Security Features for System Sharing

Casey Schaufler
SGI
Mountain View, CA USA
casey@sgi.com

Abstract

The Irix Operating System provides a number of advanced security facilities that are unsurpassed in the marketplace. Many of these facilities have been used to provide unique solutions in the super-computer environment. The Trusted Irix/CMW (Trix) feature can be used to provide extreme levels of resource sharing with data integrity and protection.

General Terms

Security, Multi-level, Sensitivity, Integrity, Access Control

Keywords

Security

1. Introduction

In the world of computer systems and especially in the smaller community of systems security there is considerable concern around protecting information from unauthorized access. The mechanisms provided to protect information are typically described in terms of denial, going to great lengths to explain how they prevent bad things from happening. All to often, this leads to the impression that serious security features in a system will result in an operating environment with the same level of comfort as a corset with cast iron stays. The reality is that while additional security mechanisms do add to the complexity of an individual person or computer, they can greatly reduce the complexity of the overall system. We tend to forget that the whole point of a security facility is to make life simpler.

The easiest to use computer system has the same principle attribute as the most secure, that being that it's turned off. A single user machine, disconnected from any network, is insignificantly more complex, unless the user is fanatical regarding organization. Complexity begins to creep in with the classic standalone mainframe, where the user has to worry about her peers, and escalates with each additional connection to other computers. By the time modern configurations are encountered, where hundreds of machines are tied together and resources are shared with people, groups, and sometimes companies which the user has never heard of, much less trusts with her data, complexity has risen to the point that security can not be contemplated without assistance from the system itself.

Military and espionage organizations have a longer history with hostile environments than do their industrial counterparts. Even today, as the general computer system environment has become less friendly, there remains added incentive for security in its traditional strongholds. A hacked web site is embarrassing, lost financial data can result in fines, downtime can impact worker productivity, but when it comes to war, lives are at stake. Further, there is no environment in which sharing of data and equipment is more important than the always alert, always taxpayer constrained military system. While it may not be attractive to consider a corporate intranet in the same terms as an aircraft carrier, many of the facilities that make it possible for Captains, seamen, and civilian contractors to share computers can be used to allow businesses, universities, and governments to share.

2. Trust Technology Features

The shared environment requires increased trust, either among those who are sharing or in the systems that provide the environment in which they are sharing. While there is little that a system provider can do about the trustworthiness of the system's users, there are facilities a system can provide which make it easier to trust. This has been understood in the industry for some time, but development cost and perceived end user reluctance to adopt them have held some vendors back.

The perception that end users are reluctant to use advanced security features may have been true in the Reagan era, but is certainly not today. Limiting use of privilege is a major concern. The ability to monitor user activity or detect intrusion is keenly interesting. User defined access control has become expected rather than tolerated. Strong user separation and integrity controls, once viewed as excessive, are seen as promising futures for the modern environment. The other commonly flawed perception is that these facilities are not available today.

The POSIX P1003.1e/2c[1] working group proposed interfaces and behaviors for these facilities. While this work never achieved standard status, it was very influential throughout the development community. The Irix feature set was developed in direct contact with the working group, and reflects this in many ways.

2.1 Capabilities

Capabilities[2] directly address the long-standing issue of an all-powerful Superuser. The traditional UNIX semantics grant any process owned by the root user privilege to violate any system security policy. With capabilities, the permission to violate security policy is separated from the user identity and broken into approximately 40 discrete exemptions that can be granted independently. New processes inherit the capability state of their parent, as is the usual behavior for process security attributes.

---

[1] Originally known as P1003.6.

[2] The term "capability" has multiple uses in the jargon of computer security. The traditional use refers to a system that provides privilege by use of one-time tickets. The use to which it is put to here derives from the POSIX P1003.1e/2c working group, who wanted to call the facility "privileges", but found the term already defined otherwise by ISSO.

The Irix system provides an augmented Superuser capability model. A process is allowed to violate a policy if it possesses the appropriate capability or if it is running as the root user. This allows for the maximum compatibility with traditional behavior while allowing a capability aware program or installation to grant privilege at a finer granularity.

The Trix system provides a pure capability model. A process is allowed to violate a policy only if it possesses the appropriate capability. This allows for the maximum protection from the abuse of privilege and separates the use of privilege from the use of the root account.

