UNIVERSITY OF CAMBRIDGE

Honware: A virtual honeypot framework for capturing CPE and IoT zero days

Alexander Vetterl and Richard Clayton

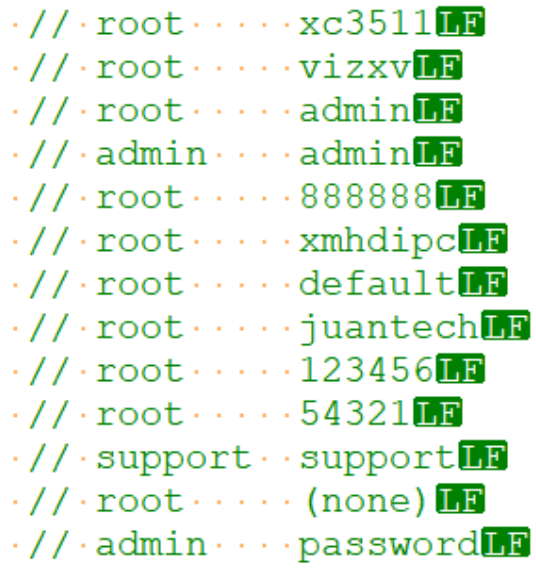
University of Cambridge

eCrime 2019, APWG Symposium on Electronic Crime Research — November 14, 2019

Introduction

Honeypot:
A resource whose value is being attacked or compromised

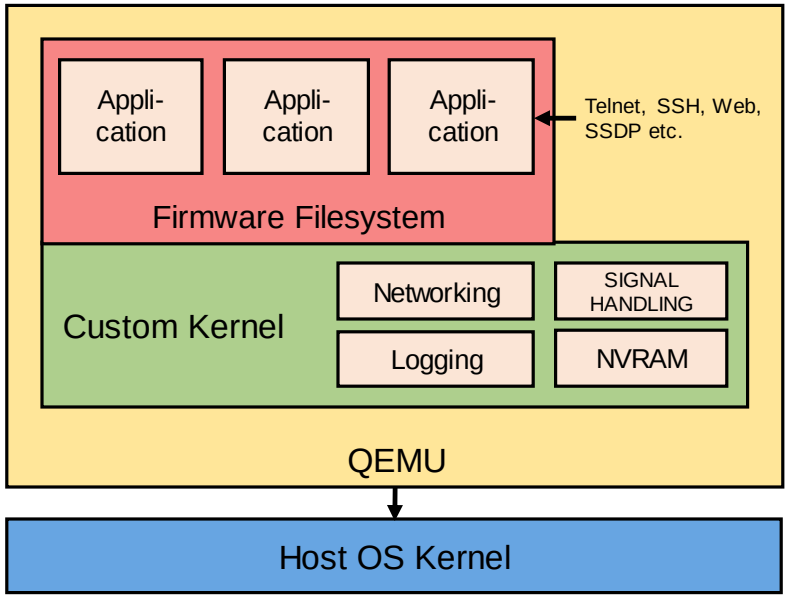
- We are good in building software honeypots for specific Malware (e.g. Mirai)
- Honeypots emulate a vulnerable device by sending appropriate strings back
- Finding vulnerable devices never has been easier
  - Stateless scanning & Shodan, Censys, Thingful



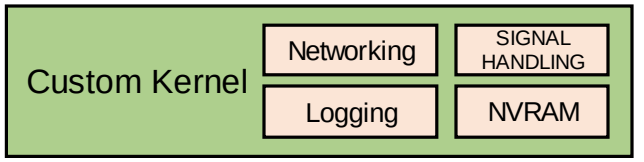
<u>Problem:</u>
Slow, iterative process only suitable for well-understood attacks

Honware: Virtualised honeypot framework

- Virtualised, because deploying and monitoring physical devices does not scale
- Aimed for Linux-based CPE and IoT devices
- We need access to the firmware image and the firmwares filesystem
- We want to run the firmwares‘ applications such as Telnet, SSH and Web servers
- Lightweight
  - <64MB RAM, <128MB disk space
- Fast: Honeypots can be set-up in minutes!



Customised pre-built kernel (1/2)

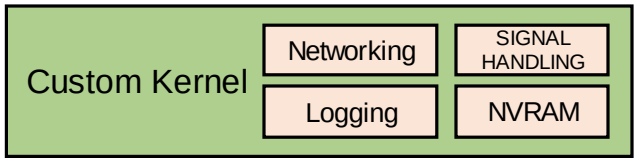


We built kernels for ARM, MIPSEB and MIPSEL

1. **Honeypot logging**
   - do_execve
2. **Signal interception**
   - SIGABRT (abort)
   - SIGSEGV (seg fault)
   - SIGPFE (floating point errors)
3. **Module loading**
   - Ignoring vermagic strings (e.g. 2.6.22-xyz)

```
httpd/317: potentially unexpected fatal signal 11.
[  170.020000] CRLF
[  170.020000] Cpu 0CRLF
[  170.020000] $ 0   : 00000000 1000a400 80800000
[  170.024000] $ 4   : 00000008 00000008 80808080
[  170.024000] $ 8   : 00000000 00000000 2aceb870
[  170.024000] $12   : 2ad2cb80 0000016d 07a99846
[  170.024000] $16   : 00000008 2ad17d5c 7fb3adb0
[  170.024000] $20   : 00480000 0045f310 0045f2fc
[  170.024000] $24   : 2ad2b4b0 2ad43d80
[  170.028000] $28   : 2ad78ac0 7fb3ad08 7fb3ada8
[  170.028000] Hi    : 0000019cCRLF
[  170.028000] Lo    : 0003c3daCRLF
[  170.028000] epc   : 2ad43dbc 0x2ad43dbcCRLF
[  170.028000]     Not taintedCRLF
[  170.028000] ra    : 2ad41d78 0x2ad41d78CRLF
[  170.028000] Status: 0000a413    USER EXL IE CR
[  170.028000] Cause : 10800008CRLF
[  170.032000] BadVA : 00000008CRLF
[  170.032000] PrId  : 00019300 (MIPS 24Kc)CRLF
```

