Intel ® Trusted Execution Technology Architectural Overview

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.
Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined.” Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information
The Intel® IA-32 architecture processors (e.g., Pentium® 4 and Pentium® III processors) may contain design defects or errors known as errata. Current characterized errata are available on request. Intel is a registered trademark of Intel Corporation and its subsidiaries in the United States and other countries. *Other names and brands may be claimed as the property of others.
COPYRIGHT © 2003, INTEL CORPORATION

Trusted Execution Technology Overview

Today's IA-32 PCs

Today's IA-32 PCs are typically linked to the Internet or Intranet and are widely used for the creation, storage, editing and sharing of personal, corporate/government, information. This enables the IA-32 PCs to play a critical role in the PC user's ability to conduct business, collaborate, access confidential data, and conduct electronic transactions with third parties.

A key reason for the IA-32 PCs being so widely used is the open nature of the platform. The architecture's interfaces are well documented and understood, allowing for a wide variety of powerful software and hardware capabilities that deliver value to corporations and end users.

The New Challenge

As the PC plays this critical role, it must operate on highly sensitive and confidential information on a regular basis. This information is typically of high value and an attractive target for hackers.

Both the sophistication and frequency of computer attacks on the client have grown steadily and experts expect that these attacks will continue to grow into the future. While firewall software, virus scanners, encryption software and other security software offer some protection; these software solutions can only do so much to protect against other, possibly malicious, software that is running at the same or higher privilege level. Hence, it is likely that software-only protections will be circumvented in the future. It is this vulnerability that is driving the need for new and innovative solutions to be developed.

Intel's Safer Computing Initiative

Intel is addressing this challenge with its Safer Computing Initiative. The goal of Safer Computing is to advance platform security and provide a hardware hardened framework for continued innovation in protection of sensitive information. A key element of this Safer Computing vision is the technology code-named Trusted Execution Technology. The goal is to protect the PC from software based attacks while maintaining the manageability, performance, versatility and backward compatibility that is critical to today's IA-32 PC. Trusted Execution technology provides hardware support for the creation of parallel, protected environments that enable a much higher level of protection for code execution and confidential information in the new software environments. The remainder of this white paper will focus on key elements of the Trusted Execution Technology Architecture.

Trusted Execution Technology Summary

Trusted Execution Technology is a set of enhanced hardware components designed to help protect sensitive information from software-based attacks. Trusted Execution features include capabilities in the microprocessor, chipset, I/O subsystems, and other platform components. When coupled with an enabled operating system and enabled applications, Trusted Execution Technology can help protect the confidentiality and integrity of data in the face of these increasingly hostile security environments. This technology provides a versatile, general-purpose safer computing environment capable of running a wide variety of operating systems and applications. Intel is initially targeting Trusted Execution Technology for applications in the business segment.

Trusted Execution Technology Architecture Overview

The Trusted Execution Technology platform delivers a number of key capabilities to the platform. These capabilities, when combined, deliver the protections that will be critical to evolve the IA-32 platform. The capabilities include:

Protected Execution: Provides applications with the ability to run in isolated protected execution environments such that no other unauthorized software on the platform can observe or compromise the information being operated upon. Each of these isolated environments has dedicated resources that are managed by the processor, chipset and OS kernel.

Sealed storage: Provides for the ability to encrypt and store keys, data or other secrets within hardware on the platform. It does this in such a way that these secrets can only be released (decrypted) to an executing environment that is the same as when the secrets were encrypted. This helps prevent attacks exploiting the vulnerability where the encrypted data has been transferred to other platforms either for normal use (thereby become decrypted) or for malicious attack.

Protected Input: Provides a mechanism that protects communication between the keyboard/mouse and applications running in the protected execution environments from being observed or compromised by any other unauthorized software running on the platform. For USB input, Trusted Execution can do this by cryptographically encrypting the keystrokes and mouse clicks with an encryption key shared between a protected domain's input manager and an input device. Only applications that have the correct encryption key can decrypt and use the transported data.

