Redpaper

David Edwards

Auditing UNIX/Linux System Use with Tivoli Access Manager for Operating Systems and Tivoli Compliance Insight Manager

Introduction

This IBM® Redpaper looks at auditing UNIX/Linux® system use with the Tivoli® Access™ Manager for Operating Systems and Tivoli Compliance Insight Manager products, and focuses only on UNIX/Linux system auditing.

Any site that has deployed a large number of UNIX® or Linux systems will be familiar with the security concerns that are entrenched in these operating systems. One of the most significant is the concern over the use of the superuser account, root, or any account with UID=0. The root user has access to any resource in the system, and where this activity is logged through system accounting or auditing, the root user has access rights to modify the audit files. The user could perform malicious changes to the system and then wipe their tracks. As many activities on a UNIX/Linux system require root authority, many sites find that the number of users who know the root password is out of control and impossible to track.

Tivoli Access Manager for Operating Systems provides operating system level access control for UNIX/Linux systems. One of the key features is the ability to control root account use. Another strength of the product is its ability to audit system use and secure the audit trail from tampering. Tivoli Compliance Insight Manager provides enterprise-wide audit and compliance reporting. Use of Tivoli Access Manager for Operating Systems with Tivoli Compliance Insight Manager can provide an effective UNIX/Linux activity auditing solution.

This paper is an introduction to Tivoli Access Manager for Operating Systems and how it provides for UNIX/Linux activity auditing. A number of privileged user use cases are performed, and the native Tivoli Access Manager for Operating Systems auditing mechanism is used to report on the use cases. Finally, this audit data is sent to the Tivoli Compliance Insight Manager and viewed using standard and custom reporting.

Tivoli Access Manager for Operating Systems (TAMOS)

This section provides an overview of TAMOS, looking at its features and high-level architecture.

Introduction to TAMOS

UNIX has several inherent problems with security when it is examined from an enterprise point of view. These include:

- There has been no inherent enterprise-strength security infrastructure in the earlier UNIX and Linux operating systems. Each vendor has addressed this issue by implementing their own forms of enhanced or hardened UNIX security, such as the Enhanced Security Option in AIX® or Security-Enhanced Linux (SELinux). While these implementations often resolve the issue of low security, they lead to different processes and tools to implement policy on disparate systems.
- Another problem centers around the concept of group users. In UNIX, when a group user account is used, such as root, all auditing is based on the group user account, not an individual user account. By its nature, this makes auditing events on the host system extremely difficult.
- Many functions in UNIX/Linux require root (or UID=0) authority to perform day-to-day tasks, so in order to perform normal functions, there is significant risk of malicious or accidental damage to the system due to the user having too much access. No access control rules apply to root, which means that not only can they access any file, they can wipe their tracks from any audit logs.

TAMOS addresses these concerns by providing a layer of authorization policy enforcement in addition to that provided by a native UNIX/Linux operating system. It applies fine-grained access controls that restrict or permit access to key system resources.

Controls are based on user identity, group membership, type of operation, time of day or day of week, and the accessing application. An administrator can control access to specific file resources, login and network services, and changes of identity. These controls can also be used to manage the execution of administrative procedures and to limit administrative capabilities on a per-user basis.

When protected resources are accessed, TAMOS performs an authorization check based on the accessing user's identity, the action, and the resource's access controls to determine whether access should be permitted or denied. This means that even if the user assumes the root identity (for example, by using `su`), but the TAMOS policy does not allow that user to access that resource, then the access is denied by TAMOS before it gets to the OS.

In addition to authorization policy enforcement, mechanisms are provided to verify defined policy and audit authorization decisions. This auditing can be set on a global basis, against specific users or against specific resources. For example, you could audit all access by root (or any UID=0 users) and against a set of high-risk files.

Access controls are stored in a policy database that is centrally maintained in the IBM Tivoli Access Manager (TAM) environment. As TAMOS on multiple servers can share the same policy database, a common policy can be built centrally and distributed. This is better than creating policy for each system and manually applying it. You can define a set of policies based on different attributes of the systems and join them to suit specific machines. For example, you may have a set of OS-specific policies (for example, one for SUSE Linux, one for AIX 5.3, one for Solaris™ 9), another set for middleware (for example, one for

WebSphere® Application Server, one for Tivoli Directory Server, and one for DB2®), and another set for applications (for example, one for the HR software, one for the accounts receivable software, and one for the POS systems). If you have three SUSE Linux machines running WAS and the POS software, they can each share the same policy centrally built from three sets of definitions. This is easier to maintain than having a unique policy for each machine that contains parallel definitions that must be kept in synch over time.

The UNIX/Linux user definitions are stored in a user registry that is also centrally maintained in the environment. TAMOS does not perform user authentication. It trusts the operating system authentication mechanism. However, it can apply additional login policy, such as checking for password expiration and enforcing a consistent password strength check.

The next section details how these functions are implemented.

TAMOS architecture

TAMOS is implemented as a series of UNIX/Linux daemons, the kernel extension, and some control files. It uses the TAM framework for policy management and the TAM user registry (such as an LDAP) for managing its users and groups. The high-level architecture for TAMOS is shown in Figure 1.

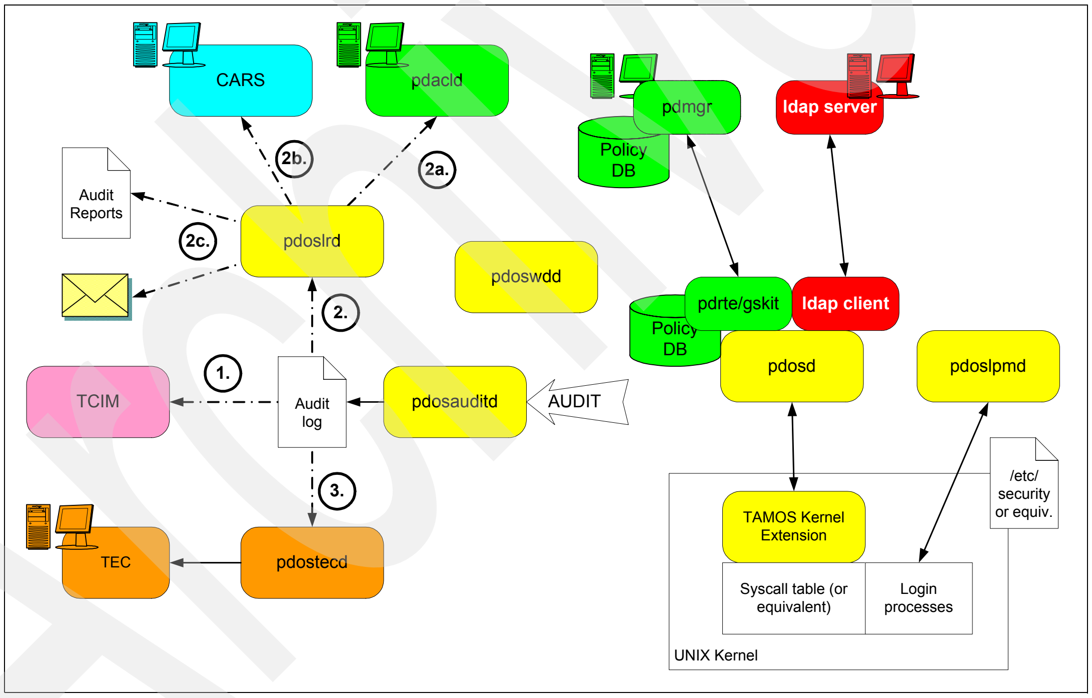


Figure 1 TAMOS components

The product has the following processes, or daemons:

pdosd This is the central authorization deamon for TAMOS. It is called by the TAMOS Kernel Extensions when a system call is made and performs authorization decisions based on the user credentials and the policy defined in the local copy of the policy database. It uses the TAM runtime (pdrte) to receive copies of the policy database from the TAM policy server (pdmgr) and the ldap client to retrieve user credentials from the user repository.

pdoslpmd	This is the Login Process Management deamon and intercepts calls from the login processes and verifies login-related policy (such as password expiration and password strength rules).
pdoswdd	This is the Watchdog daemon. It monitors what the other daemons are running and if it detects that they are not running it restarts them. This is one of the TAMOS internal security measures.
pdosauditd	This daemon captures the binary audit data generated by the other daemons (primarily pdosd) and generates audit logs. These audit logs can be viewed with the **`pdosaudview`** command (not shown), gathered by TCIM (1), distributed by the Log Router Daemon (2), or gathered by the Tivoli Enterprise Console® (TEC) daemon (3).
pdoslrd	This is the log router daemon that can route audit records on to a TAM authorization daemon for consolidation with other TAMOS system audit files (2a), the Tivoli Common Audit and Reporting Server (CARS) (2b), or other file formats such as SMTP (2c).

TAMOS is able to implement this security layer as it hooks into the system calls at the kernel level, as shown in Figure 2.

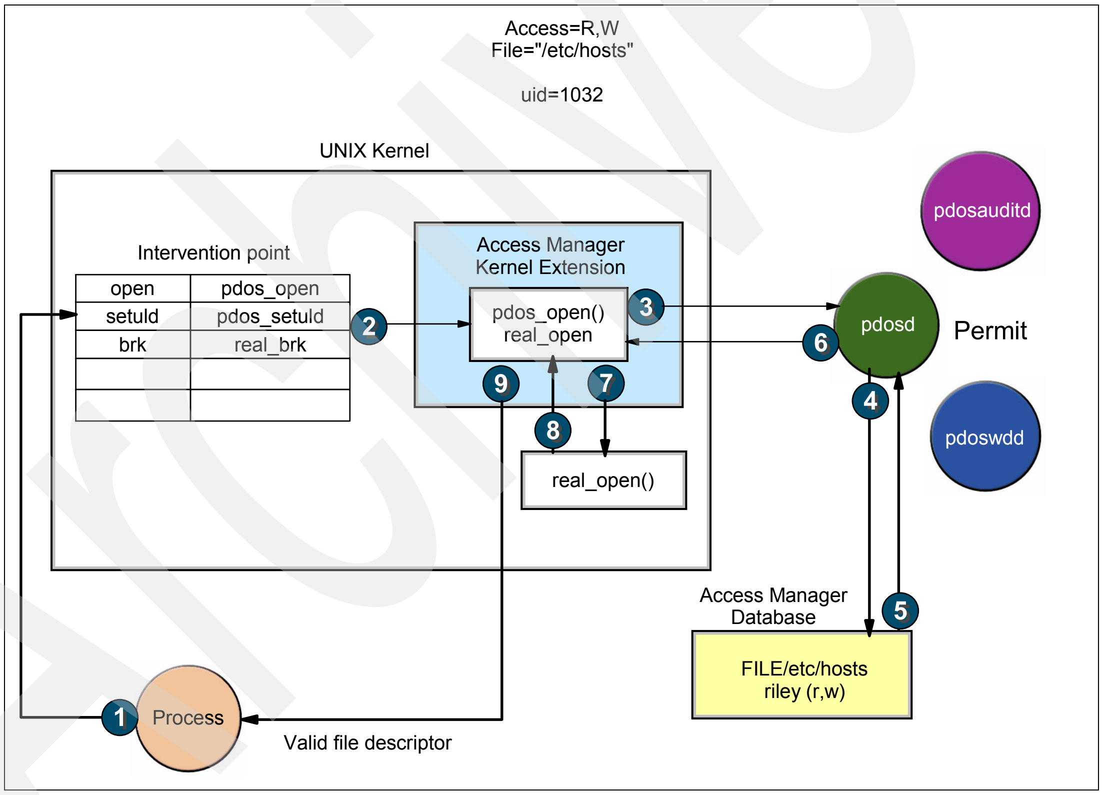


Figure 2 TAMOS kernel interception

This example shows the flow of a process attempting to access the /etc/hosts file with TAMOS installed. The steps are:

1. The process attempts to open the file, resulting in an open() call to the kernel for /etc/hosts.
2. The kernel intervention point (such as the syscall table) has been modified at startup so that the addresses for the kernel functions have been replaced with the equivalent TAMOS function. In this case, open() is mapped to the pdos_open() function in the TAMOS Kernel Extension.

3. The TAMOS kernel extension calls the TAMOS authorization daemon (pdosd), passing it the identity of the user/process, the resource being accessed, and the type of access.
4. The pdosd deamon requests the relevant policy definitions from the policy database.
5. These are processed by the pdosd daemon, which in this case allows the file access.
6. This response is returned to the TAMOS kernel extension.
7. The kernel extension calls the real kernel open function (shown as real_open() in Figure 2 on page 4).
8. The real open() function returns a file descriptor.
9. This is passed back to the application.

This approach allows TAMOS to apply access control to all requests to the kernel and apply consistent auditing.

For further information about TAMOS, see *Enterprise Security Architecture Using IBM Tivoli Security Solutions,* SG24-6014.

TAMOS and auditing

This section discusses how TAMOS fits into a broad audit solution.

Policy definition and implementation

TAMOS implements policy through the centralized TAM policy framework. It can define user roles and map them to access rights that apply to UNIX/Linux systems. As the TAM policy database may be shared across various TAM-based products, such as TAM for e-business, TAM for Business Integration, and the WAS Access Manager JACC Provider, it may be possible to implement enterprise-wide roles that apply access rights across multiple systems to cross-system users.

Roles are implemented in the form of TAM groups. Users or groups are mapped to resources in the protected objectspace via *access control lists* (ACLs) and *protected object policies* (POPs). ACLs dictate what access a user or group has on a resource. POPs dictate a level of service on that access, such as time of day restrictions or auditing levels.

The policy implemented through TAMOS covers both resource access control and login policy. The types of policy that can be implemented include:

File	Access to files, directories, soft links, hard links, and device files.
TCB	Files can also be defined as part of a Trusted Computing Base (TCB), which is monitored for changes and reacts when changes are detected (such as disabling all access to the file).
Network	Access to connect to specific ports or services on the system, and access for this machine to connect to other machines in the network.
Surrogate	Policy can be applied to all users or processes that perform any form of user switching. TAMOS also provides its own SUDO implementation that is part of the policy.
Login	Control when and from where a user can log into a system. For example, you could restrict root login to a specific terminal.
Password	Policy relating to password strength and aging.

This policy can be defined using the TAM Web Portal Manager browser-based interface or the **`pdadmin`** command-line utility. These are the same tools that would be used to define TAMeb or another TAM-related policy.

There is a set of pre-built policy definitions, called Fast Start Policy Modules, available from the TAMOS support site:

`http://www.ibm.com/software/sysmgmt/support/IBMTivoliAccessManagerforOperatingSystems.html`

At the time of writing, these modules provided policy definitions for DB2, UNIX, Oracle®, WAS, IBM HTTP Server (IHS), and TAMeb (WebSEAL). As of TAMOS 6.0 these modules are not officially supported but can still form the basis of a policy baseline.

