UNIX™ Security in a Supercomputing Environment

Vishal kaushik,Shalini Yadav
Students, Dept. of Information Technology
Dronacharya College of Engineering, Gurgaon,Haryana

***Abstract*- The UNIX™ operating system is designed for collaborative work and not for security. Vendors have modified this operating system (in some cases, radically) to provide levels of security acceptable to their customers, but the versions used in supercomputing environments would benefit from enhancements present in so-called secure versions. This paper discusses the need for security in a supercomputing environment and suggests modifications to the UNIX operating system that would decrease the vulnerability of those sites to attacks. Among the issues are additional auditing controls, changes to network programs, improved user authentication, and better application of the principle of least privilege.**

***Index Terms*- Vulnerability, authentication, enhancement, superconducting.**

I. INTRODUCTION

Supercomputing facilities are used primarily for research and development, security considerations seem out of place or unwelcome additions that serve to hinder, not advance, the effectiveness of such sites. Since computer vulnerabilities are often transitive in that penetrating one system allows 'penetration of connected systems, should a malicious person called an uftucker or cruckert break into a computer on which jobs are prepared for the super- computer, that attacker could alter the job to give him or her access to the supercomputer; similarly, he or she could modify results obtained from jobs run on the supercomputer to hinder development work or research. The "supercomputing environ- ment," includes the supercomputer and those hosts on which jobs (including research, development, and administrative func- tions) are either prepared for execution on a supercomputer, or on which the output of such a job is analyzed. Of course, not all such machines may be under the control of the supercomput- ing staff. environments using networks, the third class of mechanisms augment authentication and access over a network, and provide applications layer encryption services and authentication (where feasible). As no security is perfect, the fourth class of mechanisms logs events that may indicate possi- ble security violations; this will allow the reconstruction of a successful penetration (if discovered), or possibly the detection of an attempted penetration.

II. USER SECURITY

When a user accesses the facility or resource, his or her identity determines what he or she may do, Hence accurate identification or authentication of the user is vital.

Mechanism: To encourage users to change passwords periodically, some ver- sions of the UNIX operating system Provide a mechanism to expire passwords; others do not. However, should a password expire, the user is forced to change it at the next successful login because the system does not allow any command other than the password changing command to be used All standard versions of the UNIX operating system encrypt user passwords using a one-way function [6,27] and store the encrypted form in a world-readable file called the password file. It is not possible to determine who reads (or copies) this file. An account may have no password; in this case, anyone may access that account; no password will be requested Mechanism problems: the computer generate random passwords is not much better, since users will then write the passwords down, espe- cially when they must remember passwords for many different hosts. Generating passwords according to a set of rules works reasonably well (a common method is to make the passwords pronounceable but meaningless), but if users are assigned dif- ferent passwords for a large number of hosts they will still often write them down. Implementations of a password aging mechanism should not require the user to think of a new password instantly; as Grampp and Morris [15] point out, "the most incredibly silly passwords tend to be found on systems equipped with password aging." Further, if a user may change passwords at any time. one need only change them to satisfy the password aging mechanism, and then can change them back. If the mechanism prevents a user from changing passwords more than once within a

specified interval of time, the user may not be able to change a Poorly-chosen or compromised password accounts without passwords should not exist, especially in an installation where resources are as precious as in a supercom- puting center.

Alternatives: Except where it affects the quality of the password, the source of the password (computer or user) is not relevantt. Simple modiications to the standard UNIX password changing program can check the pro- posed password against the user's name, account name, and various dictionaries and lists of common character combinations (such as acronyms and names not in any dictionaries); this satisfies criterion 2, and makes meeting criterion 1 more likely than if the checks were omitted. As an alternative, aging mechanism should warn them before their password expires to allow time to think of another one. Many secure versions of the UNIX operating system use a "shadow password file" [14,18,26] in which the encrypted password is placed in an alternate unreadable file, prevents pass- word cracking without attempting to log in, since all systems should record failed login attempts, this increases visibility of password crackers. Modifications to the Iogin program enable the security officer to allow the login but to a restricted account (to allow security officers to trace the connection [16,39]), or to disable the attacked account. Other authentication methods may be used in place of, or in conjunction with, passwords. The challenge-response proto- col [38] requires the computer to generate some number or string (the challenge), and the user to perform some operation on it (the response). The challenge varies from instance to instance, so even if an attacker obtains one challenge and its corresponding response, subsequent challenges will be different; to prevent the relationship between challenges and responses from being deduced, the response is usually the output of some cryptographic function of the challenge. This usually requires some physical hardware (such as a calculator-like device) to obtain the response, so both knowing another user's password and access to the appropriate ha

Recomrnenduriom: UNIX systems in supercomputing envlron- ment should use a two-tier method of authentication that cannot be disabled. The 8rst is the password; either users should be allowed to select their own passwords, which are checked before being accepted by the computer. or the computer should generate a set of possible passwords and allow the user to select one (or request another set). The encrypted passwords should be kept in a shadow password file, and password aging should be used. The second tier uses another authentication method to require the user to demonstrate that he or she is not an intruder who knows the real user's password, such as a challenge- response protocol requiring an external device to obtain the reponse from the challenge. Users who fail to give a correct account name and password at the first level should still be forced to complete the second, so they cannot determine whether the failure was a bad account name, an invalid pass- word, or an erroneous response. Further, some action deemed appropriate by the site security officer should be taken for repeated failures to log in to a specific account.

