Revealing and Analysing Modem Malware

Pavel Celeda
Institute of Computer Science
Masaryk University
Botanicka 68a, 602 00 Brno
Czech Republic
celeda@ics.muni.cz

Radek Krejci
CESNET, z.s.p.o.
Zikova 4,160 00 Praha 6
Czech Republic
rkrejci@cesnet.cz

Vojtech Krmicek
Faculty of Informatics
Masaryk University
Botanicka 68a, 602 00 Brno
Czech Republic
vojtec@mail.muni.cz

***Abstract*—Malware targeting broadband devices like ADSL modems, routers and wireless access points is very frequent in recent days. In this paper, we provide a formal description of modem malware life cycle. Furthermore, we propose a set of techniques to perform detailed analysis of infected modem and we provide the binary samples of modem malware at our web repository. Description of the modem malware evolution is also included. Based on our experiences with analysing and monitoring modem malware, we report on long-term statistics of modem malware activities in campus network including a discovery of new botnet. We propose NetFlow based detection method to reveal the modem malware spreading.**

***Keywords*-modem; malware; botnet; Linux; MIPSel; security; monitoring; network; NetFlow; ISP.**

I. Introduction

End-users can protect their computers by many security tools and local networks can be protected by firewalls, intrusion detection and prevention systems. However, there is still a wide class of broadband devices, as Asymmetric Digital Subscriber Line (ADSL) modems, with a lack of proper network protection. Cui et al. [1] found over 540,000 publicly accessible broadband devices, including routers, wireless access points, modems or VoIP appliances, configured with factory default root passwords. Broadband devices contain a lightweight version of regular operating system, e.g., a BusyBox [2]. Thus, such devices are vulnerable to common malware and network attacks.

Widely deployed and often misconfigured embedded devices represent serious threat for end-users and even public and private sector. GNUCITIZEN.org reports on security issues of various embedded devices. They demonstrated [3] that there is some kind of security flaw in almost every home electronic device.

The paper is organised as follows. After the Introduction, Section II presents an overview of evolution of malware exploiting broadband devices and a formalisation of modem malware life cycle. Section III proposes a set of techniques and methods to perform a detailed analysis of the modem device infected by malware. Section IV presents statistics from long-term monitoring of modem malware. Furthermore, description of malware spreading detection method based on such network traffic statistics is included. Conclusion summarise possible usage of techniques described in this paper.

II. Modem Malware

PSYB0T was the first widespread malware directly targeting modems. It was discovered in 2009 by Australian security researcher Terry Baume [4]. PSYB0T profited from the fact that several revisions of the modem firmware was shipped with insecure web configuration interface available from the Internet. It means that no username or password was required to access the configuration interface. Remote access via SSH and TELNET was enabled with `admin` password. These flaws were fixed in later firmware revisions. PSYB0T came into awareness when it was labelled as an originator of the Distributed Denial of Service (DDoS) attack against DroneBL site [5] in March 2009.

PSYB0T was a proof-of-concept botnet exploiting typical vulnerabilities and misconfigurations of the modems. It presented simple and efficient techniques how to exploit flaws of modem firmware to break into the modems. The botnet was shot down by the botmaster in late March 2009. In that time the size of the botnet was estimated about 80–100 thousands of infected modems.

Another example of the modem malware is dhpot. It appeared shortly after PSYB0T in early 2009 in time when PSYB0T was shot down. The worm executable contains a string “`The Distributed Honeypot Project`“. There is not much information about its activities [6]. dhpot spreads via TELNET service. We acquired a dhpot sample from an infected modem in March 2011. At that time the worm infrastructure was down. Some worm artifacts were randomly spreading through Internet using last distribution server. Our dhpot analysis is based on binary file downloaded from the Trivial File Transfer Protocol (TFTP) server (`2rstt.mooo.com`).

Worm does not conduct any attacks. It is used to get information about infected modem, send it to the server and spread itself to other devices. dhpot detects a version of installed operating system, modem uptime, MAC address of the modem and its IP address. Acquired information together with login and password, used to break into the modem, are sent to the server that collects worm statistical information.

dhpot is the only observed modem malware which does not use Internet Relay Chat (IRC) and downloads files through FTP or TFTP protocols.