More details about the TAMOS policy and policy definition can be found in the *TAMOS 6.0 Administration Guide,* SG32-1709.

Audit capture

Auditing activity on a UNIX/Linux system is one of the strengths of TAMOS. You can apply audit policy consistently across a large set of disparate systems to ensure that you are capturing the same activity.

TAMOS auditing operates on three levels: auditing of *authorization activity*, auditing of process execution or file access (referred to as *auditing trace events*), and auditing for *system health*. We do not discuss the system health auditing, but the following paragraphs summarize the auditing available for the first two.

The authorization auditing is driven by the TAMOS policy enforcement. When an authorization decision is made by TAMOS and where the auditing levels match, an audit record is cut. The types of authorization auditing are:

Global	Where a level of auditing is enabled for a specific machine. It can be set to permit or deny.
Resource level	Where a level of auditing is enabled for a specific resource. These definitions can be set to permit, deny, or both, so you can capture allowed access to a resource, denied access to a resource, or all access to a resource. There is another level of granularity on resource-level auditing where specific access types can be audited. Auditing for resources can be further filtered. For example, you could specify that only create, rename, and delete operations against a file resource are audited.
User level	Where a level of auditing is enabled for a specific user. Supported audit levels for user-level audit authorization policy are: permit, deny, loginpermit, logindeny, all, and none. The loginpermit and logindeny settings control audit record generation for login authorization decisions, which are also included in the permit and deny settings.

Authorization auditing levels are generally cumulative, so if the global level is deny and there is a specific resource-level permit set, then both permit and deny audit records are cut for that resource. There are some exceptions to this.

You would use the authorization auditing where you have a well-defined security policy implemented through TAMOS. It is explicit. You are only auditing against decisions made on the policy that you have defined. The other type of activity auditing is the auditing of trace events.

Trace event auditing can be specified at a global level or at a user level. There are four settings for global-level trace event auditing:

trace_exec	Track program invocations initiated by the exec() system call. If this is set at the global level or for a very active user, this could generate a large volume of audit data.
trace_exec_l	Track program invocations initiated by the exec() system call where the accessing user's identity and effective UNIX identity do not match. For example, a user has performed a su and then executed a program.
trace_exec_root	Track program invocations initiated by the exec() system call by the root user (UID=0). This is only for users who have logged in as a uid=0 user, not someone who has assumed root identity after login.
trace_file	Track access to file system resources that are protected by TAMOS (that is, defined in the TAMOS policy that applies to the system).

You can specify equivalent settings at a user level: exec, exec_l, file, all, or none. Use of none overrides any global settings.

Note: Trace audit events are only generated for processes that descend from a login event that was detected by TAMOS. If there has been system access using a login mechanism that is *not* defined to TAMOS, no audit records are produced.

As the trace_exec (user exec), trace_exec_l (exec_l), and trace_exec_root trace settings operate on all resources, not just those defined in the TAMOS policy for a system, they can be used to monitor for unexpected behavior and tune the policy.

An effective policy management strategy is to define the standard policy and enable auditing against that policy to track violations. Then you would enable one or more trace audit settings to monitor for unexpected activity and review against the standard policy, either modifying the policy or agreeing and documenting the exceptions.

Another feature of the TAMOS policy system is the *warning mode* option. You can set a resource authorization definition to warning mode. When TAMOS detects a resource access that is denied, it allows the access but writes out an audit record indicating that it has been allowed under the warning mode. This is another effective tool to test and tune policy applied to a system. It is common to deploy new policy definitions in warning mode for a period and then review the allowed violations.

For more information see Chapter 7, "Auditing," of the *TAMOS 6.0 Administration Guide*, SG32-1709.

Audit consolidation

TAMOS has its own mechanism for audit log consolidation. It can use the TAMOS log router daemon (pdoslrd) to forward audit logs to a central location. This is shown in Figure 1 on page 3. This mechanism does not merge the audit logs. It creates a depot of different system logs on one machine that can be used for reporting.

The log router daemon can also forward events to the Tivoli Common Audit and Reporting Service (CARS), although use of CARS is being replaced by Tivoli Compliance Insight Manager (TCIM). The TAMOS Tivoli Enterprise Console (TEC) daemon can forward TAMOS audit records to TEC, although this is normally used for real-time event reporting, not historical audit and compliance reporting. These components are also shown in Figure 1 on page 3.

TCIM has two mechanisms to pull TAMOS audit log records to the TCIM server: a TAMOS actuator that runs on the TAMOS server and a TAMOS SSH actuator that runs on another server and connects to the TAMOS server via SSH. TCIM is configured to read the TAMOS binary audit logs and perform the W7 mapping for the TAMOS audit record structure.

Audit reporting

If TAMOS-only mechanisms are used for reporting, there are TAMOS commands to access the binary audit log on the local system and to access the consolidated log files.

The **pdosaudview** command is used to view the local TAMOS audit logs. It has an extensive argument list to allow filtering of data. By default it uses a keyvalue format (-F keyvalue), as shown in Example 1. It can also produce a comma-separate format (-F concise) and a readable display with an attribute and its description on each line (-F verbose).

The example shows a series of audit records relating to the root user creating a user. There is a trace exec record (A=Trace) showing the command entered and its arguments, the authorization call record (A=Check Access) followed by more trace exec records for the subsequent exec() system calls associated with the useradd.

Example 1   pdosaudview example

```
TS=Tue 30 Oct 2007 02:05:30 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/sbin/useradd
(useradd -m -u 1100 -s /bin/sh insight),APID=7406,RPSN=/bin/bash,UQ=0
TS=Tue 30 Oct 2007 02:05:30 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/dev/kazndrv,SRN=/dev/kazndrv,APID=7406,RPSN=/
usr/sbin/useradd,O=S,UQ=1
TS=Tue 30 Oct 2007 02:05:30 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/sbin/useradd.local
(/usr/sbin/useradd.local insight 1100 100
/home/insight),APID=7407,RPSN=/usr/sbin/useradd,UQ=2
TS=Tue 30 Oct 2007 02:05:30 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/bash
(/usr/sbin/useradd.local insight 1100 100
/home/insight),APID=7407,RPSN=/usr/sbin/useradd.local,UQ=3
TS=Tue 30 Oct 2007 02:05:30 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/sbin/useradd.local
(/bin/bash /usr/sbin/useradd.local insight 1100 100
/home/insight),APID=7407,RPSN=/usr/sbin/useradd.local,UQ=4
TS=Tue 30 Oct 2007 02:05:30 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/bash (/bin/bash
/usr/sbin/useradd.local insight 1100 100
/home/insight),APID=7407,RPSN=/usr/sbin/useradd.local,UQ=5
```

The pdosaudview keyvalue report format is great for an experienced user and very useful for forensic investigations, but it is not a format that would be usable by auditors and other non-technical people. The verbose report format may be usable, but can be very lengthy. For more usable reports, some scripting is required.

The **pdoscollview** command is used to view the consolidated audit log files. The arguments and report formats are basically the same as for pdosaudview.

Compliance reporting

TAMOS does not do compliance reporting. You can use the auditing settings and reporting commands to report on where access by users against resources was permitted or denied by

TAMOS, but there is no mechanism to measure this against a level of compliance. Using just the audit facilities provided by TAMOS requires extensive customization.

For compliance reporting, we recommend using TCIM.

Auditing UNIX/Linux systems with TAMOS

One of the strengths of the TAMOS product is its auditing ability. Not only can it audit the authorization decisions it makes, it can also audit all activity for different types of users, such as the root (UID=0) user or users who have performed some form of change user (such as su).

This section contains a number of test cases run on a system with TAMOS auditing enabled to show how well TAMOS can audit privileged user use on a UNIX/Linux system. The test cases are:

1. Create another privileged user. In this case we use the root account to create another account with UID=0.
2. Log in as a new user and perform suspect activity. The new user created in case #1 is used to perform an activity that could be suspect, such as editing the inittab to introduce a process to be automatically started when the system starts.
3. Root attempts to hide the audit trail. In this case root tries to remove part of the audit trail to hide activity.
4. Track su use. Many UNIX/Linux systems do not allow users to log in as root, but do allow users to su to root. On vanilla UNIX/Linux systems this may hide the original user in any audit trails. This test case aims to see how effective TAMOS is at auditing this activity.
5. Track sudo use. The super user do (sudo) program is common on many UNIX/Linux systems to give users additional access without giving them root access. This test case aims to see how effective TAMOS is at auditing this activity.

For each test case we show how the TAMOS auditing subsystem logs and reports on the activity. Prior to running the test cases, there was some environmental setup required.

Environment setup

These test cases were run on a z/Linux system running TAMOS 6.0.

On the z/Linux system there are two specific users we are interested in: root (UID=0) and a new user itsoaud. The root user is a member of the osseal-admin group, which means that it is entitled to perform TAMOS administrative functions but not audit functions. The itsoaud user is a member of the osseal-auditors group, which means that it is entitled to access the TAMOS audit data and commands, but not administrative functions. Example 2 shows the group membership as defined in TAM.

Example 2 TAMOS administrative and auditing groups in pdadmin

```
pdadmin sec_master> group list oss* 0
osseal-admin
osseal-auditors
ossaudit
osseal
pdadmin sec_master> group show-members osseal-admin
root
```

```
osseal
insight
pdadmin sec_master> group show-members osseal-auditors
osseal
insight
itsoaud
pdadmin sec_master>
```

The following sections use these users and the audit settings.

Test case #1 - create another privileged user test case

The power of the root user on UNIX/Linux is of concern to auditors. The root user can perform any function on the system irrespective of the file permissions set and can also wipe any operating system audit trail.

While there is focus on root, there also needs to be a focus on other users with UID=0. This test case looks at the root user creating another user and setting them to UID=0.

Performing the test case

We used a **useradd** command to add the new UID=0 user, as shown in Example 3.

Example 3 root test case 1 execution

```
linux11z:~/scripts # ./tamos_case1.sh
+ date
Tue Nov 13 15:30:57 EST 2007
+ useradd -g root -p secret00 -u 0 -o -r -s /bin/bash testroot
+ date
Tue Nov 13 15:30:57 EST 2007
linux11z:~/scripts # tail -3 /etc/passwd
insight1:x:1101:100::/home/insight1:/bin/sh
itsoaud:x:1105:100::/home/itsoaud:/bin/bash
testroot:x:0:0::/home/testroot:/bin/bash
linux11z:~/scripts #
```

This user is now defined on the system and can act as another root user.

Checking audit log capture

The audit records for this are captured by the TAMOS auditing mechanism and written into the binary audit.log file. The **pdosaudview** command is used to extract and format the records.

As root is not a member of the osseal-auditors group, it cannot run the **pdosaudview** command to access the audit records, as shown in Example 4.

Example 4 root attempts to view the TAMOS audit records

```
linux11z:~ # whoami
root
linux11z:~ # pdoswhoami
root
linux11z:~ # pdosaudview
-bash: /usr/bin/pdosaudview: Permission denied
```

We log in as the itsoaud user and use it to extract the audit records into a readable format, as shown in Example 5.

Example 5 itsoaud user extracts the TAMOS audit records

```
itsoaud@linux11z:/> whoami
itsoaud
itsoaud@linux11z:/> pdoswhoami
itsoaud
itsoaud@linux11z:/> pdosaudview -s now-20
Processed output is in /var/pdos/audit/text.log.
/var/pdos/audit/audit.log.2007-11-13-12-27-20 opened for processing.
/var/pdos/audit/audit.log opened for processing.
```

The **pdosaudview** command has been run without a formatting option, so it writes into a compressed text format. We specified to extract the records starting from twenty minutes ago (-s now-20). The readable audit records were written to /var/pdos/audit/text.log.

Example 6 shows the audit records produced by executing our script (tamos_case1.sh) and the **date** command in the script.

Example 6 TAMOS audit records for user add #1

```
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/root/scripts/tamos_cas
e1.sh (./tamos_case1.sh),APID=16337,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/sh
(./tamos_case1.sh),APID=16337,RPSN=/root/scripts/tamos_case1.sh,UQ=1
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/root/scripts/tamos_cas
e1.sh (/bin/sh -x
./tamos_case1.sh),APID=16337,RPSN=/root/scripts/tamos_case1.sh,UQ=2
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/sh (/bin/sh -x
./tamos_case1.sh),APID=16337,RPSN=/root/scripts/tamos_case1.sh,UQ=3
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/date
(date),APID=16338,RPSN=/root/scripts/tamos_case1.sh,UQ=4
```

Example 7 shows the second set of audit records related to the test script.

Example 7 TAMOS audit records for user add #2

```
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/sbin/useradd
(useradd -g root -p secret00 -u 0 -o -r -s /bin/bash
testroot),APID=16339,RPSN=/root/scripts/tamos_case1.sh,UQ=5
TS=Tue 13 Nov 2007 03:30:57 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/dev/kazndrv,SRN=/dev/kazndrv,APID=16339,RPSN=
/usr/sbin/useradd,O=S,UQ=6
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=28,V=P,R=1,RT=TraceFile,AN=root,AEN=root,A=Trace,P=wr,PRS=/dev/kazndrv,ARS=/
dev/kazndrv,APID=16339,RPSN=/usr/sbin/useradd,UQ=7
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/sbin/useradd.local
```

```
(/usr/sbin/useradd.local testroot 0 0
/home/testroot),APID=16340,RPSN=/usr/sbin/useradd,UQ=8
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/bash
(/usr/sbin/useradd.local testroot 0 0
/home/testroot),APID=16340,RPSN=/usr/sbin/useradd.local,UQ=9
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/sbin/useradd.local
(/bin/bash /usr/sbin/useradd.local testroot 0 0
/home/testroot),APID=16340,RPSN=/usr/sbin/useradd.local,UQ=10
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/bash (/bin/bash
/usr/sbin/useradd.local testroot 0 0
/home/testroot),APID=16340,RPSN=/usr/sbin/useradd.local,UQ=11
TS=Tue 13 Nov 2007 03:30:57 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/date
(date),APID=16341,RPSN=/root/scripts/tamos_case1.sh,UQ=12
```

The first record (highlighted) shows the **useradd** command we used, including all of the command-line options. The following record shows the TAMOS authorization check to execute the **useradd** command.

The focus on these test cases has been to ensure that activity is audited and can be reported on. As an aside, its worthwhile to look at one of these records displayed in a more readable format. The record in Example 8 was produced using the -F verbose option with pdosaudview. This is the record highlighted above.

