Integrity Monitoring and Server Hardening through AIX 6.1 Trusted Execution

Authors:
Pruthvi Panayam Nataraj
Ravi Shankar
George Mathew Koikara
Saurabh Desai

Jun 2009

IBM Corporation

CONTENTS

1 Overview

AIX 6.1 introduced multiple security features to help customers enhance the security of their environment. One of the key features is called Trusted Execution (TE). This feature not only allows customer to monitor the system for integrity violations, but also provides for locking down the system in regards to execution of programs and loading of libraries and kernel extensions. This document explains the policies of Trusted Execution and provides outlines in regards to way administrator can use to protect their system environment.

Some of the concepts explained in detail in this document include:

1. Trusted Signature Database (TSD), which stores the baseline integrity data
2. Volatile files and Trusted Signature Database
3. Monitor for non approved kernel extension loads
4. Lock down the production system
5. Trusted Execution Paths and Trusted Library Paths
6. Finding Trojan Horses
7. Relationship between tcbck and trustchk
8. Integration with Role Based Access Control /Trusted AIX.
9. Create and ship security attributes in a package

2 Introduction to System Integrity

Security of a system is multifaceted issue and requires different tools and frameworks to thwart the threats. Some of the security mechanisms provide for protection against attacks (e.g.: network traffic monitoring for attacks and taking defensive actions), and some provide active monitoring against any successful attacks and denying the attacker of powers, privileges and data access (for example, Encrypting data, stop execution of unauthorized code, etc.). One such monitoring security feature is the system integrity verification.

It is necessary that the system administrator be able to verify at any point in time that the system has not been compromised. Additionally it is required that system integrity mechanism provides means to spoil attempts by the attacker to compromise the baseline system integrity information itself. To summarize the requirements for a good system integrity function:
Integrity Measurement: Provide administrator to detect changes to the system. The changes are identified by comparing the current state to a well known previous state (also called as the **baseline**).
Lockdown: Provide means to administrator to lock down the information established as baseline. This lock down will prevent an intruder from modifying the state of the system and then Baseline such that later admin will not be able to detect the modifications.
Monitor and Protect: Provide means to monitor executions of executables, libraries, kernel extensions etc.

Figure 1 shows that Trusted Execution blocks the tampered kernel extensions from getting loaded into the OS.

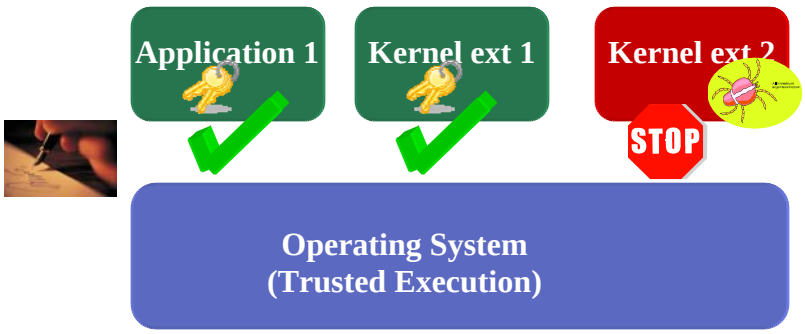

Figure 1: Trusted Execution: Keeping the malicious programs at bay

Trusted Execution provides for all of these features and more. These features are explained in detail in the following sections.

3 Integrity Measurement

It is extremely important that the system integrity be monitored regularly and reviewed by the administrator. With the availability of open source software in public domain, it is very easy for hackers and attackers to modify the source code and create Trojan Horses/malicious code and insert the same on to the target systems. It is also common that the attacker modifies either the system state or replaces important system executables and kernel extensions to complete the attack so that he/she will continue to have access to the system over a period of time.

System integrity involves capturing the good state of the system in a non-modifiable form and uses it as a reference to check the state of the system periodically. As shown in the figure below, administrator runs integrity measurement tool on a periodic basis and checks whether the system state has been modified.

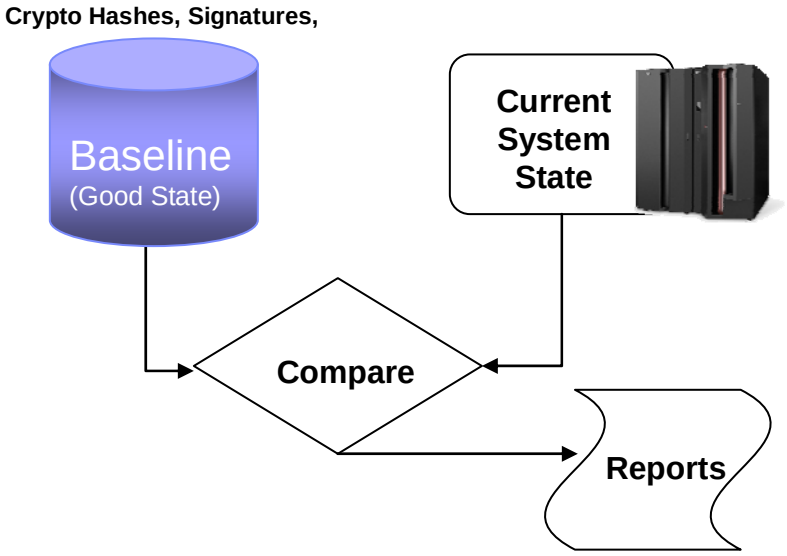

Figure 2: Integrity Measurement

AIX 6.1 ships IBM signed signatures for the important system files in the AIX operating system. These signatures are populated into a database called Trusted Signature Database (TSD). This Trusted Signature Database forms one component of the Baseline shown in the above figure. So, on a fully installed AIX system, Trusted Signature Database is populated for most AIX files and is ready to be used for integrity checking. Administrator while building their production system can add entries into the Trusted Signature Database for their custom tools and as well for as any commercial middleware or application that they plan to deploy in their production environment. Once the production level Trusted Signature Database is established, it could be used as a reference baseline for all similar systems in their environment.

