TrustCube: An Infrastructure that Builds Trust in Client

Zhexuan Song · Jesus Molina · Sung Lee · Houcheng Lee
Seigo Kotani · Ryusuke Masuoka

Fujitsu Laboratories of America
8400 Baltimore Avenue, Suite 302
College Park, Maryland 20740
USA
{zhexuan.song | jesus.molina | sung.lee | houcheng.lee
seigo.kotani | ryusuke.masuoka}@us.fujitsu.com

Abstract

In a client-server environment, typically a lot of sensitive data and/or processes (for clients as well as for the server) are maintained at the server. In order to protect the integrity of the server and prevent leakage of data to unauthorized entities, it is important to make sure that only the authorized person with properly configured authorized platforms can gain the access to the server.

In this paper, we introduce the TrustCube infrastructure. The TrustCube infrastructure is an end-to-end infrastructure that offers measurements of essential elements of clients, including person (or identity), the platform, and the environment; thus, enabling the capability for service providers to make informed decision based on the certifiable report of measurements. Under this infrastructure, a server can accurately evaluate the risk of dealing with a particular client, and handle the requests coming from that client correspondingly.

1 Introduction

Recently, more and more companies move their sensitive data into highly protected servers, and clients are accessing the data over the network. One of the major challenges is to prevent the data from leaking to wrong entities. It requires servers to block requests from unidentified visits and to control the access of authenticated users. One widely used authentication mechanism is username/password: only after a person provides the correct username and password combination, could this person access sensitive information.

However, there are still some issues that need to be addressed. First issue is about the trustfulness of the authentication mechanism. It is widely believed that username/password is a weak user authentication mechanism [Gar02], in the sense that even if the correct username/password combination is provided, it is still difficult to prove that the request is from the rightful owner of that username/password combination. Alternatively, using biometric data is considered to be a much better way to identify a person [AHGBEA07], but using biometric data still comes with its own limitation. One major concern is about its privacy. Unlike username/password, it is very difficult for a person to modify her own biometric characteristics. So once the biometrics data is compromised, there is no easy way to compensate for it.

D. Gawrock, H. Reimer, A.-R. Sadeghi, C. Vishik (Editors): Future of Trust in Computing, Vieweg (2008), 1-12

Thus it is always wise not to give the native biometrics data directly to any third parties; or, even better, not to let the data leave the client at all. We will revisit this issue later.

Secondly, after a service provider is persuaded that the request is from the authorized person, how could the service provider trust the platform the person is using? Maybe the person uses a public terminal at internet kiosk, or she plugs her USB storage device to the platform and is making copies of sensitive data files, or she connects a printer to the platform and is printing the documents out. All these cases might be unacceptable in certain applications, but the service provider needs a mechanism to find it out.

Thirdly, even if a service provider is persuaded that the request is from the authorized person, and the person is using the designated platform, without connecting any illegal peripheral devices, how could the service provider trust the environment (OS, applications, and so on) of the platform? She might already be a victim of one of countless worms and/or viruses, or she is running notorious P2P software (e.g. Winny [Fre06]), or a key logger is silently recording all her keystrokes behind the scene.

All these issues, how to trust a person, a platform and the environment of the platform, are becoming more and more important in recent days, as protecting the sensitive data becomes one of the top priorities for government and enterprises alike.

In this paper, we will present our new TrustCube infrastructure. It extensively uses the latest Trusted Computing [TCG08] technologies and addresses well on the previous three issues. The main focus of the TrustCube infrastructure is NOT to directly prevent a bad person (and/or) from using an unknown platform (and/or) that is running malicious software, which is already believed to be very hard. The main focus is to offer various measurements of essential elements of clients, including the person, the platform, and the environment, and provides the capability for *service providers* to make informed decision based on the certifiable report of measurements.

The word TrustCube or T^3 means that our infrastructure brings "trust" back to a person, a platform and environment of the platform.[1]

This paper is organized as follows. In the next section, we will introduction related works. Then we will present our TrustCube infrastructure and give a complete workflow to demonstrate how the system works. Next, we will raise some discussion about TrustCube, and finally we will conclude the paper and give some possible future works.