Example 8 TAMOS audit record in verbose mode

```
*** START OF NEW RECORD ***

Timestamp                                                    Tue 13 Nov 2007
03:30:57 PM EST
Audit Event                                                  TRACE Exec program
Audit View                                                   Trace
Audit Reason                                                 Global Audit
Audit Resource Type                                          TraceExec
Accessor Name                                                root
Accessor Effective Name                                      root
Audit Action                                                 Trace
Accessed Resource Specification                              /usr/sbin/useradd
(useradd -g root -p secret00 -u 0 -o -r -s /bin/bash testroot)
Accessor Process ID                                          16339
Running Program System Resource Name
/root/scripts/tamos_case1.sh
Audit Uniqifier                                              5
```

In this case there is extensive auditing to show that root has performed a useradd. You need to drill through the audit records to identify that a UID=0 user has been created instead of a normal user.

Test case #2 - log in as new user and perform suspect activity test case

This test case continues on from the previous one where a new UID=0 user has been created. In this test case the new user logs in and modifies the /etc/inittab as though he were introducing some malicious code to run at system startup.

Performing the test case

The first step is to log in as the new user. This is shown in Example 9.

Example 9 root test case 2 execution - part 1

```
login as: testroot
Password:
Last login: Tue Nov 13 16:09:59 2007 from 9.12.5.148
Could not chdir to home directory /home/testroot: No such file or directory
/usr/X11R6/bin/xauth:  error in locking authority file /home/testroot/.Xauthority
linux11z:/ # date
Tue Nov 13 16:31:35 EST 2007
linux11z:/ # whoami
testroot
linux11z:/ # pdoswhoami
testroot
linux11z:/ #
```

Both Linux and TAMOS consider this user to be testroot. Example 10 shows the user modifying the inittab file.

Example 10 root Test case 2 execution - part 2

```
linux11z:/etc # vi inittab
... edit file and save ...
linux11z:/etc # ls -ltr
...
-rw-r--r--   1 testroot root      1298 2007-11-13 15:30 passwd
-rw-r-----   1 testroot shadow     695 2007-11-13 16:06 shadow
-rw-r--r--   1 testroot root      1971 2007-11-13 16:32 inittab
drwxr-xr-x  59 testroot root      4096 2007-11-13 16:32 .
linux11z:/etc # date
Tue Nov 13 16:33:31 EST 2007
```

Notice that now every file in the directory has testroot as the owner after modifying the inittab file.

Checking audit log capture

Using the same approach as for the previous section, we can view the audit records associated with this test case. Example 11 shows the audit records for the SSH login, starting with the network connection (the NetIncoming record).

Example 11 testroot user logging onto system using SSH session

```
TS=Tue 13 Nov 2007 04:31:25 PM
EST,E=7,V=P,R=5,RT=NetIncoming,AN=root,AEN=root,A=Check
Access,P=C,Q=34,PBN=zlinux,PON=NetIncoming,NRH=9.12.5.148,NP=tcp,NS=22,APID=1258,R
PPN=/usr/sbin/sshd,RPSN=/usr/sbin/sshd,O=S,UQ=0
```

```
TS=Tue 13 Nov 2007 04:31:29 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/dev/kazndrv,SRN=/dev/kazndrv,APID=16927,RPSN=
/usr/sbin/sshd,O=S,UQ=0
TS=Tue 13 Nov 2007 04:31:29 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/var/pdos/lpm,SRN=/var/pdos/lpm/auth_rw.lock,A
PID=16927,RPPN=/usr/sbin/sshd,RPSN=/usr/sbin/sshd,O=S,UQ=1
TS=Tue 13 Nov 2007 04:31:29 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=l,Q=34,PBN=zlinux,PON=File/var/pdos/login,SRN=/var/pdos/login,APID=16927,
RPPN=/usr/sbin/sshd,RPSN=/usr/sbin/sshd,O=S,UQ=2
TS=Tue 13 Nov 2007 04:31:32 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=l,Q=34,PBN=zlinux,PON=File/var/pdos/login,SRN=/var/pdos/login,APID=16929,
RPPN=/usr/sbin/sshd,RPSN=/usr/sbin/sshd,O=S,UQ=0
...
TS=Tue 13 Nov 2007 04:31:32 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/dev/kazndrv,SRN=/dev/kazndrv,APID=16927,RPSN=
/usr/sbin/sshd,O=S,UQ=5
TS=Tue 13 Nov 2007 04:31:32 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/dev/kazndrv,SRN=/dev/kazndrv,APID=16927,RPSN=
/usr/sbin/sshd,O=S,UQ=6
TS=Tue 13 Nov 2007 04:31:32 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/sh (sh -c
/usr/X11R6/bin/xauth -q -),APID=16932,RPSN=/usr/sbin/sshd,UQ=7
...
```

There are a number of records related to TAMOS checking to see if sshd is considered a valid login mechanism (that is, defined to TAMOS as a login program) and whether the user is allowed to use this login process.

Notice that all records are logged against the user root. There are no records here that indicate that the user logged in as testroot. You could not tie any activity here to the testroot login.

Note also that there is a lot of *noise* generated by having the level of logging turned on. Most of the records shown above have been generated by the TraceFile and TraceExec audit settings.

Example 12 shows the audit records relating to the testroot user performing the **date**, **whoami**, **pdoswhoami**, and **ls** commands.

Example 12 testroot user performing commands

```
TS=Tue 13 Nov 2007 04:31:35 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/date
(date),APID=16951,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 04:31:42 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/bin/whoami
(whoami),APID=16953,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 04:31:45 PM
EST,E=27,V=P,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,PRS=/opt/pdos/bin/pdoswhoam
i,ARS=/usr/bin/pdoswhoami [SG] (pdoswhoami),APID=16954,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 04:31:45 PM
EST,E=28,V=P,R=1,RT=TraceFile,AN=root,AEN=root,A=Trace,P=r,PRS=/opt/pdos/lib,ARS=/
opt/pdos/lib/libosseal.so,APID=16954,RPSN=/usr/bin/pdoswhoami,UQ=1
... (7 similar messages relating to /usr/bin/pdoswhoami removed) ...
```

```
TS=Tue 13 Nov 2007 04:32:27 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/ls (/bin/ls -a -N
--color=tty -T 0 -ltr),APID=16959,RPSN=/bin/bash,UQ=0
```

Again, there is no indication that the user is testroot. Finally, Example 13 shows the user editing the inittab file, running another **`ls`** command and another **`date`** command.

Example 13 testroot user modifying /etc/inittab

```
TS=Tue 13 Nov 2007 04:32:33 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/usr/bin/vi (vi
inittab),APID=16961,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 04:32:59 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=testroot,AEN=testroot,A=Trace,ARS=/bin/ls
(/bin/ls -a -N --color=tty -T 0 -ltr),APID=16964,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 04:33:31 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=testroot,AEN=testroot,A=Trace,ARS=/bin/date
(date),APID=16969,RPSN=/bin/bash,UQ=0
```

It is important to note that the audit records for the **`vi`** command only indicate the file name entered on the command line, which in this case is inittab. If the user has not entered the full path name on the command line, you do not see it in the audit trace.

Also, the auditing has no way of knowing what has been changed in the file. It does not write before and after records.

Notice that after the file has been edited, the audit records now show the user as testroot. This matches with the change of ownership of all the files shown in Example 10 on page 13. At this point the operating system has switched to associating the user *testroot* with UID 0 rather than root. All of the files are still owned by UID 0, but the operating system now associates activity with testroot.

In summary, the TAMOS audit facility has been able to trace the activity of a user, but has not been able to identify activity for the testroot user as distinct from activity for the root user. So in this test case TAMOS auditing is not very effective.

Test case #3 - root attempts to hide audit trail test case

A key concern of the UNIX security model is that the root user can access any file, bypassing the file permission settings. This means that if the operating system produces an audit trail of activity, root can come along and delete the file or modify the file to remove the relevant records.

TAMOS implements its own auditing mechanism and can stop root from modifying the files. The exposure is that if TAMOS is not running, the root user can access the files. This test case shows the root user attempting to access the files while TAMOS is up and then stopping TAMOS to access the files.

Performing the test case

First we attempt to access the TAMOS audit files and copy them to another location, as shown in Example 14.

Example 14 root test case 3 execution - part 1

```
linux11z:~/scripts # date
Tue Nov 13 17:08:22 EST 2007
linux11z:~/scripts # cd /var/pdos/audit
linux11z:/var/pdos/audit # ls -l
/bin/ls: .: Permission denied
linux11z:/var/pdos/audit # cp audit.log /tmp
cp: cannot open `audit.log' for reading: Permission denied
linux11z:/var/pdos/audit # date
Tue Nov 13 17:08:44 EST 2007
linux11z:/var/pdos/audit #
```

In this case TAMOS has stopped root from listing the contents of the directory and copying the file to another location.

Next we shut down TAMOS and repeat the steps, as shown in Example 15.

Example 15 root test case 3 execution - part 2

```
linux11z:/var/pdos/audit # date
Tue Nov 13 17:21:20 EST 2007
linux11z:/var/pdos/audit # pdosctl -k
pdosd shutdown
pdoswdd shutdown
pdoslpmd shutdown
pdosauditd shutdown
linux11z:/var/pdos/audit # ls -l
total 29092
drwxrwx---   2 root    ossaudit    4096 2007-11-13 16:36 .
dr-xr-xr-x  25 osseal  osseal      4096 2006-03-29 12:05 ..
-rw-r-----   1 root    ossaudit  547814 2007-11-13 17:21 audit.log
-rw-r-----   1 root    ossaudit 1000127 2007-10-31 00:05
audit.log.2007-10-31-00-05-49
linux11z:/var/pdos/audit # cp audit.log /tmp
linux11z:/var/pdos/audit # date
Tue Nov 13 17:22:00 EST 2007
linux11z:/var/pdos/audit # rm audit.log
linux11z:/var/pdos/audit # date
Tue Nov 13 17:23:28 EST 2007
```

As root is a TAMOS administrator, it is allowed to shut down the TAMOS deamons. Once TAMOS is down and no longer providing access control over the files, root can view, copy, and delete the audit log files.

Checking audit log capture

Because the root user has deleted the live audit log and TAMOS has created a new one on startup, there is a large hole in the audit record flow between 12:27:19 and 17:24:15, as shown in Example 16.

Example 16 Audit records missing after delete

```
TS=Tue 13 Nov 2007 12:27:19 PM
EST,E=7,V=P,R=5,RT=Surrogate,AN=root,AEN=root,A=Check
Access,P=G,Q=34,PBN=zlinux,PON=Surrogate/Group,SN=postfix,APID=1375,RPSN=/usr/lib/
postfix/master,O=S,UQ=1
TS=Tue 13 Nov 2007 12:27:19 PM
EST,E=7,V=P,R=5,RT=Surrogate,AN=root,AEN=root,A=Check
Access,P=G,Q=34,PBN=zlinux,PON=Surrogate/User,SN=postfix,APID=1375,RPSN=/usr/lib/p
ostfix/master,O=S,UQ=2
TS=Tue 13 Nov 2007 05:24:15 PM
EST,E=35,V=H,R=6,RT=Health,AN=root,AEN=root,A=CertLife,SRN=/var/pdos/certs/pdosd.k
db:APPL_LDAP_CERT,NRH=wtsc60.itso.ibm.com,NP=tcp,NS=3392,APID=17394,AP=Expiration
Date=2017-10-23 Remaining Days=3631,O=S,UQ=0
TS=Tue 13 Nov 2007 05:24:17 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=x,Q=34,PBN=zlinux,PON=File/opt/pdos/bin/pdosshowmsg,SRN=/opt/pdos/bin/pdo
sshowmsg,APID=17524,RPSN=/opt/pdos/bin/rc.osseal,O=S,UQ=0
TS=Tue 13 Nov 2007 05:24:17 PM
EST,E=27,V=P,R=5,RT=TraceExec,AN=root,AEN=root,A=Trace,PRS=/opt/pdos/bin/pdosshowm
sg,ARS=/opt/pdos/bin/pdosshowmsg (/opt/pdos/bin/pdosshowmsg pdosr.cat
35a61082),APID=17524,RPSN=/opt/pdos/bin/rc.osseal,UQ=1
```

The size of this gap is dictated by the frequency of audit logs rolling over. But if the root user can delete the current audit log file, she can also delete the old copies.

Out of curiosity we shut down TAMOS, restored the file that was deleted, and restarted TAMOS. Example 17 shows the audit records for the **date** and **pdosctl** commands corresponding to the activity in Example 15 on page 16.

Example 17 Audit records for TAMOS shutdown

```
TS=Tue 13 Nov 2007 05:21:20 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,ARS=/bin/date
(date),APID=17360,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 05:21:24 PM
EST,E=27,V=P,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,PRS=/opt/pdos/bin/pdosctl,A
RS=/usr/bin/pdosctl [SUG] (pdosctl -k),APID=17361,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 05:21:24 PM EST,E=7,V=P,R=5,RT=File,AN=root,AEN=root,A=Check
Access,P=x,Q=34,PBN=zlinux,PON=File/opt/pdos,SRN=/usr/bin/pdosctl,APID=17361,RPSN=
/bin/bash,O=S,UQ=1
TS=Tue 13 Nov 2007 05:21:24 PM
EST,E=27,V=P,R=1,RT=TraceExec,AN=root,AEN=root,A=Trace,PRS=/opt/pdos,ARS=/usr/bin/
pdosctl [SUG] (pdosctl -k),APID=17361,RPSN=/bin/bash,UQ=2
TS=Tue 13 Nov 2007 05:21:25 PM
EST,E=24,V=A,R=1,RT=Process,AN=root,AEN=root,A=Stop,APID=7328,RPPN=/opt/pdos/bin/p
dosd,RPSN=/opt/pdos/bin/pdosd,O=S,UQ=0
TS=Tue 13 Nov 2007 05:21:27 PM
EST,E=24,V=A,R=1,RT=Process,AN=root,AEN=root,A=Stop,APID=7344,RPPN=/opt/pdos/bin/p
doswdd,RPSN=/opt/pdos/bin/pdoswdd,O=S,UQ=0
```

```
TS=Tue 13 Nov 2007 05:21:29 PM
EST,E=24,V=A,R=1,RT=Process,AN=root,AEN=root,A=Stop,APID=7336,RPPN=/opt/pdos/bin/p
doslpmd,RPSN=/opt/pdos/bin/pdoslpmd,O=S,UQ=0
TS=Tue 13 Nov 2007 05:21:31 PM
EST,E=24,V=A,R=1,RT=Process,AN=root,AEN=root,A=Stop,APID=7332,RPPN=/opt/pdos/bin/p
dosauditd,RPSN=/opt/pdos/bin/pdosauditd,O=S,UQ=0
```

Here we can see the **`date`** command followed by three records relating to the **`pdosctl -k`** command. The last records, with A=Stop, are administrative audit records showing the TAMOS processes being stopped.

