International Journal of Research in Advent Technology (E-ISSN: 2321-9637) Special Issue
National Conference "NCPCI-2016", 19 March 2016
Available online at www.ijrat.org

Router Reinforcement with OpenWrt and Cloud

Manasi Godse
Department of Computer Engineering
Pune Institute of Computer Technology
Savitribai Phule Pune University (Formerly University of Pune),
Pune, Maharashtra, IndiaPune, Maharashtra, Indiagmanasi13@gmail.com

Isha Renavikar
Department of Computer Engineering
Pune Institute of Computer Technology
Savitribai Phule Pune University (Formerly University of Pune)
isha.renavikar@gmail.com

Akanksha Chandak
Department of Computer Engineering
Pune Institute of Computer Technology
Savitribai Phule Pune University (Formerly University of Pune)
Pune, Maharashtra, India akankshachandak41@gmail.com

***Abstract*—It** is common knowledge that one cannot easily configure the router and modify it to suit one's needs. Today, every home is equipped with one or more wireless router which is used for connecting multiple devices and also used as interface between ISP(Internet service provider) and local devices. Remote access to routers is very difficult for naive users and it is not possible to modify the configuration for them. Also, the router configuration is closed source, meaning that it cannot be modified according to the user. Blocking websites, parental control and getting statistics is also difficult the traditional way i.e. normal users won't be able to get this information unless an expert intervenes. This paper proposes a system to reinforce the routers by overlaying the router with OpenWrt. Access to the router information and modifying the configuration, getting statistics, storage can be done with the help of Cloud infrastructure. The paper shows the general idea, architecture and the flow for the system that is implemented.

***Index Terms*—Router, OpenWrt, Network monitoring, Network Management.**

I. INTRODUCTION

OpenWrt provides a writeable filesystem that can be used to write modules, compile them and overlay on the router firmware to perform various operations on the router. There are various restrictions on the application selection and the configuration management that is provided by the vendor in the case of embedded devices. Using OpenWrt, one can easily use the packages provided by OpenWrt and customise the application according to one's requirements. As the name suggests, OpenWrt is open source i.e the code is publicly available which can be used by the developers to create their own application without the need to build a complete firmware image.

Cloud provides the resources to store and an on-demand access of various resources that support ubiquitous computing i.e. computing that is present everywhere. It allows the user to store and process the data over a network. Router usage includes statistics, devices connected to it, data consumed which can be stored on cloud. Also, the router can be simplified to connect with cloud and in this way, a user can have access to router as and when required. In this case, privacy plays a major role since third-party users can misuse the data. Hence the data and the processing should be done in such a way that it does not compromise the privacy of the user.

II. LITERATURE SURVEY

This section includes the previous work that is done with OpenWrt which includes the following:

- Gargoyle - Gargoyle is a free OpenWrt-based Linux distribution for a range of Broadcom and Atheros chipset based wireless routers, mainly the oldermodel Linksys WRT54G (including the WRT54GL and WRT54GS), Asus Routers and Netgear WNR3500L[3]. Among notable features is the ability to limit and monitor bandwidth and set bandwidth caps per specific IP address.

- WifiDog - WiFiDog is an open source embeddable captive portal solution used to build wireless hotspots. WiFiDog consists of two components: the gateway and the authentication server. It was written by the technical team of le Sans Fil and is included in the software package repository of OpenWrt. [4]

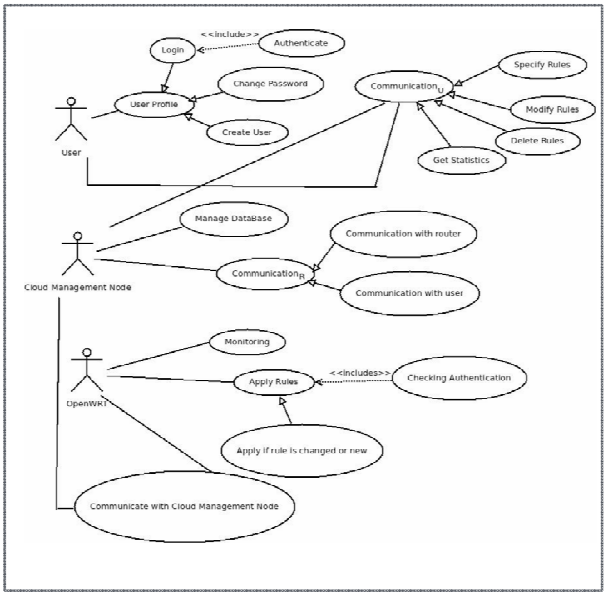
III. BASIC IDEA

The system that is to be implemented consists of the following goals and objectives:

- To provide the user with the facility of easy router configuration remotely, with advance features such as network monitoring and parental control.
- The main objective is to configure the router according to the user wherein the user can modify the policies on the router as and when required with remote access.
- The input set is of different queries that are fired by the user on the router. The queries consist of push policies on router such as blocking certain websites, providing with time bound to certain pages and to monitor the router.