2 Related Works

In this section, we will introduce works that are related to the TrustCube infrastructure. They are Trusted Computing technologies, namely Trusted Platform Module (TPM), Trusted Network Connection (TNC), chain of trust, and remote attestation.

2.1 Trusted Platform Module (TPM)

The Trusted Platform Module (TPM) is a hardware chip that offers facilities for *remote attestation* and the *secure generation of cryptographic keys*, in addition to other capabilities such as hashing, pseudo-random number generation and monotonic up-counters. The TPM is usually deployed as part of a com-

1 Please note that the TrustCube infrastructure is not limited to three aspects and can be expanded to include other aspects as necessary.

puter system's motherboard. Currently, it is estimated that around 230 million PCs are equipped with TPM by year 2009 [Kay05].

Remote attestation allows service provider to detect changes to client's platform. It works by honestly recording changes from hardware to software of the platform, and using Platform Configuration Register (PCR) extension to protect the client-side modification of the record. By looking at the record, along with the corresponding PCR values, service providers are able to know 1) whether the record is tampered, and 2) if not, whether the platform has been changed. The difficult decision about remote attestation is what to measure/verify, which we will discuss in a separate subsection.

Secure generation of cryptographic keys means that the public/private key generation functions are implemented within the TPM. Some private keys (such as Endorsement Key (EK)) never leave TPM and only authorized persons can use keys to encrypt/decrypt and sign messages. Even if a platform is lost or stolen, those keys can never be used by unauthorized persons. It is believed that using TPM key generation function is more secure that similar software solution.

Since the TPM chip only supports limited storage capabilities, inactive keys are encrypted and moved off-chip. Management of the key is performed externally by a Key and Credential Manager (KCM) [TPM08]. Keys are stored in a hierarchy structure. The root of the hierarchy tree is the Storage Root Key (SRK), which is created when creating a new platform owner. Each key has an attribute called migratable or non-migratable. A key is migratable means that KCM can back up the key and later recover it in the same or a different TPM.

Optionally, a key can be protected by a secret. Once a key is protected, only when a person provides the right secret, can she load the key into the TPM and use it. This procedure is called "key authorization." Currently, the existing key authorization mechanism for a TPM is based on a 20 byte shared secret, and is usually created from the hash of a passphrase. Trusted Computing Group has formed the Authentication Working Group to work on a solution that supports a wider range of authentication mechanisms.

In the TrustCube infrastructure, we use bioinformatics to authenticate a person to the TPM. Since the corresponding TCG specification is not available yet, a customized approach is adopted. We will modify our approach when the official specification is published in the future.

2.2 Trusted Network Connect (TNC)

Trusted Network Connect (TNC), an initiative of TCG, addresses and attempts to provide network access control that meets security requirements through open-source, non-proprietary standards. In order to ensure interoperability and compatibility with the existing network infrastructure, TNC is designed to utilize existing industry standards and protocols such as Extensible Authentication Protocol, Transport Layer Security, and RADIUS. The TNC architecture supports commonly used access mechanisms such as VPN, SSL, dial-up remote access and other networking technologies including wired and wireless networks and 802.1x infrastructure.