In December 2009 we have discovered previously unknown malware targeting modems. The Chuck Norris botnet [7] got its nickname from a comment in its source code `[R]anger Killato : in nome di Chuck Norris !`. The botnet was disclosed in February 2010 and the activity of the botnet's C&C centres was suspended on 22 February 2010. The estimated size of the botnet was over 30,000 devices in that time. Ongoing presence of the Chuck Norris botnet was documented by Marco van Berkum [8] in November 2010. Description of this improved version can be found in [9].

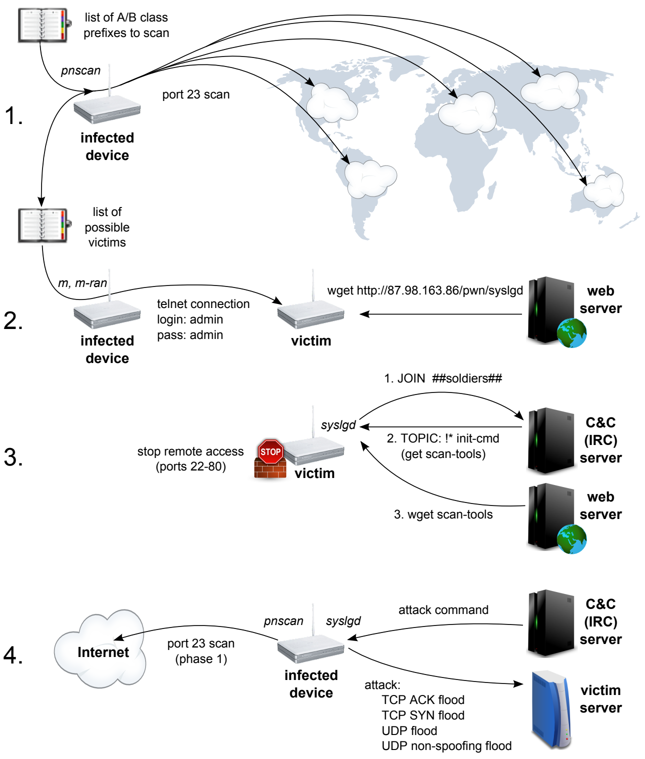

Fig. 1. The Chuck Norris botnet life cycle: *(1)* scanning for vulnerable devices in selected networks, *(2)* infection of a vulnerable device, *(3)* bot initialisation, *(4)* further scanning for vulnerable devices from a newly infected device and waiting for attack commands.

The life cycle of the Chuck Norris botnet can be divided into the four parts as shown in Figure 1. There can be seen some common characteristics typical for the most of modem malware:

Simplicity: In comparison to for example fast flux or advanced anti-debugging techniques used by PC malware, modem malware use simple techniques as IRC communication or Ultimate Packer for eXecutables (UPX) packaging [10].

Ease of Creation: Modem malware is often based on slightly modified Linux PC malware and cross-compiled for MIPSel hardware. There are publicly available malware source codes (Hydra [11], Kaiten [12]) that can be easily modified and used by any botmaster.

Command and Control: Botmasters use exclusively IRC protocol to control their botnets from dedicated C&C servers.

Ease of Removing: Modem malware so far resists in a Random Access Memory (RAM) used by modems as temporary extension storage. RAM is erased on each reboot of the device and rewritten by the firmware image from a persistent Non-Volatile Random Access Memory (NVRAM). Therefore a power cycling disinfects the modem although it does not prevent any future infection until a security configuration of the modem is changed.

Functionality Division: Main tasks of the bots are usually performed by separated binaries. The first tool performs scanning of selected networks for other vulnerable modems and controls spreading via TELNET/SSH. The second tool (bot) communicates with the C&C centre and performs several harmful functions like various types of flood attacks or interprets instructions from the C&C centre as shell commands.

Figure 1, describing Chuck Norris botnet life cycle, follows general scheme of modem malware life cycle described in Algorithm 1. As first operation after intrusion into the device, bot tries to disable any connection to the configuration interfaces. Then, the botnet binaries are downloaded and up-to-date bot is launched. Finally, bot performs orders from C&C centre while other vulnerable devices are searched for and infected.