How do you solve this dilemma of root being able to shut down TAMOS and delete the files? There are a number of options:

- Implement some form of live backup mechanism that is continuously taking a backup of the audit files that a root user cannot access.
- Implement system monitoring that alerts an operator and logs a trouble ticket when the TAMOS deamons are shutdown. This does not stop the root user from deleting files, but records the fact that they have been shut down. If only a limited number of users are defined as osseal-administrators, there will be a focus on who could have stopped the deamons. This could be supplemented by a monitoring watch on the active audit log. If it disappears then an alarm is raised.
- Remove root from the osseal-admin group so that the user is no longer considered an administrator and cannot shutdown TAMOS. However, depending on procedures, this may cause issues with system maintenance.
- Treat the root user as a special account that can only be logged into from the master console (this can be set as policy in TAM), and the root password is stored securely (such as in a safe) and only accessed by exception (with a trouble ticket raised). When root is used, it is under exceptional circumstances, and its use is closely monitored. If the audit file disappeared during this time, there would be a record of who accessed the root password and logged in as root.

Thus, this exposure can be managed procedurally.

Test case #4 -tracking su use test case

Many customers struggle with the issue of users getting the root password and su’ing to root to perform tasks. They may have a legitimate need and be very disciplined about its use. Or it may be that the root passwords are rarely changed and have become common knowledge over time.

This use case shows an ordinary user su’ing to root and performing some tasks before exiting out.

Performing the test case

This test case involves an ordinary user su’ing to root, as shown in Example 18.

Example 18 root test case 4 execution - part 1

```
login as: jsmith
Password:
Last login: Tue Nov 13 18:18:35 2007 from 9.12.5.148
jsmith@linux11z:~> whoami
jsmith
```

```
jsmith@linux11z:~> pdoswhoami
jsmith
jsmith@linux11z:~> pdoswhoami -a
1106 jsmith
jsmith@linux11z:~> su root
Password:
linux11z:/home/jsmith # whoami
root
linux11z:/home/jsmith # pdoswhoami -a
1106 jsmith
```

Before the su is executed, both the Linux **whoami** command and the TAMOS **pdoswhoami** command show the same user, jsmith. After the user executes su, the operating system considers him root, but TAMOS still tracks his login identity, jsmith (with UID 1106).

Example 19 shows the user performing a series of commands as root.

Example 19 root test case 4 execution - part 2

```
linux11z:/home/jsmith # date
Tue Nov 13 18:52:48 EST 2007
linux11z:/home/jsmith # cd ~/scripts
linux11z:~/scripts # ls
.   add_nativeid.ldif  findjava        lmod   lsrch_ssl
..  amoscfg            junct_create.pd  lsrch  tamos_case1.sh
linux11z:~/scripts # ./lsrch > /dev/nul 2>&1
linux11z:~/scripts # ls -l /etc/passwd
-rw-r--r--  1 root root 1340 2007-11-13 18:15 /etc/passwd
linux11z:~/scripts # touch /etc/passwd
linux11z:~/scripts # ls -l /etc/passwd
-rw-r--r--  1 root root 1340 2007-11-13 18:53 /etc/passwd
linux11z:~/scripts # pdosctl -a
bash: /usr/bin/pdosctl: Permission denied
linux11z:~/scripts # pdosctl -k
bash: /usr/bin/pdosctl: Permission denied
linux11z:~/scripts # date
Tue Nov 13 18:54:05 EST 2007
linux11z:~/scripts # exit
exit
jsmith@linux11z:~> touch /etc/passwd
touch: cannot touch `/etc/passwd': Permission denied
jsmith@linux11z:~> pdosctl -a
-bash: /usr/bin/pdosctl: Permission denied
jsmith@linux11z:~> whoami
jsmith
```

The user may have been able to execute some of the commands as himself, but the /etc/passwd inode entry and running pdosctl require root authority (in our environment). Notice that the user has been allowed to touch the passwd file but not run the **pdosctl** command. Access to touch the passwd file as jsmith was not allowed.

So TAMOS has allowed some access, according to the assumed authority (root), but not allowed us to run the TAMOS control command, as the user is not a member of the osseal-admin group.

Checking audit log capture

As we have a trace level of trace_exec_l enabled, trace records are cut when a user is running under anther identity. Example 20 shows the audit records for jsmith running the **date**, **ls**, and **lsrch** commands.

Example 20 Audit records for su'd user jsmith

```
TS=Tue 13 Nov 2007 06:52:48 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/bin/date
(date),APID=18552,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 06:52:58 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/bin/ls (/bin/ls -a
-N --color=tty -T 0),APID=18554,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 06:53:11 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/root/scripts/lsrch
(./lsrch),APID=18557,RPSN=/bin/bash,UQ=0
TS=Tue 13 Nov 2007 06:53:11 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/bin/sh
(./lsrch),APID=18557,RPSN=/root/scripts/lsrch,UQ=1
TS=Tue 13 Nov 2007 06:53:11 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/root/scripts/lsrch
(/bin/sh ./lsrch),APID=18557,RPSN=/root/scripts/lsrch,UQ=2
TS=Tue 13 Nov 2007 06:53:11 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/bin/sh (/bin/sh
./lsrch),APID=18557,RPSN=/root/scripts/lsrch,UQ=3
TS=Tue 13 Nov 2007 06:53:11 PM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/usr/bin/idsldapsearc
h (idsldapsearch -b o=itso -s sub -h 9.12.4.18 -p 3391 -D cn=LDAP Administrator -w
secret ibm-nativeId=*),APID=18558,RPSN=/root/scripts/lsrch,UQ=4
```

A few points to note about these audit records are:

- All show the login user name (AN=) and effective user name (AEN=), so even though the user is running as root, we can still tie activity to his login identity. This is a very powerful feature.
- When the user executes a script, such as the lsrch script, you see the full path of the executable, not just the local name. This is different from the records cut in Example 13 on page 15, where the file being edited was an argument on a command line.
- The records show the chain of commands in the lsrch script, such as the **idsldapsearch** command, not just the executed script.

The remaining audit records for this test case have not been shown. They are similar to the above.

There was no audit record showing the su to root. This was because the audit settings (trace_exec_root and trace_exec_l) only audit activity by a user who logged in as root or whose effective ID is different from his login ID. At the time of running the **su** command the user jsmith has an effective ID of jsmith.

Test case #5 - tracking sudo use test case

The sudo (super user do) program is common on UNIX/Linux systems to give users the ability to run higher-authority commands without giving them access to the user ID or password to run the commands. It uses a file to define who can run higher-authority commands, what systems they can run these on, and other settings such as whether their password is required, the target user password, or no password at all.

This test case involves an ordinary user running sudo-allowed commands to administer users, such as adding users, setting their passwords, modifying the attributes, and deleting users.

Note: TAMOS provides its own sudo function that is tied to the central TAMOS policy model. Users run the **pdossudo** command, and access to the command is checked against TAMOS SUDO resources in TAM.

Performing the test case

For this test case we turned all auditing on to see what records we got in relation to sudo. The audit levels enabled were permit, deny, admin, loginpermit, logindeny, trace_exec_root, trace_exec_l, and trace_file. For this test case TAMOS was running.

In addition to the TAMOS configuration, we needed to define the sudo configuration file, called /etc/sudoers. This is shown in Example 21.

Example 21 sudoers file for root test case 5

```
linux11z:/usr/sbin # cat /etc/sudoers
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the sudoers man page for the details on how to write a sudoers file.
#
# Host alias specification
# User alias specification
# Cmnd alias specification
Cmnd_Alias      USRADM = /usr/sbin/useradd, /usr/sbin/usermod, \
                        /usr/sbin/userdel, /usr/bin/passwd

# Defaults specification
# Defaults targetpw    # ask for the password of the target user i.e. root
# %users ALL=(ALL) ALL # WARNING! Only use this together with 'Defaults targetpw'!

# User privilege specification
# You should not use sudo as root in an SELinux environment
# If you use SELinux, remove the following line
root    ALL=(ALL) ALL

# Uncomment to allow people in group wheel to run all commands
# %wheel        ALL=(ALL)       ALL

# Same thing without a password
# %wheel        ALL=(ALL)       NOPASSWD: ALL

# Samples
# %users  ALL=/sbin/mount /cdrom,/sbin/umount /cdrom
# %users  localhost=/sbin/shutdown -h now
jsmith          ALL=(ALL)       USRADM
```

Most of the file shown in Example 21 on page 21 is the default file. The lines we configured are highlighted:

- The Cmnd_Alias statement associates the **useradd**, **usermod**, **userdel**, and **passwd** commands with the USRADM label.
- The last line associates this command label with the user jsmith (our test user). The ALL = USRADM means that jsmith can run the user administration commands on all systems where this file is used.

The test case involves jsmith running a series of user administration commands through sudo, as shown in Example 22.

Example 22 root test case 5 execution

```
login as: jsmith
Password:
Last login: Wed Nov 14 08:57:15 2007 from 9.12.5.148
jsmith@linux11z:~> date
Wed Nov 14 09:11:14 EST 2007
jsmith@linux11z:~> sudo -l
Password:
Sorry, try again.
Password:
User jsmith may run the following commands on this host:
    (ALL) /usr/sbin/useradd, /usr/sbin/usermod, /usr/sbin/userdel, /usr/bin/passwd
jsmith@linux11z:~> sudo /usr/sbin/useradd testuser
jsmith@linux11z:~> sudo /usr/bin/passwd testuser
Changing password for testuser.
New password:
Re-enter new password:
Password changed
jsmith@linux11z:~> sudo /usr/sbin/usermod -c "Test User" -p secret00 testuser
jsmith@linux11z:~> grep testuser /etc/passwd
testuser:x:1107:100:Test User:/home/testuser:/bin/bash
jsmith@linux11z:~> sudo /usr/sbin/userdel -r testuser
no crontab for testuser
jsmith@linux11z:~> date
Wed Nov 14 09:13:01 EST 2007
```

There are four user administration commands run in the following sequence; **useradd**, **passwd**, **usermod**, and **userdel**. These are a good example of commands that you would give out to ordinary users through sudo instead of giving them root access.

Checking audit log capture

The default audit log for sudo is the /usr/log/messages file written through syslog. Example 23 shows the entries in the file that relate to the test case activity.

Example 23 /var/log/messages entries for test case

```
Nov 14 09:11:06 linux11z sshd[25214]: Accepted keyboard-interactive/pam for jsmith
from 9.12.5.148 port 2015 ssh2
Nov 14 09:11:25 linux11z sudo:   jsmith : TTY=pts/2 ; PWD=/home/jsmith ; USER=root
; COMMAND=list
Nov 14 09:11:49 linux11z sudo:   jsmith : TTY=pts/2 ; PWD=/home/jsmith ; USER=root
; COMMAND=/usr/sbin/useradd testuser
```

```
Nov 14 09:11:49 linux11z useradd: new user: name=testuser, uid=1107, gid=100,
home=/home/testuser, shell=/bin/bash
Nov 14 09:12:02 linux11z sudo:   jsmith : TTY=pts/2 ; PWD=/home/jsmith ; USER=root
; COMMAND=/usr/bin/passwd testuser
Nov 14 09:12:33 linux11z sudo:   jsmith : TTY=pts/2 ; PWD=/home/jsmith ; USER=root
; COMMAND=/usr/sbin/usermod -c Test User -p secret00 testuser
Nov 14 09:12:56 linux11z sudo:   jsmith : TTY=pts/2 ; PWD=/home/jsmith ; USER=root
; COMMAND=/usr/sbin/userdel -r testuser
Nov 14 09:12:56 linux11z crontab[25258]: (root) DELETE (testuser)
```

The audit log is quite effective. It identifies the login user (jsmith), the directory she is running the command from, the user the command is being run as (root), and the command that was run. It is reasonably simple to translate these messages into TCIMs W7 format using a tool like TDI.

The TAMOS audit log also shows this activity. Some of the records associated with the **useradd** command are shown in Example 24.

Example 24   TAMOS audit log for sudo useradd command

```
TS=Wed 14 Nov 2007 09:11:49 AM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/usr/sbin/useradd
(/usr/sbin/useradd testuser),APID=25245,RPSN=/usr/bin/sudo,UQ=10
TS=Wed 14 Nov 2007 09:11:49 AM EST,E=7,V=P,R=1,RT=File,AN=jsmith,AEN=root,A=Check
Access,P=wr,Q=34,PBN=zlinux,PON=File/dev/kazndrv,SRN=/dev/kazndrv,APID=25245,RPSN=
/usr/sbin/useradd,O=S,UQ=11
TS=Wed 14 Nov 2007 09:11:49 AM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/usr/sbin/useradd.loc
al (/usr/sbin/useradd.local testuser 1107 100
/home/testuser),APID=25246,RPSN=/usr/sbin/useradd,UQ=13
TS=Wed 14 Nov 2007 09:11:49 AM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/bin/bash
(/usr/sbin/useradd.local testuser 1107 100
/home/testuser),APID=25246,RPSN=/usr/sbin/useradd.local,UQ=14
TS=Wed 14 Nov 2007 09:11:49 AM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/usr/sbin/useradd.loc
al (/bin/bash /usr/sbin/useradd.local testuser 1107 100
/home/testuser),APID=25246,RPSN=/usr/sbin/useradd.local,UQ=15
TS=Wed 14 Nov 2007 09:11:49 AM
EST,E=27,V=T,R=1,RT=TraceExec,AN=jsmith,AEN=root,A=Trace,ARS=/bin/bash (/bin/bash
/usr/sbin/useradd.local testuser 1107 100
/home/testuser),APID=25246,RPSN=/usr/sbin/useradd.local,UQ=16
```

Note that TAMOS considers this access as surrogate, the same as though the user had performed a su to root. The audit records show the login user (AN=) and the effective user (AEN=). The audit trace also shows the command arguments used by the lower level commands (such as the **useradd.local** command).

The audit trace detail for the **passwd**, **usermod**, and **userdel** commands is similar.

Auditing UNIX/Linux systems with TAMOS and TCIM

There are a number of ways in which the audit reporting can be performed for these test cases. On the most basic level you could use the **pdosaudview** command to produce a comma-separated format output and feed it into Microsoft® Excel®, Microsoft Access, or some other reporting tool.

In this book we focus on reporting in TCIM. It provides actuators to access the raw TAMOS audit data and the mapping of TAMOS records into the TCIM W7 reporting format.

Note: We noticed that TCIM was no longer collecting TAMOS audit records after a certain time. This time seemed to coincide with the test case of deleting the TAMOS audit log. This action may have corrupted a link to the file. Re-installing the actuator resolved the issue.