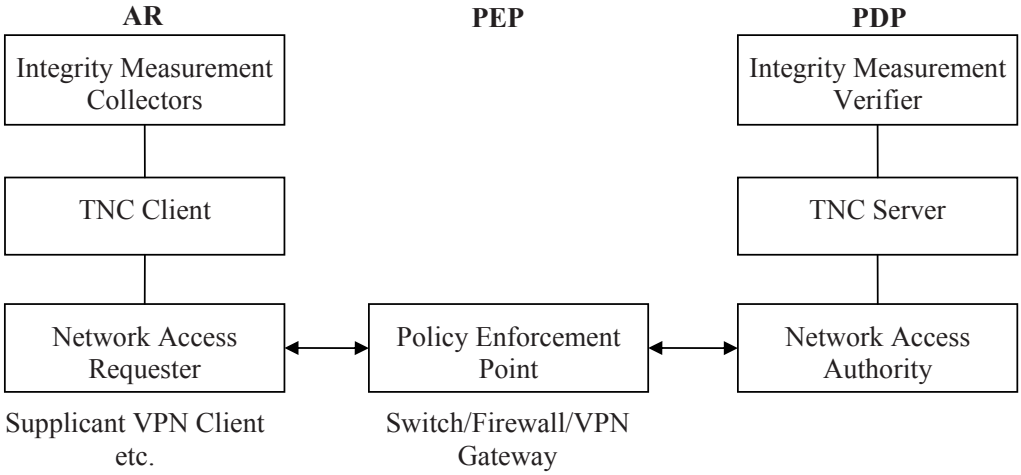

Figure 1: TNC Architecture: This diagram is based on the TNC Architecture Specification

There are three entities in TNC (Figure 1): an Access Requester (AR), a Policy Enforcement Point (PEP), and a Policy Decision Point (PDP). When an AR makes an attempt to access a protected network behind PEP, the access request is passed through an integrity evaluation process in order to determine what level of access should be granted, if any. Based on the measurements collected at the AR and the policy configuration, the PDP makes a decision. The PEP, usually a network access device such as a switch or wireless access point, enforces the decision made by PDP by granting the appropriate access to the AR. The TNC architecture allows for an option to include platform measures such as PCR values from Trusted Platform Module (TPM) of the AR.

In the TrustCube infrastructure, the procedure of building secure communication tunnel between client and server strictly follows the TNC specification.

2.3 Chain of Trust

A chain of trust is established by validating each component of hardware of software from the bottom up. It ensures that only trusted hardware and software can be used while the complete system is still flexible. At the bottom of a chain of trust is the "root of trust component." By TCG definition, root of trust (component) is "a component that must always behave in the expected manner, because its misbehavior cannot be detected."

If a component in the chain cannot validate the component at the next level, we called the chain is broken. When a chain is broken, the component at the next level and the ones at higher levels cannot be trusted, even if some validating result "seems to be correct." In this sense, to trust a measurement report about a high level component, we cannot only look at this level. In fact, we must get a complete report about each component following the chain of trust. Only if all components before this are trusted, can we really start to discuss whether or not this component is trusted.

We will discuss the chain of trust in the TrustCube infrastructure later.

2.4 Remote Attestation

As we discussed previously, remote attestation is one of the key functionalities supported by Trusted Computing technologies. An important research problem consists on determining what data should be used for remote attestation. Several attestation techniques have been proposed.

SignaCert Enterprise Trust Service (ETS) [Sig08] adopts a static scanning based solution. It is a server that can check whether the environment of a platform contains any unknown files. The basic idea is quite simple. At the server (ETS), a huge database is maintained. The database contains a long list of known files and their snapshots (digest values of files). From the client side, the current files in the environment and their snapshots are calculated and sent to the server. The server will compare the submission with the database and check whether the snapshot is correct. If the file is not in the database, or the snapshot of a file is not correct, the file is considered unknown.

TrustedVM, proposed by Vivek Haldar et al. [HVC04], is a dynamic, platform independent solution. It is a virtual machine that can dynamically retrieve enforcement and security policies from the server and execute the attestation on the programs that are running in the virtual machine. The drawback is that this solution will greatly slow down the regular operations in the virtual machine.

The Integrity Measurement Architecture (IMA) [Sai04] for Linux is implemented as part of the operating system. In IMA, a modified operating system (OS) kernel measures all applications, drivers and libraries that are loaded by the OS for verification.

In the current TrustCube infrastructure, we adopt a hybrid solution: SignaCert ETS for static scanning and IMA for dynamic measuring. More discussion will be given later. The bootloader needs to be also modified in order to maintain the chain of trust. For the current TrustCube infrastructure we use TrustedGrub [Tgrub06], which still contains some security problems, as outlined in [Kau07]. In newer version of the TrustCube infrastructure, we plan to support other bootloader such as OSLO [Kau07].

3 TrustCube Infrastructure

In this section, we will give details about the TrustCube infrastructure. First, we will introduce the components appeared in the TrustCube infrastructure, followed by a set of workflows that the TrustCube infrastructure uses to fulfill certain tasks. Finally we will present some discussion.

