PROJECT MAC

October 16, 1975

Computer Systems Research Division

Request for Comments #90

SOME THOUGHTS ON ASPECTS OF THE COMPUTER SECURITY PROBLEM

by M. D. Schroeder

October 16, 1975

Some Thoughts on Aspects of the Computer Security Problem

by Michael D. Schroeder

This note takes the form of a few short observations on the computer security problem that came out of my participation in the RAND Summer Institute on Computer Security.

On Terminology

The terms unauthorized, verified, certified, secure, and security model are part of the jargon of computer security. These terms are usually informally defined and interrelated in the following way. A secure computer system is one that prevents all unauthorized release, modification, or denial of use of the information it contains. Unauthorized means contrary to the desires of the persons responsible for the contained information. The controls on information release, modification, and denial of use implemented by a system are expressed in a security model. A system is certified if someone has verified (usually interpreted to mean formally proved) that the detailed system specifications match the security model, and has verified that the system software and hardware modules correctly implement the specifications. A certified system is presumed secure.

While these informal definitions are adequate for many circumstances, they obscure several issues. Start with the notion of unauthorized. Users and managers of a computer facility have an imprecise, intuitive notion of what unauthorized should mean. This notion is apt to become precise only after an incident occurs that proves embarrassing or awkward. Thus, defining unauthorized in terms of the desires of persons is not helpful. To be precise, the definition of unauthorized must be in terms of the abstract objects and

operations provided by a system. Unauthorized really means contrary to the security model for the system, not contrary to the desires of persons, and a secure system is better defined as one that actually prevents the acts defined as unauthorized by its security model.

This improved definition of a secure system, however, also as a problem. It suggests the possibility that a system be secure by accident. But security is a useful property only if you know a system has it. For a system to be considered secure, a convincing demonstration that it matches its security model is required. Thus, verifying that the detailed system specifications match the security model and that the implementation matches the specifications should be part of producing a secure system.

But if the definition of secure implies verification of correctness, what meaning is left for the term certified? Certification closes the connection to user desires that was broken above by altering the definition of unauthorized. Certification is what you do to a secure system to evaluate and attest to its suitability for a particular class of applications. In particular, certification is a statement by an appropriate certifier 1) that the notion of unauthorized embodied in the system security model is appropriate to the intended application, i.e., matches the social authority structure into which the system must fit, and 2) that the method of verification of correctness used to produce the secure system generates a level of confidence in the correctness that is adequate for the intended application.

Finally, it should be noted that there is a broad range of opinion on the proper form and level of detail for a security model, so that one is well advised not to use the term without defining what is meant.

On the Confinement Problem

The confinement problem currently is the topic of much interest and debate. A complete solution to the confinement problem is necessary to prevent the flow of information between two processes in a computer system. Without a solution to the confinement problem, any untrustworthy program containing a trojan horse can release the information to which it has access to a cooperating receiver process. Thus, if all information release to some class of processes must be prevented, lack of confinement forces a restriction against using unverified software to handle sensitive information.

Some progress on the confinement problem has been made. In a recent paper Lipner[1] observes that the technology being developed for modeling security properties of systems and verifying the correctness of system specifications with respect to such a security model provides a methodical technique for identifying and controlling the storage channels[2] and the legitimate channels[2] in a system. Covert channels[2] (e.g., signalling by varying the rate of usage of shared resources), however, have defied methodical control. In fact, some (including me) are convinced that covert channels can never be completely controlled in a computer system that multiplexes physical resources among processes that are to be confined from one another, and that finding covert channels with significant bandwidth (>1 character per second) will always be relatively easy in such systems. The unblockability of covert channels raises the question: why bother trying to block storage and legitimate chanels? Controlling all storage and legitimate channels only can make signalling more difficult--up the work factor if you like--but not enough

---

1 Steven B. Lipner, "A Comment on the Confinement Problem," to be presented at the ACM 5th Symposium on Operating System Principles, Austin, Texas, November, 1975.

2 Lampson's terminology, see Butler W. Lampson, "A Note on the Confinement Problem," CACM 16, 10 (October, 1973), pp. 613-615.

more difficult to matter in most sensitive applications. Program verification is the only effective weapon against trojan horse programs. Perhaps a justification is that, when all storage and legitimate channels are blocked, the gymnastics required to signal through a covert channel will require much code in trojan horse programs, and therefore such code will be easy to spot by casual verification of the programs. Also, in a system where trojan horses are not the problem, complete control of legitimate and storage channels can prevent information release through non-malicious errors. Nevertheless, the question "why bother" seems to deserve debate.

On Denial of Service

Work on the computer security problem has concentrated on the problems of unauthorized release and modification of information, and has largely ignored the problem of unauthorized denial of service. Sometimes arguments are made that denial of service is less important, but most workers avoid denial of service because it seems to be a much harder problem than release and modification. Indeed, one could argue a close relationship between denial of service and preventing the unauthorized signalling over covert channels discussed above, since both issues introduce time as a factor.

It is generally assumed that the denial of service problem is independent of release and modification by direct access. There is, however, one subtle interaction. Revocation of access, an aspect of release and modification by direct access, can be prevented by denial of service. Thus, producing a secure revocation mechanism seems to require consideration of the denial of service problem.