Figure 1 shows the use case diagram of the system which includes the actors as user, cloud management node and OpenWrt. By looking at the diagram one can easily infer the the use cases that are to be implemented. The actors in the diagram can be described as follows:

- User: This is the actual user of the system which will configure and manage the router through a UI.
- Cloud Management Node: This is the actor which will be responsible for handling database, communicating with the router and the user.
- OpenWRT: This is the actor that is responsible for making changes on the router and implementing the user queries.



1. Use case diagram

IV. GENERAL ARCHITECTURE

The architecture consists of different modules which are discussed as follows:

- MODULE 1: Mobile client UI

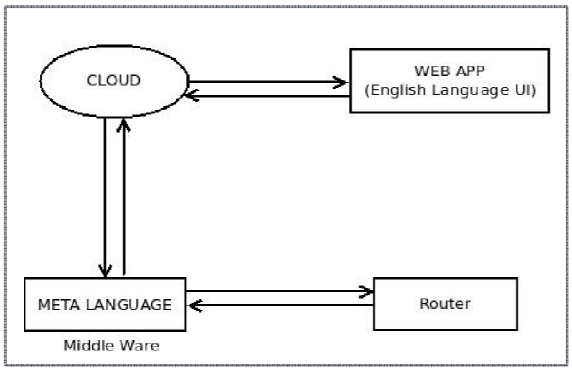
This is the user Interface of a web application which can be used by users to perform different configurations/settings of router. The configuration setting of router involves the following functions of monitoring the router and push policies. Technologies to be used: HTML5, CSS, Bootstrap

- MODULE 2: Server and Database

This module is used for the hosting the web server for the user client. Apache2 is used as the web server here along with MongoDB as database.

- It is responsible for recording user input storing it into the database and communicating the user configurations to the router. Technologies to be used:Flask

- MODULE 3: Meta language and database (on Router)This converts the user queries to the language that router understands and vice versa. The English Language UI requests for queries to Meta Language Module and accordingly reply is given back. This module works as the middleware converting the user language to a language that router understands.



2. Basic Layout

- MODULE 4: OpenWRT (Router)

OpenWRT is used for customization and modifications of router configurations. This module rests on top of the original router configuration module. It consists of the following sub-modules: Client, Checker and Monitoring.

Figure 2 shows the basic layout of the architecture.

The software interface description includes the following:

- User Interface: Web Application made using bootstrap, css and html and an Android Application.
- Server: apache2 is the server used • Backend: To be developed using flask.

V. BASIC FLOW

Authentication of the user credentials will be done which will start the communication of the user with the router through cloud. Consider an example: User A wants to block access to abc.com for 2 hours. For this, the user will first have to verify the credentials. After authentication, A will send the rule to be applied on the router. The processing will take place through cloud. The rule, on reaching the router, will be parsed and converted to the router language using the meta langauage module (that is built with OpenWrt) and then it will be applied on the router.

When the router is switched on initially, i.e. when the router boots up, its identity needs to stored on the cloud so that it can be identified. The identity will consist of the hardware address that is assigned to it since it is unique with respect to every router. For this, communication needs to be established with cloud at the time of booting up. Only after this is done, the further processing can take place.

The user will get access through authentication and then the user will be able to modify the configuration, implement certain rules. A user-friendly UI will enable the user to use the proposed system. When a rule needs to be implemented, the user will specify it and then it will be sent to the router. This will be the POST method. The POST will then be analysed by the meta language module, which consists of libcurl code. This code will extract the rule, parse it and then will be sent for authentication, to check for duplicates by calling a C function. The rules that are already applied will be stored in a policy file. Since the number of rules are limited, a flat file can be used for storage on the router. The rule will be validated and if found correct, will be applied on the router. To have evenness between the meta-language module and the actual function that will implement the rule, a structure can be created and used with a common header file. This will eliminate any ambiguity that may arise.

Following points should be satisfied by the system:

- Real-time availability of the system with suitable updating at regular intervals of time.
- All the constituting features of the software should run and update themselves independently without interfering with other features.
- The system should be able to handle the sudden burden of requestss for one router, without deviating much in response time (delay in response is assumed on over burdening the system).
- Reliable communication between the router and the user. • The router memory should be protected as any change in that may result in the malfunctioning of the application.
- The system should be secure to threats like hacking and viruses to the highest degree possible.
- Information of users such as IP addresses should be kept private so that third parties cannot gain access to this personal information in order to keep within the Data Protection Act.
- The security measures should be reliable enough and should be able to deal with at least moderate security threats. The system should use secure database and application server, which will not allow any unauthorized person to access or modify the database.

