Vulnerabilities in Router Firmware

Jayendra Khandare (jkhandar@indiana.edu) Govind Mishra (shremish@indiana.edu)

Abstract: In this paper, we have studied in detail about the security and vulnerabilities of routers and their firmware. The vulnerabilities studied in the paper includes UPnP (Universal Plug and Play) protocol, Default Settings (Username and Password), OpenSSL (a library that implements the Server Layer and Transport Layer Protocols) and Firmware upgrade and updates. Our goal in the paper is to cater and provide a solution to the vulnerabilities we have studied. We have created scripts which when executed under conditions could resolve the problems. We implemented and tested our scripts on Qemu for various router architectures.

1. INTRODUCTION

a) *Security of Commercial Routers:* In the age of the Internet, the devices that connect us to the Internet network i.e. Small Office and Home Routers (SOHO) or commercial routers needs to be secured as wireless routers are an ideal target for cybercriminals. If an attacker gains control of the router, they can control, monitor, redirect, block or otherwise tamper with a wide range of online activities which can be classified as man-in-the-middle-attack and causes the Invasion of Privacy to the user. In a recent study, it is revealed 79% of Amazon's top 25 best-selling Office/Home Office (SOHO) wireless router models have security vulnerabilities. Plus, 34% of the 50 top-selling models have publicly documented exploits that make it relatively simple for attackers to target vulnerable systems.

This paper will discuss the security protocols such as UPnP, HNAP, OpenSSL library, Firmware update, upgrade and management, Changing the Domain Name System (DNS), Working Around the Administrative Interface. These vulnerabilities and errors are the core reasons for many errors and most of the large-scale incidents are caused due to them. We have mentioned the incidents in the paper in section 2 of the paper.

b) *Working and Operation of Routers:* A router is a networking device that forwards data packets among different computer network which performs the traffic directing functions on the Internet. A data packet is typically forwarded from one router to another router through the networks that constitute an internetwork until it reaches its destination node. When a data packet comes in on one of the lines, the router reads the network address information in the packet to determine the ultimate destination. Then, using information in its routing table or routing policy, it directs the packet to the next network on its journey.

Routers can be connected to the devices in either wired or wireless fashion. For a wired connection, the device must be compatible with Ethernet port or should have an Ethernet adapter card whereas, for a wireless connection, the device signal should be compatible with the router. Typically, 802.11a, 802.11b, 802.11g and 802.11are the signal protocol used, where each signal protocol differs in the wireless data speed and the context. Wireless routers have WAP (Wireless Access Point) which enable multiple devices to connect to the router. The data in the WLAN (Wireless Area Network) is secured using the WEP (Wired Equivalent Privacy). WEP is a security protocol, specified in the IEEE Wi-Fi standard 802.11b which is designed to provide the security of level of Wired connection by encrypting the data which is sent over the WLAN.

WEP protocol uses RC4 encryption engine for encrypting data, which is also used in HTTPs. RC4s create a 24 bit of string of Initialization Vector (IVs) for all the packets in the WLAN. The IVs are the keys which separate all packets in the WLAN. The IVs were very small and hence the keys were repetitive and short for the cryptographic purposes, along with that WEP lacked in providing any cryptogenic protection to the keys. Thus, WEP is replaced with WPA and WPA2.

c) Firmware Management within Routers: Firmware is a small piece of software that makes the hardware work and do what the manufacturer intends it to does. It consists of programs written by software developers which are compatible and works on the hardware. Firmware updates are often provided because of a security breach or when a new feature is offered by the ISP (Internet Service Provider). When such an update is available the user visits the server and updates the existing firmware.

d) Findings and Contributions:

We were able to secure the issues related to UPnP, OpenSSL and BusyBox

Regarding the UPnP protocol, there are modules which are responsible for turning on and off the UPnP in a device, thus we decided to load the required modules for 420 seconds [7 minutes] and then unload them after that, and this way, the device is visible for the required duration.

For OpenSSL, we created several scripts to restrict 'sshd' daemon on various levels. Some scripts allow local network access, while others block remote access.