3.1 TrustCube Infrastructure Architecture

The TrustCube infrastructure includes modules on both the client and the server side. A general architecture is depicted in Figure 2. In this diagram, we assume that a person is using a web browser to view sensitive data in a document server (the web browser and the document server are connected in a dotted line). The diagram is similar for other types of services. In Figure 2, the solid lines mean that the connected components have direct communication while the dash lines means the connected components have logical connection or are corresponding components in client and server.

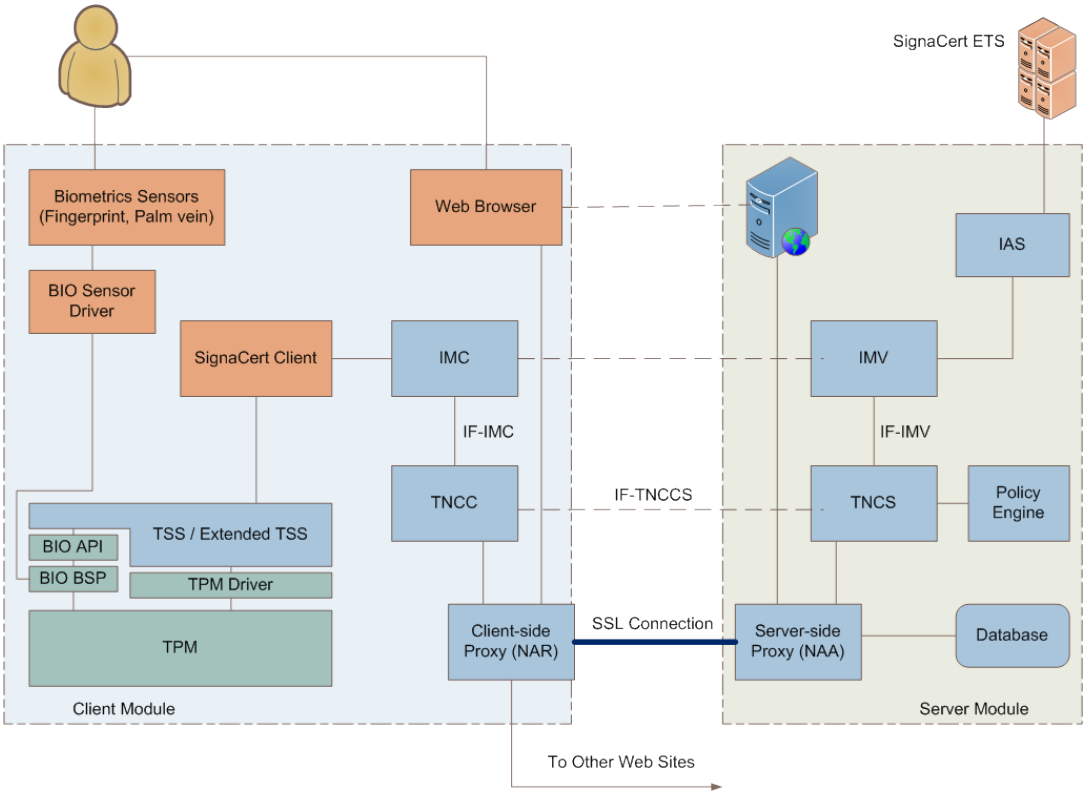

Figure 2: General architecture of the TrustCube infrastructure

The components are described as follows:

- **IMC/IMV, TNCC/TNCS** are standard TNC components. We implemented TNC 1.2 specification using the latest Java binding [TNC08].
- **Biometrics Sensors** are devices that collect person's biometrics characteristics.
- **BIO Sensor Driver, BIO API (Application Programming Interface), and BIO BSP (Biometric Service Provider)** are standard Biometrics application components [Bio05].
- **TPM, and TPM Driver** are standard TCG components.
- **TSS (TCG Software Stack) / Extended TSS**. TSS is one TCG standard component. We extended the existing TSS version 1.2 [TSS08] and added a set of functions to register/handle biometric data.
- **SignaCert client** is a module that collects the snapshots of client-side environment and generates an XML report.
- **Client-side proxy (NAR)** is one implementation of NAR. The component fulfills two functions: it initializes the TNC handshake with server; it serves as a simple proxy that sends server related requests to NAA over SSL tunnel, and relay other requests to external web sites.
- **Server-side proxy (NAA)** is one implementation of NAA. The component also fulfills two functions: it handles TNC handshake requests; it parses the server related requests and relay them to the server.
- **Database** supports Server-side proxy (NAA).