A capability set is composed of three capability vectors, the inheritable, permitted, and effective. An inheritable capability may be passed on through invoking a new program. A permitted capability may be promoted into the process's effective set on request. An effective capability indicates that the process may violate that specific policy.

Users may be granted specific capability sets based on individual needs and levels of trust. Capabilities may be associated with a user by providing an entry in the `/etc/capability` file. A user with an entry in this file is granted a default capability set and a maximum allowed set. In most cases, the default will be empty and the maximum will contain just those capabilities required to perform the special actions the user is allowed. The `auditor` user provides an example of a user with need for capabilities, being allowed to modify the audit state of the system and to terminate arbitrary processes.

Early attempts to produce systems without a Superuser removed the setuid mechanism and its ability to associate privilege with programs. The resulting systems could be made to work, but required significant deviation in serious applications from the mainline. The Irix capability mechanism avoids this problem by allowing a capability set to be associated with a program file. The capability set on the file is combined with the capability set on the invoking process to create a new capability set with which the process runs.

If the program file has no capability set the capability set is unchanged, unless the CAP_PURE_RECALC flag is set, in which case the capability set is cleared. If the program file does have a capability set, the process capability set is calculated according to this formula:

$$\text{Inheritable}_{new} = \text{Inheritable}_{file} \,\&\, \text{Inheritable}_{process}$$
$$\text{Permitted}_{new} = \text{Permitted}_{file} \mid (\text{Permitted}_{process} \,\&\, \text{Inheritable}_{new})$$
$$\text{Effective}_{new} = \text{Effective}_{file} \,\&\, \text{Permitted}_{new}$$

While the calculation may appear arcane, it provides a result allowing both capability aware and ignorant programs to function properly when correctly applied.

2.2 Security Audit Trail

No security environment is perfect as long as it includes human users and administrators. Even if the system prevents abuse it is helpful to know who is attempting to abuse it, and in what manner. This is especially true in shared environments where curiosity

sometimes gets the better of courtesy. By identifying who is not willing to work within appropriate parameters many potential incidents can be avoided.

The Security Audit Trail (SAT) system provides the ability to track every access control decision made by the system. It processes the data in real time and can record in excess of one million records per hour on a smallish[3] machine. The records generated can be pre-selected using the `sat_select` command or filtered using the `sat_filter` mechanism.

It is impossible for a system designer to determine exactly what an installation will consider an interesting access control decision. With this in mind, the Irix SAT system allows the administrator to select the events which are collected based on the type of access, the individual user, and in the case of Trix, the Mandatory Access Control information. Because even the most careful selection criteria are going to result in some false positives, the `sat_reduce` tool is provided to manually filter existing records based on users, groups, path names, time, process capabilities, command names, and event outcome.

Audit records are stored in a compact binary format. They are composed using a set of about 20 unique data tokens, each describing an important piece of security relevant information. There are four kinds of tokens; header, subject, object, and attribute. The header token defines the beginning of a record and contains the total record size, the event type, and the event's outcome. The subject token identifies the process that performed the action. One or more object tokens describe the data items acted upon. Attribute tokens are associated with subject and object tokens to describe the access control or privilege information relevant to the access check that was made.

There will be times when humans will want to examine audit records. The `sat_interpret` tool translates the binary audit record format into readable text in either of two forms. The *verbose* or *brief* form[4] is the more readable, but is quite large, as evidenced by the following example of a single record created by the `unlink` system call.

```
sat_file_crt_del,Success
        TIME                    = (05/07/2002,20:42:56)
        SYSCALL                 = unlink
        SATID                   = casey
        COMMAND                 = ex
        CWD                     = /usr/people/casey
        DEVICE                  = 15,0
        PARENT_PID              = 1096
        PID                     = 38539
        UGID                    = casey,nuucp
        UGID                    = casey,nuucp
        GID_LIST                = nuucp
        CAP_SET                 = (all=)
        MAC_LABEL               = userlow
        PATHNAME                = /usr/tmp/Ex0000038539
        OBJECT                  = BEGIN
        LOOKUP                  = /usr//tmp/@..//var//tmp//T-b//Ex0000038539
        FILE                    = 4635316,0,76
        UGID                    = casey,nuucp
```

[3] A single processor desktop system can create these volumes in extreme cases. Most sites choose more judicious audit regimens.