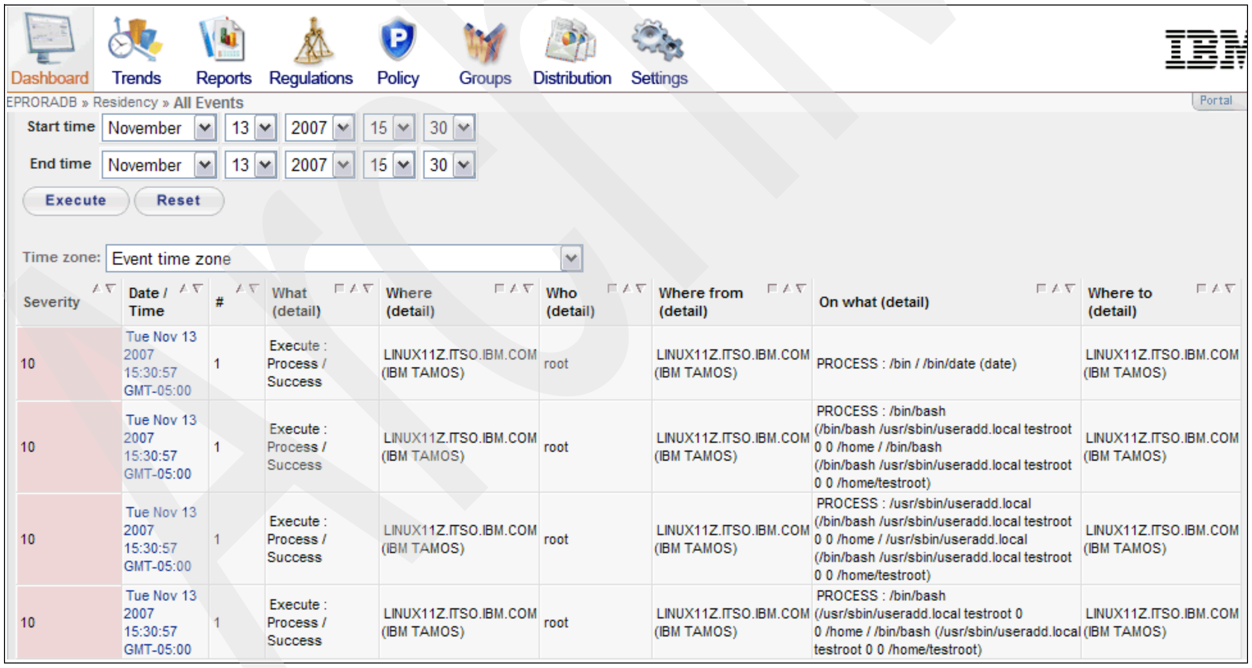
The following sections provide examples of standard and custom reporting against this test case data.

Standard TCIM reporting for TAMOS test case data

This section shows how the TCIM All Events view can show the TAMOS audit data for the test cases.

TCIM events for the creation of another privilege user test case

The first TAMOS test case ("Test case #1 - create another privileged user test case" on page 10) involved root creating another privileged user (that is, with UID=0). The TAMOS audit records for this were shown in Example 6 on page 11 and Example 7 on page 11. The TCIM mapping pulls all TAMOS audit records from TAMOS. Figure 3 shows some of these records, beginning with the first **`date`** command.



Start time: November 13 2007 15 30

End time: November 13 2007 15 30

Time zone: Event time zone

Severity	Date / Time	#	What (detail)	Where (detail)	Who (detail)	Where from (detail)	On what (detail)	Where to (detail)
10	Tue Nov 13 2007 15:30:57 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin / /bin/date (date)	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)
10	Tue Nov 13 2007 15:30:57 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin/bash (/bin/bash /usr/sbin/useradd.local testroot 0 0 /home / /bin/bash (/bin/bash /usr/sbin/useradd.local testroot 0 0 /home/testroot)	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)
10	Tue Nov 13 2007 15:30:57 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /usr/sbin/useradd.local (/bin/bash /usr/sbin/useradd.local testroot 0 0 /home / /usr/sbin/useradd.local (/bin/bash /usr/sbin/useradd.local testroot 0 0 /home/testroot)	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)
10	Tue Nov 13 2007 15:30:57 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin/bash (/usr/sbin/useradd.local testroot 0 0 /home / /bin/bash (/usr/sbin/useradd.local testroot 0 0 /home/testroot)	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)

Figure 3 TCIM report for TAMOS test case

The mapping process has categorized the TAMOS data in the W7 mapping process. The TraceExec audit record for the **`date`** command has been mapped as follows:

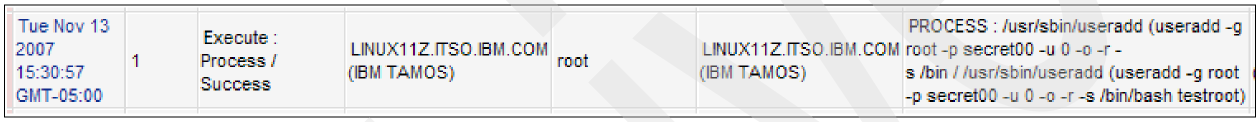
- The *when* (date/time*)* value has been derived from the time stamp in the audit record.
- The *what* value has been set to Execute: Process / Success as the TraceExec record showed a success.

- The *where* value is a combination of the server ID (host name) and the actuator (IBM TAMOS).
- The *who* value is the user issuing the **`date`** command.
- The *WhereFrom* and *WhereTo* values are the same as the where (these would be the same except for NetOutgoing and NetIncoming TAMOS records).
- The *OnWhat* value is a combination of PROCESS and the process name.

You will notice that a severity has been assigned in the TCIM load process based on some default policy associated with the TAMOS audit records.

If you look at the Date/Time field you will notice that the time is only recorded to the second, not hundredth of a second that TAMOS records them as. This can result in records appearing out of time order in TCIM. The records shown above for the **`useradd.local`** command appear before the **`useradd`** command record (see Figure 4). However, in the TAMOS audit log (Example 7 on page 11), the **`useradd`** command comes before the **`useradd.local`** commands. You should be careful associating a sequence of activities based on the sequence of records displayed in TCIM.

Figure 4 shows the **`useradd`** command from test case 1.



Tue Nov 13 2007 15:30:57 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /usr/sbin/useradd (useradd -g root -p secret00 -u 0 -o -r -s /bin / /usr/sbin/useradd (useradd -g root -p secret00 -u 0 -o -r -s /bin/bash testroot)

Figure 4 TCIM record for TAMOS user add

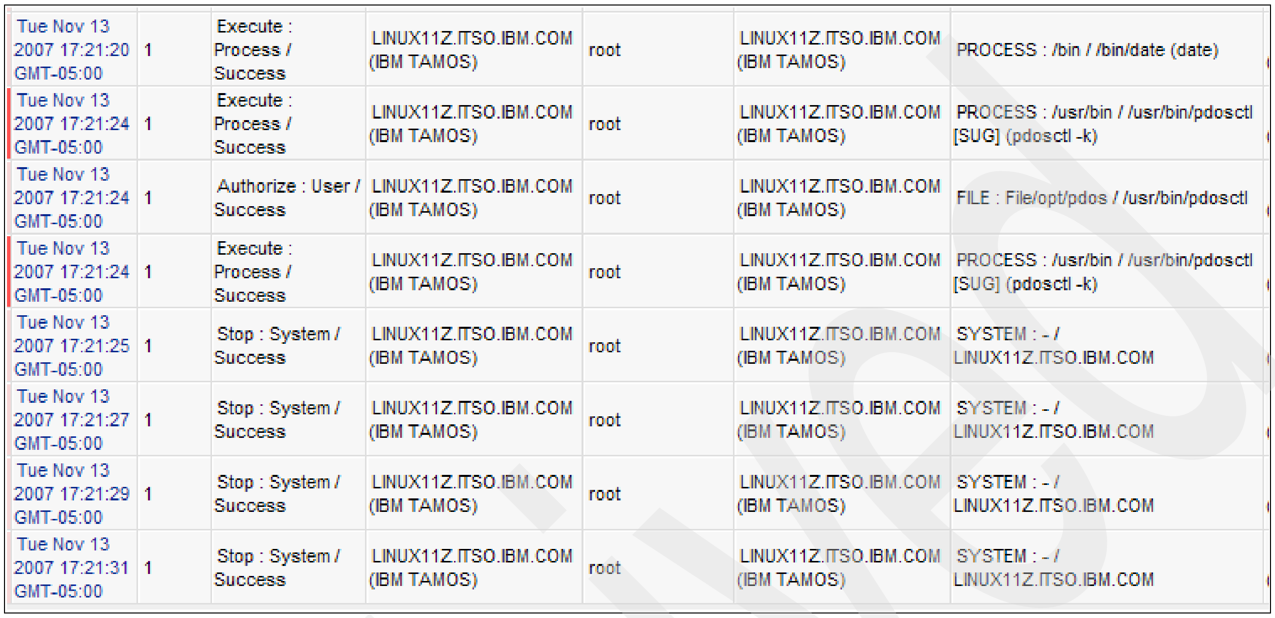
Like the TAMOS audit record, the entire command with arguments is shown. In this case it includes the password being set for the new user. If there will be this type of information available through TCIM, you need to consider who has access to view the data.

TCIM events for the new user login test case

With test case 2 ("Test case #2 - log in as new user and perform suspect activity test case" on page 13) we demonstrated that the TAMOS audit log did not identify that the new UID 0 user (testroot) logged into the system. This would not be different under TCIM.

TCIM events for attempt to hide audit trail test case

With test case 3 (“Test case #3 - root attempts to hide audit trail test case” on page 15), the root user shut down TAMOS and deleted the TAMOS audit log. We can see the shutdown commands in TCIM, as shown in Figure 5.



Tue Nov 13 2007 17:21:20 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin / /bin/date (date)
Tue Nov 13 2007 17:21:24 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /usr/bin / /usr/bin/pdosctl [SUG] (pdosctl -k)
Tue Nov 13 2007 17:21:24 GMT-05:00	1	Authorize : User / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	FILE : File/opt/pdos / /usr/bin/pdosctl
Tue Nov 13 2007 17:21:24 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /usr/bin / /usr/bin/pdosctl [SUG] (pdosctl -k)
Tue Nov 13 2007 17:21:25 GMT-05:00	1	Stop : System / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	SYSTEM : - / LINUX11Z.ITSO.IBM.COM
Tue Nov 13 2007 17:21:27 GMT-05:00	1	Stop : System / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	SYSTEM : - / LINUX11Z.ITSO.IBM.COM
Tue Nov 13 2007 17:21:29 GMT-05:00	1	Stop : System / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	SYSTEM : - / LINUX11Z.ITSO.IBM.COM
Tue Nov 13 2007 17:21:31 GMT-05:00	1	Stop : System / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	root	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	SYSTEM : - / LINUX11Z.ITSO.IBM.COM

Figure 5 TCIM records for TAMOS shutdown command

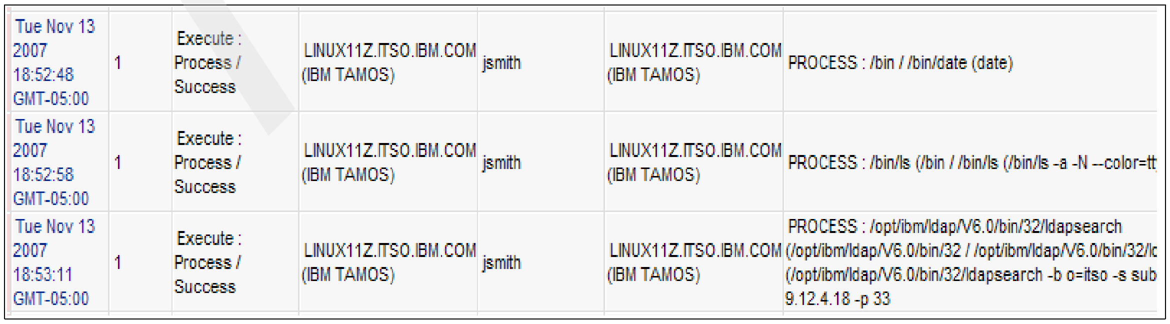
There are three records for the shutdown command **`pdosctl -k`**. These are from the TAMOS audit records shown in Example 17 on page 17.

The last four records show a successful system stop. These are misleading. The system has not stopped. The four TAMOS deamons have been stopped. Compare these records with the TAMOS audit records shown in Example 17 on page 17. They show the four TAMOS processes (pdosd, pdoswdd, pdoslpmd, and pdosauditd) being shut down. This is one case where relevant data has not been effectively mapped from the TAMOS audit records to TCIM.

Note that there are no records relating to the TAMOS audit files being copied and deleted. This is because TAMOS itself was down, and so could not cut records. To cover this auditing hole would require operating system level monitoring of the audit files. TCIM could then show the activities chronologically.

TCIM events for su tracking test case

With test case 4 (“Test case #4 -tracking su use test case” on page 18) the user jsmith logged in, su’d to root, and performed some activities. Some of these activities are shown in Figure 6.



