Objective and Scope

Effective Date: August 2008

This CUIT UNIX Standard Operating Environment (SOE) document defines the agreed system hardening requirements as well as security best practices for CUIT managed UNIX servers. The SOE establishes a baseline of security related specifications and requirements prior to any business customizations and/or operational functionality requirements.

These requirements indicate the goal state for CUIT Unix servers (Linux, Solaris, and AIX). The majority of servers comply with these requirements. Exceptions have been identified and will be addressed in accordance with priorities and available resources.

Note that ColumbiaNet terminals and the HMC control terminals for the pSeries run Linux, but are out of scope for this document.

Index

1.0 Accounts and Passwords

The CUIT ID system controls all aspects of authentication and authorization to the UNIX servers. The ID system enforces password complexity rules, password changes, and reuse history. User account information, which is stored in LDAP, identifies user's eligibility on servers. Users authenticate with their UNI password; these are stored in Kerberos, encrypted. Guidelines for choosing a complex password are explained in this document: http://www.columbia.edu/acis/faq/342.html.

Management of groups is distributed to group owners via the ID system. Because certain groups are used to control access to application-specific servers, those owners dynamically control access to these hosts. Groups must therefore be updated when staff leave or change status.

A limited number of accounts (e.g. "root") are controlled by local password files, customized per host, which are generated from a central file ("passwd.cpp"). Most passwords for these local accounts are in Kerberos; only a few are stored locally, encrypted in shadow password files readable only by root. The same password complexity rules are followed.

- Accounts and passwords are considered highly confidential and private and must not be shared with others. The only exceptions are "root" and service accounts.
- Shared passwords for service accounts and "root" are changed when staff changes. Where possible, access to service accounts is given via sudo (see below), so the account has no usable password.
- Each user is responsible for the proper use of his or her account and any activity conducted with it. This includes choosing complex passwords, protecting them, and ensuring that file protections are set correctly.
- Compromised accounts must be reported and the account disabled immediately.
- Access must be modified (e.g., removed) for change of status and /or function for an employee.
- Access privileges need to be reviewed on a periodic basis to ensure appropriateness.
- User IDs or accounts associated with former University individuals and/or affiliates are disabled in a timely manner.
- For default accounts and passwords, unused accounts are disabled and default passwords for active accounts are changed during system setup.
- Wherever possible, scripts do not include embedded or hard-coded passwords.

- Records are kept of all account creations and modifications. In the ID system, these are in the history table, viewable via "uniedit". For local password files, these are available via the "RCS" revision control system in the source tree.

NOTE:

CUIT weighed the advantages vs. disadvantages of the automated account lockout feature after preset unsuccessful attempts, which revealed that this feature might be exploited for denial of service (DoS) attacks to disrupt system operations. As a result, the automated account lockout feature is generally not enabled. See section 1.5 for a description of mitigating controls.

1.1 User Accounts

User accounts can be divided into five categories: timesharing, staff, service, root accounts, and “root” itself. All accounts have access to the command line, and a default umask of 077 (i.e., files are created with no access by group and world) or 027 (i.e., files are created with read and execute access by group and no access by world), with the CUIT UNIX Director’s authorization.

The numeric UID (user ID) is unique for all non-root accounts.

All uses of su and sudo are logged.

1.1.1 Timesharing Accounts

CUIT provides two clusters of servers, cunix and pinex, for “timesharing” or general access to multipurpose central servers.

- Any users eligible for Cyrus email have access to these servers.
- Users include over 50,000 faculty, staff, students, casuals, consultants, and recent graduates.
- Eligibility is determined by the ID system.
- Accounts are turned off 2-4 weeks after eligibility ends; sooner upon request.

When users connect to the pinex cluster, their shell is set to “pine”, a terminal-based mail client. When they quit pine, they are logged out. However, they can get access to the shell, e.g. via the pico editor within pine.

On cunix, users can also run pine, or a variety of other Unix applications. Typical usage includes Computer Science homework (compilers, etc), research (e.g. SAS or Stata), or web publishing.

