Penetration Studies

A penetration study is a test for evaluating the strengths of all security controls on the computer system. The goal of a study is to violate the site security policy. A penetration study, sometimes called a *tiger team attack* or a *red team attack*, is not a replacement for careful design and implementation with structured testing. It provides a methodology for testing the system in *toto* after it is in operation. Unlike other testing and verification technologies, it examines procedural and operational controls as well as technological controls.

Goals

A penetration test is an authorized attempt to violate specific constraints stated in the security policy or in an integrity policy. The test provides a framework in which to examine those aspects of procedural, operational, and technological security mechanisms relevant to protecting the particular aspects of system security.

A second kind of vulnerability testing tries to find some number of vulnerabilities or to find vulnerabilities within a set period of time. The strength of such a test depends on the proper interpretation of results. If the vulnerabilities are categorized and studied and if the conclusions are drawn as to the nature of the flaws, then the analysis can draw conclusions about the care taken in the design and implementation. In practice, other constraints affect the penetration study. The most notable are constraints on resources, money, and constraint on time.

Layering of Tests

A penetration test is designed to characterize the effectiveness of security mechanisms and controls to attackers. These studies are conducted from an attacker's point of view. The environment in which the tests are conducted is that in which a putative attacker would function. Different attackers have different environments. For example, insiders have access to the system, whereas outsiders need to acquire that access. This suggests a layering model for a penetrations study.

1. External attacker with no knowledge of the system.
   At this level the testers know that the target system exists and have enough information to identify it once they reach it. They must determine how to access the system. This layer is usually an exercise in social engineering and/or persistence because the testers try to trick the information out of the entity or simple dial telephone numbers or search network address spaces until they stumble upon the system. This type of testing tells very little about the security of the system itself.

2. External attacker with access to the system
   At this level, the testers have access to the system and can proceed to log in or to invoke network services available to all hosts on the network. They must then launch their attack. Typically, this step involves accessing an account from which the testers can achieve their goal or using a network service that can give them access to the system or possibly directly achieve their goal. Common forms of attack at this stage are guessing passwords, looking for unprotected accounts, and attacking network servers. Implementation flaws in servers often provide the desired access.

3. Internal attacker with access to the system.
   At this level, the testers have an account on the system and can act as authorized users of the system. The test typically involves gaining unauthorized privileges or information. At this stage, the testers acquire a good knowledge of the target system, its design, and its operation. Attacks are developed on the basis of this knowledge and access.

The usefulness of a penetration study comes from the documentation and conclusions drawn from the study and not from the success or failure of the attempted penetration.

Flaw Hypothesis Methodology

The Flaw hypothesis Methodology was developed at System Development Corporation and provides a framework for penetration studies. It consists of 5 steps:

1. Information Gathering:
   The tester becomes familiar with the system's functioning. They examine the system's design, its implementation, its operating procedures, and its use.

2. Flaw Hypothesis
   Drawing upon the knowledge gained in the first step and on the knowledge of vulnerabilities in other systems, the testers hypothesize flaws of the system.

3. Flaw Testing:
   The testers test their hypothesized flaws. If a flaw does not exist or cannot be exploited, the testers go back to step 2. If the flaw is exploitable, they proceed to the next step.

4. Flaw Generalization:
   Once a flaw has been successfully exploited, the testers attempt to generalize the vulnerability and find others similar to it. They feed their new understanding or new hypothesis back into step 2 and iterate until the test is concluded.

5. Flaw Elimination:
   The testers suggest ways to eliminate the flaw or to use procedural controls to ameliorate it.

Information Gathering and Flaw Hypothesis

In the first step of the Flaw Hypothesis Testing Methodology, the testers devise a model of the system and then explore each aspect of the design for internal consistency, incorrect assumptions, and potential flaws. They then consider the interfaces between the components and the ways in which the components work together.

The testers also examine the policies and procedures used to maintain the system, Although the design may not reveal any weaknesses, badly run or incorrectly installed systems will have vulnerabilities as a result of those errors. Sloppy administrative procedures and unnecessary use of privilege are also aspects that warrant additional investigation. Sharing accounts often enable an attacker to plant Trojan horses as does sharing of libraries, programs, and data.

Implementation problems also lead to security flaws. Models of vulnerabilities offer many clues to where the flaws may lie. Wherever the manuals suggest a limit or restriction, the testers try to violate it. Wherever the manuals describe a sequence of steps to perform an action, involving privilege data or programs, the testers omit some steps. More often than not this strategy will reveal security flaws.