Tue Nov 13 2007 18:52:48 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin / /bin/date (date)
Tue Nov 13 2007 18:52:58 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin/ls (/bin / /bin/ls (/bin/ls -a -N --color=tt
Tue Nov 13 2007 18:53:11 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /opt/ibm/ldap/V6.0/bin/32/ldapsearch (/opt/ibm/ldap/V6.0/bin/32 / /opt/ibm/ldap/V6.0/bin/32/lc (/opt/ibm/ldap/V6.0/bin/32/ldapsearch -b o=itso -s sub 9.12.4.18 -p 33

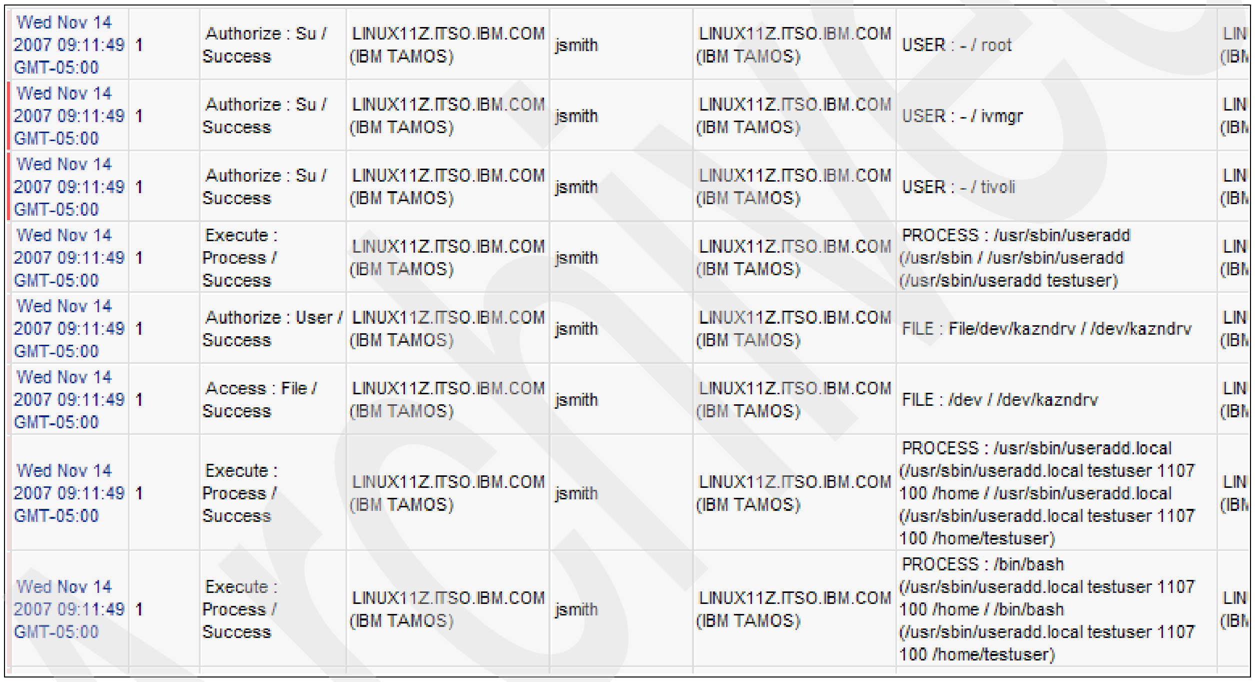
Figure 6 TCIM records for jsmith su’d to root

While these records show what activity jsmith was performing on the system, there is no indication that the user has su'd to root and was running under the higher authority. While the TAMOS audit records show that the login user ID is different from the effective user, this information has not been mapped across to the TCIM records. The W7 format does not seem to allow for two levels of who (login who and effective who) in process records.

As mentioned previously, there is no record in the TAMOS audit log of the user performing the su to root due to the level of TAMOS tracing turned on. This means that there is no record of this in TCIM. Had the original su record been audited by TAMOS and carried to TCIM, you would see the user su'ing to root and then the activity shown above and could assume that they were still running with the higher authority.

TCIM events for the sudo use test case

With test case 5 ("Test case #5 - tracking sudo use test case" on page 20) we used the sudo command to allow jsmith to perform some user administration tasks. Some of the TCIM records are shown in Figure 7.



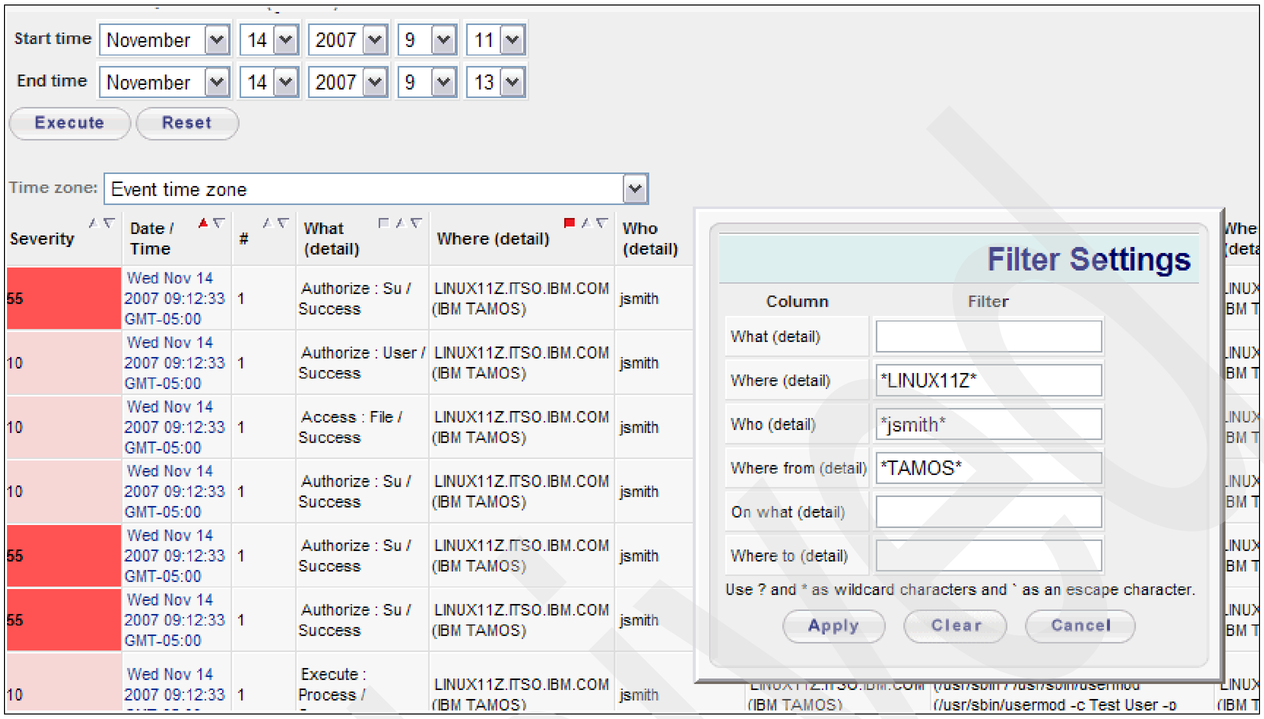
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	USER : - / root	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	USER : - / ivmgr	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	USER : - / tivoli	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /usr/sbin/useradd (/usr/sbin / /usr/sbin/useradd (/usr/sbin/useradd testuser)	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Authorize : User / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	FILE : File/dev/kazndrv / /dev/kazndrv	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	FILE : /dev / /dev/kazndrv	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /usr/sbin/useradd.local (/usr/sbin/useradd.local testuser 1107 100 /home / /usr/sbin/useradd.local (/usr/sbin/useradd.local testuser 1107 100 /home/testuser)	LIN (IB
Wed Nov 14 2007 09:11:49 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	PROCESS : /bin/bash (/usr/sbin/useradd.local testuser 1107 100 /home / /bin/bash (/usr/sbin/useradd.local testuser 1107 100 /home/testuser)	LIN (IB

Figure 7 TCIM records for jsmith using sudo

Unlike with the su test case, the sudo commands show records in TCIM. These are the "Authorize : Su / Success" records shown in Figure 7. However, there is no indication in the execute records that the commands were run through sudo.

As with the TAMOS audit records, you can see the command line arguments, such as for the **`useradd.local`** command, so care needs to be taken with access to this data.

It is worth highlighting the search filtering capabilities of TCIM. Figure 8 shows the filtering options to show the records for the last test case.



Start time November 14 2007 9 11

End time November 14 2007 9 13

Time zone: Event time zone

Severity	Date / Time	#	What (detail)	Where (detail)	Who (detail)
55	Wed Nov 14 2007 09:12:33 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith
10	Wed Nov 14 2007 09:12:33 GMT-05:00	1	Authorize : User / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith
10	Wed Nov 14 2007 09:12:33 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith
10	Wed Nov 14 2007 09:12:33 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith
55	Wed Nov 14 2007 09:12:33 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith
55	Wed Nov 14 2007 09:12:33 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith
10	Wed Nov 14 2007 09:12:33	1	Execute : Process /	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	jsmith

Filter Settings

Column	Filter
What (detail)	
Where (detail)	*LINUX11Z*
Who (detail)	*jsmith*
Where from (detail)	*TAMOS*
On what (detail)	
Where to (detail)	

Use ? and * as wildcard characters and ` as an escape character.

Figure 8 TCIM filtering options

On the top left of Figure 8 are the start and end time filters. We also used the filter settings accessible through the small box icons in the column headers to further reduce the data shown (note that if there is a filter applied to one of the columns, this box is shown as red).

Custom TCIM reporting for TAMOS test case data

It would be common in most deployments to build custom policy to enable better searching of data and to enable TCIM automated actions. This section details some custom policy created to highlight some of the events created in the test cases.

Custom policy - highlight su to root

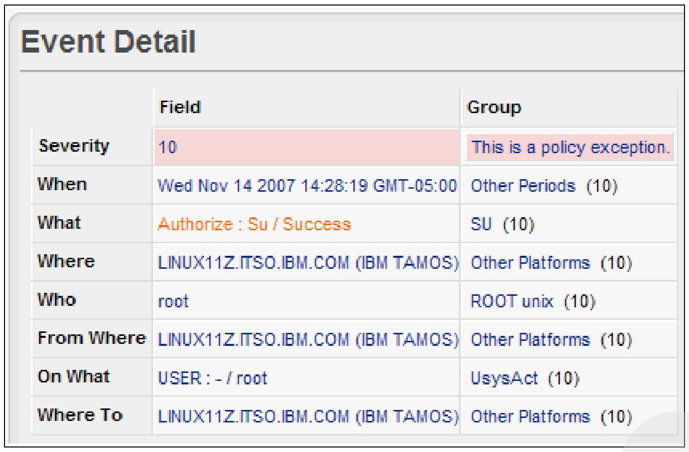
The first condition we wanted to highlight was when a user su's to root. We need to be careful, as there are many audit records cut when system processes run as root, so we need to make an exception for system accounts.

TCIM policy is built from a set of groupings that can be applied to policy rules and attention rules. The groupings are defined using the TCIM Console and involve record filters based on boolean logic. Getting the logic correct in the UI can be challenging until you understand how the conditions work.

The simplest approach is to define what conditions you want in your grouping and look at the that you have available. For this example we want to:

- Capture all records where a su to root is performed.
- Discard any records where the originator is a system account.

Figure 9 shows one of the su records in TCIM that we can use to help build the filters.



Event Detail

	Field	Group
Severity	10	This is a policy exception.
When	Wed Nov 14 2007 14:28:19 GMT-05:00	Other Periods (10)
What	Authorize : Su / Success	SU (10)
Where	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	Other Platforms (10)
Who	root	ROOT unix (10)
From Where	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	Other Platforms (10)
On What	USER : - / root	UsysAct (10)
Where To	LINUX11Z.ITSO.IBM.COM (IBM TAMOS)	Other Platforms (10)

Figure 9 Event detail for su record

The two fields we are interested in are the What field, showing the su activity, and the On What field, showing the user performing the su.

We need to set up filters that show records of where the What field contains "Authorize : Su / Success" and the OnWhat field contains "USER : - / *name*", where *name* is not one of the system accounts.

We look at the What condition first. You see that each field and group in the event detail is a link. If you hover the mouse over the data in the What Field cell, a pop-up shows the structure of the field:

```
mainclass:eventclass/successclass - Show event list for Event type "Authorize : Su
/ Success"
```

In this What field:

- mainclass = Authorize
- eventclass = Su
- successclass = Success

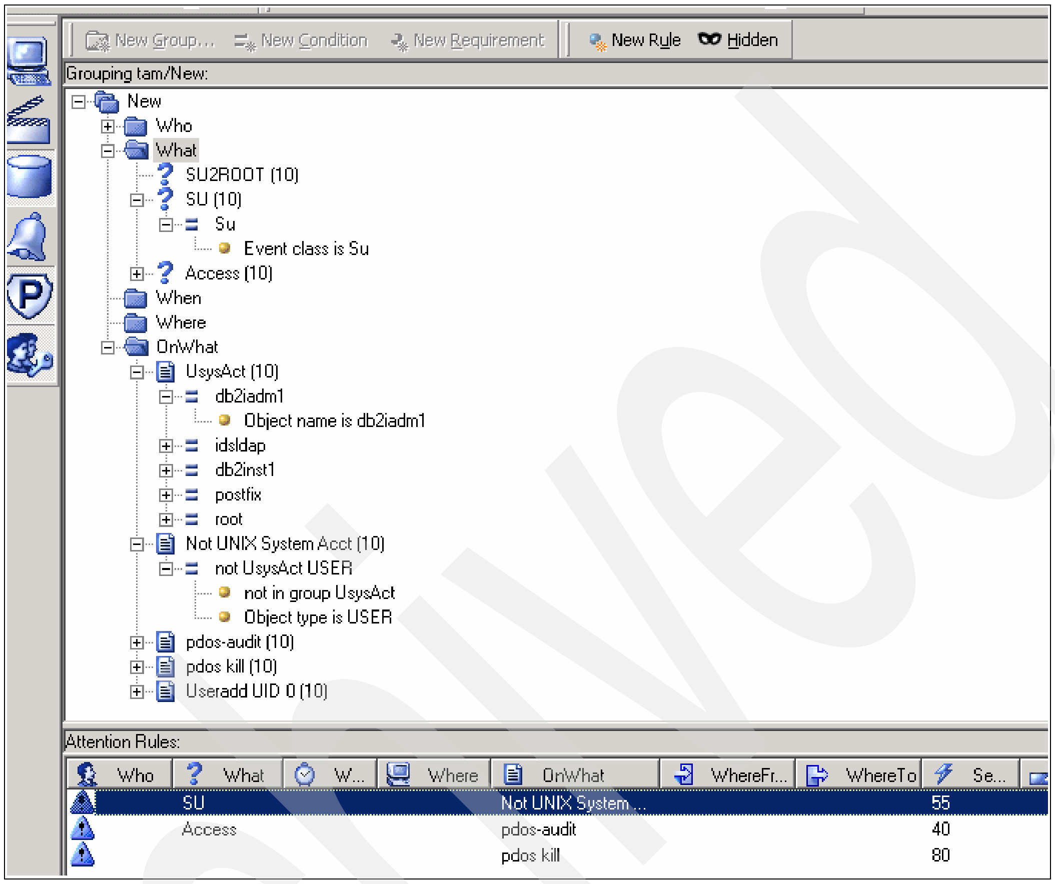
Using the same approach for the On What field we can see that:

- type = USER
- path = -
- name = root

These arguments can be used in the TCIM console to define the grouping filters. The filters we want to apply for this are:

- What; eventclass = Su
- OnWhat; type = USER and name not in (list of system accounts)

The first is simple to implement. The second requires that an OnWhat group be created containing all of the system accounts. Then a condition can be created where the type is USER and the name is not in this group. The implementation of these is shown in Figure 10.



Grouping tam/New:

```
New
  Who
  What
    SU2ROOT (10)
    SU (10)
      Su
        Event class is Su
    Access (10)
  When
  Where
  OnWhat
    UsysAct (10)
      db2iadm1
        Object name is db2iadm1
      idsldap
      db2inst1
      postfix
      root
    Not UNIX System Acct (10)
      not UsysAct USER
        not in group UsysAct
        Object type is USER
    pdos-audit (10)
    pdos kill (10)
    Useradd UID 0 (10)
```

Attention Rules:

Who	What	W...	Where	OnWhat	WhereFr...	WhereTo	Se...
	SU			Not UNIX System ...			55
	Access			pdos-audit			40
				pdos kill			80

Figure 10   Grouping conditions for su to root policy

Towards the top of Figure 10 there is the What condition, where Eventclass is Su.