For BusyBox protocol, after studying several firmware architectures we concluded that we need only a few numbers of modules to be necessary for Busybox and only restricting to these modules would reduce the architecture complexity and functionality of the router.

2. ROUTER SECURITY BREACH INCIDENTS

a) McAfee Labs [1] discovered that a banking malware Pinkslipbot has used infected machines as control servers since April 2016, even after its capability to steal personal and financial data from the infected machine has been removed by a security product. Pinkslipbot controls a large botnet of more than 500,000 infected machines and steals over a half-million records every day. The machines are of home users, whose computers are behind a network address translation router. Network Address Translation [3] is the method of mapping one IP address space into another by modifying network address information. To do so, Pinkslipbot uses universal plug and play (UPnP) to open ports, allowing incoming connections from anyone on the Internet to communicate with the infected machine. As far as we know, Pinkslipbot is the first malware to use infected machines as HTTPS-based control servers. Later, McAfee provided a patch [2] which then blocked any intrusion from the router to the machine. This worked perfectly only for their own specific problem.

Microsoft Windows Operating System was targeted by a computer worm known as Conficker or Kido [4]. The Conficker worm infected 9 million computer including government, business and home computers in over 190 countries. It was an executable-based Universal Plug and Play malware which reaches out and opens an incoming port through the firewall and performs port forwarding. Microsoft provided a patch which limited the port forwarding but the solution was limited to this exact malware.

b) More than 1,000 Linksys home and Small Office routers were detected with a malicious new worm called "The Moon" [5]. This code which derives its name from a 2009 science-fiction movie "moon" spreads from routers to routers. The worm was attacking the routers whose remote-administration features also known as HNAP were open. Home Network Administration Protocol (HNAP), a management tool found on some consumer-grade routers that transmit sensitive information about the router over the Web at http://[router IP address]/HNAP1/, and grants full control to remote users who provide administrative usernames and passwords. Though, this also exposes the user to vulnerabilities where an attacker could easily exploit this administrative feature and enter the system. The worm appears to be able to connect to a command-and-control server, from where an attacker can manipulate the compromised systems.

The worm works by remotely calling a router's Home Network Administration Protocol or HNAP. It then uses a known vulnerability in the router's Common Gateway Interface (CGI) script to gain administrative control. The Moon also resets some routers to use Google's DNS (domain name system) servers at Internet Protocol addresses 8.8.8.8 and 8.8.4.4. Once the worm infects a router, it scans the Internet for other Linksys routers to infect. Though, a simple reboot or choosing to switch off the HNAP removes the worm.

c) OpenSSL, an open-source code library that implemented the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. For SSL to work, the computer needs to communicate to the server via sending 'heartbeats' that keep informing the server that client (computer) is online (alive). The Heartbleed bug [6], is a bug in OpenSSL's library which causes the memory leak from the server to the client and from client to server, and thus allowed the attackers to read portions of the affected server's memory, potentially revealing user's data, that the server did not intend to reveal. It opens doors for the cyber criminals to extract sensitive data directly from the server's memory without leaving any traces.

d) The Russian government-sponsored actors are using the compromised devices to perform man-in-the-middle attacks that extract passwords, intellectual property, and other sensitive information and to lay the groundwork for potential intrusions. The alert identified multiple stages in the hacker campaign which includes

Reconnaissance, in which the hackers identify Internet-exposed network ports used for telnet, simple network management protocol, Cisco Smart Install, and similar services. Then, delivery of traffic to vulnerable devices that cause them to send configuration files that contain cryptographically hashed passwords and other sensitive data and then they exploit the user based on the credentials they have obtained. Finally, command and control, where the attackers masquerade as legitimate users and established a connection through a previously installed backdoor.

e) WikiLeaks secret documents from vault7 showed that home routers from 10 manufacturers, including Linksys, DLink, and Belkin, can be turned into covert listening posts that allow the Central Intelligence Agency to monitor and manipulate incoming and outgoing traffic and infect connected devices.

CherryBlossom infects routers by identifying their make and model and injecting malicious firmware into them. This kind of hack, when successful, is nearly impossible to detect because it infects the hardware itself and is not something anti-virus software is capable of checking. These can remotely affect the routers even when they have strong administrative passwords.