1.1.2 Staff Accounts

Staff accounts are typically for UNIX administrators, developers, and application supports. These accounts allow access to the timesharing hosts plus various back end hosts.

- Different sets of staff have access to different hosts, e.g. CUL staff are allowed to connect to back end hosts owned by the Libraries.
- More critical hosts generally have fewer users.
- In some cases, access is given via local password files (see section 1.0) with Kerberos passwords.
- In some cases, access is via "types" in the ID system, edited via uniedit.
- In some cases, access is by Unix groups, via the access.conf file. This allows for more centralized granting and revocation of access as staff responsibilities change.

1.1.3 Service Accounts

Individual accounts should not be used for production processes. Therefore, service accounts are used, such as "oracle".

- Generally, access to the service account for management is via sudo.
- File transfers are accomplished via ssh keys and use of scp.
- When necessary, a password is assigned to the service account, but this is avoided whenever possible.

1.1.4 Root Accounts

For accountability reasons, individual root accounts are used instead of sharing the password for a single root account.

- Like other staff accounts, root accounts are not necessarily valid everywhere.
- Root accounts are named based on the individual's Unix username or UNI, e.g. melissa's root account is rmelissa, jf659's is rjf659. Occasionally the result is modified, e.g. to truncate to 8 characters.
- Each root account has its own password. Only the individual owner of the account knows this password.
- Use of root accounts must follow the guidelines in the "Root Lecture" at https://www1.columbia.edu/sec/acis/sy/systems-manual/Security-Incidents/root-lecture.txt.
- Root accounts are given to vendors only under special circumstances, with management approval, and only on a those hosts where the vendor is responsible for the software.
- Before getting a root account, staff must have experience working with CUIT servers and knowledge of our environment. Optimally, they should have watched someone else's use of root and in turn be supervised the first few times they use the account.

1.1.5 Root

The "root" account (aka superuser) is utilized exclusively by the operating system and members of the Unix Systems Group for system related tasks on the servers and network.

- Access is given by sharing the password.
- The password is changed whenever a staff member who knows it leaves.
- The password may not be written down or stored in plain text. PGP can be used.
- A list exists of users who have the password.
- "root" is only used when the individual root account is unavailable, e.g. due to network problems or in single-user mode.
- "root" can only be used from the console.

1.2 File Management

Packages and patches are centrally managed with cfengine, up2date, and locally written packages called opium and cpp2hosts. These ensure that fixes and upgrades are applied consistently across hundreds of servers, but also allow for control of differences between hosts. Changes to cpp files, as well as cfengine and opium config files, are recorded using RCS (revision control system).

Available updates are monitored via industry mailing lists and in some cases (e.g. up2date) automated updates. Appropriate updates are applied. Software is kept upgraded to currently supported versions, except when it would cause a disruption to a production service.

1.3 Switching Users, Assuring Operational Continuity

Certain policies help assure operational continuity and allow for appropriate granting and revocation of extended access.

1.3.1 Suid/Sgid

User-writable filesystems are mounted with the "nosuid" option. Suid and sgid files not provided by the operating system are avoided where possible. Alternatives include cron jobs or damons running as the intended user, and sudo.

1.3.2 Sudo

Sudo is used, as noted above, to give access to service accounts. It is also used to give more limited access to run specific commands as root or other service accounts. Access is controlled by groups where possible. This centralizes access management with the group owner. (See groups description in 1.0.)

1.3.3 Assuring Operational Continuity

Production jobs, e.g. batch or cron jobs, run as service accounts, not individual staff accounts. Programs and scripts for production or general staff use should be in centralized or application-specific directories, not in home directories. In this way, when an individual's role changes or when they leave the University, their account can be removed without negative impact.

1.4 TCP/IP Network Services

Only those network services which are needed are running on each host. This is controlled via centrally-managed inetd.conf files (see above) and RCS'd init files.

1.4.1 Required Services

All hosts run **sshd** to allow administrators and developers to connect.

Most hosts run NFS clients, which involves services like **rpc.lockd** to run on both client and server. Because lockd is an RPC process, it can run on a variety of ports.