Trusted Execution functions and policies on AIX are supported through a single command **trustchk** which acts as a policy manager as well as integrity measurement tool.

Command can be used by the administrator to setup the policies as well as to do the integrity checks.

Figure 2 shows how Trusted Execution can compare the system against the baseline and generate reports. One can execute trustchk as follows to do the integrity measurements (Note that this check can be done on an AIX 6.1 or later system without doing any additional setup):

trustchk -n ALL	Checks the integrity of the system and produces a report. If you had taken up a backup of Trusted Signature Database (say on a CD etc), one can use -F option to specify an alternate Trusted Signature Database to check against.

This will report any anomalies found on the system with regards to the base line. Example report is shown below:

```
…..
Verification of attributes failed: hash
Verification of attributes failed: signature
Verification of stanza failed: /usr/sbin/pstat
…..
```

Here message indicates that hash and signatures of file /usr/sbin/pstat do not match as captured in the Baseline.

Note that typically trustchk command should be able to scan all the files in the baseline within few minutes.

Recommendations:

1. Verify the integrity of the system periodically. E.g.: Add a job to cron tab to do the check and mail the report to administrator on a daily basis.
2. Take a backup of the production system Baseline on a DVD and use that as a reference for checks frequently.

3.1 Trusted Signature Database (baseline) Management

For any integrity measurement to be successful it is necessary to establish a clear baseline for the system and manage the same. It is important that a baseline be established for a production level server and there after any changes to baseline be carefully controlled. AIX by default stores the baseline related information in a file called */etc/security/tsd/tsd.dat*. The command trustchk provides interfaces to manage this base line file.

AIX provides for excellent support in regards to establishing and managing the baseline information for the server.

- AIX ships SHA256 Hashes and signatures (SHA256 hashes signed using a RSA 2048 bit key) for various system files that are critical for system operation and

hence need to be monitored on the system. These hashes/signatures (and other file related attributes) are automatically installed during the installation of the various packages and are captured in the Trusted Signature Database . Along with the installation of signatures and such information, installation process also installs any certificates related to the Trusted Execution that are part of the package.

- Command trustchk can be used to add/modify/delete entries to the Trusted Signature Database . For example if the administrator feels that he/she needs to monitor few files related an application or middleware, they can insert the entries related to the same into the baseline. Note that while doing so, administrator can provide their own private key/certificate pair to sign the application/middleware files related hashes.
- Administrator can choose to lockdown the baseline file on a production system. This would mean that no modifications to the baseline will be allowed in the system (even by root). These policies will be covered in detail in a later section.
- Trusted Signature Database database has a limited set of entries in it. These entries are added based on the security attributes each file has. For example, if a file is has SUID bit set, it will have an entry in Trusted Signature Database . Administrators will have to decide on exact set of entries that they want to have in the Trusted Signature Database . For example, the Trusted Signature Database can be configured such that it only allows a small set of DB2 applications to run on the system.
- AIX provides for a framework for ISVs to use to ship integrity information in regards to their software as part of the packages. This properly formatted integrity information will be recognized by the installp command on AIX and the integrity information will be automatically populated in the baseline database. More information on the framework will be covered in a later section.

It is expected that the administrator install all the software in regards to their production/deployment environment on an AIX server and then start establishing the baseline. Review the software installed in addition to the AIX install and mark the files related to the additional software to be in the baseline using the following steps:

1. Make a list of the software installed in addition to AIX. Identify the following types of files that may need to be monitored for integrity checking
    a. Any file that is a setuid/setgid application
    b. Any configuration files that would need monitoring.
    c. Any executables and libraries that are important (e.g.: root only executables, libraries etc)
2. Create a private key/certificate (public) pair. One can use openssl to generate the pair. See the box “How to generate private/certificate key pair?” box below.

3. For the list of files, add entries into the tsd.dat base line database.

Note 1

How to generate private/certificate key pair ?

Trusted Execution requires that you provide the private key and the corresponding certificate to manage the entries in the tsd.dat baseline file. One can do the same using OpenSSL tool as described in the following steps.

1. Install openssl(OpenSSL is part of the AIX expansion pack)
2. Create a private key (key will be created in the PEM format): *openssl genrsa -out TSDprivkey.pem 2048*
3. Create the corresponding certificate in DER format(valid for 3650 days/10 years): *openssl req –new –x509 –key TSDprivkey.pem –outform DER –out TSDcertificate.der –days 3650*
4. Convert the private key format from PEM to DER: *openssl pkcs8 –inform PEM –in TSDprivkey.pem –topk8 –nocrypt –outform DER –out TSDprivkey.der*
5. Invoke the trustchk to include an entry for file /usr/bin/yourfile: *trustchk -s TSDprivkey.der -v TSDcertificate.der –a /usr/bin/yourfile*

Note 2

How to find the files that are not in Trusted Signature Database but running running on system ?

TE treats all the files that are not present in Trusted Signature Database as untrusted. If STOP_UNTRUSTD policy is enabled, execution of such files gets blocked. To configure the Trusted Execution effectively, it is required for the administrator to know the list of such untrusted files executing on the system. This list can be easily found by using following steps.