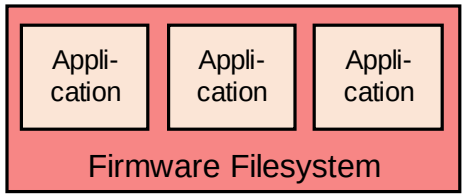
Customised pre-built kernel (2/2)



4. **NVRAM (non-volatile memory)**
   - Set LD_PRELOAD to the path of our own nvram implementation
   - Intercept nvram_get and nvram_set calls
5. **Network configuration**
   - Look for bridge configuration: br0 and ra0
   - If that fails, the kernel will execute a default configuration (customisable by users!)
     - Necessary interfaces
     - Assign static IP addresses

```
nvram_set: lan_ipaddr = "192.168.1.250"CRLF
nvram_set: lan_dns = "0.0.0.0"CRLF
nvram_set: lan_gateway = "0.0.0.0"CRLF
nvram_set: pre_lan_ipaddr = "0.0.0.0"CRLF
nvram_set: pre_lan_netmask = "255.255.255.0"CRLF
nvram_set: lan_netmask = "255.255.255.0"CRLF
nvram_set: lan_proto = "dhcp"CRLF
nvram_set: lan_wins = ""CRLF
nvram_set: lan_domain = ""CRLF
nvram_set: lan_lease = "60"CRLF
nvram_set: lan_stp = "1"CRLF
nvram_set: lan_route = ""CRLF
nvram_set: lan1_ifname = ""CRLF
nvram_set: lan1_ifnames = ""CRLF
nvram_set: lan1_hwnames = ""CRLF
nvram_set: lan1_hwaddr = ""CRLF
nvram_set: lan1_dhcp = "0"CRLF
nvram_set: lan1_ipaddr = "192.168.2.1"CRLF
nvram_set: lan1_netmask = "255.255.255.0"CRLF
nvram_set: lan1_gateway = "192.168.2.1"CRLF
```

Step 1: Extracting firmware images



Binwalk

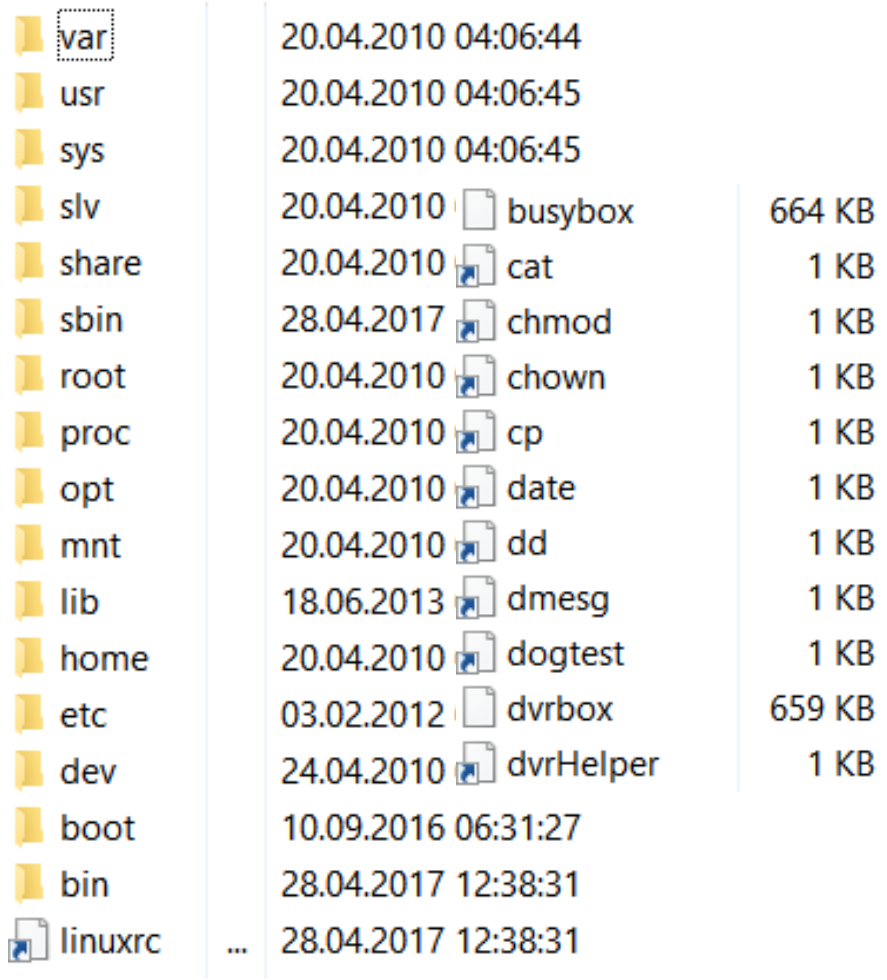
- Looking for standard Linux filesystem structure (bin, usr, proc etc.)