Once installed, CherryBlossom turns the device into a "Fly Trap" that beacons a CIA-controlled server known as a "Cherry Tree." The beacon includes device status and security information that the CherryTree logs to a database. In response, the CherryTree sends the infected device a "Mission" consisting of specific tasks tailored to the target. CIA operators can use a "Cherry Web" browser-based user interface to view Flytrap status and security information, plan new missions, view mission-related data, and perform system administration tasks.

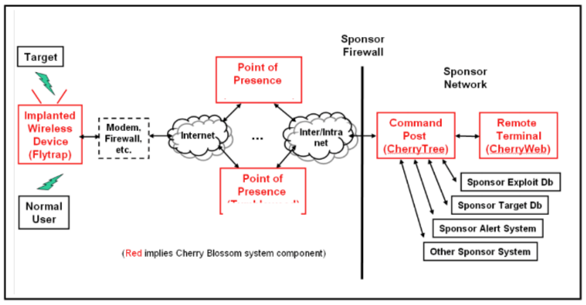
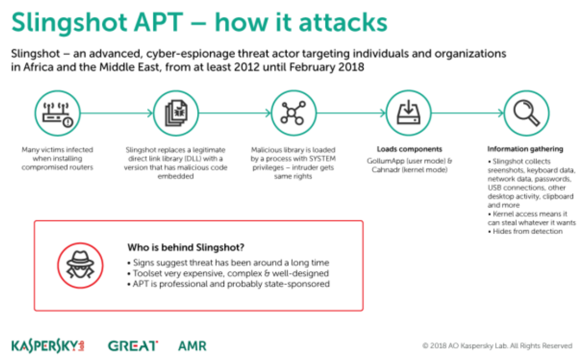


Figure 1: Cherry Blossom Architecture (U)

The Missions leveraged the OpenSSL where Mission can target connected users based on IPs, e-mail addresses, MAC addresses, chat user names, and VoIP numbers. Mission tasks can include copying all or only some of the traffic; copying e-mail addresses, chat user names, and VoIP numbers; invoking a feature known as "Windex," which redirects a user's browser that attempts to perform a drive-by malware attack; establishing a virtual private network connection that gives access to the local area network; and the proxying of all network connections.

f) Security researchers at Kaspersky Lab have discovered what's likely to be another state-sponsored malware strain, and this one is more advanced than most. Nicknamed Slingshot, the code spies on PCs through a multi-layer attack that targets MikroTik routers. It first replaces a library file with a malicious version that downloads other malicious components and then launches a clever two-pronged attack on the computers themselves. One, Canhadr runs low-level kernel code that effectively gives the intruder free rein, including deep access to storage and memory; the other, Gollum App, focuses on the user level and includes code to coordinate efforts, manage the file system and keep the malware alive.


Slingshot APT – how it attacks
Slingshot – an advanced, cyber-espionage threat actor targeting individuals and organizations in Africa and the Middle East, from at least 2012 until February 2018
Many victims infected when installing compromised routers
Slingshot replaces a legitimate direct link library (DLL) with a version that has malicious code embedded
Malicious library is loaded by a process with SYSTEM privileges – intruder gets same rights
Loads components
GollumApp (user mode) & Cahnadr (kernel mode)
Information gathering
• Slingshot collects sreenshots, keyboard data, network data, passwords, USB connections, other desktop activity, clipboard and more
• Kernel access means it can steal whatever it wants
• Hides from detection
Who is behind Slingshot?
• Signs suggest threat has been around a long time
• Toolset very expensive, complex & well-designed
• APT is professional and probably state-sponsored
KASPERSKY lab
GREAT
AMR
© 2018 AO Kaspersky Lab. All Rights Reserved

Slingshot also stores its malware files in an encrypted virtual file system, encrypts every text string in its modules, calls services directly (to avoid tripping security software checks) and even shuts components down when forensic tools are active. To run its code in kernel mode in the most recent versions of operating systems that have Driver Signature Enforcement, Slingshot loads signed vulnerable drivers and runs its own code through their vulnerabilities. If there's a common method of detecting malware or identifying its behavior, Slingshot likely has a defense against it. It's no wonder that the code has been active since at least 2012. Currently, the microTek routers could be made safe with an update but Slingshot may have a wider range than microTek routers.