1. trustchk –p TE=ON CHKEXEC=ON CHKSHLIB=ON CHKSCRIPT=ON CHKKERNEXT=ON
   This will enable the verification of all kinds of files getting loaded
2. Note that we have not turned on the policies STOP_UNTRUSTD and STOP_ON_CHKFAIL. That means the files will not be stopped from execution.
3. However, Trusted Execution logs a message to syslog when an untrusted file runs on system. Administrator can look into syslog for messages with “Trusted Execution: ” tag and find out the files that are untrusted.
4. Once such list is generated, administrator can decide whether to add those files into Trusted Signature Database or block them from execution.

Caution: Note that all the files that do not have an entry in Trusted Signature Database will be considered as untrusted. Enabling STOP_UNTRUSTD run time policy (discussed later) will stop all such files from execution. So a detailed analysis needs to be made before adding/deleting files from Trusted Signature Database and enabling the kernel policies.

4 Securing Trusted Signature Database

Key requirement for the integrity measurement to be successful is to guard against any modifications to the base line itself by the intruder. One can do this many ways. One of the easiest ways to achieve is to use a once-writeable media such as CD R/DVD R to store the Trusted Signature Database and use this CD/DVD as the reference Trusted Signature Database while checking for integrity periodically.

Another option to guard Trusted Signature Database against modification is to use the lock down policy in AIX 6.1 Trusted Execution. This disallows all writes to the Trusted Signature Database file and is described in the next section.

It is required that Trusted Signature Database be managed and maintained at a site/institution level. Create the baseline on one production system and use the same every where the same baseline for checking. Also it is extremely important to the Trusted Signature Database be in sync with the current system state. Any updates to the system files (if done non-installp way).

Note 3

How to protect Trusted Signature Database from modifications ?
To block modifications to TSD, enter the following command:

trustchk –p TE=ON TSD_LOCK=ON

Now, the /etc/security/tsd/tsd.dat is protected from all kinds of file modifications

How to take backup of Trusted Signature Database?
To take a backup, simply copy the /etc/security/tsd/tsd.dat to a protected media location. Later, to do the integrity checks using the backup Trusted Signature Database run following command:
trustchk -F <backup-tsd> –n ALL

5 Trusted Execution policies

Trusted Execution provides a policy based mechanism to determine the action when it encounters an object (file, library or executable) which is found to be suspect based on the various attributes of the object.

If a malicious user breaks into root account, he can turn OFF the Trusted Execution policies and change the system configuration. He could even modify the Trusted Signature Database contents. Hence, Trusted Execution provides locking capabilities to prevent system from any such damages.

Table 1

TE	Enables or disables Trusted Execution. Policies can only be activated when the Trusted Execution option is set to ON.
TSD_FILES_LOCK	Protects all the trusted files from modifications. That

	means unlink, rename, write or mount operations are not allowed on the trusted files. **Caution**: The config files, which get updated by daemons or any other commands, could be part of Trusted Signature Database . Enabling this policy could prevent daemons from writing to such files. Hence config files should always be marked as VOLATILE so that this lock policy is not enforced on them.
TSD_LOCK	Protects the Trusted Signature Database itself. If this is ON, any modifications to the /etc/security/tsd/tsd.dat will not be allowed. **Caution**: Note that trustchk command will also fail to add/delete any entries to Trusted Signature Database .
LOCK_KERN_POLICIES	LOCK_KERN_POLICIES can be used on a production system to protect the other Trusted Execution policies from being turned OFF. Once this policy is enabled, none of the other policies can be turned OFF. However, they can always be turned ON. This is based on the assumption that security can always be extended, but never compromised. If this policy is ON and any other policy has to be turned off, then first turn off LOCK_KERN_POLICIES and reboot the system. Then the other policies can be changed.
CHKEXEC	This policy makes sure that integrity of the trusted binaries is check during load time.
CHKSHLIB	The integrity of the trusted shared libraries is checked before loading them into the memory for execution.
CHKSCRIPT	The integrity of the trusted shell scripts is checked before loading them into the memory.
CHKKERNEXT	The integrity of the kernel extensions is checked before loading them into memory.
STOP_UNTRUSTED	This policy blocks the entities (which can be executable/libraries/scripts or kernel extensions) from loading if they are not in Trusted Signature Database . Hence, it works in combination with CHK* policies. For example, if CHKSCRIPT=ON and STOP_UNTRUSTED=ON, then those scripts that do not belong to Trusted Signature Database will not be executed. Stops the loading of files that do not belong to the Trusted Signature Database .

	TROJAN Stops the loading of files that do not belong to the Trusted Signature Database and have one of the following security settings: ▪ Have suid/sgid bit set ▪ Linked to a file in the Trusted Signature Database ▪ Have entry in the **privcmds** Database ▪ Be linked to a file in the **privcmds** database
STOP_ON_CHKFAIL	This policy stops the loading of the trusted files (same as the entities above) if integrity checks fail for them. It works in combination with CHK* policies. For instance, CHKEXEC=ON and STOP_ON_CHKFAIL=ON. Then any executable for an integrity check has failed will not be loaded.
TEP	Using this policy, we can set the value of the Trusted Execution path, enable it, or disable it. The Trusted Execution path is a list of directory paths separated by a colon. Once this policy is configured, then only the executables belonging to the set path are allowed to execute.
TLP	Like TEP, using this policy we can set the value of the trusted library path, enable, it or disable it. Trusted Library Path is colon-separated directory path. Once it is enabled, then only the libraries that belong to the Trusted Library Path will be loaded into the memory.