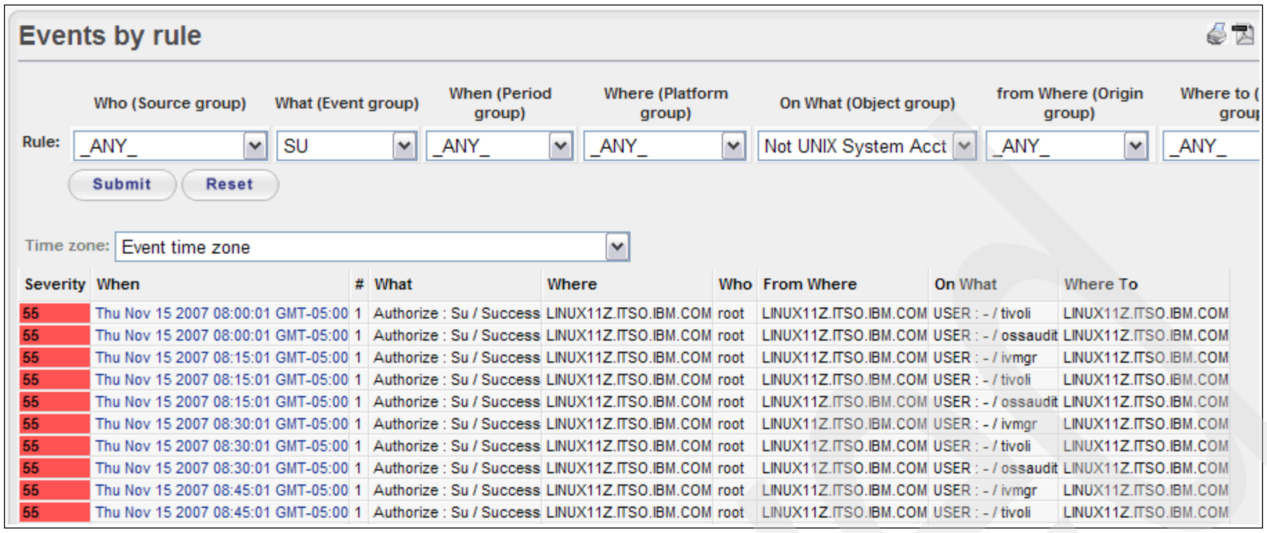
Under the OnWhat branch there is a *group* called UsysAct that contains the accounts we defined as UNIX system accounts. Each one of the *conditions* under this grouping is joined with an OR. Each of these conditions contains a single *requirement* with the object name as *name*. So the group UsysAct's object name is db2iadm1 OR the object name is idsldap OR the object name is db2inst1, and so on.

Under this is a group called Not UNIX System Act. This contains a single condition of not UsysAct USER. This condition has two requirements:

- Not in group UsysAct
- Object type is USER

Requirements are joined with an AND. So the grouping statement is Object type is USER and user not in group UsysAct.

These policy rules can now be used to filter the display in the TCIM iView. Figure 11 shows some records with the SU What rule and the Not UNIX System Accts On what rule.



Events by rule

	Who (Source group)	What (Event group)	When (Period group)	Where (Platform group)	On What (Object group)	from Where (Origin group)	Where to (grou
Rule:	_ANY_	SU	_ANY_	_ANY_	Not UNIX System Acct	_ANY_	_ANY_

Time zone: Event time zone

Severity	When	#	What	Where	Who	From Where	On What	Where To
55	Thu Nov 15 2007 08:00:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / tivoli	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:00:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / ossaudit	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:15:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / ivmgr	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:15:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / tivoli	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:15:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / ossaudit	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:30:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / ivmgr	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:30:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / tivoli	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:30:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / ossaudit	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:45:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / ivmgr	LINUX11Z.ITSO.IBM.COM
55	Thu Nov 15 2007 08:45:01 GMT-05:00	1	Authorize : Su / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	USER : - / tivoli	LINUX11Z.ITSO.IBM.COM

Figure 11 TCIM Events by Rule report with custom policy applied

Note that the report still contains system accounts, such as tivoli and ivmgr. You need to further tune the report to add these to the list of system accounts.

You can see that the severity for these events is 55. This has been set by creating an attention rule, as shown in Figure 10 on page 30.

Custom policy - highlight access to the TAMOS audit files

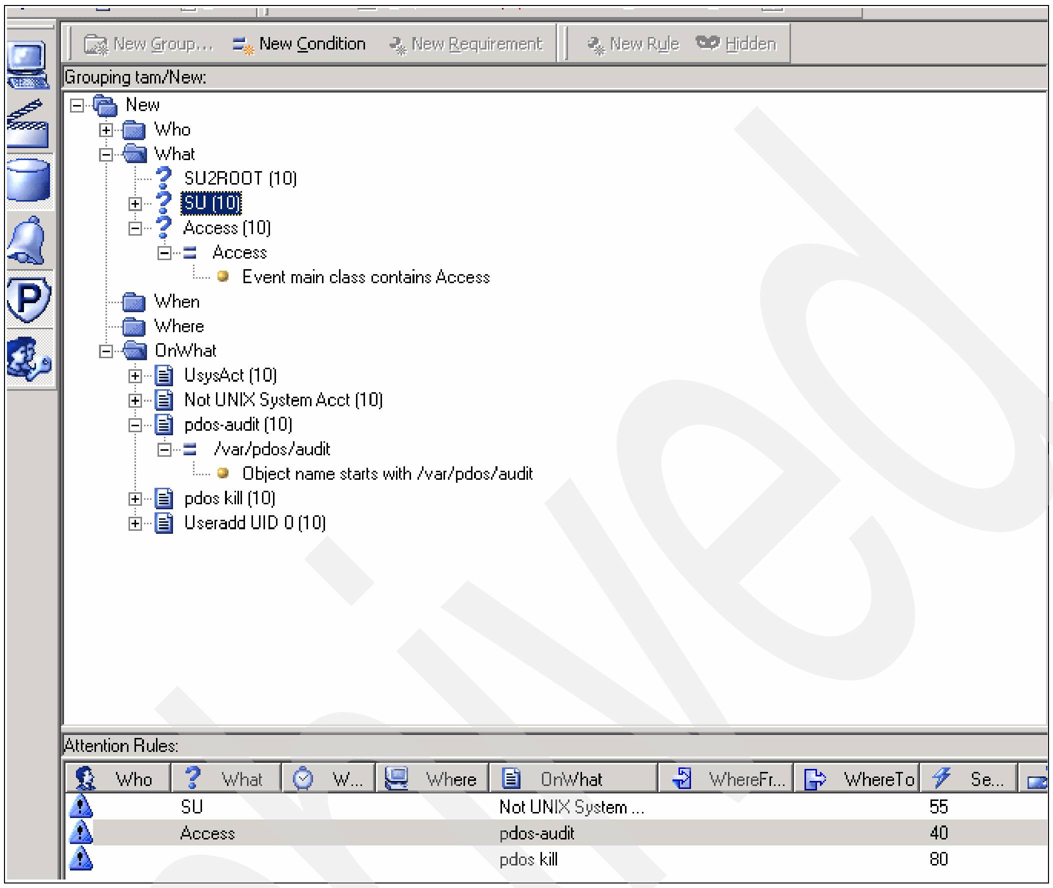
The next condition we want to highlight was when someone accesses the TAMOS audit files. TAMOS tightly controls the files, but you might want to monitor access against them.

The policy rule that we want to apply is to show all access to files in the /var/pdos/audit directory.

Using the same approach as in the previous section, we define two conditions:

- What: event main class contains access
- OnWhat: object name starts with /vaer/pdos/audit

The implementation of these into TCIM groups, conditions, and requirements is shown in Figure 12.



Grouping tam/New:

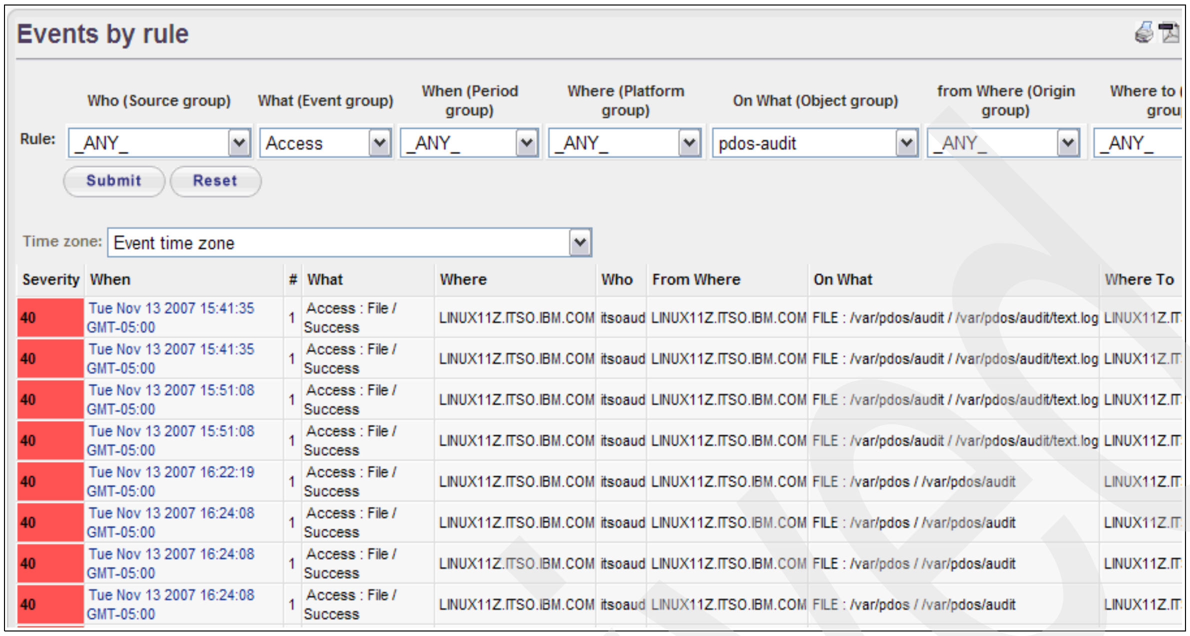
```
New
  Who
  What
    SU2ROOT (10)
    SU (10)
    Access (10)
      Access
        Event main class contains Access
  When
  Where
  OnWhat
    UsysAct (10)
    Not UNIX System Acct (10)
    pdos-audit (10)
      /var/pdos/audit
        Object name starts with /var/pdos/audit
    pdos kill (10)
    Useradd UID 0 (10)
```

Attention Rules:

Who	What	W...	Where	OnWhat	WhereFr...	WhereTo	Se...
	SU			Not UNIX System ...			55
	Access			pdos-audit			40
				pdos kill			80

Figure 12   Grouping conditions for access to TAMOS audit files policy

There is only a single requirement for each condition and a single condition for each group. Figure 13 shows these filter rules applied in the TCIM iView Events by Type report.


Events by rule

	Who (Source group)	What (Event group)	When (Period group)	Where (Platform group)	On What (Object group)	from Where (Origin group)	Where to grou
Rule:	_ANY_	Access	_ANY_	_ANY_	pdos-audit	_ANY_	_ANY_

Submit Reset

Time zone: Event time zone

Severity	When	#	What	Where	Who	From Where	On What	Where To
40	Tue Nov 13 2007 15:41:35 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos/audit / /var/pdos/audit/text.log	LINUX11Z.IT
40	Tue Nov 13 2007 15:41:35 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos/audit / /var/pdos/audit/text.log	LINUX11Z.IT
40	Tue Nov 13 2007 15:51:08 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos/audit / /var/pdos/audit/text.log	LINUX11Z.IT
40	Tue Nov 13 2007 15:51:08 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos/audit / /var/pdos/audit/text.log	LINUX11Z.IT
40	Tue Nov 13 2007 16:22:19 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos / /var/pdos/audit	LINUX11Z.IT
40	Tue Nov 13 2007 16:24:08 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos / /var/pdos/audit	LINUX11Z.IT
40	Tue Nov 13 2007 16:24:08 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos / /var/pdos/audit	LINUX11Z.IT
40	Tue Nov 13 2007 16:24:08 GMT-05:00	1	Access : File / Success	LINUX11Z.ITSO.IBM.COM	itsoaud	LINUX11Z.ITSO.IBM.COM	FILE : /var/pdos / /var/pdos/audit	LINUX11Z.IT

Figure 13   TCIM Events by Rule report with custom policy applied

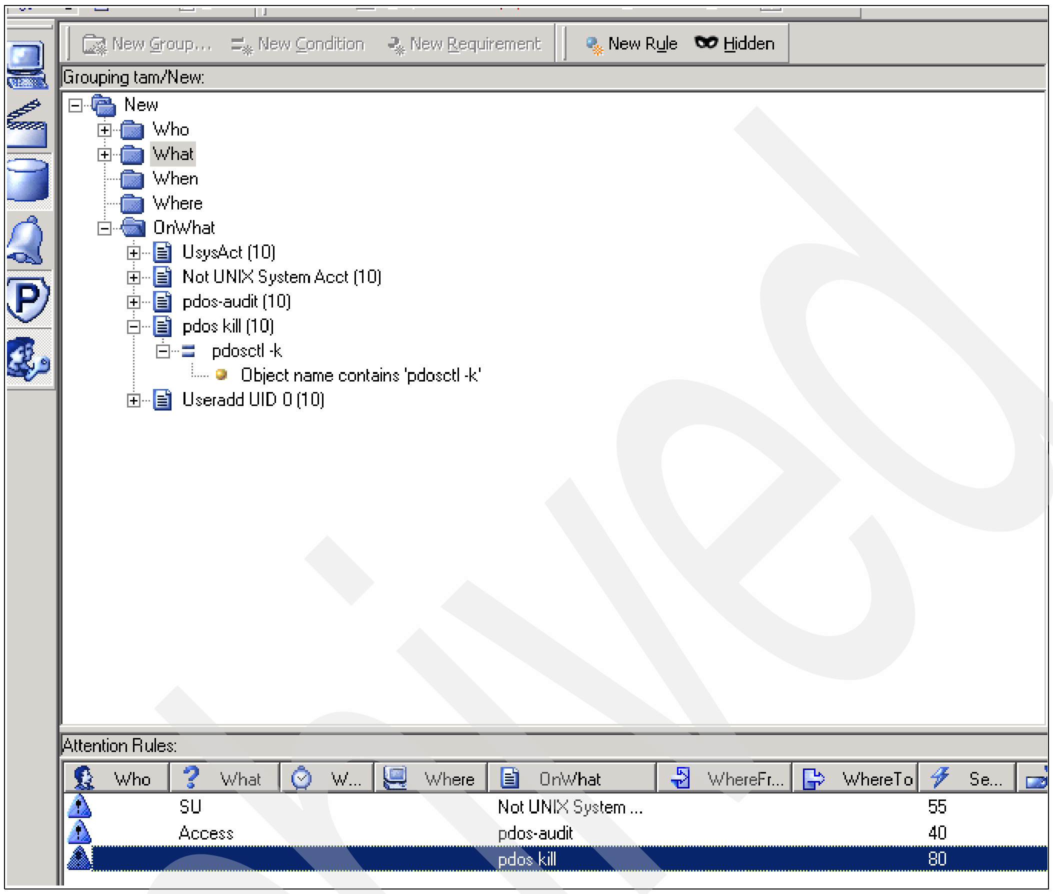
An attention rule was built with the combination of the Access What group and the pdos-audit On What group with a severity of 40.