- **Policy Engine** supports TNCS in making decisions about whether or not to give access to a client and/or under which trust level.
- **IAS (integrated authentication service)** helps in the TNC handshake procedure. While a client starts a TNC handshake, it will send the complete report about person, platform and environment. IAS verified the consistency and the correctness of the report. However, IAS will NOT make any decision about whether the request from the client should be approved or not.
- **SignaCert ETS** helps IAS to verify the correctness of the environment section of the report.

3.2 TrustCube Infrastructure Workflow

Before a client is activated, it must complete a one-time registration procedure, which is called "registration phase." Before a client can access any sensitive data in a server, it needs to pass person/platform/environment authentication, we call this procedure "authentication phase." And finally the person starts to work on the sensitive data in the server, and we call this period "operation phase." Figure 3 shows the phase change state diagram.

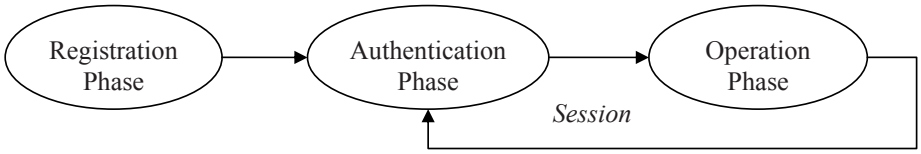

Figure 3: Phase changes in the TrustCube infrastructure

Please note that the communication between the client and the server in all three phases must be secure and must guarantee integrity and authenticity. This can be done using various cryptographic protocols, such as SSL.

3.2.1 Registration Phase

During the registration phase, the information about person, platform and environment is collected and stored in IAS for future authentication purpose. The registration sequence is person first, platform next. Environment registration is optional.

During the person registration, the person's biometric reference data, such as fingerprints, eye retinas and irises, and palm vein patterns, are collected, encrypted and stored in the client platform as a BLOB. Meanwhile, a hash function, such as SHA-1 is applied on the BLOB and the output is used as a secret. Next, the system administrator generates a signing key, which we called person's identity key, or K_{PI}, if the person does not have her identify key created before. If the user already has her identity key in another client, the key will be migrated to this platform. The key, newly created or migrated, is attached under the SRK, and protected by the secret generated from the person's biometric reference data.

If K_{PI} is newly created, it needs to be certified by a Certificate Authority (CA) and a copy of the certificate is stored in IAS for future verification. This step is similar to the AIK registration defined in [SKAE05]. However in the TrustCube infrastructure, we dropped steps related to Privacy CA because the CA we used is controlled by us instead of a public CA mentioned in the specification.

Two steps are involved in the platform registration. The first step is to create and register a signing key which we called platform key, or K_p. K_p is certified by a CA and a copy of the certificate is stored in IAS for future verification. K_p is *not* protected by any secret and any person who has access to the platform can access this key and use it to sign a report. This step needs to be done only once for each platform.

Please note that K_{PI} is migratable and K_P is non-migratable.

The other step of the platform registration is to collect and register certain PCR values (PCR 0 – PCR 6) as the platform's hardware snapshots. This is crucial since it represents the components in the chain of trust before OS. Later, when the hardware setting of the platform is modified, or new hardware is added, we must re-do this step.

The environment registration is to harvest the snapshots of files that will be used in the client. This registration is optional, in the sense that only when the software that will be run in the platform is not known beforehand, is it necessary to register them. When we talk about registering software, we mean to create snapshots (e.g. SHA1 hash value) of all files the software contains. The list of <file, snapshot> pairs is collected, and then added to the white list. For an enterprise setting when multiple platforms are running similar software, this step needs to be done only once.

After the registration, the client is ready to use.