6 Volatile Files and Trusted Signature Database

A good system baseline not only includes the system related executables, kernel extensions, libraries but also includes other important files that are critical for correct system operation. Many of the system settings including multiple security settings are captured in configuration files. Note that unlike executables and others, these configuration files could get modified during the life of a system. For example /etc/passwd file is an important file in the system capturing the user configurations. While it is important to monitor this file, the monitoring aspects could not include factors such as size, hash etc (since the file will change users are being added/deleted/modified). Files such as /etc/passwd are termed as volatile files on the system. These types of files are monitored for integrity for file attributes such as owner, permission bits etc and are not monitored for size, hash/signature etc

AIX ships many of the system files such as /etc/passwd in Trusted Signature Database marked as volatile files. For your environment if more files need to be added to Trusted Signature Database , one can do so using the trustchk command with the files to be marked as VOLATILE.

Caution: Note that if policy TSD_FILES_LOCK is ON for, volatile files could pose functional problems. For example if /etc/passwd file is locked, user management functions/commands such as mkuser will not work for creation of local users. Hence such files need to be marked VOLATILE so that Trusted Execution does not lock such files.

Note 4

How to add VOLATILE files into Trusted Signature Database?

For making a file volatile, use size=VOLATILE while adding file to Trusted Signature Database and do not specify the key or certificate flags

trustchk –a /home/mydir/myfile size=VOLATILE

This will make the file /home/mydir/myfile a trusted file. The security attributes of this file will be verified during trust check verifications.. Also, if TSD_FILES_LOCK policy is enabled, then modifications to the file are not allowed. However, the run time checks will not be applicable for the file for the size.

7 Finding Trojan Horses

Command trustchk will support scanning for Trojan horses in AIX 6 TL4. When trustchk is invoked with the right options and provided with directory/set of directories to scan, then it will go through all the files on the particular target file system and scan for any programs that are not part of Trusted Signature Database , but are suspect from a privilege escalation point of view. Some of the criteria used for such a scan are:
Files will be tagged as potential suspect if it has any of the following properties:

1. File is a setuid or setgid program (owner is root/group security etc), but is not listed in Trusted Signature Database .
2. File is owned by root, but is not in the Trusted Signature Database .
3. File is a privileged command and is not in Trusted Signature Database .
4. File is a symbolic link to privileged command (RBAC) and not in Trusted Signature Database .

For runtime monitoring this can be set using the `'STOP_UNTRUSTED'` policy to the `'TROJAN'` value.

How to find Trojans on the system ?

trustchk -n tree

This will scan entire system to find all the possible vulnerable programs that are not present in TSD. To scan a particular directory, the path can be provided next to tree option.

How to block Trojan executables from running on system?

trustchk -p TE=ON CHKEXEC=ON STOP_UNTRUSTD=TROJAN

This will block all the executables which are not in Trusted Signature Database and match the Trojan properties as described above.

Note that above features are available in AIX 6 TL4.

8 Monitor and stop malicious program loads

Trusted Execution provides for monitoring and stopping malicious programs before they could harm the system.

TE can be configured to monitor the execution of kernel extensions, programs, shell scripts and libraries. When the files actually get loaded, Trusted Execution verifies the hash values of the file with respect the base line hash in Trusted Signature Database . If the verifications fail, then file will be stopped from execution. Trusted Execution also does the runtime verification of other attributes like owner, group and mode during loads.

Note that during any fileset installation, Trusted Execution policies TE, TEP and TLP need be turned OFF.

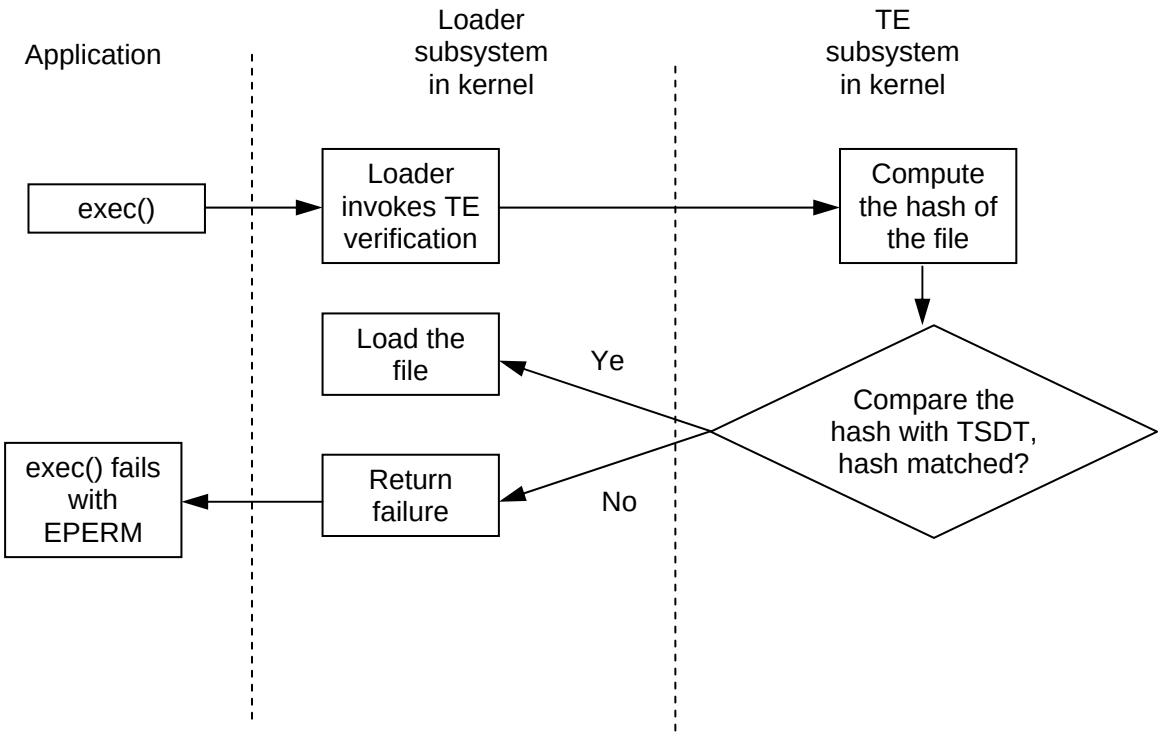

Figure 3

Note 5

How to configure runtime verification?