Algorithm 1 Modem Botnet Life Cycle

```
$PORT$ := 23 {TELNET port}
$NETS$ := predefined list of network prefixes
$CREDENTIALS$ := login/password combinations
$MODEMS$ := initially empty set of vulnerable modems

$BOT$:
daemonize() {go to background}
$IRCConnection$ := IRCConnect(C&C server, IRC channel)
while $COMMAND$ := receiveCommand($IRCConnection$) do
    if $COMMAND$ == Quit then
        break;
    end if
    performCommand($COMMAND$)
end while

$BOT-LIFECYCLE$:
{Disable connections to a configuration interface(s)}
system ("iptables -I INPUT 1 -p tcp –dport $PORT$ -j DROP")
{Download bot executable file}
wget http://C&C server/bot
{Start ./bot in background}
BOT
{Scan for vulnerable modems}
for all $NET$ in $NETS$ do
    $IPS$ := IP addresses based on $NET$ prefix
    for all $IP$ in $IPS$ do
        if testConnection (IP on PORT) == OK then
            $MODEMS$ := IP {add another vulnerable modem}
        end if
    end for
end for
{Try to break into vulnerable modems}
for all $MODEM$ in $MODEMS$ do
    for all $CREDENTIAL$ in $CREDENTIALS$ do
        if login($MODEM$, $PORT$, $CREDENTIAL$) == OK then
            execute $BOT-LIFECYCLE$ at $MODEM$
        end if
    end for
end for
```

III. Infected Modem Analysis

One approach to study malware is to perform analysis of the infected modem. Through this, we are able to observe botnet activities and it is possible to detect new C&C server(s) and distribution site(s). This Section shows analysis which uses modem built-in commands and can be performed by any user.

A. Remote Access to Modem Administration

Devices with enabled remote access via TELNET, SSH, or HTTP(S) are potentially vulnerable. To determine modem public IP address, check from your local network a website which provides you the IP information[1]. To check remote access, use your actual IP address and type commands listed in example bellow.

```
# telnet -l admin <<IP>>
# ssh root@<<IP>>
# curl http://<<IP>>
# curl https://<<IP>>
```

Alternatively, you can access your modem from local network using default IP address, usually 192.168.1.1. Typically, the firewall policy for local network is less strict than for the Internet interface. In case you have established the connection, try default credentials[2] for your device to log in. If you cannot connect, try to power cycle your device to "disinfect" it before you check the access again. The malware often blocks connections to TCP ports 22–80 to hide own activities. See Section III-C how to prevent blocking remote access.

B. Modem Operating System

BusyBox provides a fairly complete environment for any small or embedded system. It is a multi-call binary that combines many common Unix utilities into a single executable. Nowadays, the BusyBox is used in many Linux-based modems. Some of the devices provide access to a shell as shown in example below.

```
# busybox
BusyBox v0.61.pre (2006.12.15-07:55+0000) multi-call
binary

Usage: busybox [function] [arguments]...
   or: [function] [arguments]...

Currently defined functions:
   [, ash, busybox, cat, chgrp, chmod, chown, cp,
   date, dd, df, echo, false, free, grep, hostname,
   id, ifconfig, init, insmod, kill, ln, login, ls,
   lsmod, mkdir, modprobe, mount, mv, passwd, ping,
   ps, pwd, reboot, rm, rmmod, route, sh, sleep,
   sync, tar, test, tftp, touch, true, tty, umount,
   wget, whoami, yes
```

Once the bot gains access to a modem, it will try to download malware via `wget` or other download programs like `ftpget` or `tftp`. Other used commands are `sh`, `chmod`, `ln`, `mkdir`, and `rm`. Generally, the modem file system [13] is read-only. To store files, the RAM-based file system mounted at `/var` must be used. Everything in `/var` will be lost on device reboot.

```
# ls -alF /var/* | grep '*'
-rwxrwxrwx    1 0    0   220326 Sep 29 15:35 hidr*
-rwxr-xr-x    1 0    0    39228 Sep 28 22:00 mzn*
-rwxr-xr-x    1 0    0    53301 Sep 29 21:58 scanzz*
-rwxrwxrwx    1 0    0    41531 Sep 29 15:11 suka*
-rwxr-xr-x    1 0    0    45580 Sep 28 11:22 .z.sh*
```

To hide malware presence, some bots add dot character in front of the name of the file or directory, e.g., `.scan`, or they use `...` as a directory name. Other common technique is to remove stored binary files, once they are executed. Process information pseudo-file system still provides information about executed commands. `/proc/*/exe` shows process number of removed file. `ps` command provides further information about the running process, including command path.

```
# ls -la /proc/*/exe | grep var
ls: /proc/7821/exe: No such file or directory
/proc/2593/exe -> /var/tmp/suka
ls: ls:/proc/2617/exe: No such file or directory
/proc/2621/exe -> /var/tmp/mzn

# ps
 PID  Uid VmSize Stat Command
2621 root    636 S    /var/tmp/mzn <- bot client!!
7821 root    644 S    -bash        <- bot client!!
```