Creating an ext2 filesystem

- Copying the firmwares‘ structures and files
- Typically very small (<128MB)

Identifying the architecture based on ELF headers

- e.g. Busybox binary
- Used to select the appropriate kernel



Step 2: Modifying filesystem & preparation

Supports custom configurations

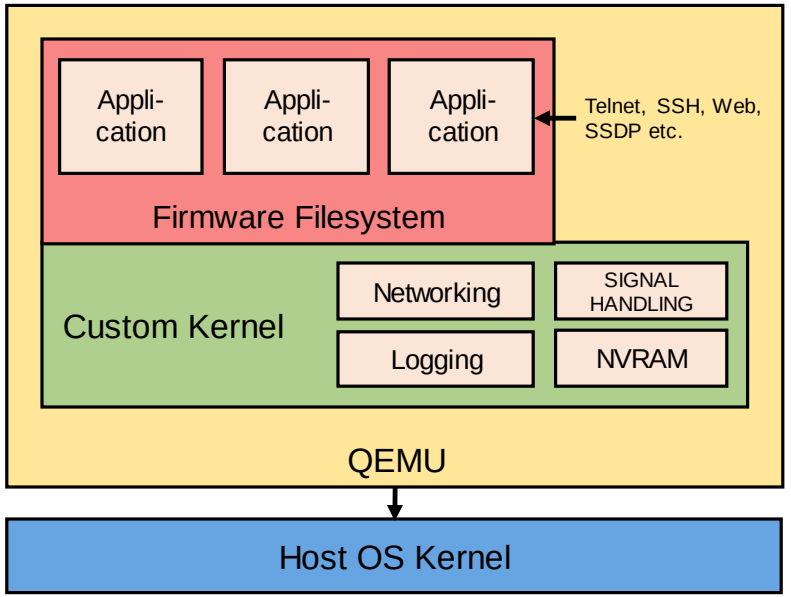
- Modified do_execve to execute, if present, /sbin/boot.sh through the kernel function call_usermodehelper

NVRAM emulation

- Added as kernel module

Network configuration

- Re-route incoming packets on the host ethernet interface to the QEMU tap interface and
- Post-route the packets back to the host



Evaluation

- Extraction
- Network reachability
- Responding services

- 23,035 firmware images from Firmadyne (2016)
- As of March 2019, 8,387 images can still be downloaded

- Timing attacks

- Looked for self-identifying devices
- Repeated measurements for three protocols: FTP, Telnet and HTTPS

- Case studies

- Deployed multiple honeypots on the Internet
- Four case studies which show that devices can be rapidly emulated

Eval. 1: Extraction and network reachability

#	Brand	Available (2019-03/2016-02/Δ)		Extracted (honw./firm./Δ)		Network reach. (honw./firm./Δ)	
1	Actiontec	0/14	14↓	–	–	–	–
2	Airlink101	0/15	15↓	–	–	–	–
3	Apple	0/9	9↓	–	–	–	–
4	Asus	1/3	2↓	1/1	←	1/0	1↑
5	AT&T	3/25	22↓	0/2	2↓	–	–
6	AVM	0/132	132↓	–	–	–	–
7	Belkin	123/140	17↓	49/49	←	9/0	9↑
8	Buffalo	97/143	46↓	6/7	1↓	2/1	1↑
9	CenturyLink	13/31	18↓	7/4	3↑	7/0	7↑
10	Cerowrt	0/14	14↓	–	–	–	–
11	Cisco	0/61	61↓	–	–	–	–
12	D-Link	1443/4688	3245↓	537/498	39↑	272/115	157↑
13	Forceware	0/2	2↓	–	–	–	–
14	Foscam	44/56	12↓	5/5	←	–	–
15	Haxorware	0/7	7↓	–	–	–	–
16	Huawei	13/29	16↓	0/3	3↓	–	–
17	Inmarsat	0/47	47↓	–	–	–	–
18	Iridium	0/17	17↓	–	–	–	–
19	Linksys	32/126	94↓	26/26	←	15/1	14↑
20	MikroTik	4/13	9↓	–	–	–	–
21	Netgear	1396/5280	3884↓	639/629	10↑	384/187	197↑
22	On Networks	0/28	28↓	–	–	–	–
23	Open Wir.	0/1	1↓	–	–	–	–
24	OpenWrt	756/1498	742↓	714/705	9↑	674/0	674↑
25	pfSense	214/256	42↓	–	–	–	–
26	Polycom	612/644	32↓	0/24	24↓	–	–
27	QNAP	8/464	456↓	–	–	–	–
28	RouterTech	0/12	12↓	–	–	–	–
29	Seiki	0/16	16↓	–	–	–	–
30	Supermicro	0/150	150↓	–	–	–	–
31	Synology	1977/2094	117↓	1866/239	1627↑	–	–
32	Tenda	6/244	238↓	4/3	1↑	2/0	2↑
33	Tenvis	9/49	40↓	6/6	←	6/4	2↑
34	Thuraya	0/18	18↓	–	–	–	–
35	Tomato	362/2942	2580↓	362/362	←	217/0	217↑
36	TP-Link	463/1072	609↓	171/171	←	147/95	52↑
37	TRENDnet	336/822	486↓	134/100	34↑	87/37	50↑
38	Ubiquiti	26/51	25↓	20/19	1↑	11/0	11↑
39	u-blox	0/16	16↓	–	–	–	–
40	Verizon	0/37	37↓	–	–	–	–
41	Western Dig.	0/1	1↓	–	–	–	–
42	ZyXEL	449/1768	1319↓	103/67	36↑	69/20	49↑
	Total	8387/23035	14648↓	4650/2920	1730↑	1903/460	1443↑