Protected graphics: Provides a mechanism that enables applications running within the protected execution environment to send display information to the graphics frame buffer without being observed or compromised by any other unauthorized software running on the platform. This is done by creating a more protected pathway between an application or software agent and the output display context (such as a window object).

Attestation: Enables a system to provide assurance that the Trusted Execution's protected environment was correctly invoked. It also provides the ability to provide a measurement of the software running in the protected space. The information exchanged during an attestation function is called an Attestation Identity Key credential and is used to help establish mutual trust between parties.

Protected Launch: Provides for the controlled launch and registration of the critical OS and system software components in a protected execution environment.

Trusted Execution Technology Hardware Overview

Implementation of a Trusted Execution Technology-enabled platform requires a number of hardware enhancements (see Figure 1). Key hardware elements of this platform are:

Processor: Extensions to the IA-32 architecture allow for the creation of multiple execution environments, or partitions. This allows for the coexistence of a standard (legacy) partition and protected partition, where software can run in isolation in the protected partition, free from being observed or compromised by other software running on the platform. Access to hardware resources (such as memory) is hardened by enhancements in the processor and chipset hardware. Other processor enhancements include: (1) event handling, to reduce the vulnerability of data exposed through system events, (2) instructions to manage the protected execution environment, (3) and instructions to establish a more secure software stack.

Chipset: Extensions to the chipset deliver support for key elements of this new, more protected platform. They include: (1) the capability to enforce memory protection policy, (2) enhancements to protect data access from memory, (3) protected channels to graphics and input/output devices, (4) and interfaces to the Trusted Platform Module [Version 1.2].

Keyboard and Mouse: Enhancements to the keyboard and mouse enable communication between these input devices and applications running in a protected partition to take place without being observed or compromised by unauthorized software running on the platform.

Graphics: Enhancements to the graphic subsystem enable applications running within a protected partition to send display information to the graphics frame buffer without being observed or compromised by unauthorized software running on the platform.

The TPM v. 1.2 device: Also called the Fixed Token, is bound to the platform and connected to the PC's LPC bus. The TPM provides the hardware-based mechanism to store or 'seal' keys and other data to the platform. It also provides the hardware mechanism to report platform attestations.