3. RELATED PREVIOUS STUDIES

The authors in this paper investigated the web interface of home routers i.e. considered the HNAP protocol of the routers. The Authors used the XSS and UI redressing attacks in their studies on 10 different manufacturers (TP-Link, Netgear, Huawei, D-Link, Linksys, LogiLink, Belkin, Buffalo, Fritz!Box, and Asus). The home routers configure themselves over a web interface, where the integrated webserver contains different options and input fields which are used by the attackers. Thus, after implementing these attacks, the attacker gets the access of DNS server or default IP gateway and in turn obtaining full control over all the data traffic and he can even reboot the device to make it unavailable for a specific time or may even redirect the traffic.

Attackers in the paper were mainly the web attackers who do not have any physical access to the routers. Authors used two attack types which are XSS (Cross Site Scripting) and UI Redressing. The paper has used two variants of XSS such as Reflected XSS and Stored XSS.

In Reflective XSS, the attacker sends the HTTP GET or POST request using a form which includes the default user credentials and malicious JavaScript code that runs on the web interface. The user thus if makes any change on the browser asks for the routers IP address from the web interface and in the snapshot the attacker also has the credentials. Then the attacker can change either the administrative controls or the routing path. The other type is stored XSS where the attacker has preloaded the malicious JS code in a section of the website.

The other kind of attack used by the authors was UI Redressing, which is a technique where the attacker aims to modify the look and behavior of the web page. Thus, it forces the user to make changes to get the default screen and thus he does the action intended by the attacker. Attackers use clickjacking method, which is loading the website inside an iFrame on the attacker's page and make the iFrame element transparent. Thus, the attackers use social engineered techniques to lure the user and spill out important information, which is then leveraged by the attacker.

The countermeasures provided by the paper are:

Randomization of the Default Login Data, that is an attacker has a 55% chance of a correct admin info in case the default configuration is not changed, thus a randomly generated an initial login data. SSL/TLS are the signed certifications, which ensures that attacker cannot inject its own certificates

Input Validation, attacks such as reflective XSS and stored XSS use the input sinks which are not validated properly, there are whitelist and blacklist which disallow the usage of any special characters by encoding them so that they cannot be used to execute malicious code.

This paper discusses in detail about several framing attacks called frame busting which refers to the code intended to prevent the web page from loading in original frames. Also, it discusses that those existing solutions at hand leak private user information. These framing attacks steal the routers WPA secret key and accurately locate them on a map. The automated attack was carried out on eight different brands of routers using a default password: Belkin, Netgear, DLink, Linksys, Buffalo Zyxtel, SMC, TrendNet. Hence, an attacker can create an accurate world map of WPA keys needed to access private WIFI networks.

At first, Authors fingerprinted the browser and scanned the local area network to find the router. To locate the router, port scanner looks for all probable IP addresses especially in the range of 192.168.*.1 to 192.168.*.254, as routers use a limited set of IP addresses.

When the routers IP is found, the attacker performs tests to identify the router and logs into them.
Router Identification involves trying a default password based on the router model and browsers feedback to get the username and password. Now, attackers use this and this to send forms to the user to extract the WPA key and MAC address. These details can be used by the attackers to

The authors leveraged some of the shortcomings of the browser and network, which if improved could lead to more secured routers. The UPnP protocol if disabled could protect the router but still, it sends out the response that it is closed, changing it to stealth mode will secure the router as it will not send any information to any solicited devices. The other shortcoming leveraged by the authors was that web browsers such as Firefox, Chrome and IE8 where the browser lets the attacker know if the web interface of router is using HTTP, Locating the router and constant feedback whether a password is used are correct supports the attacker's intentions and hence makes the routers vulnerable.

