lserv2设计

本文档旨在lserv2的设计进行说明。（同时是某课的作业）
November 8, 2014
与本文档相关的源代码文件为lserv2.c以及lclie2.c，本文档的源文件为lserv2.tex，lserv2的配置文件目录为config。均可从下列地址获得。

http://github.com/CDLuminate/lsock_learn/

Contents

1 编译和运行

这是针对linux编写的Socket程序，在终端下执行

```
$ make
```

即编译好。然后执行

```
$ ./server2
```

即在预定义的端口（2333）上开启了服务器。若要使用服务器，可用Netcat或者telnet这个工具，即

```
$ nc localhost 2333
```

与该服务器进行互动。（lclie2.c也能互动）
具体可以使用的指令参见下文。

1.1 配置

端口 可用 -p 选项指定端口

账户 在 config 目录中进行配置

2 设计

2.1 实现概要

在实现了C/S的相互通信的基础上[1]，在服务器端添加了功能， 让服务器不仅能接收并打印收到的信息，还能解析客户端发送的信息。
lserv2采用的模式是

客户端 客户端只负责发送和接收信息，剩下的所有处理过程都在服务器端发生。 用户使用客户端的时候，用户输入指令，指令发送到服务器，服务器解析并进行相应反馈。

服务器 接收指令，解析，反馈。具体功能参见下文。

这样的设计有一点好处：将客户端的功能缩减到极致（只剩下信息收发功能），于是可以预防比如 "对客户端进行逆向工程，把身份认证部分填充以 0x90[2]，于是。。。"这样的情况。亦即预防 调皮的客户端，甚至不怀好意的重新实现过的客户端。

2.2 具体功能

以下命令都可以在客户端中手工输入，同时服务器能够反馈:

1. GET: 回传一段预定义的HTML代码
2. USER: 指令格式为 USER ¡username¿，登记用户名
3. PASS: 指令格式为 PASS ¡password¿，登记密码
4. QUIT: 退出程序（客户端和服务器均会退出）
5. LOGOUT: 登出
6. SEC: 打印预定义的秘密信息，如果没有成功登录过则会发出警告
7. : 如果遇到其他指令，均不进行额外操作，服务器会提示不支持该指令

[1]也即，服务器和客户端可以在stdin中写入信息， 然后发送给对方，对方在stdout中打印这条信息。如此实现互动。
[2]针对x86架构

3 安全分析

3.1 缓冲区溢出

lserv2 在实现之初就开始预防缓冲区溢出，具体体现在比如：

1. 避免使用 sprintf, strcpy, strcmp 等函数，一律换成更能预防溢出的函数 snprintf, strncpy, strncmp 等。

2. instruction 缓冲区有 1024 Byte， 但是进行 read() 调用时最大读取 1023 Byte， 一定程度上能够预防某些调皮的客户端进行溢出测试。

因此作者本人也深刻怀疑对该程序进行缓冲区溢出研究的难度，但鉴于没有经验，无法评论。

3.2 暴力破解

为方便研究，作者使用了固定4位的用户名和密码。因此暴力破解"最多"只需要尝试

$$Try = 256^4 \tag{1}$$

种可能性。实际需要测试的值数量小于上述值。因此暴力破解不失为一种良方。

3.3 逆向

客户端功能极其精简，逆也逆不出有价值的东西来。毕竟Netcat也能与Server正常通信。 服务器端逆向不太符合实际情景。

3.4 中间人攻击/入侵

出于C/S通信采用明文，因此只要获得 C/S/Route 之间任意一方的控制权，进行抓包， 就能够抓取认证信息。

3.5 配置文件

用户名和密码均放于配置文件中，文件泄漏会导致风险。

4 参考以及原型

4.1 参考书籍

《UNIX网络编程》

4.2 设计模式

TCP并发服务器，指令驱动

5 其他

更多细节均在程序注释中写明。 lserv2[3]以及本文档(.tex)遵循MIT许可证。

5.1 changelog

Date: ?

1. 增加一个读取配置文件（用户名和密码）的部分，避免用户名和密码写死的尴尬。
2. 增加getopt部分，可以更改监听端口。

Date, 2014/11/08

1. 增加select()调用，使lserv2阻塞在select而不是直接阻塞在read()和write()调用。

[3]https://github.com/cdluminate