C. Modem Firewall

The modem may be infected by several botnets simultaneously if the TELNET port remains open. Typically, the first malware operation after gaining access is to set IP table rule which will block remote connections to TCP ports 22–80.

```
# iptables -I INPUT 1 -p tcp --dport 22:80 \
  -s ! 127.0.0.1 -j DROP
```

This will effectively block all attempts to remotely remove or analyse malware inside infected device. To avoid such situation, a small shell daemon can be used to check firewall rules and reopen closed TELNET port.

```
# ( while true; do \
   iptables -nL INPUT --line-numbers | \
   grep '1   ' | grep ACCEPT | grep 'dpt:23' || \
   iptables -I INPUT 1 -p tcp --dport 23 -j ACCEPT;\
   sleep 10; \
  done ) &
```

Connection tracking provides information about hosts trying to connect to the modem administration interface. It can be network discovery scans or TELNET brute force attacks. Resulting list of IP addresses can be used to traceback attack sources.

```
# while true; do \
   cat /proc/net/ip_conntrack | grep 'dport=23 ' | \
   grep CLOSE ; sleep 2; echo -n .; \
  done
```

D. Memory Analysis

Modems do not store any persistent logs which can provide used shell commands. Physical memory analysis is the only way how to determine information as malware commands, distribution sites, IRC messages, etc. Memory stores partial information even after the reboot. This way, malware artifacts can be extracted even from the disinfected devices.

The `dd` program can be used to capture the contents of physical memory using `/dev/mem` file. In the recent Linux kernels, `/dev/mem` is no longer available by default. On the other hand, all modems that we analysed provide `/dev/mem`. Following command will create image of physical memory and store it on a remote server.

[1] http://www.whatismyip.org

[2] http://www.routerpasswords.com

```
# dd if=/dev/mem bs=1M | \
  ssh user@192.168.1.20 dd of=/tmp/mem-copy.bin
```

The /proc/iomem shows the map of the system memory for each physical device. This is important if the system RAM does not start at address 0. dd is not able to capture memory without correct offset (starting address).

```
# cat /proc/iomem
00000000-13ffffff : reserved
14000000-1401ffff : System RAM
14020000-14ffffff : System RAM
  14020000-141cb7bf : Kernel code
  141dc300-141f7fff : Kernel data
a8610000-a86107ff : eth0
```

We search for a “wget” pattern to extract botnet distribution sites from the current content of memory.

```
# dd if=/dev/mem skip=335544320 bs=1 | grep ’wget’
```

To extract botnet IRC communication from the actual contents of memory, we search for “PRIVMSG” or “NOTICE” strings used as IRC user commands.

```
# dd if=/dev/mem skip=335544320 bs=1 | grep PRIVMSG
# dd if=/dev/mem skip=335544320 bs=1 | grep NOTICE
```

E. Malware Samples

Malware samples can be downloaded from botnet distribution sites. What to do if they are down or the botmasters blocked access from malware researchers’ network? In such case, we must extract the malware samples from infected device. The modem file upload possibilities are quite limited. We use hexdump to convert a binary file to hexadecimal dump which we copy from terminal output and convert them back to original file.

```
# cat /var/.scan/m | hexdump -v -e ’"\\\x" 1/1 "%02x"’
# echo -e -n "‘cat terminal_output_file‘" > m
```

We maintain an archive[3] of modem malware samples for other malware researchers. The archive includes all modem malware binaries, we describe in this paper.

IV. Malware Network Activities

To protect our campus network, we have developed and deployed network monitoring system based on NetFlow information (see Figure 2). The network-based approach allows us to see all activities against and from our network. We use NetFlow [14] as an input for the security analyses and the anomaly detection systems we work on.

