Smart Router

Thibault Debatty [1] Pietro Michiardi [2] Olivier Thonnard [3] Wim Mees [1]

[1]Royal Military Academy, Brussels, Belgium

[2]EURECOM, Campus SophiaTech, France

[3]Symantec Research Labs, Sophia Antipolis, France

Introduction

The number of IoTs devices present in our houses is constantly growing up. Smart-TVs, smart-lights,IP-cameras and number others are all connected to the Internet to be managed from outside our homes. But how can we, in as SOHO context, monitor these devices to be sure they only do the work they should. So how to be sure they are not botnets ?

Goal

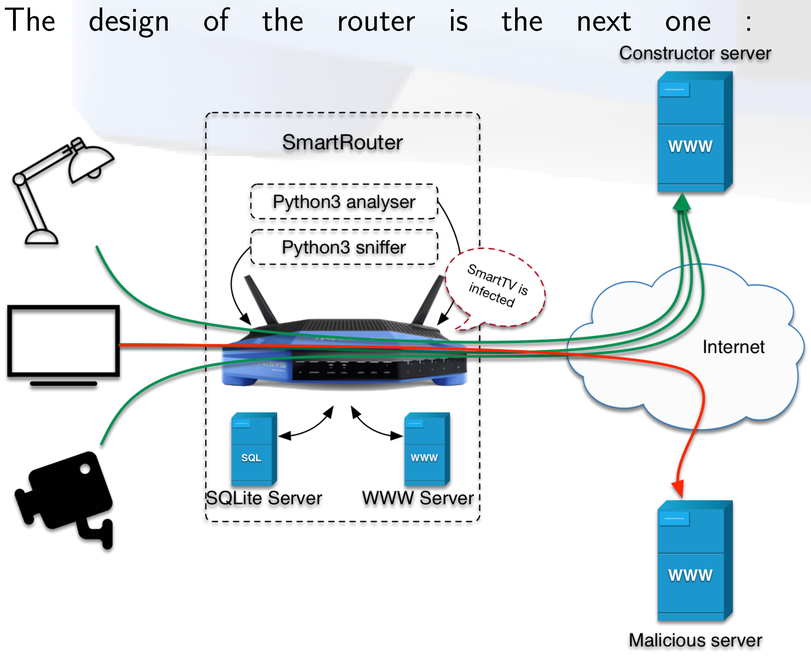
The goal of this project is to monitor IoT traffic of a SOHO (Small Office, Home Office) network. By monitoring it, this project can send alerts when an IoT has an unusual network traffic. For example, if it suddenly begin to send random request to random IP addresses around the world.

Structure

This project was initially designed from Ubuntu machine but was then adapted to work on an OpenWRT distribution which is more adapted from network embedded devices. This project was developed in Python3.

Design

The design of the router is the next one :



OpenWRT (`https://openwrt.org`)

The OpenWrt Project is a Linux operating system targeting embedded devices. So it was one of the possible choices for the Linux distribution. More specifically, OpenWRT was a compatible distribution for the router used for this project. The routers used are Linksys WRT1200ac.

Scapy

The main goal of the project is to have a router which executes 2 main tasks :

- Sniff : all SYN packets of TCP/80 && TCP/443 are sniffed and stored in an SQLite3 database. To be able to know which IP corresponds to which domain, DNS packets are also sniffed and then data are correlated to make IP correspond to a domain. If no domain corresponds to an IP, IPs are DNSreversed to try to have a domain.
- Analyze : after having sniffed packet, a process must be launched to analyze all traffic and determine if the traffic is legit or malicious.

Installation

The installation is very simple thanks to auto-deploy scripts. A Readme is present to the Github the guide the installation.

Like routers used for this project do not have a lot of storage, a USB stick is needed to expand data storage. All this procedure is also explained in the Github(`https://github.com/RUCD/smart-router`).

One of the auto-deploy script is the following :

```
sh -c "$(curl -fsSL https://raw.githubusercontent.com/
RUCD/smart-router/master/docs/setupScripts/setupSR.sh)"
```

Alerting

Initially, two different types of alerting had been designed. On via a web browser, and another one, optional, like an application or something like that.

Slack alerts

Slack alert have the role of the initial application alert. A slack application has been integrated in a workspace. So with it, the Smart Router send alert to a specific channel, thanks to slack tokens.

This slack integration allow the receive notification like initially wanted, but without having to develop a specific application for it.

Web server A web server using Laravel framework was initially designed to assure the web server role. But due to portability issues and high resource consumption, alert are simply read on a text file and displayed vie a simple web server.

Future Work

Lots of modules can be developed in future work, for example we can imagine :

- More protocol sniffed : for now, only web and DNS traffic is sniffed, but it could be interesting to have a generic sniffer which allowed to sniff everything traffic,
- Web configurations : all configuration is done in command line, so a more user-friendly web interface configuration page could be a great idea,
- Target sniffing : a good idea will be to be able to detect IoTs and filter only this traffic. Actually, all traffic is sniffed, but some can be blacklisted not to be sniffed,
- Target analysis : actually, all traffic is reanalyzed every time an analysis is run. A better analysis will be to analyze only new traffic since the last analysis,
- Lots of other integration are possible to have better interfaces and configuration modules.

Conclusion

This project has a lot of other possibilities then experimented here. It also has a lot of optimization and machine learning possible.