Custom policy - highlight use of the TAMOS process kill command

The next condition we want to highlight is when someone uses the TAMOS kill command (**`pdosctl -k`**). For normal operation there should be no need to kill the processes, so this situation is worth highlighting.

The policy rule we want to apply is to show all execution of **`pdosctl -k`**. This only required a single OnWhat condition, where the object name contains pdosctl -k.

The implementation of these into TCIM groups, conditions, and requirements is shown in Figure 14.



Grouping tam/New:

- New
  - Who
  - What
  - When
  - Where
  - OnWhat
    - UsysAct (10)
    - Not UNIX System Acct (10)
    - pdos-audit (10)
    - pdos kill (10)
      - pdosctl -k
        - Object name contains 'pdosctl -k'
    - Useradd UID 0 (10)

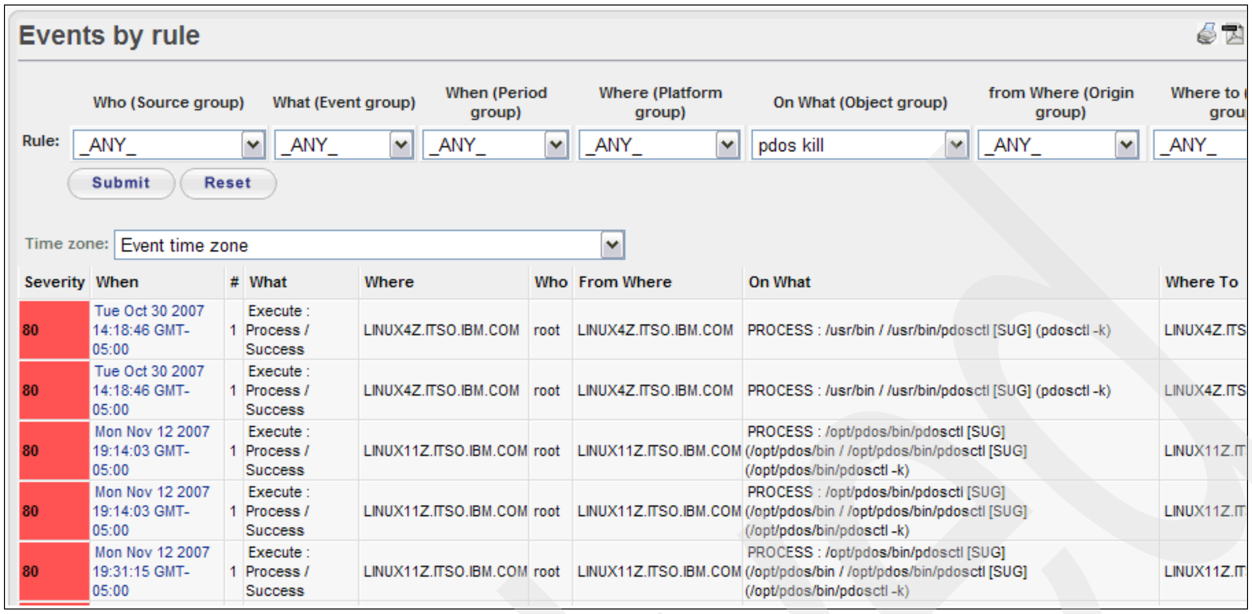
Attention Rules:

Who	What	W...	Where	OnWhat	WhereFr...	WhereTo	Se...
⚠	SU			Not UNIX System ...			55
⚠	Access			pdos-audit			40
⚠				pdos kill			80

Figure 14 Grouping conditions for running the TAMOS kill command policy

The single requirement of the object name contains pdosctl -k under the pdos kill group. Note that this requires the command to have a single space between the pdosctl and the -k strings. You could also code two separate requirements under the same condition with one for the object name containing pdosctl and the other for the object name containing -k.

The TCIM Events by Rule report with this grouping applied is shown in Figure 15.



Events by rule

	Who (Source group)	What (Event group)	When (Period group)	Where (Platform group)	On What (Object group)	from Where (Origin group)	Where to grou
Rule:	_ANY_	_ANY_	_ANY_	_ANY_	pdos kill	_ANY_	_ANY_

Time zone: Event time zone

Severity	When	#	What	Where	Who	From Where	On What	Where To
80	Tue Oct 30 2007 14:18:46 GMT-05:00	1	Execute : Process / Success	LINUX4Z.ITSO.IBM.COM	root	LINUX4Z.ITSO.IBM.COM	PROCESS : /usr/bin / /usr/bin/pdosctl [SUG] (pdosctl -k)	LINUX4Z.ITS
80	Tue Oct 30 2007 14:18:46 GMT-05:00	1	Execute : Process / Success	LINUX4Z.ITSO.IBM.COM	root	LINUX4Z.ITSO.IBM.COM	PROCESS : /usr/bin / /usr/bin/pdosctl [SUG] (pdosctl -k)	LINUX4Z.ITS
80	Mon Nov 12 2007 19:14:03 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	PROCESS : /opt/pdos/bin/pdosctl [SUG] (/opt/pdos/bin / /opt/pdos/bin/pdosctl [SUG] (/opt/pdos/bin/pdosctl -k)	LINUX11Z.IT
80	Mon Nov 12 2007 19:14:03 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	PROCESS : /opt/pdos/bin/pdosctl [SUG] (/opt/pdos/bin / /opt/pdos/bin/pdosctl [SUG] (/opt/pdos/bin/pdosctl -k)	LINUX11Z.IT
80	Mon Nov 12 2007 19:31:15 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	PROCESS : /opt/pdos/bin/pdosctl [SUG] (/opt/pdos/bin / /opt/pdos/bin/pdosctl [SUG] (/opt/pdos/bin/pdosctl -k)	LINUX11Z.IT

Figure 15 TCIM Events by Rule report with custom policy applied

An attention rule was built with the combination of the Access What group and the pdos-audit On What group with a severity of 80.

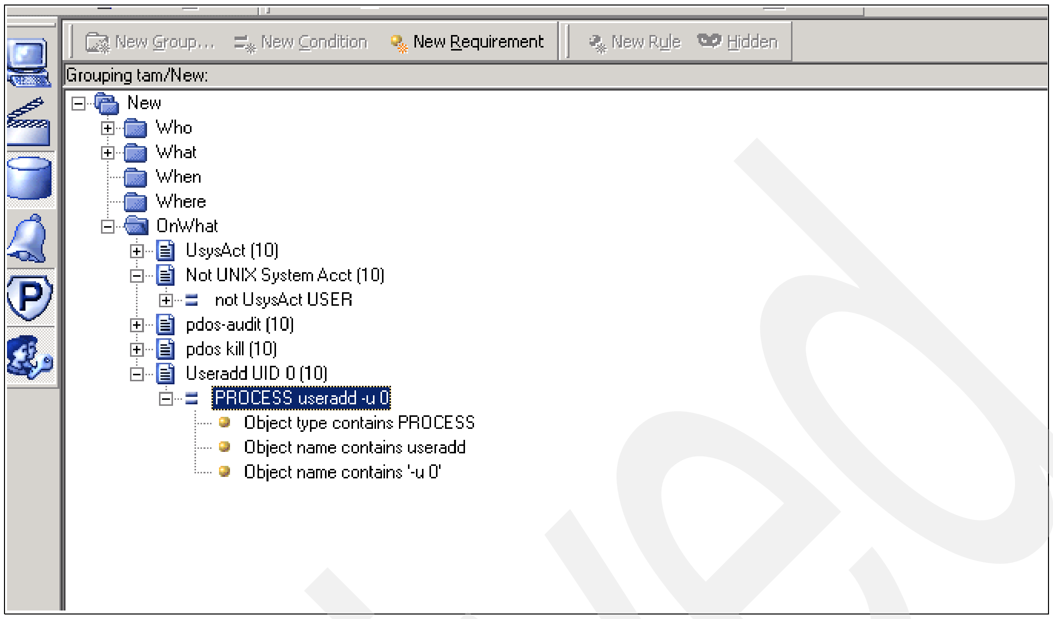
Custom policy - highlight creation of UID=0 users

The next condition we want to highlight is when someone creates a user using the **useradd** command and sets the UID to be zero. This is an exception condition that should be highlighted.

The policy rule we want to apply is to show all executions of **useradd** where the argument `-u 0` is entered. There are three conditions required:

- OnWhat;: object type is PROCESS
- OnWhat: object name contains useradd
- OnWhat: object name contains -u 0

The implementation of these conditions into a TCIM group is shown in Figure 16.

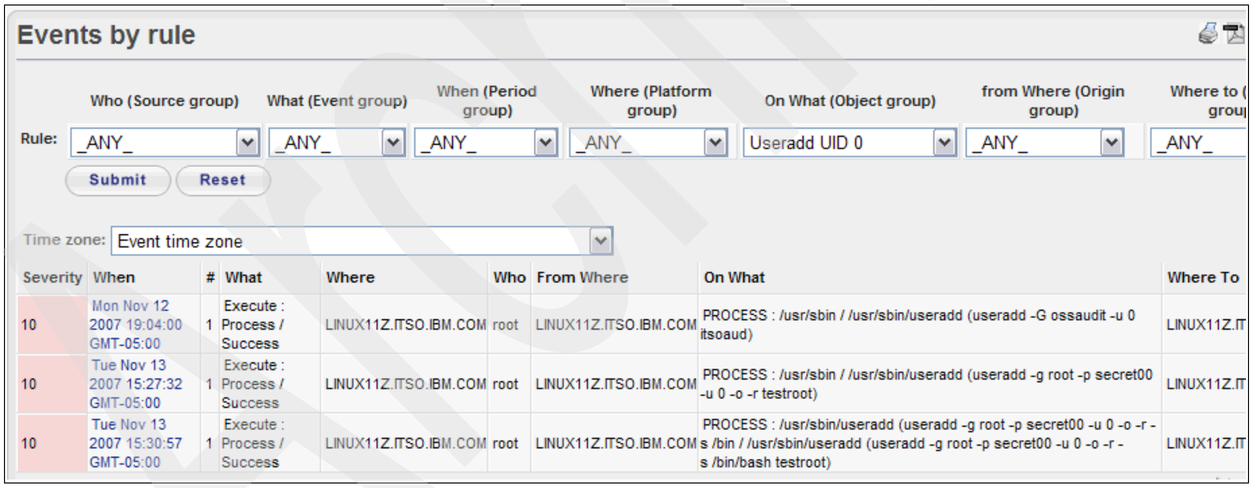


```
Grouping tam/New:
New
  Who
  What
  When
  Where
  OnWhat
    UsysAct (10)
    Not UNIX System Acct (10)
      not UsysAct USER
    pdos-audit (10)
    pdos kill (10)
    Useradd UID 0 (10)
      PROCESS useradd -u 0
        Object type contains PROCESS
        Object name contains useradd
        Object name contains '-u 0'
```

Figure 16   Grouping conditions for executing a useradd with UID=0 policy

There is only a single condition under the group. This condition contains three requirements that are joined with an AND.

The TCIM Events by Rule report with this grouping applied is shown in Figure 17.



Events by rule

	Who (Source group)	What (Event group)	When (Period group)	Where (Platform group)	On What (Object group)	from Where (Origin group)	Where to (grou
Rule:	_ANY_	_ANY_	_ANY_	_ANY_	Useradd UID 0	_ANY_	_ANY_

Submit   Reset

Time zone: Event time zone

Severity	When	#	What	Where	Who	From Where	On What	Where To
10	Mon Nov 12 2007 19:04:00 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	PROCESS : /usr/sbin / /usr/sbin/useradd (useradd -G ossaudit -u 0 itsoaud)	LINUX11Z.IT
10	Tue Nov 13 2007 15:27:32 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	PROCESS : /usr/sbin / /usr/sbin/useradd (useradd -g root -p secret00 -u 0 -o -r testroot)	LINUX11Z.IT
10	Tue Nov 13 2007 15:30:57 GMT-05:00	1	Execute : Process / Success	LINUX11Z.ITSO.IBM.COM	root	LINUX11Z.ITSO.IBM.COM	PROCESS : /usr/sbin/useradd (useradd -g root -p secret00 -u 0 -o -r -s /bin / /usr/sbin/useradd (useradd -g root -p secret00 -u 0 -o -r -s /bin/bash testroot)	LINUX11Z.IT

Figure 17   TCIM Events by Rule report with custom policy applied

In this example we did not set an attention rule.

These four examples have shown how to implement custom policy for the TAMOS audit events. The approach can be used for any type of event for any source. One of the key things we want to highlight with these examples is the way to implement the OR and AND conditions within the groups, conditions, and requirements.

The team that wrote this IBM Redpaper

This paper was produced by a Tivoli specialist working at the International Technical Support Organization in Poughkeepsie, New York.

David Edwards is a Consulting IT Specialist with the Tivoli Global Response team. He is based in Australia. He has twenty years of experience in IT, covering areas as diverse as application development, CICS® systems programming, distributed systems management, and working as a Product Specialist for Tivoli Security products. He holds a Bachelor of Science (Chemistry and Applied Mathematics) degree from Monash University and a graduate degree in computer science from Swinburne University. He has written extensively on the Tivoli Systems Management and Security products, including co-authoring four IBM Redbooks®, as well as authoring a number of IBM Redpapers and developerWorks® articles.

Thanks to the following people for their contributions to this project:

Paola Bari, Richard Conway, Roy Costa
International Technical Support Organization, Poughkeepsie

Klaus Heinrich Kiwi
IBM Brazil

Notices

This document REDP-4402-00 was created or updated on March 3, 2008.

Send us your comments in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

Redpaper™

Trademarks