General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

Produced by the NASA Center for Aerospace Information (CASI)

DATA PROTECTION BY
SELF-AWARE COMPUTING SYSTEMS

W. M. McKEEMAN

CEP REPORT, VOLUME 2, No. 6

June 1970

The Computer Evolution Project
Applied Sciences
The University of California at
Santa Cruz, California 95060

N71-19599
(ACCESSION NUMBER)
8
(PAGES)
CR-117126
(NASA CR OR TMX OR AD NUMBER)
(THRU)
63
(CODE)
08
(CATEGORY)
FACILITY FORM 602

DATA PROTECTION BY

SELF-AWARE COMPUTING SYSTEMS

W. M. McKEEMAN

ABSTRACT

We propose mechanisms whereby access to proprietary data can be made difficult to the unauthorized user. Programs must be able to check that they are running on the right computer with the right users and be able to react with alarm when a check fails.

Problem Statement

It is illegal to burglarize a safe; it is also hard to do. If we achieve legal safeguards for computer data, we also must make it hard to violate them. If we do, then most people will not bother to try; if they do try, it will be demonstrated that the violation was intentional.

Among the many situations where we would like some measure of protection are:

(1) Proprietary Programs

In order to make possible the sale of programs for less than their development cost, we must prevent unauthorized use of the ones sold, by other people, or on other machines, or in different organizations.

(2) Files on a Public Device

Even though a memory is public (the disk), the information in a file may need protection either to prevent it from being disturbed or because it is confidential.

Keys

An individual key is associated with a person; a device key is associated with a machine; an environment key is associated with a measurable attribute of the surroundings. A key may be either specific or arbitrary. For example:

	Specific	Arbitrary
Individual	Fingerprint, voice pattern signature, social security number	password
Device	Machine manufacturer's serial number	Name of purchaser
Environment	Time and date, local magnetic field	

A key expression is a logical combination of the presence or absence of a set of keys. Let K_1, K_2, . . . be variables that are true only if the corresponding key is present. Then

$$(K_1 \wedge K_2) \vee (K_3 \wedge K_4) \quad . \ . \ .$$

expresses the interlock on ICBM launch sites where both members of a specific set of pairs must be present to activate.

$$(\overline{K_1} \vee K_2) \wedge (K_1 \vee \overline{K_2})$$

expresses a necessary lockout on an airline washroom if K_1 belongs to a man and K_2 to a woman, etc.

A key can be checked at any time, but usually as soon as access to the protected data is attempted. When a key check fails, an alarm is activated.

Alarms

Alarms come in increasingly vigorous versions. For example:

(1) Access to data refused.
(2) Further access to computer complex refused.
(3) Illegally accessed file destroyed.
(4) Operator notified.
(5) Owner of data notified.
(6) Police notified.
(7) Machine destroyed.
(8) Local environment destroyed.

Which alarm is appropriate depends upon the circumstances. (7) or (8) would be applicable as protection to military classified data.

Protection

Protection is a kind of control; it must then necessarily

lie somewhere on the path of access to the data. The main problems are in deciding where best to place it and how to check it.

There is some difference in program protection and data protection. A program is active and can check itself, data normally cannot (the chicken that laid the golden egg could squawk when stolen, the egg could not). Program loses this attribute when treated like data (cooked chicken doesn't squawk about being eaten).

No system of locks is foolproof. The general solution to a lock is to step outside its domain of applicability:

(1) If you don't know the safe combination, feel or listen for the tumbler action;

(2) if the tumblers are too good, drill the lock off;

(3) if the safe is boobytrapped, kidnap the owner to open it;

(4) if the owner refuses, torture his children, etc.

We measure the success of a lock by:

(1) The ratio of difficulty in illegal and authorized access.

(2) Probability of success in obtaining access with randomly selected keys.

(3) Probability in detecting access before damage is done.

Once an alarm is enplanted, our next line of defense is to prevent an alarmectomy. The procedure can be carried as many levels as desired although two is probably plenty.

Implementation

For the remainder of this note, we are concerned with techniques for achieving protection.

A machine readable ID card and the associated reader can be used to establish individual identity. This technique is directly applicable to console use in a remote access environment where billing information is needed. The card reader may also confiscate cards that are being illegally used as in a gas station set-up.

Any specific individual key needs recognition. Here we play the probabilities; the recognizer must be sufficiently likely to accept the authorized individual, sufficiently likely to reject all others.

A program can be made to run only on a specific computer if provision is made to read the machine serial number. The instruction is

CAB

Compare And Burn, which compares the actual machine number with the proffered key and activates an alarm if they are not equal. Slightly more general, the purchaser could request some one arbitrary bit pattern for all his machines. Then any programs he buys will run on all his computers. The preparation of the program will, of course, involve the machine dependency.

An alarmectomy protection for CAB is provided by

SAB

Sum And Burn, which checksums the program space and compares with the appropriate compiler generated constant. A further level of protection is provided if the SAB does not appear explicitly but is subtly manufactured by the program during execution. A different kind of alarmectomy backup for CAB is to flag at compile time all code as "program only". The computer will cause a Burn to be executed if code is fetched as data.

If a program is to be run only on a specific day or at some specific time, the clock can provide a key. If the machine is not to be moved, the local magnetic field provides a key, etc.

There are various ways to sound the alarm. One particularly useful way is to clear the program space and registers

to prevent the memory-dump gambit for getting around the locks. Related, when a file has to be moved from one device to another, the old copy must be destroyed or at least marked "inaccessible". Furthermore, no peeking in transit can be allowed.

Summary

The same techniques long used in protecting other valuables can be applied to computer systems. The essential ploy is to make the probable cost of illegal access much higher than the probable reward while keeping the probable cost of legal access correspondingly low. Some additions to present hardware are needed.