CHAPTER 4
Data and Application Security
Part 1

Dr. Mohamed Marie

Topics in this Chapter:

- Types of Applications
- Application Models and Technologies
- Application Threats and Countermeasures
- Security in the Software Development Life Cycle
- Application Security Controls
- Databases and Data Warehouses

Types of Applications

In this section the following types of applications will be discussed:

- Agents
- Applets
- Client-server
- Distributed
- Web applications

Types of Applications

Agents

- **Agents** are small standalone programs that are part of a larger application. Agents carry out specific functions, such as remote status collection or remote system management.

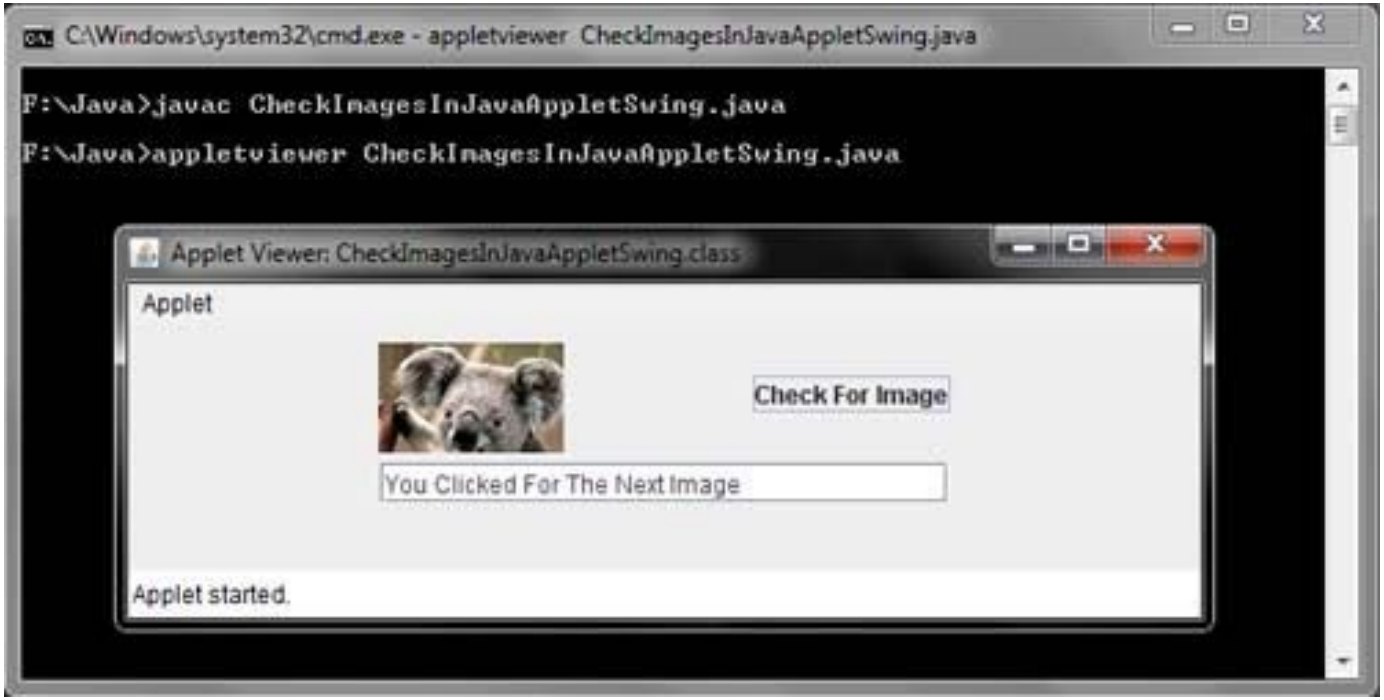
Some examples of agents include:

- **Anti-virus.** You could consider the anti-virus program on a workstation or server as an agent in an enterprise environment that includes a central management console.
- **Patch management.** An agent on each server periodically queries the OS on the existence of software patches, and will install patches when commanded to do so from the central patch management server.
- **Configuration management.** A central server tracks and manages the OS configuration of each server and workstation by communicating to agents on those managed systems. Agents will collect configuration information and pass it back to central servers; agents will also perform configuration changes upon command.