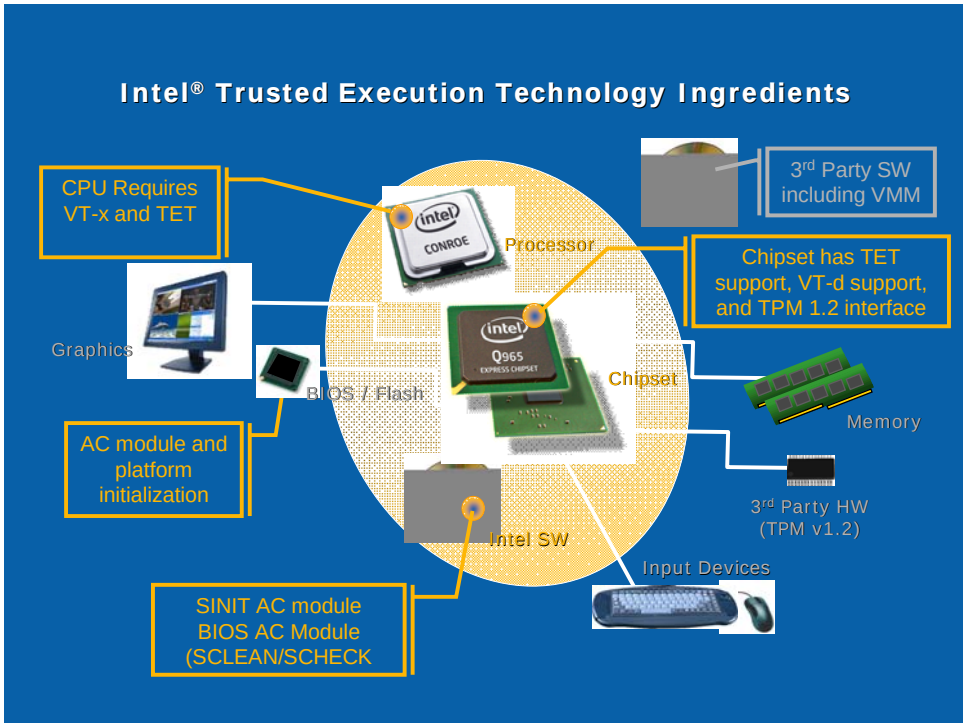

Figure 1: Key Hardware Enhancements

The Trusted Execution Technology Protection Model

Trusted Execution Technology provides a set of capabilities that can be utilized in many different operating environments (Figure 2). One proposed architecture provides a protection model similar to the following:

A **standard partition** that provides an execution environment that is identical to today's IA- 32 environment. In this environment, users will be able to run applications and other software just as they do on today's PC. The standard partition's obvious advantage is that it preserves the value of the existing code base (i.e. existing software does not need modification to run in the standard partition) and potential future software that is less security conscious. Unfortunately, it also retains the inherent vulnerabilities of today's environment.

A **protected partition** provides a parallel and co-existing environment that will run hardened software that makes use of the hardware-based security foundation enabled by Trusted Execution Technology. Within this environment, different applications can run in isolation, free from being observed or compromised by software running in the standard partition and other applications running in the protected partition. A protected partition requires a Trusted Execution-capable processor, chipset, and a domain manager to provide domain separation. The TPM device protects secrets stored

in a Trusted Execution platform when the protected partition is not running. The Trusted Execution Technology protection model can support any domain manager, and future, enhanced OS kernel.

Applications can be written to execute within the protected partition or, in most cases, make use of both partitions. In the latter case, much of the application code could still reside within the standard partition (this code manages the human interface and handles I/0) and services written to manipulate secure or sensitive information, would move to modules written for the protected partition.
The Trusted Execution Technology protected partition is hardened against software attacks because:

It's domain separation allows hardened software to run in memory pages that are protected from viewing or modification by unauthorized applications.

It's memory protection prevents DMA engines from reading or modifying protected memory pages.

It's protected graphics processes application data from the protected partition such that it is not visible either to software in the standard partition or other software running in the protected partition.

It provides a trusted channel to keyboard and mouse that prevents keyboard snooping nd/or modification of user's keystrokes or mouse movements.

