Position: "Insider" is Relative

Matt Bishop
Dept. of Computer Science
University of California at Davis
Davis, CA 95616-8562
+1 (530) 752-8060
bishop@cs.ucdavis.edu

1. Background

Some observations will clarify what follows.

1.1 Security Policy

A security policy defines "security" for a given site or set of sites. Most security policies provide for trusted users to whom the policy either does not apply or to whom some parts of the policy do not apply. For example, in a traditional Bell-LaPadula model with strong tranquility, labels of entities do not change. In practise, this is too restrictive, so a trusted user (the site security officer) is allowed to set and change labels. Indeed, in their demonstration that Multics satisfies the model [1], Bell and LaPadula explicitly defined trusted users as subjects against whom the *-property is not enforced. The users are trusted not to violate that property.[i]

Most computer systems provide such a user as a matter of convenience. The best-known examples are those of the "superuser" or "root" on UNIX and UNIX-like systems, and "Administrator" on Windows systems. The rationale is that such users can intercede when something goes wrong, and repair the damage or prevent further damage. A security policy may constrain actions based upon procedural mechanisms, but typically there are few if any enforcement mechanisms controlling the use of these accounts.

1.2 Trust and Assumptions

All security rests on trust. Security policies embody trust in a number of ways. First is that when trusted entities violate the rules of the policy, they do so for good cause. If this assumption is incorrect, a trusted entity may cause damage or loss against which the policy is to guard—in short, the policy and its enforcement mechanisms would be ineffective. Second is that the security policy partitions all states of the system into "allowed" and "disallowed" states. If there are states not described by the security policy, this assumption is violated because it is unclear whether the states are "allowed" or "disallowed."

Policies defined non-rigorously—for example, in a natural language like English, French, or Russian—also make assumptions about environment and other legal and cultural constraints that inhibit certain enforcement mechanisms, or require the use of others. For example, if a country requires that all cryptographic keys be registered with the police, any cryptographic mechanisms must be augmented with an enforcement mechanism to transmit the key to the police. Similarly, if a culture values privacy, enforcement mechanisms requiring users to reveal personal information will not work well.

NSPW 2005 Lake Arrowhead CA USA

2. Redefining the Insider Threat

Consider the notion of an "insider" first. Merriam-Webster's dictionary [4] defines an "insider" as a "member of a group, category, or organization: as ... a person who is in a position of power or has access to confidential information; or ... one (as an officer or director) who is in a position to have special knowledge of the affairs or to influence the decisions of a company." In the context of a threat, this definition means that the insider uses that access, knowledge, or information to do something nefarious. So the insider must be trusted not to violate the confidences entrusted to her.

In terms of computer security, the insider is one who has some property that distinguishes her from others. This property requires that the insider be able to take action that would violate the security policy *were it done by an untrusted user*. The insider is trusted to do so only when appropriate. As an example, consider the proof for *get-read* satisfying the Bell-LaPadula mode. The "trusted subject" is a subject whose current security level does not dominate that of the object.[ii] Were the subject to act nefariously, the *-property would not hold.

The specific property, or properties, that distinguish an insider from other users, varies among different situations. Example properties are:

1. Having physical access to a computer;
2. Having a login on a computer; and
3. Having administrator access to the computer.

Combine this with the notion of a policy. A security policy makes specific rules about what is, and is not, allowed. An attack must violate some rule in the security policy. Trusted entities may violate some rule in the security policy, but such a violation by a trusted entity is not considered an attack. So enforcement mechanisms do not block a trusted entity from violating the security policy.

We therefore propose the following definition:

Definition. An *insider with respect to rules R* is a user who may take action that would violate some set *R* of rules in the security policy were the user not trusted. The insider is trusted to take the action only when appropriate, as determined by the insider's discretion.

Definition. The *insider threat* is the threat that the insider may abuse her discretion by taking actions that would violate the security policy when such actions are not warranted.

For completeness, we also define the insider problem.

Definition. The *insider problem* is the problem of dealing with the insider threat.