Algorithm 2 describes the method to detect malware spreading. The use of the TELNET protocol should be discontinued for security related shortcomings and replaced by SSH protocol. Any TELNET activity, especially on the public Internet, is suspicious. TELNET is the main protocol used to break into insecure modems. Therefore, we focus on TELNET connections (TCP port 23). Other conditions to designate a certain IP address as an attacker are:

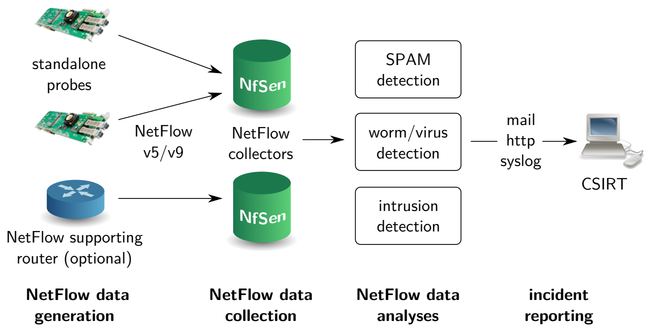

Fig. 2. Flow-based network security monitoring system. Generated NetFlow data are stored into NetFlow collectors and used for network forensics by university security team.

1) A particular flow contains TCP SYN flag only – attacker only checks for opened TELNET port and do not send more than one packet to the host.
2) TCP SYN packet size is 60 bytes – the flow length (corresponding to the single SYN packet in the flow) of 60 bytes is a fingerprint of Linux TCP/IP stack used in modems.

Algorithm 2 Malware Spreading Detection

```
$FLOWS$ := observed flow information

{Select suspicious IP addresses}
for all $FLOW$ in $FLOWS$ do
    if $FLOW.TCP.dstport$ == 23 then
        if SYN in $FLOW.TCP.flags$ then
            if not ACK in $FLOW.TCP.flags$ then
                if $FLOW.Bytes$ == 60 then
                    $IPS.add(FLOW.IP.srcaddr)$
                end if
            end if
        end if
    end if
end for
```

Here we provide a long term statistics acquired both from the campus network and from the single host outside the campus network. These statistics are based on the Algorithm 2. They were collected in the period from October 2009 till July 2011 and contain NetFlow data representing TELNET scans. This behaviour is typical for botnets targeting modems. Example of such malware is the Chuck Norris botnet, which was active in the watched period.

Figures 3 and 4 show the number of total scans aggregated by days in different networks. We can see significantly larger number of scans targeting campus network. This is caused by the large address space covered by campus network, containing B class subnet instead of one particular IP address in the case of single host statistics.

The number of scans performed by a single attacker corresponds also with this fact. In the case of campus network, each attacker performed 194 scans in average. It roughly corresponds to the horizontal scan exploring C class subnet of the campus network. In the case of host outside campus network, there was only one scan per attacker in average.

[3] http://dior.ics.muni.cz/∼celeda/malware-samples/

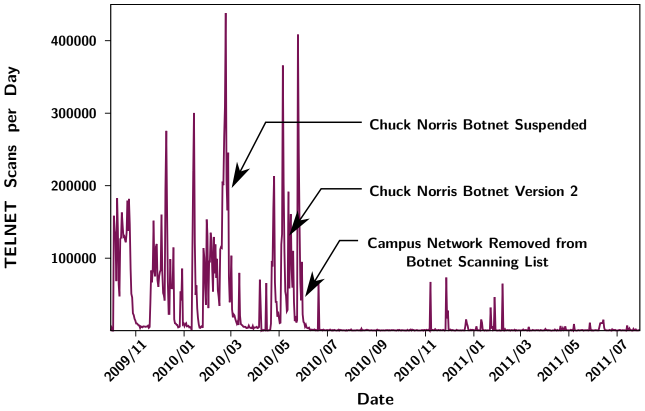

Fig. 3. Number of TELNET scans – campus network 147.251.0.0/16.

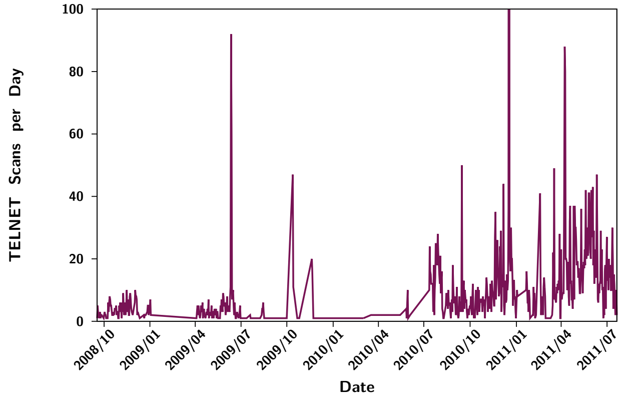

Fig. 4. Number of TELNET scans – single host outside campus network.