[4] In earlier versions of the tool these were separate options.

```
MODE                    = rw-------
MAC_LABEL               = userlow
OBJECT                  = END
```

The same record in the *linear* form takes up considerably less space, can be run through processing tools such as `grep` and `sed`, but is harder to read. The line breaks here are not actually present in the record.

```
sat_file_crt_del,Success TIME=(05/07/2002,20:42:56) SYSCALL=unlink
SATID=casey COMMAND=ex CWD=/usr/people/casey DEVICE=15,0 PARENT_PID=1096
PID=38539 UGID=casey,nuucp UGID=casey,nuucp GID_LIST=nuucp CAP_SET=(all=)
MAC_LABEL=userlow PATHNAME=/usr/tmp/Ex0000038539 OBJECT=BEGIN
LOOKUP=/usr//tmp/@..//var//tmp//T-b//Ex0000038539 FILE=4635316,0,76
UGID=casey,nuucp MODE=rw------- MAC_LABEL=userlow OBJECT=END
```

Audit log file creation is managed by `satd`. This system daemon can be directed to place newly generated audit records into specific files or collections of files into designated directories. The daemon can be configured to maintain a collection of destinations and use any of three modes to determine which should be used based on amount of space available in each. Audit files are kept into 4 megabytes by default[5], with the size configurable to suit the local needs.

2.3 Access Control Lists

The Irix system offers an extension to the traditional file permission bit scheme, the Access Control List (ACL). ACLs provide a finer granularity than do permission bits, allowing access to be specified not just for the file owner, owning group, and world but also for specific users and groups. Where it is possible to exclude access to a file to a single user using permission bits alone, it requires group list management well in excess of that typically provided on any given site. With ACLs this is a trivial operation.

An important attribute of the Irix ACL scheme is the relationship between ACLs and traditional file permission bits. One very important compatibility requirement is that the code sequence

```
stat(path, &statbuf);
chmod(path, 0);
chmod(path, statbuf.st_mode);
```

results in `path` having exactly the same access afterwards as it did before even if the file has an ACL. Were this requirement not meet cases could arise that allowed more access than was intended either by the original ACL setting or the program or user setting the permission bits.

The file permission bits can be viewed as a small, tightly constrained ACL. Written out in ACL format, one useful set might look like this

```
u::rwx,g::rx,o::x
```

[5] In 1990 this was as large a file as could be conveniently manipulated.

with the file owner having read, write, and execute permission, the owning group read and execute, and all others having only execute permission[6]. An ACL containing only those "base" entries may be set, and the access will be identical to that of a file with those bits. Further, the ACL entries and the mode bits will be kept in sync, with changes to either the permission bits or the ACL entries begin reflected in the other.

The ACL does get more complicated as entries are added beyond the base entries. Adding any entry beyond the base requires that a *mask* entry also be added. The mask entry supplants the owning group in the file permission bits and is used to limit the access on all of the entries beyond the owner and other entries. To explicitly deny access to the user `casey`, who is not the file owner, the ACL would look like this

```
u::rwx,g::rx,o::x,m::rx,u:casey:-
```

with the mask set to limit all extended entries and the explicit user entry for `casey` specifying no access. To additionally permit members of the group `ludite` read access the ACL could be

```
u::rwx,g::rx,o::x,m::rx,u:casey:-,g:ludite:rx
```

again with the mask applied to the explicit group entry, preventing execute access. Entries are processed in order of specificity. Thus, `casey` will not be granted any access to the file even if he is in the `ludite` group.

2.4 Mandatory Access Control

The features discussed to this point are all available in Irix. The Trusted Irix (Trix[7]) product is distinguished from Irix by the addition of Mandatory Access Control (MAC). MAC differs from the traditional access scheme in that it is not at the discretion of the user. Users are not allowed to specify who can access a file. That determination is made based on the MAC labels associated with the process and the file. Users are not allowed to change these labels.

There are two components to the Trix MAC policy. The sensitivity component is based on the Bell & LaPadula policy made popular by the United States Department of Defense Trusted Computer Systems Evaluation Criteria[8] of 1985. The sensitivity policy protects information from users who should not see it. The integrity component is derived from a policy described by Biba, and is the inverse of the sensitivity policy. The integrity policy protects users from information they should not see.

