Chapter 4
Computer Network Vulnerabilities

4.1 Definition

System vulnerabilities are weaknesses in the software or hardware on a server or a client that can be exploited by a determined intruder to gain access to or shut down a network. Donald Pipkin defines system vulnerability as a condition, a weakness of or an absence of security procedure, or technical, physical, or other controls that could be exploited by a threat [1].

Vulnerabilities exist do not only in hardware and software that constitute a computer system but also in policies and procedures, especially security policies and procedures, that are used in a computer network system and in users and employees of the computer network systems. Since vulnerabilities can be found in so many areas in a network system, one can say that a security vulnerability is indeed anything in a computer network that has the potential to cause or be exploited for an advantage. Now that we know what vulnerabilities are, let us look at their possible sources.

4.2 Sources of Vulnerabilities

The frequency of attacks in the last several years, and the speed and spread of these attacks, indicate serious security vulnerability problems in our network systems. There is no definitive list of all possible sources of these system vulnerabilities. Many scholars and indeed many security incident reporting agencies such as Bugtraq: the mailing list for vulnerabilities, CERT/CC: the U.S.A. Computer Emergency Response Team, NTBugtraq: the mailing list for Windows security, RUS-CERT: the Germany Computer Emergency Response Team, and U.S.DOE-CIAC: the U.S. Department of Energy Computer Incident Adversary Capability, have called attention to not only one but multiple factors that contribute to these security problems and pose obstacles to the security solutions. Among the most frequently mentioned sources of security vulnerability problems in computer networks are design flaws, poor security management, incorrect implementation, Internet technology vulnerability, the nature of intruder activity, the difficulty of
fixing vulnerable systems, the limits of effectiveness of reactive solutions, and social engineering [2].

### 4.2.1 Design Flaws

The two major components of a computer system, hardware and software, quite often have design flaws. Hardware systems are less susceptible to design flaws than their software counterparts owing to less complexity, which makes them easier to test; limited number of possible inputs and expected outcomes, again making it easy to test and verify; and the long history of hardware engineering. But even with all these factors backing up hardware engineering, because of complexity in the new computer systems, design flaws are still common.

But the biggest problems in system security vulnerability are due to software design flaws. A number of factors cause software design flaws, including overlooking security issues all together. However, three major factors contribute a great deal to software design flaws: human factors, software complexity, and trustworthy software sources [3].

#### 4.2.1.1 Human Factors

In the human factor category, poor software performance can be a result of the following:

1. **Memory lapses and attentional failures**: For example, someone was supposed to have removed or added a line of code, tested, or verified, but did not because of simple forgetfulness.
2. **Rush to finish**: The result of pressure, most often from management, to get the product on the market either to cut development costs or to meet a client deadline can cause problems.
3. **Overconfidence and use of nonstandard or untested algorithms**: Before algorithms are fully tested by peers, they are put into the product line because they seem to have worked on a few test runs.
4. **Malice**: Software developers, like any other professionals, have malicious people in their ranks. Bugs, viruses, and worms have been known to be embedded and downloaded in software, as is the case with Trojan horse software, which boots itself at a timed location. As we will see in Section 8.4, malice has traditionally been used for vendetta, personal gain (especially monetary), and just irresponsible amusement. Although it is possible to safeguard against other types of human errors, it is very difficult to prevent malice.
5. **Complacency**: When either an individual or a software producer has significant experience in software development, it is easy to overlook certain testing and other error control measures in those parts of software that were tested previously in a similar or related product, forgetting that no one software product can conform to all requirements in all environments.