Critical to this step is the identification of the structures and mechanisms that control the system. Throughout all of this, the testers draw upon their past experiences with the system, with penetrating systems in general, and on flaws that have been found in other systems.

Flaw Testing

After the testers have hypothesized a set of flaws, they determine the order in which to test the flaws. The priority of the testing is a function of the goals of the test. If the testing is to uncover major design or implementation flaws, hypothetical flaws that involve design problems or flaws in system-critical code will be given a very high priority. If the testing is to uncover the vulnerability of the system to outsider attack, flaws

related to external access protocols and programs will be given a very high priority and flaws affecting only internal use will be given a low priority.

When a system must be tested, it should be backed up and all users should be removed from it. The tester then verifies that the system is configured as needed for the test and takes notes of the requirements for detecting the flaw. The tester then verifies the existence of the flaw. In many cases, this can be done without exploiting the flaw. In some cases it cannot. The latter cases are often political in which the system developers or managers refuse to believe that the flaw exists until it is demonstrated.

Flaw Generalization

As testers successfully penetrate the system, classes of flaws begin to emerge. When the testers combine their discoveries, a generalization of the types of flaws that exist in the system can be determined.

Flaw Elimination

The flaw elimination step is often omitted because correction of flaws is not part of the penetration testing. However, flaws that are uncovered by the testing must be corrected. Proper correction of a flaw requires an understanding of the context of the flaw as well as of the details of both the flaw and its exploitation. This implies that the environment in which the system functions is relevant to correction of the flaw.

If the flaw is a design flaw, then understanding how the flaw can be exploited becomes critical because all that the site can do is to try to block those paths of exploitation or to detect any attacker who tries to exploit the flaw.

Penetrating a UNIX System

The first goal is to gain access to the system. Our target is a system connected to the Internet. We begin by scanning the network ports on the target system. The table below shows some typical ports utilized on a UNIX machine.

ftp	21/tcp	File Transfer
telnet	23/tcp	Telnet
smtp	25/tcp	Simple Mail Transfer
finger	79/tcp	Finger
sunrpc	111/tcp	Sun Remote Procedure Call
exec	512/tcp	remote process execution
(rexecd)		
login	513/tcp	remote login (rlogind)
shell	514/tcp	rlogin style exec (rshd)
printer	515/tcp	spooler (lpd)
uucp	540/tcp	uucpd
nfs	2049/tcp	networked file system
xterm	6000/tcp	x-windows server

The key to penetrating any system is an understanding of that system. Most UNIX systems use some form of sendmail as their SMTP service. That program will display information about itself when a connection is made to sendmail. This information tells the tester what set of attack will likely be successful.

Penetrating a Windows NT System

We try the same approach as above, i.e. port scanning. From this result we discover that we are dealing with a Windows NT system. In this case, we try guessing the administrator password. We get lucky and guess that the administrator password is admin. We now have administrator privilege on the local system. We want to see if we can gain the same privilege on other system in the domain.

We examine the local system and discover that the domain administrator has installed a service that is running with the privileges of a domain administrator. We obtain a program that will dump the local security authority database and load it onto the system. After executing it, we obtain the service account password. Using this password, we acquire domain administrator privileges and can now access any system in the domain.

This penetration test uncovered a serious administrative problem. A sensitive account had a password that was easy to guess. This indicates a procedural problem within the organization.. The system administrator was too busy or just forgot to choose a good password. Two generalizations are appropriate. First other systems should be checked for weak passwords. Secondly, the organization's security policies should be reviewed as should its education of its system administrators and its mechanisms for publicizing the policies.

Debate

Considerable debate has arisen about the validity of penetration studies for testing system security. At one end of the spectrum are some vendors who report that "after 1 year of our system being on the Internet, no one has successfully penetrated the system" implying, and in some cases stating, that this shows that their product is quite secure. At the other end is the claim that penetration testing has no validity and only rigorous design, implementation, and validation comprise an adequate test of security. The truth lies somewhere between these two extremes. When properly done, penetration tests examine the design and implementation of security mechanisms from the point of view of the attacker. The knowledge and understanding gleaned from such a viewpoint is invaluable.

Conclusion

Penetrating testing is a very informal, non-rigorous technique for checking the security of a system. Two problems with the Flaw Hypothesis Methodology are its dependence on the caliber of the testers and its lack of systematic examination of the system. High-caliber testers will examine the design systematically, but all too often the testing degenerates into a more scattered analysis.