Each Trix user is assigned a clearance, or set of MAC labels they're allowed to use, and a default label, in the `/etc/clearance` file. When the user logs in to the system a label from this set is assigned to the shell process, and this label is inherited by all of its child processes. Access is granted based on a special relationship called *dominance*. A process

[6] This is mode 0751 for those of us who haven't forgotten the old days of octal.
[7] Trix is both an American fruit-flavored breakfast cereal and an Australian dishwashing liquid. All other attempts at nicknames for the product have failed. We're in diverse company.
[8] The TCSEC, or Orange Book, for its brightly colored cover.

whose MAC label dominates a file's is allowed read access, and a process whose MAC label equals a file's is allowed write access.

The sensitivity component of a MAC label allows users to be separated hierarchically with the use of *levels*, or compartmented using *categories*. A label with a level equal to or higher than another can dominate it, allowing a scheme in which one set of user can read the information of another set of users as well as their own. Trix supports 256 separate site definable levels. In the case of two labels with disjoint categories neither dominates the other, resulting in a case in which neither group can see the other's information. Trix supports 65,536 site definable categories, of which any given label can have as many as 250 at any time. There are also special MAC labels defined for system use.

3. Attribute Aware Networking

One of the saddest oversights in the history of the computer industry is that the Internet Protocol was not designed to extend beyond a single, friendly work group. While there had been some attempts to provide limited information about the participants involved in a network communication[9], and high level cryptographic authentication schemes abound, it wasn't until the Trusted Systems Interoperability Group (TSIG) work with the Security Attribute Modulation Protocol (SAMP) that access control could truly be distributed across a network. Combined with a generalized scheme for defaulting information that is not transmitted or available for transmission, SAMP allows for the realization of the network system concepts that have been so long sought after.

3.1 System Composition

Multiple computers can be easily combined into a system when all of the attributes required to make access checks are shared between them. Even if these attributes are assigned on a machine basis reasonable policies can be implemented, allowing significant resource sharing in a reasonably safe environment.

3.2 CXFS

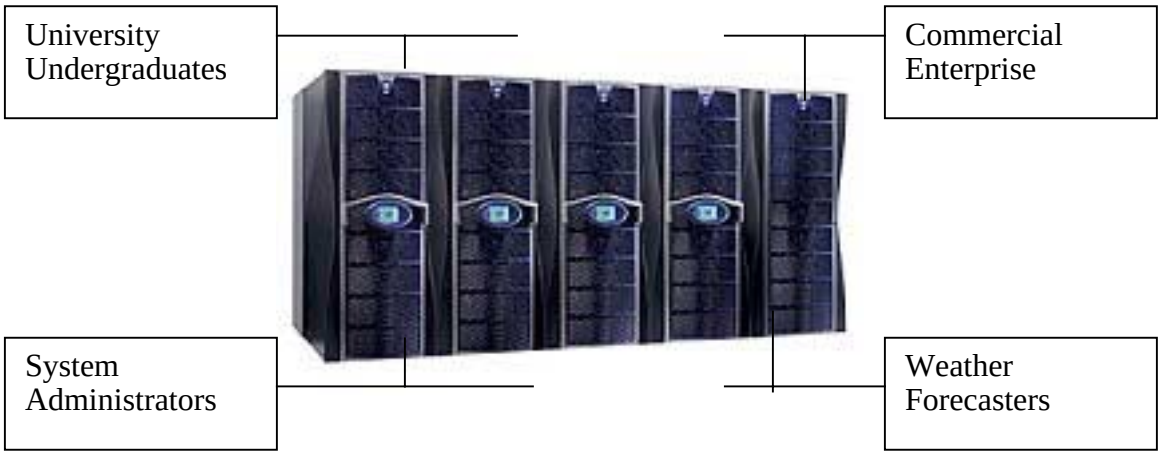
CXFS provides a fully attribute aware cluster file system. The protocol includes provision for systems that use MAC, capabilities, and ACLs.

[9] IPSO, RFC 1108, RIPSO, RFC 1038

4. Sharing Scenarios

We have a number of interesting cases in which Trix is being used to enable system sharing between groups with limited confidence in their neighbors. These are all real configurations that are in use today.