3.2.2 Authentication Phase

The TrustCube infrastructure uses a token based authentication system. Authentication phase happens at the beginning of each session. After a successful authentication, a token is generated and assigned to the client. The token will be attached with any further requests from the same client.

Generally speaking, the authentication phase is a TNC handshake procedure. The procedure is triggered when a person launches the client-side proxy (NAR). As the TNC specification specifies, an IMC will be loaded. This IMC does the following things

- It prompts the person to scan her biometric characteristics (e.g. fingerprint, palm vein), and compares the scanned sample with the biometric reference data using BIO API [Bio05]. If the match fails, the handshake procedure halts. If the scanned data successfully matches the record, using the same hash function we used in the registration phase on the biometric reference data, the secret is calculated and the person unlocks K_{PI}. This step has a potential loophole if the biometric reference data and the hash function are retrieved by a bad person through other channel and the secret is leaked. A possible solution is to store the biometric reference data in hardware (such as advanced security chip or smart card) and to do the matching and secret calculation in the same hardware. Currently, we are working on the smart card approach.
- It retrieves the dynamically scanned snapshots collected by the operation system.
- It launches a separate thread to run a static scanning on the environment. This thread utilizes the SignaCert client module to harvest the snapshots of files at certain directories. The selection of directories to be scanned is defined in a policy file stored in SignaCert ETS and the server can modified the policy file based on its settings. The scanned result is an XML document signed by the SignaCert client.
- It generates a complete report, which includes environment scan report, TPM PCR values, TPM monotonic counter value, the current timestamp, and so on, and signs the report with both K_P and K_{PI}. The counter value and the timestamp are included to avoid the client using the same request more than once.
- The complete signed report is encoded into the TNCC request batch and sent over to the server. After the server verifies the correctness of the batch, the signed report is sent over to IAS through IMV.

- IAS (Integrated Authentication Server) verifies the report and returns the authentication results back to IMV. Please note that IAS does NOT make any decision about whether the request should be accepted or denied. The following methods are used in IAS
- If the report is correctly signed by K_P, the platform is the registered one.
- If the report is correctly signed by K_{PI}, the person is the registered one. This is based on the assumption that only the person which can provide the correct biometrics information can unlock the K_{PI} and use it to sign the report.
- If the PCR 0 – 6 values are identical to the registered ones, the components in the chain of trust before operating system are not compromised.
- If the counter value has been seen before, the request has been used before and will be denied immediately.
- If the difference between the current time and the timestamp in the report is greater than the predefined threshold, it will be reported as a potential, but not certain, issue.
- The snapshots collected by the OS are relayed to the SignaCert ETS for verification. Any unknown and/or known malicious measurements will make the environment report untrustworthy.
- The static scanning report is also relayed to SignaCert ETS for verification. Any unknown and/or known malicious signatures will be identified.

The output of IAS is then sent to the policy engine. The policy engine applies the pre-defined policies and determines the following issues.

- Should the current request be approved?
- If approved, which trust level should be assigned to the client?
- If not approved, what is the problem and how the client should fix the problem?

Sample policies include: 1) if the platform or user verification fails, the request will be denied, or 2) if any unknown item in the OS measurements is identified, the request will be denied, or 3) if unknown files are identified, the trust level for the client is 1 (low), or 4) if the correctness of the request cannot be verified (i.e. the signature does not match the request), the request will be denied, and so on. System administrator can always modify the policies after the TrustCube infrastructure is deployed to further tune the system.

Based on the decision of the policy engine, the server-side proxy (NAA) will either assign an access token to the client, or send a deny message which also includes the reason to the client. The access token is a randomly generated unique number. At the server side, this token is linked with the request, the decision from the server, and a valid period. In order for a client to get service beyond the valid period, the client must submit a new authentication request.

3.2.3 Operation Phase

After the client receives the token from the server, it is ready to visit the sensitive data. In this subsection, we are using a browser as the client-side application, but the same idea can be applied to other applications as well.

All HTTP requests from the browser go through the client-side proxy. The client-side proxy and the server are connected by a SSL tunnel. If the destination of the packets is other than the server, the proxy will reply the message to its original destination. Otherwise, the HTTP requests will be put in a special packet and sent to the server-side proxy using the SSL tunnel. The token is attached with the packet.