Types of Applications

Applets

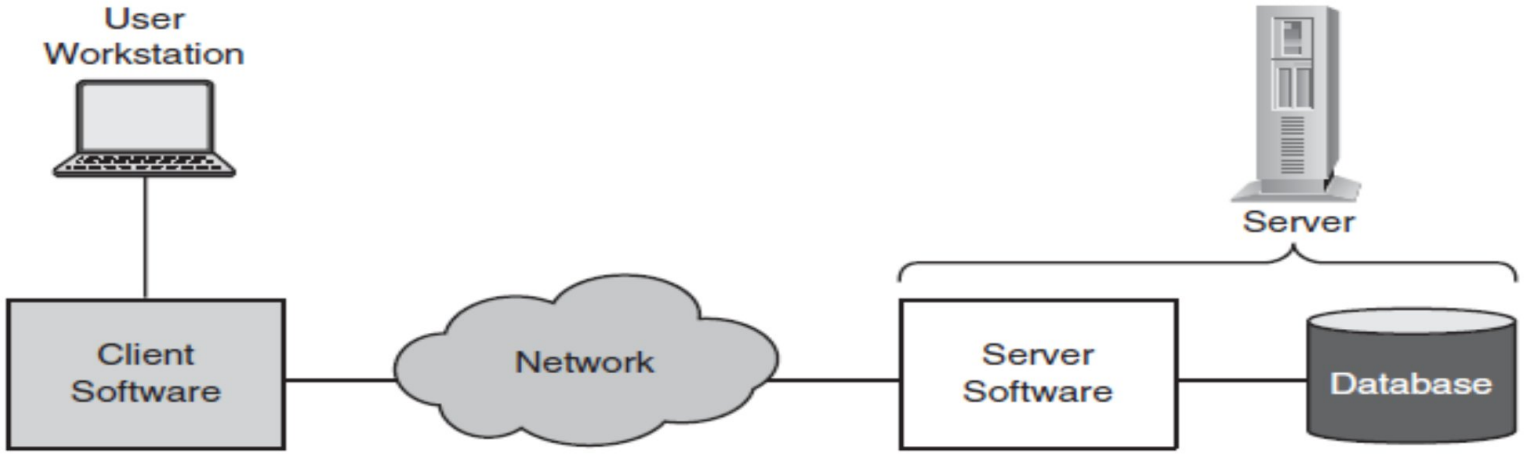
- An **applet** is a software program that runs within the context of another program. Unable to run on its own, an applet performs a narrow function.
- The most common use of applets is within Web browsers.
- Examples of Web browser applets include media players such as Flash and Shockwave players, and content viewers such as Adobe Reader. Figure 3-1 shows a Java applet running in a Web browser window.



Types of Applications

Client-server Applications

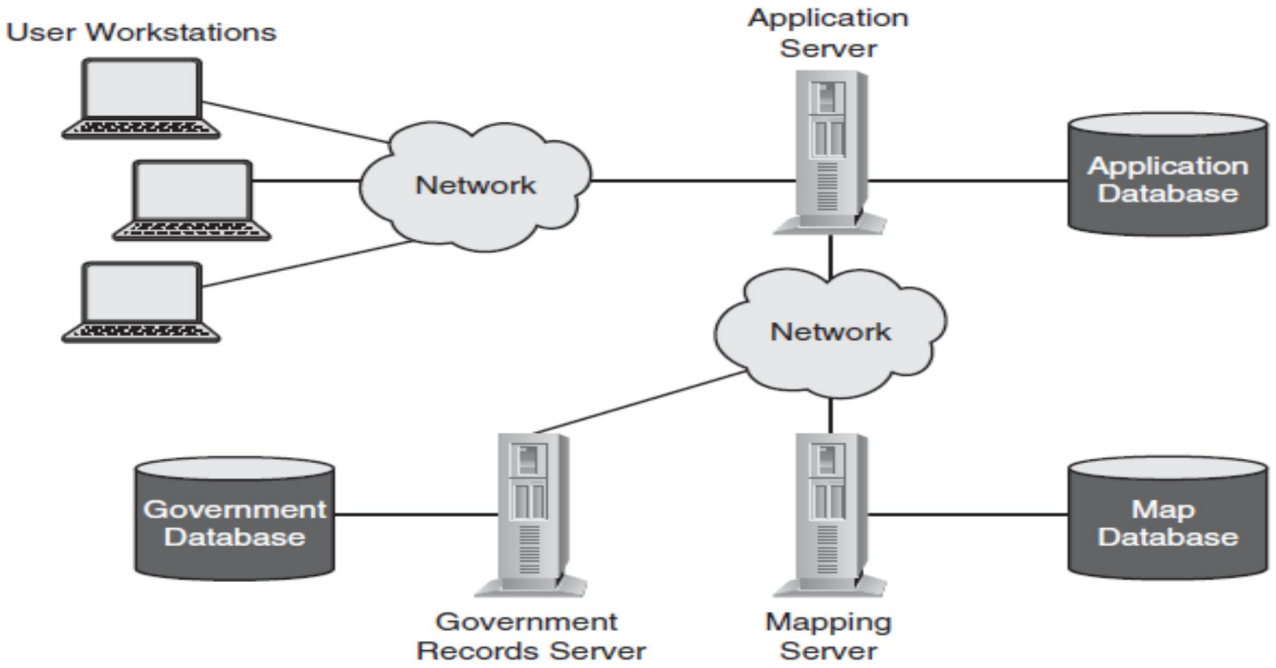
- The software components in client-server applications are not centralized, but instead are present in two places: clients and servers. Clients and servers usually communicate with each other via networks. Specific characteristics of clients and servers are explained here.
- **Client characteristics.** Client software is the part of the application used by humans, and primarily contains user interface logic that displays instructions and data, accepts input data from a keyboard or other device, and accepts instructions or directives from users.
- **Server characteristics**. In typical client-server applications, the server component performs database updates and communicates with clients. Server components typically do not have user interface logic but instead run as daemons or services.



