Language-Based Protection

- Specification of protection in a programming language allows the high-level description of policies for the allocation and use of resources.

- Language implementation can provide software for protection enforcement when automatic hardware-supported checking is unavailable.

- Interpret protection specifications to generate calls on whatever protection system is provided by the hardware and the operating system.
Protection in Java 2

- Protection is handled by Java Virtual Machine (JVM)

- A class is assigned a protection domain when it is loaded by the JVM

- The protection domain indicates what operations the class can (and cannot) perform

- If a library method is invoked that performs a privileged operation, the stack is inspected to ensure the operation can be performed by the library
## Stack Inspection

<table>
<thead>
<tr>
<th>protection domain:</th>
<th>untrusted applet</th>
<th>URL loader</th>
<th>networking</th>
</tr>
</thead>
<tbody>
<tr>
<td>socket permission:</td>
<td>none</td>
<td>*.lucent.com:80, connect</td>
<td>any</td>
</tr>
<tr>
<td>class:</td>
<td>gui:</td>
<td>get(URL u):</td>
<td>open(Addr a):</td>
</tr>
<tr>
<td></td>
<td>get(url); open(addr); ...</td>
<td>doPrivileged { open('proxy.lucent.com:80'); } &lt;request u from proxy&gt; ...</td>
<td>checkPermission (a, connect); connect (a); ...</td>
</tr>
</tbody>
</table>
Chapter 15: Security - Objectives

- To discuss security threats and attacks
- To explain the fundamentals of encryption, authentication, and hashing
- To examine the uses of cryptography in computing
- To describe the various countermeasures to security attacks
The Security Problem

- Security must consider external environment of the system, and protect the system resources.
- Intruders (crackers) attempt to breach security.
- **Threat** is potential security violation.
- **Attack** is attempt to breach security.
- Attack can be accidental or malicious.
- Easier to protect against accidental than malicious misuse.
Security Violations

- **Categories**
  - Breach of confidentiality
  - Breach of integrity
  - Breach of availability
  - Theft of service
  - Denial of service

- **Methods**
  - Masquerading (breach authentication)
  - Replay attack
    - Message modification
  - Man-in-the-middle attack
  - Session hijacking
Standard Security Attacks

1. Normal
   - Sender
   - Communication
   - Receiver

2. Masquerading
   - Sender
   - Communication
   - Attacker
   - Communication
   - Receiver

3. Man-in-the-middle
   - Sender
   - Communication
   - Attacker
   - Communication
   - Receiver
Security Measure Levels

- Security must occur at four levels to be effective:
  - Physical
  - Human
    - Avoid social engineering, phishing, dumpster diving
  - Operating System
  - Network

- Security is as weak as the weakest link.
Program Threats

- **Trojan Horse**
  - Code segment that misuses its environment
  - Exploits mechanisms for allowing programs written by users to be executed by other users
  - **Spyware, pop-up browser windows, covert channels**

- **Trap Door**
  - Specific user identifier or password that circumvents normal security procedures
  - Could be included in a compiler

- **Logic Bomb**
  - Program that initiates a security incident under certain circumstances

- **Stack and Buffer Overflow**
  - Exploits a bug in a program (overflow either the stack or memory buffers)
C Program with Buffer-overflow Condition

```c
#include <stdio.h>
#define BUFFER SIZE 256
int main(int argc, char *argv[]) {
    char buffer[BUFFER SIZE];
    if (argc < 2)
        return -1;
    else {
        strcpy(buffer, argv[1]);
        return 0;
    }
}
```
Layout of Typical Stack Frame

- **frame pointer**
- **bottom**
- **top**

- **return address**
- **saved frame pointer**
- **automatic variables**
- **parameter(s)**

The stack frame grows from the bottom towards the top.
Modified Shell Code

```c
#include <stdio.h>
int main(int argc, char *argv[])
{
    execvp("/bin/sh","/bin/sh", NULL);
    return 0;
}
```
Hypothetical Stack Frame

(a) Before attack

(b) After attack

- return address
- saved frame pointer
- buffer(BUFFER_SIZE - 1)
- ... (empty)
- buffer(1)
- buffer(0)

- address of modified shell code
- NO_OP
- ... (empty)

Copied from buffer (BUFFER_SIZE - 1) to modified shell code.
Program Threats (Cont.)

- Viruses
  - Code fragment embedded in legitimate program
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro
    - Visual Basic Macro to reformat hard drive
      ```vba
      Sub AutoOpen()
      Dim oFS
      Set oFS = CreateObject("Scripting.FileSystemObject")
      vs = Shell("c:command.com /k format c:", vbHide)
      End Sub
      ```
Program Threats (Cont.)

- **Virus dropper** inserts virus onto the system
- Many categories of viruses, literally many thousands of viruses
  - File
  - Boot
  - Macro
  - Source code
  - Polymorphic
  - Encrypted
  - Stealth
  - Tunneling
  - Multipartite
  - Armored
A Boot-sector Computer Virus

- Virus copies boot sector to unused location X
- Virus replaces original boot block with itself
- At system boot, virus decreases physical memory, hides in memory above new limit
- Virus attaches to disk read-write interrupt, monitors all disk activity
- Whenever new removable R/W disk is installed, it infects that as well
- It blocks any attempts of other programs to write the boot sector
- It has a logic bomb to wreak havoc at a certain date
System and Network Threats

- Worms – use *spawn* mechanism; standalone program
- Internet worm
  - Exploited UNIX networking features (remote access) and bugs in *finger* and *sendmail* programs
  - **Grappling hook** program uploaded main worm program
- Port scanning
  - Automated attempt to connect to a range of ports on one or a range of IP addresses
- Denial of Service
  - Overload the targeted computer preventing it from doing any useful work
  - Distributed denial-of-service (**DDOS**) come from multiple sites at once
The Morris Internet Worm

- Grappling hook
- Worm

Target system

- rsh attack
- Finger attack
- Sendmail attack
- Request for worm

Infected system

- Worm sent