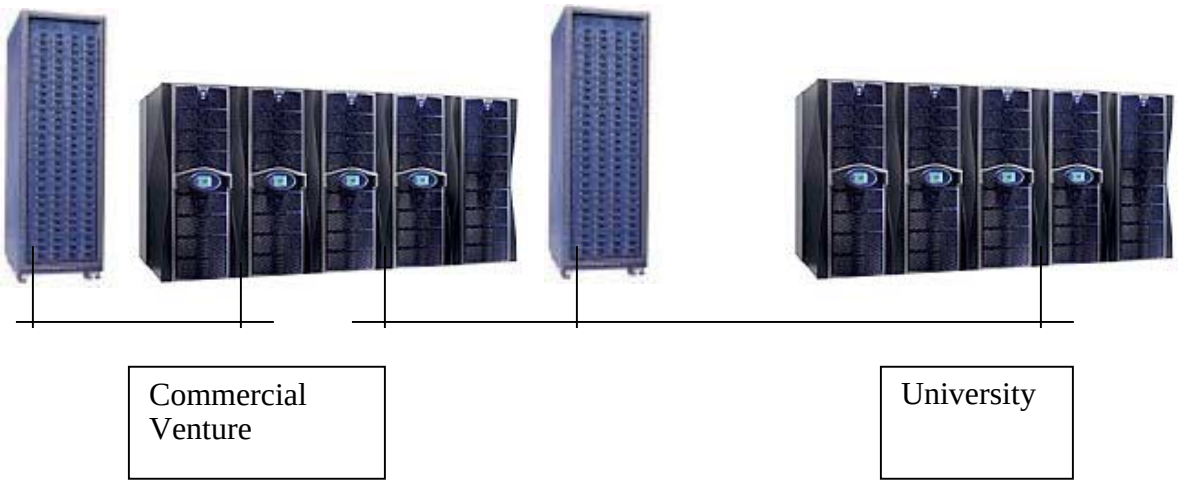
4.1 Simple Shared System



Users access the system from four different dedicated single level networks. The administration network is reserved for system level access. Each of the other networks is designated to a particular MAC category, the users being separated by those categories. An interesting artifact of this configuration is that the weather forecasters who are also students may not use their work systems to perform their academic roles.

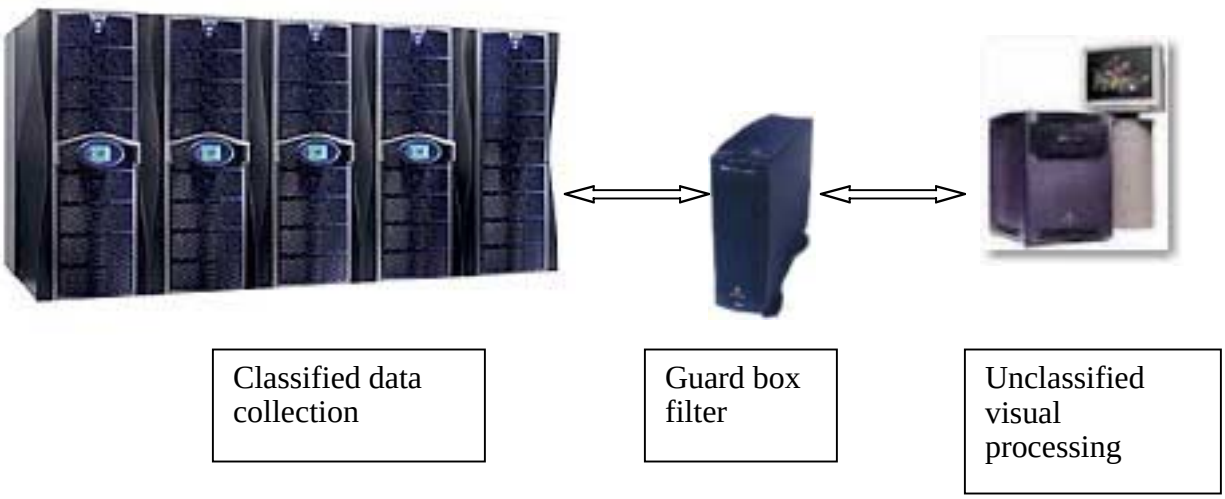
This configuration is almost exactly the classic timesharing system. It differs only in that users access the system over networks rather than terminal lines. Resources limits and job scheduling assist in ensuring that all system resources are shared fairly.