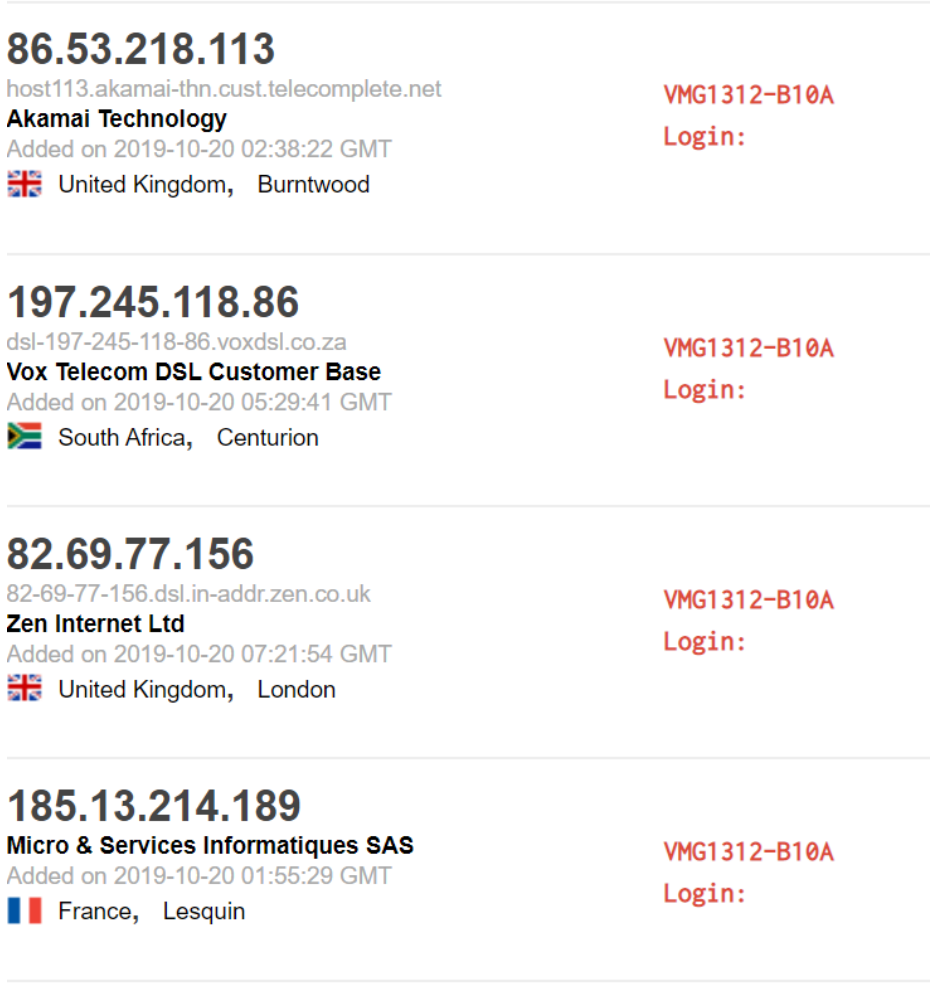
Evaluation 1: Responding services

- Significantly more services respond on their listening ports
- Telnet, HTTP, dhcp and UPnP are the most common services
- Forcing network configuration is key (failed dhcp, missing nvram values etc.)

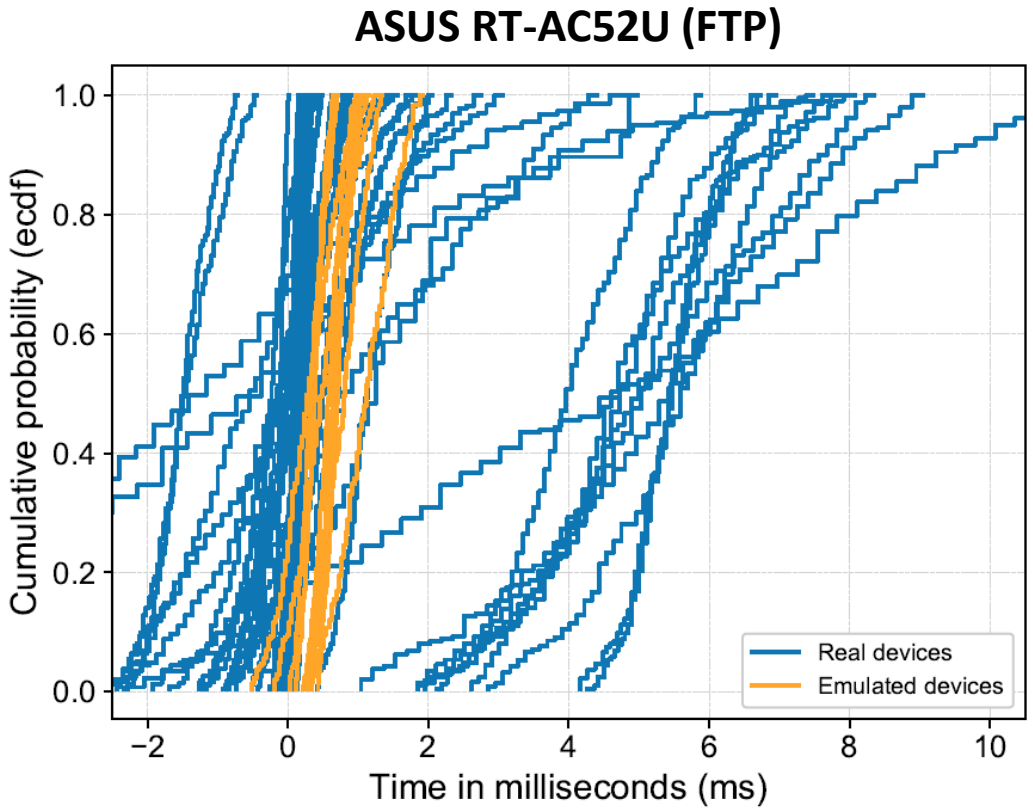
Prot.	Port/Service	Honware	Firmadyne	Δ
TCP	23/telnet	879	149	730↑
TCP	80/http	676	293	383↑
UDP	67/dhcp	316	160	156↑
UDP	1900/UPnP	239	128	111↑
UDP	53/various	239	174	65↑
TCP	3333/dec-notes	222	102	120↑
TCP	5555/freeciv	203	57	146↑
TCP	5431/UPnP	177	48	129↑
UDP	137/netbios	154	82	72↑
TCP	53/domain	139	73	66↑
TCP	443/https	107	105	2↑
UDP	5353/mdns	102	34	68↑
UDP	69/tftp	104	26	78↑
TCP	1900/UPnP	56	60	4↓
TCP	49152/UPnP	53	62	9↓