4. DESIGN AND IMPLEMENTATION

1.UPnP -
By default, UPnP feature is always on for ease of access. This feature helps a device identify and communicate with other devices in the network range which is a very serious issue. To avoid such a security risk, this feature is omitted from business grade devices, but it is still there in commercial devices like SOHO routers. To handle this problem, we could restrict the access to this feature and provide a web-interface which will require admin access to make the connection. This doesn't solve the actual problem but restricts it.
Basically, the modules are available which are loaded whenever the router system boots and mostly these modules are loaded by default. Hence, to avoid this problem, we have provided some scripts which will load the required module for a specific time (420 seconds[7Mins]) when admin asks to and after this period, these modules are unloaded. Without loading these modules, no external device can use UPnP feature.

2.OpenSSL –
For most of the times when manufacturer releases a new update or a patch, this issue somehow manages to remain unnoticed. In 2017, researchers found 'the Heartbleed bug' which when exploited allows the adversary to get access to secret keys used for certificates, usernames and passwords without leaving a trace. Fortunately, OpenSSL has fixed this bug and released a new version of the software. Also, CherryTree project exploits running instances of OpenSSL which are there on any commercial router to setup the flytrap and Cherry Blossom mechanism. This mechanism allows them to monitor data traffic, and credentials. The best option to address this issue is to restrict the OpenSSL server, so that only the local network can access it. We can also provide the user with an option in the webpage of router settings to start or stop it based on the requirement. This won't affect the devices connected to the router. Hence, if some device wants to use it, they can utilize this functionality. But only admin will be able to turn it on or off for the router.

3.Restricting BusyBox –
This utility is touted as the swiss army knife of embedded Linux. It provides a lot of Unix utilities which are used by various processes for example: mkdir, sendmail, chroot, ftpput, etc. The number of utilities provided through busybox is continuously increasing as the time goes by. This constant evolution possesses a threat in disguise because some of them are not needed for functioning of the devices but are provided nonetheless. Basically, nobody takes the trouble to cut out the unnecessary utilities. Some of the firmware architectures we studied had kept in mind this mistake and hence, added only the required modules while compiling their busybox. Hence, their

builds were light. On the other hand, most of the firmware architectures don't think about this factor carefully and create heavy builds. This practice needs to be avoided.

4. Weak passwords –
On all the commercial routers, the admin user account is set up by default. The values are hardcoded into the system. To tackle this problem, there is not much we can do. The ideal case would be to use dual authentication or begin with a strong password. According to some researches, dual authentication is possible but will be expensive. The new system created would be required to satisfy the needs. The manufacturers will not be willing to go through the whole deal. Anyhow, most of the users don't visit the webpage of server configuration that much, hence this will be kind of a waste of money. Alternatively, the other method would be less expensive and onetime thing, hence end-users would try this. The second method we are proposing is to put a strong password defined by user at the first use. So ideally, when the router is setup at user's place, before any-communication takes place, the user would be required to setup a strong password without which the router won't connect to the internet. This initial setup will be mandatory and one-time thing. Before this setup, only local communication(intra-network) will be active.

5. User being unaware about firmware updates –
The firmware updates or patches provided by the manufacturer are not very regular. Mostly the updates are provided only if some researchers have found some issues in the device. Here, we cannot blame the manufacturer for the delay, because to provide an update first you need to find the faults in the existing system. Going further, even if the firmware updates are available the end users are unaware of it and hence, the update is not performed. To tackle this case, in our implementation we have added a checkpoint where the router checks the specified location for an update after a fixed period and sets the flag which indicates that a new update is available. If the ROM has the device has enough space for downloading and storing the update it does so and in the next reboot session, it performs the update. SOHO routers are used in offices and homes where it is possible to have a computer with wired connection. We could set up a messaging service and notify user about such a case.

6. HNAP –
This concept and its implementation can be considered as a feature for the service providers and manufacturers and a vulnerability to users. On the manufacturer side, using this mechanism provides accurate topology discovery, custom task extensibility and other advantages to keep track and provide customer support. On the consumer side, it helps in troubleshooting if some configuration has gone awry. Also, not just manufacturers are able to keep track using this, any adversary can exploit to this functionality and cause some serious issues. So instead of keeping the HNAP server running all the time, it would be beneficial if we could somehow make this user dependent. So that, whenever the user needs some help from the manufacturer / service provider regarding troubleshooting, he could activate this from the web-interface.