Once the server received the packet, it will first retrieve the token and validate it. If the token is not valid or expired, a HTTP 401 error message will be returned immediately over the tunnel, then relay to the browser. For a valid token, the initial HTTP requests are rewritten by attaching the token into the request URL. Then, the rewritten HTTP requests are sent to the document server.

The document server retrieves the token from the request URL. From the token, it finds out the trust level of the request. Based on the trust level, the corresponding service is provided. All responses from the document server are sent back to the browser through the server-side proxy, the client-side proxy path.

After the person finishes the browsing, she closes the client-side proxy (NAR). The client-side proxy will send a "bye-bye" message to the server and the server immediately invalidates the token. This concludes the session.

3.3 Discussion

In this subsection, we would like to discuss some issues related to the TrustCube infrastructure.

3.3.1 Chain of Trust

As we discussed previously, it is important that the chain of trust is maintained at the client side in order to provide a certifiable report about the person, the platform, and the environment. In the TrustCube infrastructure, the chain of trust is maintained in the following way.

The root of trust is the TPM. Follow the chain of trust, components below the operating system are measured and corresponding PCR values are extended with the measurement. If the values are identical, we will infer that the platform is not compromised. The operating system is a trusted one, and we are using IBM's IMA [Sai04] on a Fedora system. In a trusted OS, the executed files and loaded libraries are measured before being loaded into memory. Of course, our TrustCube client side modules (include fingerprint drivers) are measured as well. The trusted OS extends the measurement to PCR 8 to make sure that these measurements cannot be compromised without detection. Those measurements are part of the request and sent to the server. The server first checks the measurements and make sure they are identical to what have been saved in IAS, or using third party services, such as SignaCert ETS, to do the check. If the result is positive, the current running environment at the client is trusted, thus the data the client sent is also trusted. Finally the server will do the regular authentication based on the data.

Please note that any mistakes that causes the chain of trust to be broken, such as wrong values of PCR 0 – PCR 6, or unknown files in the operating system, will make the finally request untrustworthy, and cause the request to be rejected by the server.

3.3.2 Authentication of the TPM

In the current TPM specification, using passphrase is the only method for a person to unlock a key in the TPM. However, as we mentioned before, we would like to introduce other mechanisms, such as biometrics, smart card, to receive stronger protection. This requirement is currently being studied by the newly formed TCG Authentication Working Group.

However, currently since the BIO BSP is running as a software module, it is in a different level of the chain of trust from the TPM. If, for some reason, the chain of trust broke in the middle, certain TPM functions will become unavailable.

A possible solution for the problem is to create a “super security chip” or design a smart card which implements both the TPM functions and BIO BSP. In this way, BIO BSP works at the same level as the TPM in the chain of trust and below the operating system. This solution also requires certain extensions from the TSS.

In the current TrustCube infrastructure, we are implementing this idea in a customized way. We will be glad to adopt any specifications from the Authentication Working Group when they become available.

3.3.3 OS Measurements and Static Scanning Report

The TrustCube infrastructure needs both OS measurements and environment static scanning report. The difference is that OS measurements include executable files and libraries that are loaded by OS and environment static scanning report includes the snapshots of disk files at certain directories, no matter if they are loaded or not.

OS measurements are important in the sense that they show the current OS status. If any unknown or malicious measurements are detected, this OS instance is untrustworthy, so are the data it collected. However, only using OS measurement has two drawbacks: firstly, if a malicious file stays in the system but is not loaded yet (this is very common for certain type of virus which comes with infected files), it cannot be detected; and secondly, if the malicious file is a script and the damage is caused after it is executed by a innocent program, it cannot be identified either.

Scanning the whole environment may detect this sort of incumbent malicious files. The problem here is that it is not easy to decide which directories should be includes in the scanning policy. The major bottleneck here is at the disk I/O. One of our experiments shows that just to scan the files in the Windows directory on a standard Windows XP installation takes more than 30 seconds. Among them, 99.99% of the time is to read files from the disk. If we include too many directories in the scanning policy, the initial waiting time might to be too long.