Evaluation 2: Timing attack

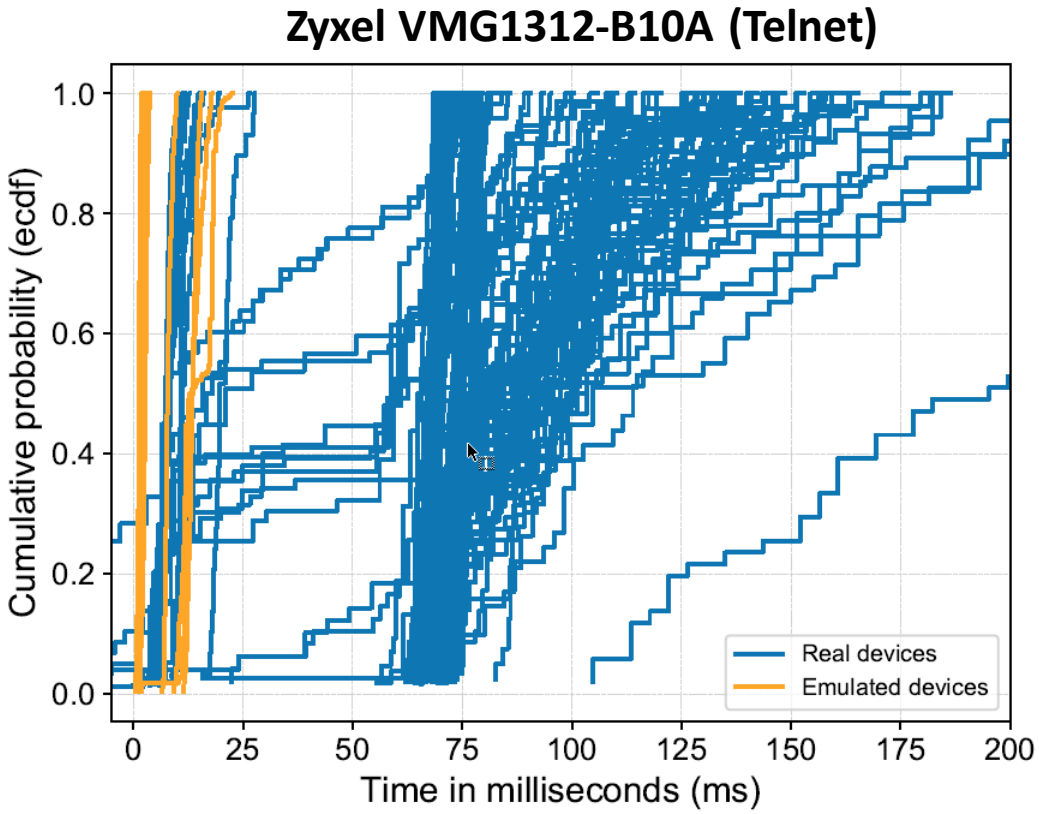
- Attackers can use timing differences to detect honeypots
- Using Shodan, we looked for three self-identifying devices (“banner”)
- We set up a total of 30 honeypots, ten for each device, on two cloud providers
- We measure the time the applications take to respond to our requests
- RTT is calculated and is subsequently used to adjust the timing information



Evaluation 2: Timing attack (FTP and Telnet)