Trusted execution can verify the integrity of the files listed in Trusted Signature Database during their execution. Any file that is tampered, will not be allowed to run.

1. To verify the integrity of all executables(listed in TSD) and stop them if found tampered:
   trustchk –p TE=ON, CHKEXEC=ON STOP_ON_CHKFAIL=ON
2. To allow only the trusted kernel extensions that are listed in Trusted Signature Database to run on the system:
   trustchk –p TE=ON CHKEXEC=ON CHKKERNEXT=ON STOP_UNTRUSTD=ON

Caution: Note that all the files that do not have an entry in Trusted Signature Database will be considered as untrusted. Enabling STOP_UNTRUSTD run time policy will stop all the files from execution. So a detailed analysis needs to be made before adding/deleting files from Trusted Signature Database and enabling the kernel policies.

9 Trusted Paths

Trusted Execution can be used to define a set of paths which have the trusted binaries that can be allowed to run. When a binary is about to get loaded, Trusted Execution will check if it belongs to one of the defined Trusted Paths. If it's not from a trusted path, then Trusted Execution will block it from execution.

Trusted Execution Path (TEP): Trusted Execution allows only the executables from these paths to be executed.
Trusted Library Path (TLP): Trusted Execution allows only the libraries from these paths to be loaded.

Note 6

How to list trusted paths?

1. trustchk –p tep
   This will list the colon separated Trusted Execution Paths and the state of TEP(ON/OFF).

How update trusted paths?

2. trustchk –p tlp= /usr/lib:/usr/ccs/lib:/lib:/var/lib:**/home/my-dir**

10 Care to be taken while setting Trusted Execution policies

Trusted Execution policies are enforced by kernel security subsystem and loader. Once they are enabled, kernel starts monitoring all the files that are executing on the system. Based on the policies set, it will take actions like blocking the file from execution. The policies like STOP_UNTRUSTD, TEP and Trusted Library Path need to be enabled after detailed analysis files that need to run on the system.

STOP_UNTRUSTD will block all the files that are not in Trusted Signature Database from execution. Note that not all files that run in the system are in Trusted Signature Database . Users have to carefully identify the trusted files and add them to Trusted Signature Database before setting this policy. Note that enabling this can cause problems even during system shutdown/reboot.

TEP and **TLP** maintain a list of trusted paths. If TEP or TLP is enabled, all the files that are not part of these paths will be blocked from execution. So the TEP paths have to be carefully chosen before enabling the policies.

11 Supported cryptographic algorithms

Trusted Execution uses the IBM CLiC (CryptoLite for C) for cryptographic needs. So, clic.rte needs to be installed from AIX expansion pack before enabling the Trusted Execution kernel policies. The supported algorithms are as follows:

SHA1
SHA128
SHA256
MD5
MD2

12 Relationship between tcbck and trustchk

AIX has supported integrity monitoring mechanism previously through Trusted Computing Base (TCB). One of the components of Trusted Computing Base is tcbck program that mainly supported for integrity verification. To use tcbck program a customer was required to choose Trusted Computing Base install option during installation. This resulted in computing of checksums for various AIX files and they were captured as part of the database related to Trusted Computing Base (called sysck.cfg). Integrity measurements were done using checksums which do not provide a high level of protection against modifications. Trusted Execution as compared to that does not require a separate install option to be chosen. Trusted Execution related database Trusted Signature Database is created during installation of various packages and does not add any significant overheads to the install. Additionally, Trusted Execution is based off SHA256 hashes and the related signatures. And also Trusted Execution provides for many more active monitoring related features. Hence it is believed that trustchk will be much more sophisticated and convenient to use for system administrators and will replace tcbck as part of the overall Trusted Computing Base features.

The Table 2 captures some of the key differences between the tcbck and trustchk programs.

Table 2

TCB	**Trusted Execution**
Offline integrity verification	• Offline/Runtime integrity verification • Trusted Paths • Lock down mode etc
Verifies checksums	Verifies the digital signatures and SHA256 hashes
tcbck	trustchk (supports most of features of tcbck)
/etc/security/sysck.cfg	/etc/security/tsd/tsd.dat

13 Integration with Role Based Access Control and Trusted AIX

Trusted Execution (TE) has been designed to be aware of Role based Access Control (RBAC) introduced in AIX 6.1 and Trusted AIX.

The Role Based Access Control attributes for privileged commands in the privileged command database can be verified against the Trusted Signature Database . When trustchk is invoked with the –r option, only the Role Based Access Control attributes are verified from the Trusted Signature Database ,

On a Trusted AIX system, the trustchk –l option can be used to verify only the labels of the filesystem objects against the ones stored in Trusted Signature Database .

If a file being added to the Trusted Signature Database using the –a option without any attributes specified, belongs to the Privileged Command Database, then the Role Based Access Control attributes are populated from the Privileged Command database into the Trusted Signature Database .
Similarly, in Trusted AIX system, the existing labels of the file object will be taken from the filesystem.