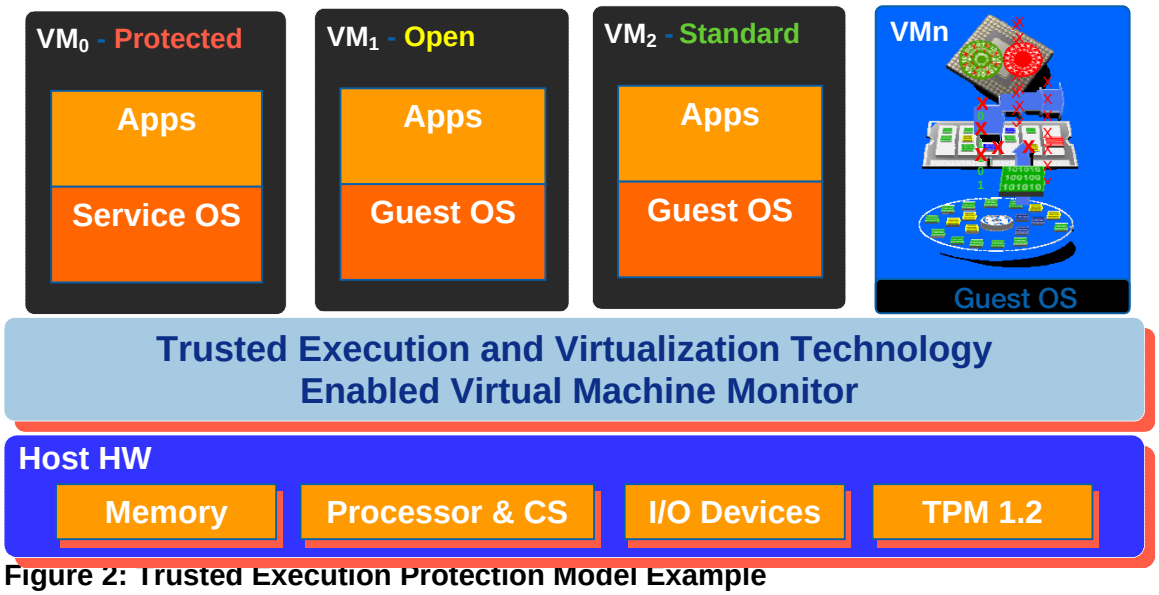

Figure 2: Trusted Execution Protection Model Example

More Architectural Details on a Protected Environment

Booting up a protected partition

Trusted Execution Technology supports the ability to launch protected environments without platform reboot, and legacy software is able to run unmodified in a standard partition. Typically, a protected partition is launched by a request to an appropriate software component that is aware of the Trusted Execution Technology. In response to such a launch request, memory spaces are allocated for the protected partition and marked protected. The domain manager is loaded into the designated memory spaces and registered by an authenticated code module (AC). The launch of a protected execution environment occurs in stages. These are designed to ensure that the processor and the chipset recognize and participate in the launch, that all participants launch the same environment, and that there is no interference with the launch process. The stages include:

1. Ordinary software running on a Trusted Execution Technology processor executes a new SENTER instruction to initiate the launch process. This new instruction triggers a sequence of hand-shakes. At

the conclusion of this first round of hand-shakes, the processor and chipset are ready to be brought into a protected environment.
2. The processor loads an authenticated code module into internal private memory, authenticates it, registers its identity in a platform configuration register (PCR) in the TPM, and then invokes it. The AC checks that there is no improperly configured hardware, enables memory protection for the proposed domain manager, records the identity of the domain manager in a TPM PCR, then transfers execution control to the domain manager.

Exiting a protected partition

When a protected partition is no longer needed, TRUSTED EXECUTION TECHNOLOGY supports the take-down of the protected environment. This is again performed in stages.
1. The domain manager is responsible for cleaning up the protected partitions, ensuring that no secrets are left behind in either memory or registers. These actions include re-sealing secrets to be placed in persistent storage, and scrubbing the contents of protected partition pages.
2. The domain manager invokes a new instruction SEXIT to exit. The SEXIT instruction triggers a sequence of hand-shakes and then exits the protected environment.

Special event protections

Most normal system events, including exceptions and interrupts, are handled within the protection boundaries established by the partitions. Such events may be serviced within the partition, or may be trapped to the domain manager for service (depending on the nature of the event).
However, certain abnormal system events can potentially result in a transfer of control to agents running outside a protected partition, creating a potential venue of attack to confidential data residing in memory. Trusted Execution Technology processors and chipsets include hardware support that can detect and handle these events in a manner that does not permit the exposure of secrets or any tampering with protected execution.
For example, certain system conditions could force a system reset without permitting the domain manager to first remove secrets from memory. Trusted Execution Technology hardware protections provide that, following an unanticipated reset, memory that might contain secrets is scrubbed before it can be accessed by un-trusted software.