7. Incoherency in integrated server architecture –
The integrated server architecture of the firmware we studied is very dubious to understand and hence incorporating that in our implementation was a hassle. Figure.2 shows how complicated the structure is and differences among various config files.

5. EVALUATION PLAN

Out of the 7 methods mentioned in the previous section, we were able to successfully emulate 3 methods using the Qemu emulator. We couldn't make a head-way because to test some methods like HNAP, etc. because for that multiple devices should be connected to router which was not possible for us with Qemu.

The setup for insuring a strong admin password is feasible, but due to lack of time we were not able to finish the required scripts for that.

Keeping track of the newly released updates and patches proved to be a tough one. There is no fix location where we can get this information and keeping track of the empty space available and utilizing it is a tough thing to emulate. We started with 1.0 version of XXX router firmware and by running our partial script, we downloaded the update. The procedure after that is complex and hence time consuming. That's why we were not able to get it done with.

Restricting busybox is not that much of the trouble. The source code of busybox is public and compiling it is very easy. The only important task in this procedure is to sort through various packages available and decide which to include. Some of these packages are important to general functioning of the router and hence, should be added no matter what. If the manufacturers come up with a list to do this based on their specific model, it would be very helpful. Now they don't have something like this. One the router architecture we studied had a busybox executable of size 4 MB which is just too much. On the other hand, based on our elementary studies we created and used a busybox executable of 300 kB which provided basic functionalities.

Table 1. (at the bottom) shows the variations in size of busyboxes used by various architecture based on our study.

Though the incoherency of the server architecture should be addressed someday, we do not see it happening anytime soon. It will require a lot of work from manufacturers. As this work will require a convention to follow, that convention needs to be defined first. So anyhow, it is a long way from realizing anytime soon.

Table 2. (at the bottom) shows the incoherent structures of various architectures.

Although Qemu can be considered as a perfect emulator, our understanding of it is limited. Hence, we created the scripts and executed them manually for UPnP and OpenSSL and verified the differences in the running processes on the router to analyze the results.

The script we came up with takes care of the OpenSSL problem and restricts the access [except for localhost]. Although the execution was manual, it does the job and can be incorporate in any server architecture. It can also be modified to block all the communication as well. There were some problems with the execution if 'sshd' is already running. Hence, sometimes we need to stop the daemon first and them run the script.

Figure 1. (at the bottom) shows the successful execution of the script which takes care of OpenSSL.

There are some processes on router which take care of UPnP feature. The modules are loaded at the time of initiation. The code for that is usually located in "/etc/init.d/rcS" or "/etc/rc.d/rcS" depending on the architecture.
To include our scripts in a fully functional router, all we need to include them in the "init" scripts mentioned above and then provide some interface on webpages to toggle the switches which execute the reverse scripts. Although adding the scripts is very easy and can be done by changing only a few lines, adding interface to provide the toggle switch is kind of difficult, because of the varieties available in the server and becomes difficult to keep track. In our implementation, we were able to provide such a switch in a server which used PHP.

6. CONCLUSION

Although we were successful in securing issues related to busybox, OpenSSL, and UPnP, the study we conducted was not able to be conclusive about all the proposed ideas due to lack of time and understanding of the embedded architecture. We came to know about the incoherence in server architectures which is a problem if you are not familiar with them. It makes automation difficult and more time consuming. If someone could standardize the server architecture used by router firmware, it would be a very good initiative.

Secondly, it is apparent from the various types and functionalities provided by various busyboxes that manufacturers don't spend much of the time in devising / compiling their versions of it. They don't have to add all the available Unix utilities, instead they should just add what is required and nothing more.

Thirdly, our initial plan of using dual authentication mechanism is feasible but impractical since it requires a whole new system to be devised and maintained by the manufacturers. As this system is going to be expensive to maintain, none of the manufacturers will opt for it. Our alternative to cater to this problem is feasible and not expensive. It only requires creating a couple of new modules and incorporate them in the existing framework.