14 Create and ship security attributes in a package

AIX 6.1 introduced a framework to distribute security attributes for various files as part of the packages. For example an ISV will be able to use this framework to ship signatures for their software's critical files and executables. This section provides guidelines for an ISV/software vendor about how they can create security attributes information and then ship it as part of their packages.

AIX 6.1 security framework is illustrated in the following figure.

```
/usr/bin/chuser:
        owner = root
        …..
        size =
        cert_tag =
        signature =
        hash_value =
        ….
```

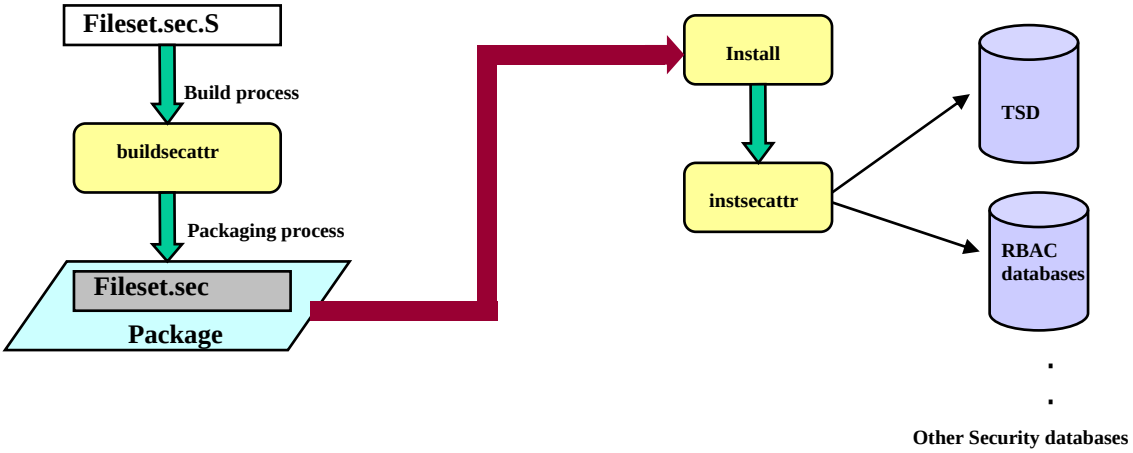

Figure 4: Processing of security attributes.

The security attributes for files are shipped as part of a special file in the fileset. The attributes supported and processed by the tools are listed in Table 3.

Following is example content of a <fileset>.sec file:

```
/usr/bin/xxx:
        owner = root
        group = security
        mode = TCB,SUID,550
        type = FILE
        hardlinks = /usr/bin/yyy
        symlinks =
        size = 10897
        cert_tag = 00cffb75b878cd48f8
        signature =
3889fc73e534ba2bfc1a9cf1a17220b2f9ee9434f6d94c53394e14744b9a59a3ed3e805
bbca85f28046ab75da4e20cad0fd01145921523aa4c00026110f62a46538381ba81ac81
3a5d39fb1871a8a01170c125287ba4543eab1c51c2dbf15d3cdb80052ba28229060e6e0
817ab3182c9149f90eaab7d3ae6a037bb6ca64d9272
        hash_value =
3931a650486e5131ba96e2d2a00e569b962a3c3217b3fff3328dba2e9aa2411a
        minslabel = SLSL
        maxslabel = SLSL
        intlabel = SHTL
        accessauths = aix.security.user
        innateprivs =
PV_AU_ADD,PV_AU_ADMIN,PV_AU_PROC,PV_DAC_R,PV_DAC_W,PV_DAC_X,PV_SU_UID
```

```
inheritprivs =
authprivs =
secflags = FSF_EPS
t_accessauth = aix.security.user.audit
t_innateprivs =PV_MAC_,PV_MIC
t_inheritprivs =
t_authprivs =
t_secflags =
```

Table 3: Security attributes for objects processed by install tools

Attribute	Type	Description	Target
owner	Static	Owner of the file, Should be same as that in fileset.il file	TSD
group	Static	Group of the file, Should be same as that in fileset.il file	TSD
mode	Static	File modes, Should be same as that in fileset.il file TCB: - Trusted Computing Base file *SUID - SUID bit set for the file* SGID - SGID bit set for the file	TSD
hardlinks	Static	Hardlinks that are pointing to the file, Should be same as that in fileset.il file	TSD
symlinks	Static	Symbolic links that are pointing to the file, Should be same as that in fileset.il file	TSD
type	Static	Type of the file FILE: Normal file(executable, library, config file etc) DIR: Directory HLINK: Hard link (for hlinks, the size, hash_value and signature should be marked VOLATILE) SLINK: Soft link	TSD
size	Dynamic	Size of the file, will be populated from buildsecattr tool. Should be marked as VOLATILE, if the file keeps changing. Check v/V flag in fileset.il	TSD

