7TH DOD/NBS COMP. SEC. CONF. SEP 24-26 1984

ON THE INABILITY OF AN UNMODIFIED CAPABILITY MACHINE TO ENFORCE THE *-PROPERTY

W. E. Boebert

Honeywell Systems and Research Center

Minneapolis, MN

ABSTRACT

It is shown that, in the absence of additional mechanisms, a machine based on capability addressing is incapable of enforcing a common class of security policies. This circumstance results from an intrinsic property of such machines: the right to exercise access carries with it the right to grant access. This result was, to the author's best knowledge, discovered by the PSOS [1] design team circa 1979, and is summarized here owing to increased interest in capability machines and the enforcement of this particular class of security policies.

UNMODIFIED CAPABILITY MACHINES

A "capability machine" [2] is one in which there exist distinguished objects of the form (Name, Mode), where Name is the name of a storage object such as a segment, and Mode is an access mode such as Read or Write. Capabilities are protected from tampering and are passed between programs in order to provide controlled sharing of storage objects. In order for a program to exercise access to a storage object, the program must possess a capability granting the desired access. "Possess" in this sense means that the program has access to the storage object which contains the capability. An "unmodified capability machine," in the sense used here, is one in which capabilities are the sole mechanism for controlling access by programs to storage objects.

SECURITY POLICIES

The Simple Security Property and the *-Property

The Simple Security Property and the *-Property are components of a commonly encountered class of mandatory security policies. Such policies define restrictions on the flow of information within a computer system. The restrictions are based on attributes associated with programs in execution (often called subjects) and storage objects (often called objects). A simplified version of these properties is presented below; the interested reader is referred to [3] and [4] for more details.

The attribute associated with a subject is its "clearance," a value which expresses the trustworthiness of the user on whose behalf the program is executing. The attribute associated with a storage object is its "classification," a value which expresses the sensitivity of the information it contains.

The Simple Security Property states that Read access is permitted if and only if clearance is greater than or equal to classification. The *-Property states that Write access is permitted if and only if classificat is greater than or equal to clearance.

Trojan Horse Attacks

These restrictions are imposed to prevent what are called "Trojan Horse attacks," in which a malicious program abuses the clearance it temporarily possesses (as the consequence of executing on behalf of a trusted user) in order to compromise the information to which that user has a legitimate right of access. In the typical Trojan Horse attack, a program written by a malicious party is made publicly available. An unwitting user invokes the program, bestowing upon it (for the period of the invocation) the clearance level of the user. The program then performs, in addition to its publicly known function, a clandestine examination and transfer of information to which its author does not have legitimate access.

The appropriateness of the Simple Security Property is obvious, since it states that no program may access information whose sensitivity exceeds the trustworthiness of the user on whose behalf the program is executing. The *-Property is less obvious; it exists to prevent the trivial circumvention of the Simple Security Property by means of Write access. Without the *-Property, it would be possible for a malicious program to write sensitive information (to which it temporarily has legitimate access, as a consequence of being invoked unwittingly by a trustworthy user) into a storage object of low classification. Such information could then be read later by a program executing on behalf of a user of low trustworthiness, thereby compromising the sensitive information. The restriction imposed by the *-Property prevents this "de facto declassification."

POLICY ENFORCEMENT ON AN UNMODIFIED CAPABILITY MACHINE

Consider an omniscient oracle which executes on a pure capability machine for the purpose of enforcing a security policy consisting of the Simple Security Property and the *-Property. Any program wishing access to a storage object must appeal to the oracle, which compares the clearance of the program to the classification of the storage object and grants or denies access accordingly, by setting the Mode value of the capability which is returned as the response to the appeal. Thus a program executing on behalf of a user with a high clearance which requests access to a storage object of low classification would receive a capability with only the Read access set, in accordance with the *-Property.

SUBVERSION OF THE ENFORCEMENT MECHANISM

In a pure capability machine, the intentions of such an oracle can be subverted by the following attack: A malicious program executing on behalf of a user with a low clearance requests a capability which grants Read and Write access to a storage object of equally low classification. We will call this object low_object and the capability RW_low_object. Such a request is naturally granted by the oracle. The program places RW_low_objec in low_object. At some later time, a user with a high clearance unwittingly invokes a Trojan Horse program. The Trojan Horse program requests a capabil

granting read access to a storage object of high classification. We will call the object high_object and the capability R_high_object. This request will also naturally be granted. Finally, the Trojan Horse program requests a capability granting Read access to low_object. This request will be granted by the oracle, in accordance with the Simple Security Property. We will call this last capability R_low_object.

The Trojan Horse program then uses R_low_object to fetch RW_low_object from low_object. A malicious program now simultaneously possesses R_high_object and RW_low_object, and is therefore able to transfer information in violation of the *_Property.

Note that this attack succeeds even if the *-Property is further restricted to state that writing can only occur if clearance equals classification. The attack can be stopped only if both reading and writing are restricted to cases where clearance equals classification, which is of course the trivial case of no flow whatever.

CONCLUDING OBSERVATIONS

The attack is made possible by an inherent attribute of pure capability machines: the right to exercise access carries with it the right to propagate that access. Thus even if an omniscient oracle correctly creates capabilities, it cannot control their further propagation. If extra mechanisms are imposed to impose this control, the machine is no longer an unmodified capability machine.

REFERENCES

[1] Neuman, P.G., et al, A Provably Secure Operating System: The System, Its Applications, and Proofs, Computer Science Laboratory Report CSL-116, SRI International, Menlo Park, CA, 7 May 1980.

[2] Dennis, J.B. and Van Horn, E.C., "Programming Semantics for Multiprogrammed Computations," CACM, IX, 3, March 1966, pp. 143-155.

[3] Landwehr, C.E., "Formal Models for Computer Security," ACM Comp. Surv. XIII, 3, Sept. 1981, pp. 247-248.

[4] Department of Defense Trusted Computer System Evaluation Criteria, Department of Defense Computer Security Center, 15 Aug. 1983.