We studied various versions of various architectures. The updates usually are provided if some major

threats are found by researchers and published in the media. While the fixes are provided for only high-risk threats, all moderate to low risk issues just remain unattended.
For example: HNAP bug and OpenSSL (Heartbleed bug) issues came to limelight around the same time, manufacturers and OpenSSL took it seriously and provided a fix within months. On the other hand, some manufacturers provided fixes for some of their devices 2 years later. Most of their devices are still vulnerable. This issue needs to be addressed more firmly.

7. LIMITATIONS

1. Emulation on Qemu is not fully functional.
2. Due to variations in integrated server architecture, using webpages to execute scripts is impossible.
3. The scripts do not detect location of various required libraries and executables now, we will provide thorough script within a few days at GitHub repo.
4. All scripts are run manually on the server.
5. Lack of any automation what-so-ever.
6. Not tested with real hardware.

8. FUTURE SCOPE

- Enforced password modifications.
  - *Initially right after setup*
  - *Then before each access request*
- The architecture is overwhelming to understand as of now but given enough time, the process can be automated.
- Scripts can be added to integrated server architecture and can be directly used through default web - interface.
- Someone could formulate a methodology to bring order in this chaos of incoherency.
- All the required scripts and libraries are available at GitHub repo.
- We will be able to automate the whole procedure in a couple of weeks. Hopefully, by the end of this month we would finish automation.

9. REFERENCES

[1] https://www.mcafee.com/hk/downloads/free-tools/pinkslipbot.aspx

[2] https://securingtomorrow.mcafee.com/mcafee-labs/mcafee-discovers-pinkslipbot-exploiting-infected-machines-as-control-servers-releases-free-tool-to-detect-disable-trojan/

[3] https://en.wikipedia.org/wiki/Network_address_translation

[4] https://en.wikipedia.org/wiki/Conficker

[5] https://www.tomsguide.com/us/malware-spreading-worm-linksys,news-18316.html

[6] http://heartbleed.com/

LIST OF FIGURES

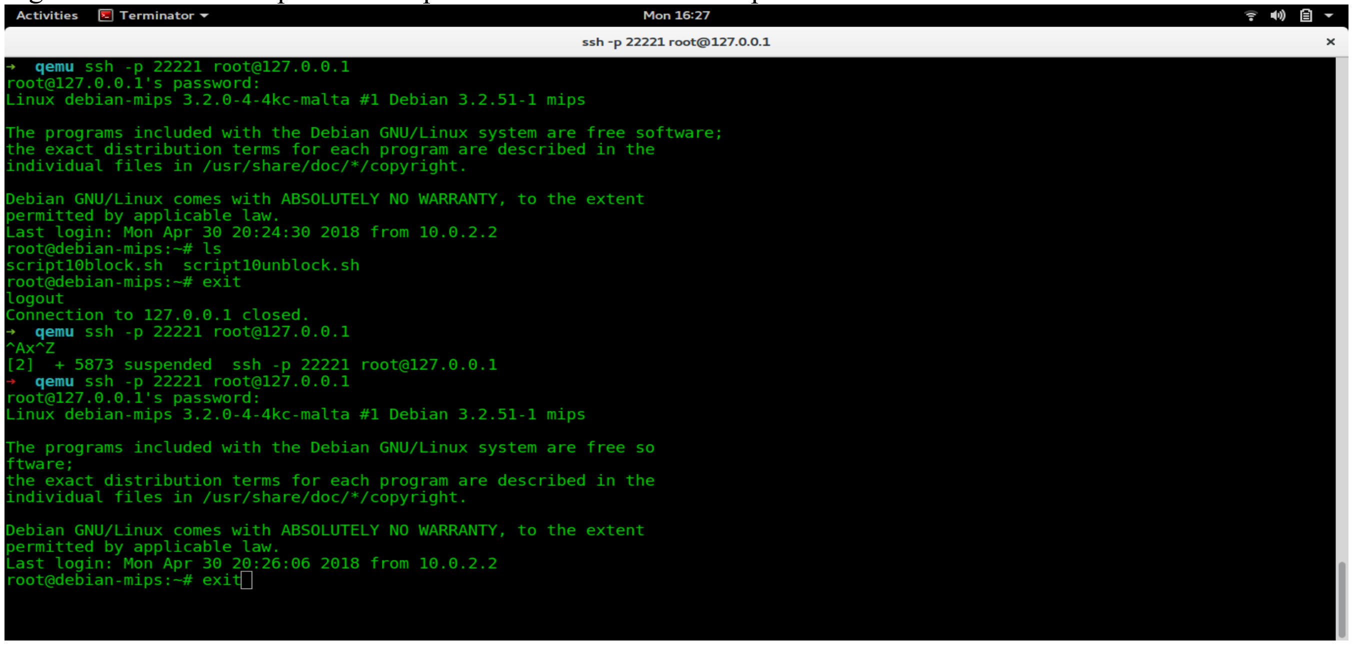
Table 1. Variations in busybox executables