4.2 CXFS in the center



In this situation a university owns a large system, which it shares with a commercial venture. The commercial venture's large data sets are protected from student access by MAC categories. CXFS file systems allow high speed access between the venture's machines and the university's. The venture maintains a separate facility for the data, which can be physically isolated from the university using a mechanical network disconnection device.

4.3 Guard Box



In the guard box system a privileged application inspects and downgrades information before passing to into a less controlled environment. In this case MAC levels are used on the guard box system to prevent data flow from the classified network to the unclassified. The guard box application uses capabilities to enable appropriate information downgrade without permitting any other privileged operations. User access to the guard box machine is forbidden, with continuous automated and frequent manual checks of the security audit trail ensuring that such access is not attempted.

5. Summary

The advanced security features from the trust technology realm provide an environment in which the viability of sharing is greatly improved over what has been available in the past. SGI's commitment to compatibility means that a site is not giving up the interfaces and facilities it requires to achieve usual objectives. The long-term commitment to assurance through evaluation provides evidence that these features will perform as advertised throughout their deployment.

National Information Assurance Partnership

Common Criteria Certificate

Silicon Graphics, Inc.

The IT product identified in this certificate has been evaluated at an accredited testing laboratory using the Common Methodology for IT Security Evaluation (Version 1.0) for conformance to the Common Criteria for IT Security Evaluation (Version 2.1). This certificate applies only to the specific version and release of the product in its evaluated configuration. The product's functional and assurance security specifications are contained in its security target. The evaluation has been conducted in accordance with the provisions of the NIAP Common Criteria Evaluation and Validation Scheme and the conclusions of the testing laboratory in the evaluation technical report are consistent with the evidence adduced. This certificate is not an endorsement of the IT product by any agency of the U.S. Government and no warranty of the IT product is either expressed or implied.

Product Name: IRIX
Version and Release Numbers: Version 6.5.13 with patches 4354, 4451, and 4452
Evaluation Platform: Origin 200 workstation and Origin 3000 server
Assurance Level: EAL3

Name of CCTL: Science Applications International Corporation
Validation Report Number: CCEVS-VR-02-0019
Date Issued: 30 April 2002
Protection Profile Identifier: Controlled Access Protection Profile, Version 1.d, October 8, 1999

Director
Information Technology Laboratory
National Institute of Standards and Technology

Information Assurance
Director
National Security Agency

National Information Assurance Partnership

Common Criteria Certificate

Silicon Graphics, Inc.

The IT product identified in this certificate has been evaluated at an accredited testing laboratory using the Common Methodology for IT Security Evaluation (Version 1.0) for conformance to the Common Criteria for IT Security Evaluation (Version 2.1). This certificate applies only to the specific version and release of the product in its evaluated configuration. The product's functional and assurance security specifications are contained in its security target. The evaluation has been conducted in accordance with the provisions of the NIAP Common Criteria Evaluation and Validation Scheme and the conclusions of the testing laboratory in the evaluation technical report are consistent with the evidence adduced. This certificate is not an endorsement of the IT product by any agency of the U.S. Government and no warranty of the IT product is either expressed or implied.

Product Name: Trusted IRIX/CMW
Version and Release Numbers: Version 6.5.13 with patches 4354, 4451, and 4452, 4373, and 4471
Evaluation Platform: Origin 200 workstation and Origin 3000 server
Assurance Level: EAL3 Augmented

Name of CCTL: Science Applications International Corporation
Validation Report Number: CCEVS-VR-02-0020
Date Issued: 10 May 2002
Protection Profile Identifier: Labeled Security Protection Profile, Version 1.b, October 8, 1999

Director
Information Technology Laboratory
National Institute of Standards and Technology

Information Assurance
Director
National Security Agency

6. References

1. Bell, D. E. and LaPadula, L. J. *Secure Computer Systems: Unified Exposition and Multics Interpretation*, MTR-2997 Rev.1, MITRE Corp, Bedford, Mass., March 1976.
2. DoD 5200.28-STD *Department of Defense Trusted Computer System Evaluation Criteria*. December 1985.
3. [RFC1038] "Revised Internet Protocol Security Options, RIPSO," RFC 1038, Network Information Center.
4. [RFC1108] " Internet Protocol Security Options, IPSO," RFC 1108, Network Information Center.