计算机网络原理实验报告

学院 计算机学院 专业 计算机科学与技术 班级 10012006

学号 2020303245 姓名 夏卓 实验时间： 2022/11/19

一、 实验名称：

TCP 端口扫描

二、 实验目的：

理解 TCP 端口扫描的含义及其实现方法，分析并比较采用单进程与多进程实现方法对扫描效率的影响。

三、 实验环境：

Win10,Intelx86

四、 实验内容及步骤：

实验内容：

1. 采用单进程完成对0~1024范围内的TCP端口扫描,并检验结果是否正确，将正确的结果在屏幕上打印，统计实验检测完成时间并打印。
2. 采用多进程完成对0~1024范围内的TCP端口扫描,并检验结果是否正确，将正确的结果在屏幕上打印，统计实验检测完成时间并打印。

实验步骤：

(1) 理解 TCP 端口扫描的含义及其实现方法

在网络环境中，不同计算机进程之间的通信通常采用 C/S 模式，服务器在周知端口上开启一项服务，等待客户机连接请求，建立 TCP 连接，并提供服务。本实验利用操作系统提供的 connect()系统调用来判断 TCP 应用服务器的某个端口是否开放。若端口处于侦听状态，则 connect()成功返回，同时表明该 TCP 端口开放；否则，表明该 TCP 端口未开放，即没有提供服务。

这个技术最大的优点是，该系统调用不受任何限制，系统中任何用户都有权限使用这个调用。但存在的问题是，该调用容易被目的端日志系统记录而被管理员发现。若扫描程序采用单线程方式，则扫描速度很慢，故可以利用多线程的扫描方式加快扫描速度。

(2) 设计并实现单进程下的 TCP 端口扫描程序。

单进程下的 TCP 扫描是串行进行的，即是从 0 号端口到 1024 号端口依次进行尝试连接，若连接成功说明端口开放连接成功，否则说明端口不开放。每次都需要创建新的 socket 进行连接，另外可以添加计时函数对运行时间进行统计，由于单进程运行时间较慢，可以对每次扫描端口

进行计时：

```
for (int i = 0; i < 1024; i++)
{
    clock_t start_time = clock();
    ScanTcpPort(i);
    printf("scan port%d ", i);
    clock_t end_time = clock();
    cout << "cost time: " << (double)(end_time - start_time) / CLOCKS_PER_SEC << "s" << endl;
}
```

(3) 设计并实现多线程下的 TCP 端口扫描程序。

与单进程相比，多线程的效率更高，代码参考书 P266。在主函数中，需要等待子进程全部结束后再退出，因此需要进行等待：

```
while (ThreadCount > 0)
{
    Sleep(100);
    WSACleanup();
}
```

另外在创建线程时，需要统计目前并发运行的总线程数，这里用到了原子操作，当线程数大于设置的最大并发数时需要停止创建，每个线程中运行的函数与单线程中的类似：

```
for (int i = 0; i < 1024; i++)
{
    while (ThreadCount >= MaxThreadNum)
        Sleep(20);
    DWORD ThreadId;
    CreateThread(NULL, 0, ScanTcpPort, (LPVOID) new short(i), 0, &ThreadId);
}
```

五、 实验结果：

单进程端口扫描：

由于时间过长，这里显示了扫描每个端口所需花费的时间，可以看出大概扫描每个端口都需要 2s 的时间，25 号端口是开放的。

```
scan port20 cost time: 2.034s
scan port21 cost time: 2.046s
scan port22 cost time: 2.045s
scan port23 cost time: 2.043s
scan port24 cost time: 2.051s
TcpPort:25 open
scan port25 cost time: 0.002s
scan port26 cost time: 2.05s
scan port27 cost time: 2.04s
```

多线程端口扫描：

可以看出相比与单进程，多线程端口扫描的效率更高，检测结果也一致。

```
(base) PS D:\大学\大三上\计网\实验5\端口扫描> .\mul_port_scan 10.30.217.218
Open TCP Ports:
TcpPort:25
TcpPort:135
TcpPort:80
TcpPort:110
TcpPort:139
TcpPort:445
TcpPort:902
TcpPort:912
cost time: 4.35s
```

六、 实验总结

端口扫描的基本原理

向目标主机的所有或者需要扫描的特定端口发送特殊的数据包，若该端口号对应的网络服务对外提供该服务就会返回信息，扫描器通过分析其返回的信息判断目标主机端口的服务状态，发现特定主机提供了哪些服务，进而利用服务的漏洞对网络系统进行攻击。

TCP 端口扫描类型

1. 全扫描

扫描主机尝试使用三次握手与目标主机的某个端口建立正规的连接，若成功建立连接，则端口处于开放状态，反之处于关闭状态。全扫描实现简单，且以较低的权限就可以进行该操作，实现的方法通常使用操作系统提供的connect()函数。但是在流量日志中会有大量明显的记录。

2. 半扫描

半扫描也称 SYN 扫描，在半扫描中，仅发送 SYN 数据段，如果应答为 RST，则端口处于关闭状态，若应答为 SYN/ACK，则端口处于监听状态。不过这种方式需要较高的权限，而且现在的大部分防火墙已经开始对这种扫描方式做处理。

3. FIN 扫描

FIN 扫描是向目标发送一个 FIN 数据包，如果是开放的端口，会返回 RST 数据包，关闭的端口则不会返回数据包，可以通过这种方式来判断端口是否打开。这种方式并不在 TCP 三次握手的状态中，所以不会被记录，相对 SYN 扫描要更隐蔽一些。

教师评语：

成绩：_____ 教师签名：_____ 批阅日期：_____