More on Attestation and Trust

Unsealing and sealing secrets

Trusted Execution Technology provides the capability to seal and unseal secrets with the assistance of a TPM v.1.2 device. This capability ensures that a secret generated by one domain manager or environment is not available to another domain. The basis of this protection lies with the TPM's *Storage Root Key (SRK)*, a public/private key pair. The SRK private key never leaves the TPM. Any data encrypted with the SRK public key can only be decrypted by the corresponding SRK private key. As the private key never leaves the TPM, only this TPM may decrypt this data. The TPM provides a SEAL operation, which permits data and a list of PCRs to be encrypted into a blob using a TPM storage key. The resulting encrypted blob may be stored anywhere. A corresponding UNSEAL operation decrypts the blob, but will not expose decrypted data unless the saved PCR values match current PCRs. This operation permits a domain manager to seal data to the current PCR values representing its current protected environment; the resulting blob can only be unsealed to expose the data if the identical domain manager is running.
Typically, a domain manager generates its own bulk encryption key, to be used in software, and seals this key using the TPM. The bulk encryption key is then used to encrypt all secrets managed by this domain manager, and may also be used to encrypt secrets for the applications running in the protected partition.

Establishing Initial Trust

A sealed secret can only be unsealed and accessed by the same domain manager environment. If a secret known only to a user was sealed to an environment that the user chose to trust, then if this

secret can be re-displayed the user knows the same trusted environment is currently running. A similar method uses a secret shared with a remote agent, allowing the remote agent to know that the same trusted environment is currently running. But that leaves the question of how the user or remote agent determines that the environment should be trusted before a shared secret exists. To put the question more succinctly: how do we determine initial trust?

Trusted Execution Technology supports an optional, verifiable reporting mechanism, called attestation. Attestation permits either the user or, optionally, a remote agent to measure the currently running environment using measurement and reporting mechanisms supported by the TPM. Based upon these reported measurements, the user or remote agent may use this information to decide whether to trust the current platform environment.

For a remote agent, the attestation process involves standard cryptographic methods. A remote agent generates a random value (called a nonce or challenge), and sends it to the system to be tested. At that system, the TPM creates a record containing the nonce and the current PCR values (which represent the currently running domain manager environment). The TPM signs this record with its private key and the signed record is returned to the remote agent, along with the TPM's public key and credentials. The remote agent may examine the credentials to determine that this public key does; in fact, represent a real TPM, then uses the public key to verify the signature on the record and, then extracts the data from the record. The extracted data may now be checked against various lists to determine if this is an environment acceptable to the remote agent. Attesting the environment of a local machine to a human user is more challenging, given that most humans cannot perform cryptographic calculations in their heads. There are at least three methods a user may choose from to identify the local machine environment and make a trust decision:

1. Assuming that a system is in its original state (as delivered from an OEM that a user trusts), the user may simply choose to trust this initial configuration. The user would be advised to create a secret (e.g. a favorite phrase or quote) to be sealed to this environment. As long as this secret can be displayed to the user on subsequent boots, the user has confidence that the same environment is running.
2. A portable token capable of cryptographic operations may be used to act as a "remote agent like" proxy for the user. This token can be loaded with measurements of valid environment configurations at the local retailer. Such a portable token can then be connected to the PC and performs attestation of the user system in a manner identical to that described for remote agents. The portable token could then report pass/fail.
3. The user may request that a remote agent perform attestation of the system. However, this leaves the problem of how the remote agent safely reports this information back to the user, given that the user cannot (yet) trust the software environment on the system. There are at least two methods of achieving this:

    If the user has a portable token, the remote agent's results can be communicated using cryptographically secured protocols to the portable token which displays the result for the user.

    The remote agent provides the results "out-of-band", perhaps using an automated phone menu or mail.

Summary

Security threats targeted towards the personal computer are expected to increase in both sophistication and frequency. In order to protect sensitive information against these threats, Intel plans to enhance the IA-32 PC architecture with Trusted Execution Technology. This technology will provide a significant step in the future advancement of the platform by providing a hardware-hardened security framework while preserving legacy software environment. The innovations of Trusted Execution Technology include domain separation, protected execution, sealed storage, a trusted channel to graphics and input devices, mechanisms for authenticating the launch of a protected environment and attestation of platform identity. These building blocks will help enable a new, open,

software architecture for security that can help protect a user’s or corporation’s sensitive information from software based attacks.