As a result of these definitions, the "insider problem" is tied to a lack of enforcement mechanisms. Specifically, an insider is an entity about which the policy makes assumptions, specifically that the entities are trusted to act in specific ways. Because of the nature of these assumptions, the enforcement mechanisms will not detect violations of the trust. This characterizes the insider problem: a violation of trust.

3. Model of the Insider

Return to our example of three users, one with physical access to the system but no account on the system; one with an ordinary user account on the system; and one with administrative privileges on her account on the system. Assuming the system requires physical access to use, the three rules with respect to which each is an insider will be:

R_1 = "physical access"

R_2 = "physical access, user account"

R_3 = "physical access, user account, administrative privileges"

These three rules form a linear hierarchy. To people who do not satisfy rule R_1, any person satisfying any rule is "more inside" than they, and so would be classified as an insider. Similarly, for people satisfying rule R_1 but not meeting rule R_2, any user who meets R_2 or R_3 is an insider with respect to the people meeting R_1 only. Hence being an insider is relative to some other set of people. Further, as anyone meeting R_3 also meets R_2 and R_1, and any person meeting R_2 also meets R_1, there is a natural, linear hierarchy of insiders.

More generally, let $I(R_i)$ be the set of users who have the property described in R_i. Then $I(R_1) \leq I(R_2) \leq I(R_3)$. To the members of $I(R_i)$, the members of $I(R_j)$, $i < j$, are insiders. Hence one cannot say that an entity is an "insider." Instead, one must say that the entity is an "insider with respect to the rule set R" or, when there is an inclusive relationship between the rule sets restricting two distinct entities, that one entity is an insider with respect to the other entity.

4. Conclusion

This position paper presents a different paradigm for the "insider threat." Specifically, an insider is a trusted entity that is given the power to violate one or more rules in a given security policy. Enforcement mechanisms are not applied against those trusted users. The insider threat occurs when a trusted entity abuses that power. The key issue in the insider threat is to determine what uses constitute abuse of that power.

Further, handling the insider threat requires alternate enforcement mechanisms, because those predicated upon the security policy will fail by definition of the "insider." Either the policy must be augmented to eliminate (some of) the trust, or enforcement mechanisms external to the policy must be defined and implemented.

This suggests that different levels of enforcement mechanisms could be used to detect insider abuse, or inhibit it. For example, a specification-based intrusion detection system [2]0 could have several specifications for a single program, each specification corresponding to a set of rules used to define levels of insiders. Then the system could determine which level is appropriate for a user, and use the appropriate specification. Similarly, an access control mechanism could use the sets of rules to define the nature and type of access allowed to a user constrained by the appropriate set of rules. The mandatory component of the Multics ring-based access control system [3] did something like this, by constraining the type of access to data based upon the ring number of the accessing procedure and the access bracket of the data segment.

This view of the insider problem leads to an interesting observation. The insider problem is not "a" problem. Rather, it is a continuum of problems, ranging from the case of a rogue user with little to no privileges to the case of an official with a large number of privileges. What distinguishes the insider problem from others is that policy-based enforcement mechanisms will not work, because the explicit granting of trust creates an exception that those mechanisms honor.

5. ACKNOWLEDGMENTS

Thanks to Karl Levitt, Todd Heberlein, Sean Peisert, and the anonymous referees. This work was supported by award CCR-0311723 from the National Science Foundation to the University of California, Davis.

6. REFERENCES

[1] D. Bell and L. La Padula, Secure Computer System: Unified Exposition and Multics Interpretation, ESD-TR-75-306, ESD/AFSC, Hanscom AFB, Bedford, MA 01731 (1975).

[2] C. Ko, M. Ruschitzka, and K. Levitt, "Execution Monitoring of Security-Critical Programs in Distributed Systems: A Specification-Based Approach," *Proceedings of the 1997 IEEE Symposium on Security and Privacy* pp. 175–187 (May 1997).

[3] E. Organick, *The Multics System: An Examination of Its Structure*, The MIT Press, Cambridge, MA (1972).

[4] Merriam-Webster Collegiate Dictionary, available at http://search.eb.com/dictionary

[i] See [1], Table 1, p. 78.

[ii] The model requires that the maximum security level of the subject dominate the security level of the object, even when the subject is trusted.