VI. IMPLEMENTATIONDETAILS

. The input taken from the admin of the router is the mac address of the router so that when the router boots up, the corresponding user will be mapped to the corresponding router. The URL of the website to be blocked is taken as the next input from the user. This URL is converted into its corresponding IP address using the nslookup command.

When the IP address of the website to be blocked is sent to the router, the following steps take place:

1. First, rule is converted into a hash function using a hash algorithm i.e. djb2 (Daniel J. Bernstein) which has short number of lines as compared to others and is fast and has reasonable distribution of hash values.

2. If the hash value is already present i.e. if the rule is already present in the rule file, the rule wont be applied else it will be applied on the router.

3. Rule is applied using iptables library in C i.e. libiptc and libip4tc. This library provides the functions which can add, read and delete rules in the iptables.

4. The above three steps (checking in the first step) is performed every 1 min by the use of pthread library. The timer thread is executed every one minute so as to check if a rule needs to be deleted or is to be added.

Steps involved in making OpenWrt on VirtualBox as an Access Point:

1. A separate NIC is needed that will act as an AP since the machine's NIC will be bridged with the OpenWrt so as to provide network connectivity to it.

2. The separate NIC used is TP-LINK TL-

WN722N USB Adapter. For the USB adapter to be recognised by the OpenWrt on the virtual box, the modules to be included during the make of the .vdi file using make menuconfig are usb2, usb2-pci, horst, ath-9k and ath-9k-htc.

3. To be able to use OpenWrt as an AP, the changes are made to the network, wireless, dhcp and firewall files with reference to the OpenWrt as Routed AP [13]

4. After completing the settings, using the 'ifup wifi' command followed by 'wifi', OpenWrt SSID is broadcasted and devices can be connected to it thereby successfully acting as an AP.

5. Using 'iwinfo interface_name assoclist', the list of connected devices (MAC address) can be displayed which can be further mapped to the IP address using the dhcp.leases file so as to implement the rule for the corresponding device.

Communication between Cloud and OpenWrt:

Since the web server is hosted on www.cloudmesh.in, to enable OpenWrt to send and receive data, the libcurl library is used in C. Using libcurl enables us to communicate with Cloud without the use of a server on OpenWrt and hence the router resources are saved since the load of not running a web server exits. The router polls the server every one minute to get the policies that need to be applied by running a thread every one minute.

VII. FUTURE WORKAND CONCLUSION

To implement the proposed system, the working of OpenWrt has to be understood. To overlay the router functionality, modules have to written with OpenWrt and then image file will be generated which will be deployed on the router. Also, the cloud functionality, communication, network usage have to be considered. The paper describes the system w.r.t. only one router whereas after deployment 'n' number of routers will be connected with a variety of devices where conflicts can occur which need to be worked on. The future scope of the system can be extended to support the Internet of Things (IoT) devices which will ease the merging of these devices into the existing infrastructure.

Router Reinforcement with OpenWrt and Cloud attempts to bridge the gap between user and the router management process. It aims to establish parental control, network monitoring, setting the boundaries of the router and get- ting the usage statistics. Moreover, the major constraint of being able to manage the router by being connected to the router is overcome by the usage of Cloud. Hence, remote access can be provided to the user to connect to the router.

REFERENCES

[1] An OpenWRT solution for future wireless homes Palazzi,
[2] Claudio E. ; Brunati, M. ; Roccetti, M.
[3] OpenWRT operating system based controllers for mobile robot and building automation system students projects realization Kciuk, M.
[4] Kissling, Kristian (17 Jul 2009). "Gargoyle: Web Interface for Router Configuration"
[5] Lenczner, Michael. "Wireless Portals with Wifidog"
[6] J. F. Kurose, K. W. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, Addison Wesley Longman, Boston, MA, USA, 2001
[7] Computer Networks (5th Edition) by Andrew S. Tanenbaum and David J. Wetherall (Oct 7, 2010)
[8] Teach your router new tricks with DD-WRT or OpenWrt
[9] Serdar Yegulalp
[10] OpenWRT website: http://openwrt.org/
[11] OpenWRT wiki, http://kb.netgear.com/app/products/model/aid/ 2598
[12] OpenWRT wiki, http://wiki.openwrt.org/oldwiki/openwrtdocs/ hardware/netgear/wgt634u
[13] Meraki White Paper: Wireless guest access at the office.
[14] OpenWRT wiki, http://wiki.openwrt.org/about/start