GetPhysicalPresence… methods).
SpecVersion	string	The version of the Trusted Computing Group (TCG) specification that the TPM supports. This value includes the major and minor TCG specification version, the specification revision level, and the errata revision level. All values are in hexadecimal. For example, a version information of "1.2, 2, 0" indicates that the device was implemented to TCG specification version 1.2, revision level 2, and with no errata. When the data is unavailable, "Not Supported" is returned.

The following are the methods exposed:

Name	Description
AddBlockedCommand	Adds a TPM command to the local list of commands blocked on Windows.
ChangeOwnerAuth	Changes the TPM owner authorization value.
Clear	Resets the TPM to its factory-default state.
ConvertToOwnerAuth	Converts a user-provided passphrase to a 20-byte owner authorization value that can be used to interact with the TPM.
CreateEndorsementKeyPair	Creates a 2048-bit endorsement key pair on the TPM.
Disable	Allows the TPM owner to disable the TPM.
DisableAutoProvisioning	Disables auto provisioning of the TPM if it is currently enabled.
Enable	Allows the TPM owner to enable the TPM.
EnableAutoProvisioning	Enables auto provisioning of the TPM if it is currently disabled.
GetOwnerAuth	Gets the owner authorization information for a TPM if one is available in the registry.
GetPhysicalPresenceConfirmationStatus	Indicates if confirmation from a physically present user is required for a given physical presence operation.
GetPhysicalPresenceRequest	Gets and returns the pending TPM physical presence operation. Use the SetPhysicalPresenceRequest method to request an operation.
GetPhysicalPresenceResponse	Gets and returns the results from a TPM physical presence operation that was performed.
GetPhysicalPresenceTransition	Indicates the user action that is needed to perform a TPM physical presence operation.
GetSrkADThumbprint	Gets the Storage root key thumbprint for a given modulus of the public portion of the TPM Storage Root Key.
GetSrkPublicKeyModulus	Gets the modulus of the public portion of the TPM Storage Root Key.
ImportOwnerAuth	Imports the owner authorization information for a TPM that is already owned into the operating system registry.
IsActivated	Indicates whether the TPM is activated.
IsAutoProvisioningEnabled	Indicates if auto provisioning of the TPM is enabled.
IsCommandBlocked	Indicates whether the TPM command can run on this operating system.
IsCommandPresent	Indicates whether a TPM command is supported by this computer.
IsEnabled	Indicates whether the TPM is enabled.
IsEndorsementKeyPairPresent	Indicates whether the TPM has an endorsement key pair.
IsOwned	Indicates whether the TPM has an owner.
IsOwnerClearDisabled	Indicates whether the TPM owner can clear the TPM.
IsOwnershipAllowed	Indicates whether a TPM owner can be installed.
IsPhysicalClearDisabled	Indicates whether a TPM physical presence operation can clear the TPM.
IsPhysicalPresenceHardwareEnabled	Indicates whether this computer supports a dedicated hardware path to signal physical presence.
IsReady	Indicates whether the TPM is ready for use.
IsReadyInformation	Indicates whether the TPM is ready and provides additional information on the state of the TPM.
IsSrkAuthCompatible	Indicates whether the Storage Root Key (SRK) authorization is compatible with Windows.
Provision	Attempts to provision the TPM to a completely ready state and will take the ownership of TPM if it is not already owned.

RemoveBlockedCommand	Removes a TPM command from the local list of commands blocked by Windows.
ResetAuthLockOut	Resets the time-out period or other mechanism that TPM manufacturers implement to protect against dictionary attacks on the TPM.
ResetSrkAuth	Resets the Storage Root Key (SRK) authorization value to be compatible with Windows.
SelfTest	Performs a self-test of the TPM and returns the result.
SetPhysicalPresenceRequest	Requests a TPM physical presence operation to run.
TakeOwnership	Installs an owner for the TPM.

8.0 Additional Sources of TPM information

The TPM standard	https://trustedcomputinggroup.org/work-groups/software-stack/ and https://trustedcomputinggroup.org/resource/tcg-software-stack-tss-specification/
Free eBook on TPM	https://link.springer.com/book/10.1007%2F978-1-4302-6584-9
Microsoft TPM software stack	https://github.com/Microsoft/TSS.MSR
Microsoft TPM recommendations	https://docs.microsoft.com/en-us/windows/security/information-protection/tpm/tpm-recommendations
Microsoft TPM toolkit	https://www.microsoft.com/en-us/download/details.aspx?id=52487&from=http%3A%2F%2Fresearch.microsoft.com%2Fen-us%2Fdownloads%2F74c45746-24ad-4cb7-ba4b-0c6df2f92d5d%2F
Microsoft TPM Base Services	https://docs.microsoft.com/en-us/windows/desktop/TBS/tpm-base-services-portal
Old lib for TPM called TrouSerS	http://security.polito.it/trusted-computing/trousers-for-windows/
Open source TPM tools	https://github.com/tpm2-software/tpm2-tools
Excellent Primer on TPM use	https://blog.hansenpartnership.com/using-your-tpm-as-a-secure-key-store/
NIST SP800-164 Hardware Security in Mobile Devices	https://csrc.nist.gov/csrc/media/publications/sp/800-164/draft/documents/sp800_164_draft.pdf
Article on TPM	https://www.electronicdesign.com/embedded/standardizing-trust-embedded-systems