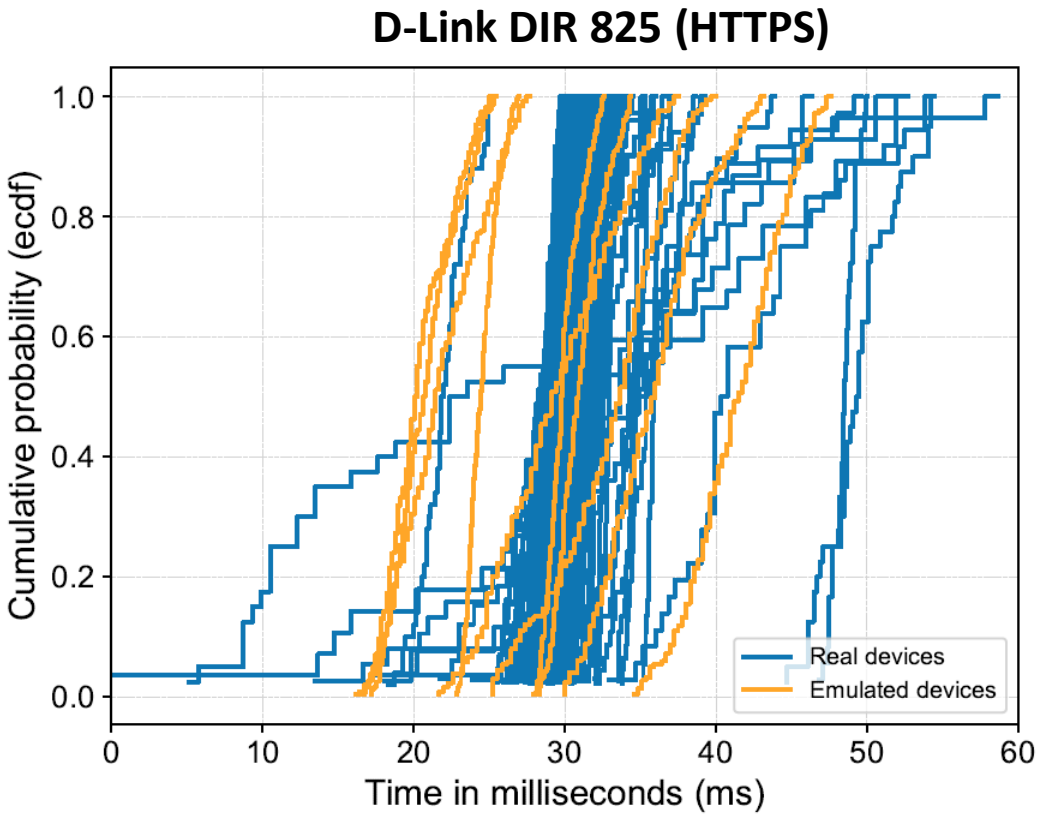


Time between resource request (carriage return) and login message

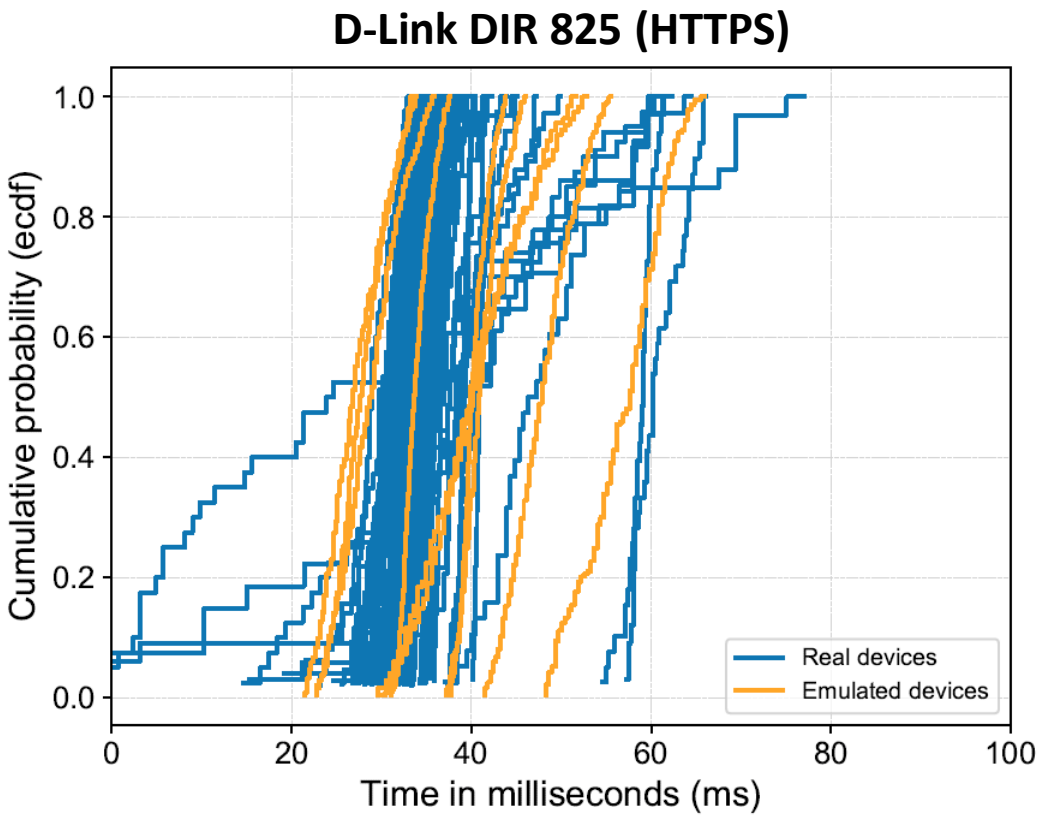


Time to Login message

Evaluation 2: Timing attack (HTTPS)



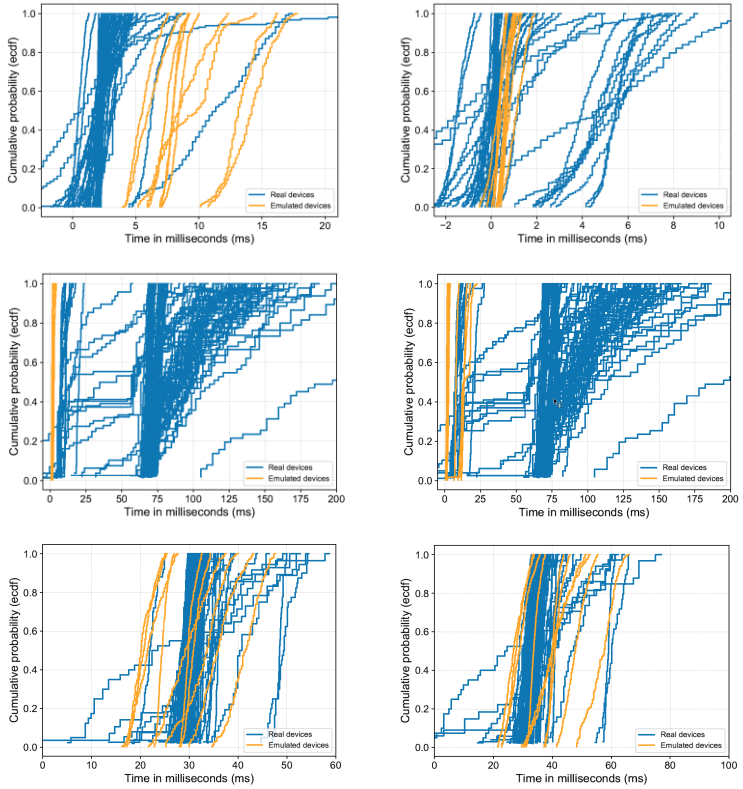
Time to complete the TLS handshake



Time between ClientHello and resource received (web page)