cert_tag	Dynamic	Certificate tag, will be populated from buildsecattr tool	TSD
signature	Dynamic	Signature of the file, will be populated from buildsecattr tool. Should be marked as VOLATILE, if the file keeps changing. Check v/V flag in fileset.il	TSD
hash_value	Dynamic	Cryptographic hash of the file, will be populated from buildsecattr tool. Should be marked as VOLATILE, if the file keeps changing. Check v/V flag in fileset.il	TSD
minslabel	Static	Minimum sensitivity label for the object	TSD
maxslabel	Static	Maximum sensitivity label for the object. This attribute is not applicable to regular files and FIFO	TSD
intlabel	Static	Integrity label for the object	TSD
accessauths	Static	Access authorization on the object	Trusted Signature Database & Role Based Access Control privcmds
innateprivs	Static	Innate privileges for the file	Trusted Signature Database & Role Based Access Control privcmds
inheritprivs	Static	Inherit privileges for the file	Trusted Signature Database & Role Based Access Control privcmds
authprivs	Static	Privileges that will be assigned to the user if he/she has the given authorizations	Trusted Signature Database & Role Based Access Control privcmds
authroles	Static	Roles to be authenticated before the command can be executed	Trusted Signature Database & Role Based Access Control privcmds
euid	Static	Effective UID to be assigned to the process when invoked by authorized user.	Trusted Signature Database & Role Based Access Control privcmds
egid	Static	Effective GID to be assigned to the process when invoked by authorized user.	Trusted Signature Database & Role Based Access Control privcmds
ruid	Static	Real UID to be assigned to the process when invoked by authorized user.	Trusted Signature Database & Role Based Access Control privcmds

secflags	Static	Security flags associated with the object	Trusted Signature Database & Role Based Access Control privcmds
t_accessauth	Static	Trusted AIX/MLS specific access authorizations	Trusted Signature Database & Role Based Access Control privcmds
t_innateprivs	Static	Trusted AIX/MLS specific innate privileges for the file	Trusted Signature Database & Role Based Access Control privcmds
t_inheritprivs	Static	Trusted AIX/MLS specific Inherit privileges for the file	Trusted Signature Database & Role Based Access Control privcmds
t_authprivs	Static	Trusted AIX/MLS specific privileges that will be assigned to the user if he/she has the given authorizations	Trusted Signature Database & Role Based Access Control privcmds
t_secflags	Static	Trusted AIX/MLS specific file security flags associated with the object	Trusted Signature Database & Role Based Access Control privcmds

Here is a description of how the <fileset>.sec files are generated in AIX build environment:

- A security attribute file <fileset>.sec is generated during the build for each fileset which contains the static and computed attributes
- Dynamic attributes like size, hash and signature are computed during packaging
- A signing key which available in build environment is used to generate signatures.
- Certificate corresponding to signing key will be at /etc/security/certificates
- Note that tool for computing the hash, signature and size is not provided by AIX and ISVs should have their own mechanism for generating <fileset>.sec files

This <fileset>.sec file is shipped as part of liblpp.a in the fileset in the path usr/lpp/<fileset>/inst_root/ of the installp package.

Here is a description of how <fileset>.sec file is processed during installation:

- The tool /usr/sbin/instsecattr processes the <fileset>.sec files during installation of filesets
- This processing automatically happens when there is a <fileset>.sec file in the fileset.
- The security attributes are distributed to various security DBs (Trusted Signature Database and privcmds DBs) during installation
- Note that during installation, Trusted Execution policies need be turned OFF

An alternative way to have the security attributes installed would be to ship the contents in the post install configuration scripts for the packages. This is method is also suitable for RPM based packages. A brief sequence of steps on how to setup using this approach is given below.

- Create a temporary security attribute file (say /tmp/<fileset>.sec) from the post install configuration script. This file should contain all the attributes values as required as shown in the code snippet below.
- Invoke instsecattr tool on this security attribute file

Code snippet

```
#!/bin/sh
#Post install configuration file
cat > /tmp/myfileset.sec <<EOF
/usr/bin/xxx:
        owner = root
        group = security
        mode = TCB,SUID,550
        type = FILE
        hardlinks = /usr/bin/yyy
        symlinks =
        size = 10897
        cert_tag = 00cffb75b878cd48f8
        signature =
3889fc73e534ba2bfc1a9cf1a17220b2f9ee9434f6d94c53394e14744b9a59a3ed3e805
bbca85f28046ab75da4e20cad0fd01145921523aa4c00026110f62a46538381ba81ac81
3a5d39fb1871a8a01170c125287ba4543eab1c51c2dbf15d3cdb80052ba28229060e6e0
817ab3182c9149f90eaab7d3ae6a037bb6ca64d9272
        hash_value =
3931a650486e5131ba96e2d2a00e569b962a3c3217b3fff3328dba2e9aa2411a
        minslabel = SLSL
        maxslabel = SLSL
        intlabel = SHTL
        accessauths = aix.security.user
        innateprivs =
PV_AU_ADD,PV_AU_ADMIN,PV_AU_PROC,PV_DAC_R,PV_DAC_W,PV_DAC_X,PV_SU_UID
        inheritprivs =
        authprivs =
        secflags = FSF_EPS
        t_accessauth = aix.security.user.audit
        t_innateprivs =PV_MAC_,PV_MIC
        t_inheritprivs =
        t_authprivs =
        t_secflags =
EOF

# Delete the older entries if they exist
/usr/sbin/instsecattr –d /tmp/myfileset.sec

# Add the new entries
/usr/sbin/instsecattr –a /tmp/myfileset.sec
#End of script
```

Note:

While shipping updates to the filesets care should be taken to ship the new sizes of updated binaries to the target system or else such binaries should be marked VOLATILE. Marking executables as VOLATILE is not recommended.

15 Hosting Trusted Execution policies on LDAP server

Trusted Execution supports hosting the Trusted Signature Database database and policies on a centralized LDAP server. This feature will be supported from AIX 6 TL4. An LDAP server will host the Trusted Signature Database database and policy files for multiple clients, which helps central management. These central policies could be used as a master copy during offline verifications also.