One possible improvement is to implement a background environment scanning function. It runs as a background process and collecting environment report before the real authentication happens. During the scanning, any changes in the directory will be reflected as well. During the authentication phase, the update-to-date environment report will be used without delay.

3.3.4 Dynamic Environment Monitoring

Another possible improvement is to introduce a dynamic monitoring module in the client. Currently, after the client passes the authentication and the token is assigned, any further changes on the client side are not known by the server. If some malicious software is invoked or some unknown devices are connected during the operation phase, the server does not have any control.

A dynamic monitoring module in the client may help. The module will monitor the real-time changes of the platform and environment and report the changes to the server. Based on the changes, the server can either do nothing, or lower the trust level, or deny any future request from the client. Please note that the server will NOT increase the trust level, because the dynamic monitoring module cannot prove that the system is more trustworthy than the module itself.

4 Conclusion and Future Works

In this paper, we introduced the TrustCube infrastructure. The infrastructure extensively uses the Trusted Computing technologies and allows the server to make judgment based on the certifiable report about the person, the platform and the environment. The infrastructure is very flexible and can be used in almost all applications as an independent module to enhance their security.

Future works include a dynamic monitoring module and the improvement of the environment scanning module. Furthermore, we will extend the concept to other platforms, including routers, disk drivers, TV sets, node controllers, sensors, and so on. These platforms are potential stepping stone for Distributed Denial-of-service (DDoS). For a platform, it is tragic to be an accomplice of an evil deed and we believe that our TrustCube infrastructure is a good cure for it.

References

[Gar02] Garfinkel, Simson**.** Web Security, Privacy and Commerce, 2nd Edition. s.l. : O'Reilly Media, Inc., 2002. ISBN 0596000456.

[AHGBEA07] M1.4 Ad Hoc Group on Biometric in E-Authentication (AHGBEA). Study Report on Biometrics in E-Authentication. Washington, DC : InterNational Committee for Information Technology Standards, INCITS Secretariat, Information Technology Industry Council (ITI), 2007.

[Fre06] Freire, Carl. Virus spreads data, scandal over Winny. MSNBC. [Online] June 12, 2006. http://www.msnbc.msn.com/id/13283771/.

[TCG08] Trusted Computing Group. [Online] https://www.trustedcomputinggroup.org/home 2008.

[Kay05] The Future of Trusted Computing. Kay, Roger. s.l. : GovSec, 2005.

[TPM08] Trusted Computing Group. Trusted Platform Module (TPM) Specification. https://www.trustedcomputinggroup.org/specs/TPM, 2008.

[Sig08] SignaCert. SignaCert Enterprise Trust Server. http://www.signacert.com/products/enterprise-trust-server/, 2008.

[HVC04] Semantic Remote Attestation - A Virtual Machine directed approach to Trusted Computing. Haldar, Vivek, Chandra, Deepak and Franz, Michael. San Jose, California : 3rd Virtual Machine Research & Technology Symposium, 2004.

[Sai04] Design and Implementation of a TCG-based Integrity Measurement Architecture. Sailer, Reiner, et al. San Diego, California : s.n., 2004. 13th Usenix Security Symposium.

[Tgrub06] TrustedGRUB. http://www.prosec.rub.de/trusted_grub.html, 2006

[Kau07] OSLO: improving the security of trusted computing. Kauer, Bernhard. Boston, Massachusetts: 16th USENIX Security Symposium, 2007.

[TNC08] Trusted Computing Group. Specification, Trusted Network Connect (TNC). https://www.trustedcomputinggroup.org/specs/TNC, 2008.

[Bio05] ISO/IEC 19784-1. Information technology - Biometric application programming interface - Part 1: BioAPI Specification (version 2.0, international). 2005.

[TSS08] Trusted Computing Group. TCG Software Stack (TSS) Specifications. https://www.trustedcomputinggroup.org/specs/TSS, 2008.

[SKAE05] Trusted Computing Infrastructure Working Group. Subject Key Attestation Evidence Extension. https://www.trustedcomputinggroup.org/specs/IWG/IWG_SKAE_Extension_1-00.pdf, 2005.