Evaluation 2: Timing attack conclusion

- Emulation does not generally slow down applications
  - Low-cost cloud instances > CPE/IoT devices
- Where emulation is faster, it would be possible to artificially slow responses
- Internet inherently introduces jitter, network delays and artefacts
  - Increases time and effort to mount such attacks



Attackers need to perform a significant amount of measurements to identify the discrepancies and fingerprint the honeypot

Case Study 1 - DNS hijacking attack

Whilst emulating a router from ipTIME, we observed a DNS hijacking attack

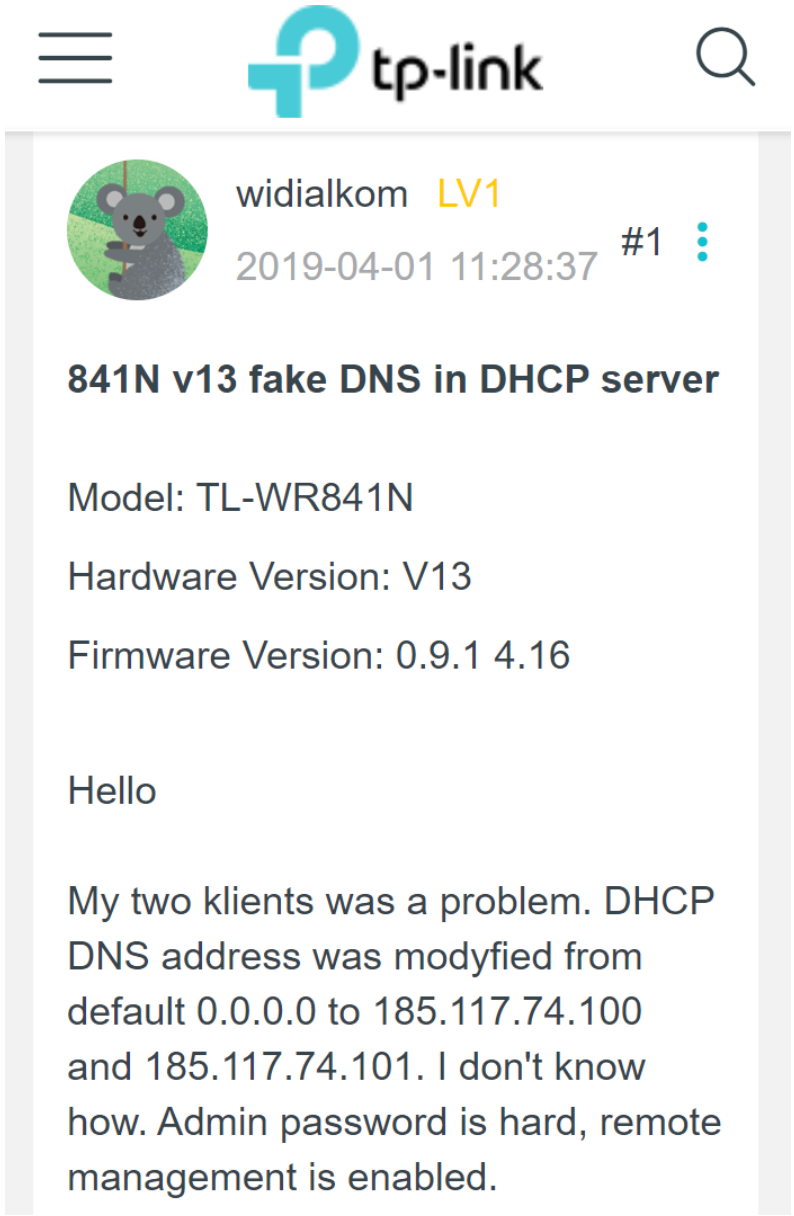
GET /cgi-bin/timepro.cgi?tmenu=netconf&smenu=wansetup&act=save&wan=wan1&ifname=eth1&sel=dynamic&wan_type=dynamic&allow_private=on&dns_dynamic_chk=on&userid=&passwd=&mtu.pppoe.eth1=1454&lcp_flag=1&lcp_echo_interval=30&lcp_echo_failure=10&mtu.static.eth1=1500&fdns_dynamic1=185&fdns_dynamic2=117&fdns_dynamic3=74&fdns_dynamic4=100&sdns_dynamic1=185&sdns_dynamic2=117&sdns_dynamic3=74&sdns_dynamic4=101 HTTP/1.1