Types of Applications

Distributed Applications

- **Distributed applications** have software components running on several separate systems in a wide variety of architecture including two-tier, three-tier, and multi-tier. Usually, distributed systems are designed in a way to physically or logically separate different functions in the application. There are many possible reasons for this separation, including scalability, performance, geographical, and security.



Types of Applications

Web Applications

Web applications provide several significant advances over client-server applications including:

- **Thinner clients**. The browser becomes the client software, which works with all of the enterprise's web applications.
- **Better network performance.** More business logic resides on the server, and only display logic resides on the workstation, significantly reducing demands on the network. This enables more users to use the application without incurring meltdowns on the network.
- **Lower cost of ownership.** The organization only needs to make sure that workstations have a reasonably current version of a web browser. The administrative overhead related to maintaining versions client software components for all of the organization's client-server applications is eliminated.
- **More terminal types supported.** Users are no longer locked into a hardware or OS platform, but can access applications using a variety of terminal types including Windows, UNIX, and Apple workstations, and also mobile devices such as smartphones and PDAs.
- **Any user can access the application.** Because a web application requires only a browser as a client, any user anywhere in the world can potentially access the application.

Application Models and Technologies

- Computer systems and application programming languages are generally built upon models that give the language some form and structure. Four models that have been the most popular are:
  - Control flow,
  - Structured,
  - Object-oriented, and
  - Knowledge-based.

Application Models and Technologies

Control Flow Languages

- The earliest computer languages, known as control flow, were sequential in nature—that is, they executed statements one after the other. Most languages used some variation of an "if-then" construct as well as a "goto" construct to alter the sequence of instructions. The disadvantage of control flow is the difficulty in verifying a program's integrity. Excessive use of "goto" statements turned linear logic into "spaghetti" code that was difficult to analyze and understand. The "goto" statement was demonized, and structured languages won favor.

Structured Languages

- Programming languages with procedural structure were developed to overcome the deficiencies of control flow applications with their "goto" statements. Structured languages used subroutines or functions and relied less on goto (some structured languages do not have goto at all). Structured languages tend to be structured in "blocks" of code that are bracketed by keywords such as if…fi, BEGIN…END, {…}, if…then…else…endif, and so on. The flow of logic in structured languages tends to be hierarchical rather than linear, which tends to make analysis and verification somewhat easier.

Application Models and Technologies

Object Oriented Programming

- Object oriented (commonly known as OO, and pronounced oh-oh) programming is a completely different approach to computer languages than the structured languages in use such as BASIC and C. Object oriented programming has a particular vocabulary that is used to describe how components are named and assembled into programs. These terms are:
  - Class
  - Object
  - Method
  - Encapsulation
  - Inheritance
  - Polymorphism

Application Models and Technologies

Object Oriented Programming

- **Class:** A **class** defines the characteristics of an object, including its characteristics such as attributes, properties and fields, plus the methods it can perform.
- **Object:** An **object** is a particular instance of a class. The class superhero defines all superheroes and lists their characteristics. The object Superman is one particular superhero. The object Superman is an instance of the class superhero.
- **Method:** A **method** defines the abilities that an object can perform. It may contain instructions, as well as input variables and output variables. A method is similar to a function or subroutine in structured programming. It consists of some instructions or calculations, and communicates using message passing. The method fly() is one of Superman's methods.
- **Encapsulation: Encapsulation** refers to the implementation details in a method that are concealed. For example, the code for Superman's fly() method contains several other methods like propulsion() and steering() that other objects do not need to be concerned about.
- **Inheritance:** The term **inheritance** refers to the characteristics of a subclass that inherit attributes from their parent classes. And in turn, subclasses can introduce their own attributes that are passed to their subclasses.
- **Polymorphism:** The characteristic of **polymorphism** allows objects of different types to respond to method calls differently, depending upon their type. For instance, a call to a compute-tax() method will result in different behavior depending upon the country (type) where the transaction takes place (there are not only different tax rates, but different taxable goods, and some people are taxed at different rates).

Application Models and Technologies

Knowledge-based Applications

- **Knowledge-based systems:** are applications that are used to make predictions or decisions based upon input data. They include feedback mechanisms that enable them to learn and refine their guidance, improving their accuracy over time. The objective of knowledge-based systems is the ability for a system to possess some of the qualities of human reasoning.
- **Neural Networks:** Neural networks are so-named because they are modeled after biological reasoning processes that humans possess. A neural network (NN) consists of interconnected artificial neurons that store pieces of information about a particular problem. Neural networks are given many cases of situations and outcomes; the more events the neural network is given, the more accurately it will be able to predict future outcomes.
- **Expert Systems:** Expert systems accumulate knowledge on a particular subject, including conditions and outcomes. The more samples that the expert system is able to obtain, the greater is its ability to predict future outcomes.