Figure 3 illustrates how the bots stopped their activity after botnet disclosure. We released the botnet information to the security community including TF-CSIRT[4] in February 2010. Afterwards, the botmasters removed our campus network from their scanning lists. We can see significant decrease in number of scans in the second half of the year 2010.

Later we have observed blocked access to the botnet distribution and C&C servers from networks outside the set of networks scanned by bots. These way botnet owners try to reduce possibility of studying botnet by security experts.

V. Conclusion

As we have demonstrated in the Section IV, there is a high potential to detect suspicious network activity from the ISPs point of view. ISPs can start with instructions and best-practices for managing security situation in the network described in some Internet Engineering Task Force (IETF) documents [15], [16]. According to these documents, ISPs are supposed to use some combination of malware detection methods, like network/application traffic flow analysis or Intrusion Detection Systems (IDS), to protect their customers.

In this paper, we have described several techniques to get as much as possible information about modem malware. With our experiences from analysing modem malware, we have shown a general characteristics and behaviour pattern of the current generation of the modem malware. Furthermore, we also provide a number of malware samples for other researchers.

Acknowledgements

This material is based upon work supported by the Czech Ministry of Defence under Contract No. SMO02008PR980-OVMASUN200801 and also supported by the “CESNET Large Infrastructure” project LM2010005 funded by the Ministry of Education, Youth and Sports of the Czech Republic.

References

[1] A. Cui and S. Stolfo, “A quantitative analysis of the insecurity of embedded network devices: Results of a wide-area scan,” in *Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC’10)*, New York, NY, USA, 2010, pp. 97–106.

[2] E. Andersen. (2011) Busybox – the swiss army knife of embedded linux. [Online]. Available: http://www.busybox.net

[3] GNUCITIZEN. (2008, Feb.) Router hacking challenge. [Online]. Available: http://www.gnucitizen.org/blog/router-hacking-challenge/

[4] T. Baume. (2009) Psyb0t information page. [Online]. Available: http://baume.id.au/psyb0t

[5] Nenolod. (2011) Network bluepill – stealth router-based botnet has been ddosing dronebl for the last couple of weeks. [Online]. Available: http://www.dronebl.org/blog/8

[6] (2009, Mar.) Random attempts to hit port 23 (telnet). [Online]. Available: http://forums.whirlpool.net.au/forum-replies.cfm?t=1164229

[7] P. Čeleda, R. Krejčí, J. Vykopal, and M. Drašar, “Embedded malware – an analysis of the chuck norris botnet,” in *Proceedings of the 2010 European Conference on Computer Network Defense*, Los Alamitos, CA, USA, 2010, pp. 3–10.

[8] M. Berkum. (2010, Nov.) Full disclosure mailing list archives – ssh scans, i caught one. [Online]. Available: http://seclists.org/fulldisclosure/2010/Nov/228

[9] P. Čeleda and R. Krejčí, “An analysis of the chuck norris botnet 2,” Institute of Computer Science, Masaryk University, Brno, Tech. Rep., Mar. 2011. [Online]. Available: http://www.muni.cz/ics/research/cyber/files/cnb-2.pdf

[10] M. Oberhumer, L. Molnár, and J. Reiser. (2011) UPX - the Ultimate Packer for eXecutables. http://upx.sourceforge.net/.

[11] (2008) Hydra – Mass DDoS Tool. [Online]. Available: http://data.nicenamecrew.com/papers/malwareforrouters/resources/dlink-automatic/hydra-2008.1.zip

[12] (2001, Dec.) kaiten.c irc ddos bot. [Online]. Available: http://packetstormsecurity.org/irc/kaiten.c

[13] P. Lougher. (2011) SQUASHFS - A squashed read-only filesystem for Linux. http://squashfs.sourceforge.net/.

[14] B. Claise, “Cisco systems netflow services export version 9,” RFC 3954, Oct. 2004. [Online]. Available: http://tools.ietf.org/html/rfc3954

[15] J. Livingood, N. Moody, and M. O’Reirdan, “Recommendations for the remediation of bots in ISP networks,” Internet-Draft, Sep. 2011. [Online]. Available: http://tools.ietf.org/html/draft-oreirdan-mody-bot-remediation

[16] C. Chung, A. Kasyanov, J. Livingood, and N. Moody, “Comcast’s web notification system design,” RFC 6108, Feb. 2011. [Online]. Available: http://tools.ietf.org/html/rfc6108

[4] http://www.terena.org/activities/tf-csirt/