Backup software (Veritas, TSM) often requires a network service to listen for connections from the central backup system.

Hosts listen for connections from our monitoring system **symon**. In some cases, **rup** is used for monitoring. **Snmpd** is also required.

1.4.2 Application-Specific Services

Application-specific services include web servers, database servers, Kerberos, and LDAP. These are run only on the appropriate hosts.

Ntpd is run only on the time server host, which synchronizes with a local stratum 1 time server (which in turn synchronizes with a satellite). However, **ntp** is run on all hosts to keep the time in sync. This is critical for Kerberos authentication to work, and is also helpful in comparing events for diagnosis of issues.

Sendmail generally listens to the network only on mail hosts. On other hosts it is run in queue-only mode, or when invoked by a program to send a specific message. When needed, the individual hosts are MX'd to the central servers.

1.4.3 Deprecated Services

The rsh ports **shell** and **login** are still in use by centralized maintenance scripts. We are working to update these scripts to use **ssh**.

1.4.4 Disallowed Services

As no passwords may be sent in plain text, any such services are disabled. The exception is **ftpd**. This runs on ftp.columbia.edu for anonymous ftp only. It also runs on stage.ais.columbia.edu, for file transfers from vendors and other third parties. (Vendors are encouraged to use a secure protocol but cannot always comply.)

For privacy and security reasons, ports such as **finger** and **rwhod** are closed.

1.4.5 Filters

Additionally, ports are filtered via a variety of methods. We use TCP wrappers (hosts.allow and hosts.deny), and where appropriate IP tables or IP chains. In the Enterprise Zone (see below), most ports are closed by default and opened as needed, via network ACLs (aka firewalls).

Filters are maintained centrally for consistency, including customization by host as needed.

1.5 Security Monitoring

All systems connected to the Columbia University network are monitored by PAIRS (Point-of-contact And Incident Response System) and GULP (Grand Unified Logging Program). More sensitive servers have additional monitoring and restrictions.

1.5.1 PAIRS

The PAIRS system was developed at Columbia University and was built around the concept that a compromised computer system will behave differently. It is an anomaly based intrusion detection system. PAIRS operates by processing all NETFLOW data generated at the borders of the University network. This data represents all traffic going into or leaving the University. By looking at these traffic patterns, we are able to determine if a system has been compromised and take corrective action.

1.5.2 GULP

The GULP system was also developed at Columbia University and is used to process all authenticated log information produced. It correlates the information with DHCP so we can use it to tie an authenticated login to an individual computer system. As part of this process, any brute force password attacks are noted and if the attacker gains access to a system, Security is notified immediately.

1.5.3 Enterprise Zone

Sensitive business applications / systems are located in an area known as the "Enterprise Zone". This is an isolated set of subnets that have strong ACLs associated with each server. The ACLs are set up so that only the ports needed for the applications residing on the server are exposed, and where possible, they are also restricted by IP address. These ACLs greatly limit the attack surface.

In addition to the restrictions of the Enterprise Zone, these systems are monitored with the Columbia Fingerprint Service. The Fingerprint service monitors the network traffic going to and from each machine looking for changes in the traffic patterns. It also looks for changes in the port definitions on each server for evidence of new or unexpected services being started.

1.6 System Monitoring

Servers are monitored via a central system called symon (though research is being done into a next generation system). Symon ensures that systems remain in working order by detecting problems and alerting staff. In some cases, symon can fix simple problems, e.g. by restarting a process.

Monitors include:

- **Ping** to detect that a host is alive and on the network.
- **CPU load**.
- Host-specific services such as **httpd**.
- **Memory** usage, including **swap**.
- **Disk** fullness.
- **Mounted filesystems** matching config files.
- **Time** synchronized with central clock.

Staff is on-call 7x24 to respond to issues called out by symon or by our central helpdesk and operations staff.

Version Tracking History

Change history comments	Version	Date
Initial UNIX SOE and Security Best Practices created by Melissa Metz, Joel Rosenblatt, and Larry Lee	V1	7/08
Updated section 1.1 – User account to include umask 027	V1.1	1/09