There will be 2 different versions of Trusted Signature Database and Trusted Execution policy files for local(/etc/security/tsd/ and LDAP(/etc/security/tsd/ldap). /etc/nscontrol.conf file can be used to specify that LDAP version has to be referred by trustchk command during various operations such as add/modify/verify attributes.

tetoldif command can be used to convert the tsd.dat and tepolicies.dat files to LDAP format. This command reads the **/etc/security/ldap/sectoldif.cfg to determine the Trusted Signature Database DB name and sub-trees where the data has to be exported to.**

Trusted Execution Policies on LDAP
This will enable the system administrators to enforce the Trusted Execution policies on various clients by just changing policy file on LDAP.

To convert a Trusted Execution database into ldif format
tetoldif -d cn=aixdata –p

How to add the ldif file into LDAP server:
Redirect the output of above command to a temporary file and then upload that file using ldapadd command.

Trusted Signature Database on LDAP
As Trusted Signature Database is a very critical component and contains all the security attributes, it is required to take a backup periodically. There could be chances that

privileged users can tamper this database itself. Hence administrators can keep a copy of the Trusted Signature Database on LDAP and use if for verifying the integrity of the system periodically.

To convert a Trusted Signature Database into ldif format:
```
tetoldif -d cn=aixdata -s
```

How to add the ldif file into LDAP server:
Redirect the output of above command to a temporary file and then upload that file using ldapadd command.

16 Frequently asked questions

1. Why are some of the AIX shipped libraries marked volatile and not shipped with signatures?
   - This is due to the fact some of the AIX libraries are updated during install with object codes (during install updates) and hence it is difficult to recalculate the signature without the private key. However, it is encouraged that the system administrator signs these binaries with their custom private/public key. Why does switching from locked down mode require a reboot ?
   - Most AIX 6.1 security features have been designed such that the system administrator can increase the security of the system without a reboot. However any change in security settings that results in reduction of security level requires a reboot. So an intruder would need to change the setting and reboot to make the security level to be level. Here it is assumed that reboot is a significant event not to be missed by the administrator. A security conscious administrator can further prevent the intruder from rebooting by inserting a password promoting program in the boot path, so that the intruder is stopped in his/her tracks while trying to reboot the system. Note that this level of security does come at the cost of usability.
2. Why we need to turn of Trusted Execution during installation of filesets?
   The reason is that if some files which are used by install scripts could be replaced during installation(like /usr/bin/cp). This will cause run time verification of such newly installed commands fail. /usr/bin/cp might get invoked when install is going on and as the new binary hash does not match with one in kernel Trusted Signature Database tables, it may fail to execute.

3. What to do if trustchk reports errors?
   Answer: Analyze if the errors reported are serious in nature. Typically the Trusted

Signature Database is modified during installation of filesets. If no installation has been done lately and trustchk reports errors it could mean that the files in question have been modified. ‘trustchk' reports which attribute of the object has caused the error.

4. What to do if want to make sure trustchk is run at every boot?
   Answer: Add ‘trustchk –t ALL’ to one of the boot scripts. /etc/rc.security.boot.

5. If a binary is failing to load because dependent library is not in Trusted Signature Database ?
   Answer: Turn off TE. Add the library that is failing to Trusted Signature Database and turn on TE.

6. How to recover the system when none of the binaries are loading?
   Answer: Boot in single user mode and do one of these:
   a. Comment the “trustchk –b” line in /etc/rc.security.boot script and then reboot
   b. From AIX 6 TL4, you can find the Trusted Execution policies in /etc/security/tsd/tepolicies.dat. Edit the file and turn off TE.

8 WPAR restricting binaries

17 Glossary

TE	Trusted Execution
TSD	Trusted Signature Database, which hosts the security attributes for trusted files
LDAP	Lightweight Directory Access Protocol: Used as a database for storing the information and provides ways to access them quickly
TEP	Trusted Execution Path: Trusted Execution allows only binaries in the paths listed in TEP to execute
TLP	Trusted Library Path: Trusted Execution allows only libraries in the paths listed in Trusted Library Path to get loaded
RBAC	Role Based Access Control: Provides fine-grained access control to the system resources
SHA256	Secure Hashing Algorithm: Default algorithm used to generate the cryptographic hashes

© IBM Corporation 2006
IBM Corporation
Systems and Technology Group
Route 100
Somers, New York 10589

Produced in the United States of America
February 2006
All Rights Reserved

This document was developed for products and/or services offered in the United States. IBM may not offer the products, features, or services discussed in this document in other countries.

The information may be subject to change without notice. Consult your local IBM business contact for information on the products, features and services available in your area.

All statements regarding IBM future directions and intent are subject to change or withdrawal without notice and represent goals and objectives only.

IBM, the IBM logo, AIX, AIX 5L, DB2, Tivoli are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both. A full list of U.S. trademarks owned by IBM may be found at: http://www.ibm.com/legal/copytrade.shtml.

UNIX is a registered trademark of The Open Group in the United States, other countries or both.

POSIX is a registered trademark of IEEE.

Microsoft and Windows are registered trademarks of the Microsoft Corporation.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. In the United States and/or other countries.

Kerberos and Project Athena are trademarks of Massachusetts Institute of Technology.

Other company, product, and service names may be trademarks or service marks of others.

Copying or downloading the images contained in this document is expressly prohibited without the written consent of IBM.

Information concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of the non-IBM products should be addressed with those suppliers.

The IBM home page on the Internet can be found at: http://www.ibm.com.

The IBM System p5 and @server p5 home page on the Internet can be found at: http://www.ibm.com/systems/p.