III. Access Control, Integrity, And Least Privilege Threats

Access control mechanisms regulate the ability to access resources such as files, devices, and processes. If the access control mechanisms are inadequate, users may be able to modify or alter data they should not have access to, or may not be able to access data they are authorized to. Standard UNIX Access Control Mechanism: Standard UNIX sys- tems use a simplification of the traditional access control list. Each object is assigned a uer and a group identification (typi- cally, the user identification is that of the owner of the object, and the group identification is either that of the owner or of some related object). Three sets of permissions are owned. Neither method is acceptable in pm&e. Hence when a subset of users of the system must access a file, its owner usually gives that access permission to all users. Thus the mechanism fails to support the principle of least privilege [36]. The existence of an omnipotent user also violates this principle, because once an attacker has gained access to that account he or she can do anything. The setuid mechanism makes access to other accounts all too easy if not programmed with care, because programs using that mechanism often fail to do adequate checking. The only mechanism to check the integrity of files is a - trols on the object allow the action. (How this is done varies from system to system.) By judicious selection of security lev- els, sets of compartments, and setting

of the access control lists, users (and the system administrator) can control access to any granularity required, thus these mechanisms enable enforcement of the principle of least privilege. Almost all versions of secure UNIX have modified or eliminated the notion of the superuser, especially by partitioning the superuser functions among several users [11,14,20]. In some cases these users are not permitted to log in, but must act in single-user mode; in others, they must access restricted com- mand interpreters to perform their function. This eliminates the problem of system administrators logging in as the superuser and acting in ways that do object's set of integrity compartments is a subset of the 695 subject's set of integrity compartments, and the discretionary access controls on the object allow the action. The system administrators are free to use, or not use, any of the imple- mented aspects. Recommendations: The discretionary access control mechanism should be access control Lists. perhaps with the standard simplification being used should the user attempting access not be named in the list associated with the object. The superuser functions should be split over multiple accounts (each with less privilege), and the system should use a mandatory access con- trol mechanism to limit the effects of penetrations. If setuid and setgid programs are to be present, they should lose their privileged setting when any process writes to them. The setuid and setgid privilege should be granted only through a trusted path, so no user would unknowingly surrender privileges. Finally, a multilevel integrity mechanism should be in place. However, the aspects of the multilevel mechanism to be used should be at the discretion of the system administrator, and he or she should be able to activate (or deactivate) them on a per- user bar&t.

IV. Network Privacy And Authentication Threats

As most users communicate with supercomputing environments over networks, authenticating their identity and providing tools for private communication with the computer and with other users enables accurate identification of those entitled to use specific resources and allows work done on the supercomputer, and its results, to be private. The very general problem of authorization and secrecy in networks is discussed elsewhere [21,40]; the discussion here is confined to tools speciiic to the UNIX operating system and to the 4.2 and 4.3 Berkeley user's security clearance should be a function of the t The rule against "reading down" would most likely not be applied to all users because. as Gasser points out. it "is probably more suited to containing errors" than threats to security [13, p. 701. But the rule against "writing up" would he. host from which he or she is connecting to the supercomputing environment. Methods of breaking the standard UNIX encryption pro- gram are very widely known [34]. Electronic mail as supplied provides neither privacy nor authentication. While the body of letters may be encrypted using the program described above, the result must be expanded (to seven bit characters) to comply with the intemet mail proto- col [33]. Further, it is not possible to determine whether the named sender did in fact send the letter without an out-of-band mechanism (such as a telephone cab.)

V. Alternatives To The Standard Berkeley UNIX Network Tools

Many sites only allow system administrators to designate hosts as trusted. Others allow users to designate those hosts under the administration's control as trusted, but no others (for exam- ple, at the NAS Project. the computer "prandtl.nas.nasa.gov" could he trusted, but "icefloe.dartmouth.edu" could not. since it is not under NAS control). Still others restrict this ability to unprivileged users; administrative or superuser accounts may not use this feature. Many sites have

VI. Conclusion

This paper presented a brief overview of some security mechanisms in conventional UNIX systems, discussed their deficiencies, described tools that improve or augment their effectiveness, and recommended changes to increase the difficulty of an attacker trying to penetrate undetectably a com- puter within the supercomputing environment. Supercomputer manufacturers offering versions of the UNM operating system are aware of these needs, and at least one supplies a system enhanced with many of the features described above [lo]. Many of the recommended tools are similar to, or identi- cal to, parts of commercial secure versions of the UNIX operating system designed to meet criteria outlined in the Trusted Com- puter System Evaluation Criteria

[31]. However, those criteria should be followed only if dictated by site (or administrative) policy. Instead administrators of supercomputing environments should implement those recommendations they believe will be most effective within their specific environment. In this way they can achieve that delicate balance between security and the user-friendliness their user community require