Manufacturer	Model	Busybox size (in kb)
DLink	DIR 615B	353
DLink	DSR 150B	Unknown
DLink	DSR 500N	1126
DLink	DSR 1000N	1126
DLink	DIR 866	431 + 93
DLink	DIR 866 DDWRT	292
NetGear	JNR 1010	350
NetGear	MR314	Unknown
NetGear	R 6120	451
NetGear	R 6700	370
NetGear	WGT 624v4	398
NetGear	XR 500	525
NetGear	WNAP 320	1433
OpenWRT(default)		227

Table 2. Incoherent structures in various architectures

Maker	Model	Server	Server Tech	FS path	UPnP
Dlink	DIR 615B	ASP	/sbin/httpd	/www/	/sbin/miniupnpd
Dlink	DSR 150B	HTML	/sslvpn/bin/httpd	/sslvpn/www/	/bin/upnpd
Dlink	DSR 500N	HTML	/sslvpn/bin/httpd	/sslvpn/www/	/bin/upnpd
Dlink	DSR 1000N	HTML	/sslvpn/bin/httpd	/var/www/	/bin/upnpd
Dlink	DIR 866	HTML	/usr/bin/httpd	/etc/www compressed	/usr/sbin/upnp
Dlink	DIR 866 DD	HTML	/usr/bin/httpd	/etc/www compressed	/usr/sbin/upnp
NetGear	JNR 1010	PHP	/usr/sbin/mini-httpd	/usr/www/	/sbin/miniupnpd
NetGear	MR314	unknown	Unknown	unknown	Unknown
NetGear	R 6120	HTML	/usr/sbin/mini-httpd	/www.eng/	/usr/sbin/miniupnpd
NetGear	R 6700	HTML	/sbin/htmlget	/www/	/usr/sbin/upnpd
NetGear	WGT 624v4	HTML	/usr/sbin/httpd	/www/	/usr/sbin/upnpd
NetGear	XR 500	HTML	/usr/sbin/uhttpd	/www/	/usr/sbin/miniupnpd
NetGear	WNAP 320	PHP	/sbin/lighttpd	/home/www/	NOT FOUND
OpenWRT		NONE	/usr/bin/lighttpd	/var/www/	/sbin/miniupnpd

Figure 1. execution snapshot of script which blocks/unblocks OpenSSL



```
→  qemu ssh -p 22221 root@127.0.0.1
root@127.0.0.1's password:
Linux debian-mips 3.2.0-4-4kc-malta #1 Debian 3.2.51-1 mips

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Apr 30 20:24:30 2018 from 10.0.2.2
root@debian-mips:~# ls
script10block.sh  script10unblock.sh
root@debian-mips:~# exit
logout
Connection to 127.0.0.1 closed.
→  qemu ssh -p 22221 root@127.0.0.1
^Ax^Z
[2]  + 5873 suspended  ssh -p 22221 root@127.0.0.1
→  qemu ssh -p 22221 root@127.0.0.1
root@127.0.0.1's password:
Linux debian-mips 3.2.0-4-4kc-malta #1 Debian 3.2.51-1 mips

The programs included with the Debian GNU/Linux system are free so
ftware;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Apr 30 20:26:06 2018 from 10.0.2.2
root@debian-mips:~# exit
```