/sbin/iptables -t nat -A PREROUTING -i br0 -d 192.168.0.1 -p udp --dport 53 -j DNAT --to-destination 185.117.74.100

yahoo.com

118.30.28.10
AS41718: China Great Firewall Network Limited Company

>40 IPs with the same certificate



widialkom LV1

2019-04-01 11:28:37 #1

841N v13 fake DNS in DHCP server

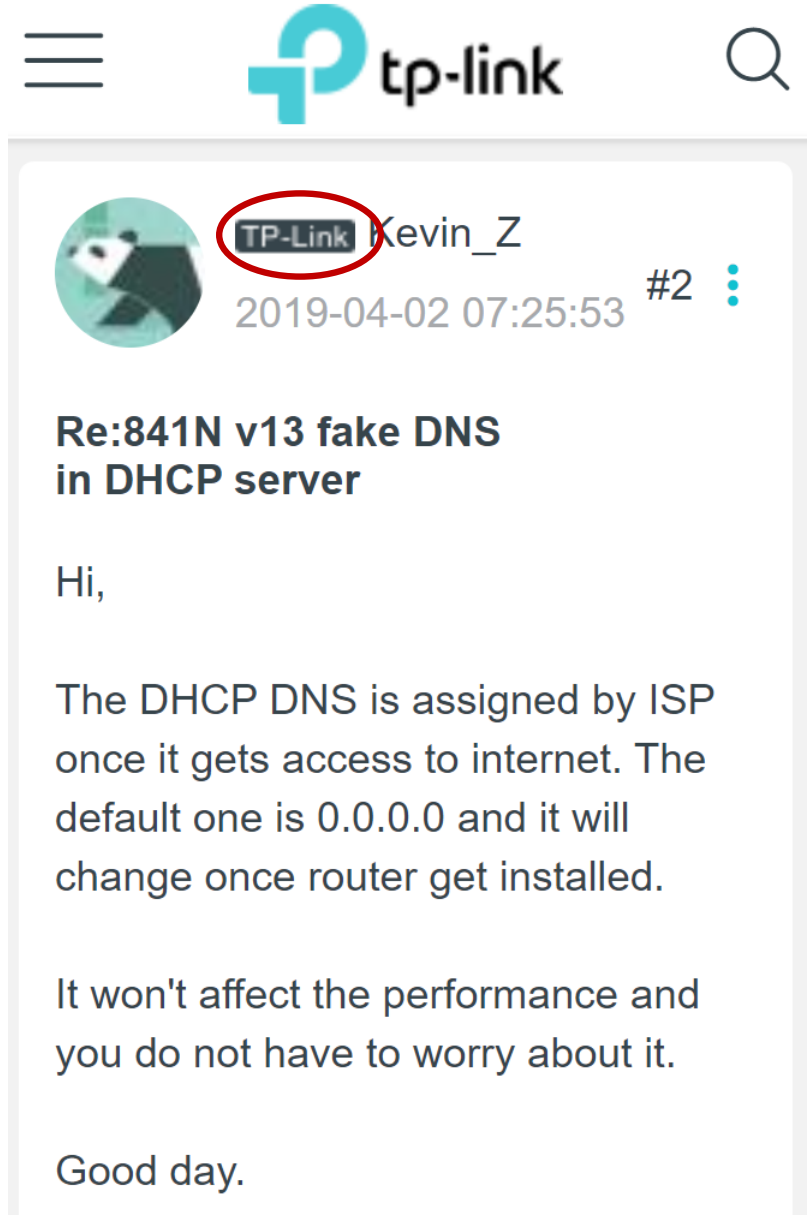
Model: TL-WR841N

Hardware Version: V13

Firmware Version: 0.9.1 4.16

Hello

My two klients was a problem. DHCP DNS address was modyfied from default 0.0.0.0 to 185.117.74.100 and 185.117.74.101. I don't know how. Admin password is hard, remote management is enabled.



TP-Link Kevin_Z

2019-04-02 07:25:53 #2

Re:841N v13 fake DNS in DHCP server

Hi,

The DHCP DNS is assigned by ISP once it gets access to internet. The default one is 0.0.0.0 and it will change once router get installed.

It won't affect the performance and you do not have to worry about it.

Good day.

Case Study 2: ThinkPhP Malware

- Emulating an ADSL modem router from TP-Link
- Non-validated input allows attackers to run arbitrary code
- >50k devices affected
- We make malware available to the defender community considerably faster than traditional honeypots

#Seen	Filename	Country	First seen		Detection ratio
			Honware	Virustotal	Virustotal
52	Tsunami.x86	DE	2019-23-02	unknown	5/67
35	cayo4	DE	2019-28-02	2019-21-03	10/68
34	Tsunami.x86	RO	2019-19-02	unknown	5/67
8	X86_64	CA	2019-28-02	unknown	0/66
6	shiina	US	2019-28-02	unknown	7/67
5	Tsunami.x86	US	2019-27-02	unknown	0/66
5	Tsunami.x86	US	2019-24-02	unknown	2/67
5	lessie.x86	NL	2019-26-03	2019-23-02	2/66
4	Tsunami.x86	ZA	2019-26-03	2019-01-03	13/71
4	Tsunami.x86	US	2019-18-02	unknown	4/67
3	Tsunami.x86	DE	2019-23-02	unknown	0/66
3	Tsunami.x86	US	2019-21-02	unknown	2/66
2	cayo4	NL	2019-22-02	unknown	0/66
2	x86	US	2019-19-02	unknown	0/66
2	Tsunami.x86	US	2019-27-02	unknown	1/66

Conclusion

Framework to deploy honeypots for CPE/IoT devices

- We use the real services/applications which are shipped with the device
- Avoids misconfigurations, missing features/commands

Better than existing emulation strategies in all areas

- Extraction, network reachability, listening services

Capable of detecting vulnerabilities at scale

- Four cases which show that devices can be rapidly emulated
- Rebalancing the economics of attackers by cutting the attackers' ability to exploit vulnerabilities for considerable time

Q & A

Alexander Vetterl

alexander.vetterl@cl.cam.ac.uk