REPUTATION SYSTEMS IN WIFI NETWORKS

Lukáš Žilka
Bachelor Degree Programme (1), FIT BUT
E-mail: xzilka07@stud.fit.vutbr.cz

Supervised by: Marek Kumpošt
E-mail: kumpost@fit.vutbr.cz

ABSTRACT

The main purpose of this paper is to review and continue with the previous work of Ing. Petr Blahák [1] and Ing. Petr Hirš [2]. Firstly I will perform an analysis and review of the completed parts of the system (also known as Repunet) and point out pros and cons of my predecessors' solution. Elements of previous work did not cooperate, and we found putting it together as quite difficult; the matter of future extension was not covered either, therefore, we decided to rework all existing parts – with respect to extensibility - and prepare a new framework, that will remedy all drawbacks of the original design. Lastly, I will describe an automated VMWare image building process which attends to the creation of a bootable system containing a fully working version of Repunet System.

1 INTRODUCTION

The Reputation System in Wifi Networks project, (latere termed "Repunet") was founded by Ing. Petr Blahák. He devised its concept and basic principles.
Wireless networks, in general, are prone to attacks, due to the posibility that anyone in the range of a signal can try to break through the security measures to the network without being noticed. This is considered to be a huge negative factor and inspired Petr Blahák to propose a solution, based on analysis of the connected clients' behaviour – reputation. The system collects data from access points in a network and identifies an attacker from the results of the analysis of the collected data. The data contains total traffic activity transmitted over various ports and protocols, such as signal-to-noise variance, suspicious activities, port or ip scanning, etc...

2 ORIGINAL DESIGN

Repunet was implemented as a series of data mining scripts and a web visualisation of the collected data. The data mining scripts were designed to periodically read the states of the set of access points. The access points had to contain a custom Linux distribution and only *Z-Com XI-626* wifi cards. If the scripts were found to cooperate with any other unrecognized access point type, they would have to be rewritten. The read state is then stored in a database. No optimizing measures as to the database were taken, which is the source of unmaintainable

growth of the database size. This uncaring approach to the database and the rigid data mining scripts along with the database structure, result in the current problems with data mining today. The visualisation framework was built upon the data mining scripts' database structure. It offers automatic drawing of network topology, a single-client signal chart, multi-client signal variance chart, user roles, action log, data browser, etc... It has a worker-friendly and cohesive user interface with built-in language localization. It would serve as a good starting point for the next generation of Repunet, if it were not written in PHP language on its own, since no other supportive framework is used. Most of the code is bound to a specific database structure and even though it uses XSL templates and XML configuration, it has been proven inadaptable and unmaintainable for the future use.

3 CURRENT DESIGN

3.1 DATA MINING

The current method for data mining has proven itself unsatisfying, mainly because it can operate a particular hardware only. There is no possibility for those scripts to be utilized, and – even with minor configuration changes – made to read a status of a different access points.
The idea is to implement the data mining scripts in order to become flexible and highly adjustable. That would ensure using SNMP as a data source and a modular daemon as a run-time environment.
The daemon will perform the following: ask a particular module when to launch it (cron-like); encapsulate the database access so that it is not dependant on any particular data scheme; provide SNMP access; threading and freeze-detection system; run-time plugging, unplugging and manual execution.

3.2 WEB-INTERFACE

Making adjustments, I made the design as adjustable and unbound as possible. Since the web-interface by Ing. Petr Hirš makes it extremely difficult to extend the present functionality, after a very careful investigation and testing, I chose a Django [3] web-framework. It offers a very flexible environment for a web developer and includes many useful functions. In Django, each component of the system is considered to be a separate application, thus, it can easily be plugged in/out or reused elsewhere. That fits the modularity Repunet needs.

3.3 DATAMAPPER AND TOPOLOGY VIEWER

To enable Repunet to be adjustable, a datamapper must be defined. This helps the other modules – topology viewer, data viewer, etc. – to interact with the data in the database without any essential knowledge of the underlying database schema. The mapper module maps chosen fields from the database table to a virtual model. The model is defined by the module which uses it and therefore, knows how to use it.
Networks are much easier to manage when they are familiar. This module offers a fully automated topology viewer, i.e. overview, detail view, zoom, etc., and recognizes just the routing table entries at the access points. At this time two ways of view are available. One uses SVG format with JavaScript to draw the map, access points and links among them. The second provides integration with the Google Maps system and displays the topology on a real map.

4 AUTOMATED VMWARE IMAGE BUILDING

4.1 FUNDAMENTALS

The VMWare Company does not provide an easy solution for automated creation of machines. My aim was to automate the building of a VMWare machine with Linux distribution for Repunet or any other software. I created a solution which can be used for various purposes by anyone needing to automatize a building process of a VMWare machine.

4.2 IMPLEMENTATION

I estabilished the following basic goals for the automated building process: Debian Linux as a base; additional software packages; zero user interaction; flexibility; automated copy of selected configuration files and other software; VMWare-playable right away.
I chose bash and GNU toolkit for the implementation because it is presnt in every Linux distribution and, if not present, can be easily obtained.
Enter following parameters to a configuration file: location of any 2 loopback devices it can use on your system; location of target virtual disk file; virtual disk size and partitions; list of additional Debian packages which could be obtained by *apt-get* command; list of configuration and any additional files desired to be copied to the target system.
As first, a base layout containing partitions and filesystem of the new virtual disk is set up. Afterwards it allows the *debootstrap* tool install the base Debian system, runs GRUB script to write MBR and install itself to the virtual disk, then downloads and installs the additional packages, along with their dependencies. Then the configuration and the additional files are copied to the image. At this stage, the script is completed with a product containing a fully working and bootable VMWare image.

5 CONCLUSION AND FUTURE DIRECTIONS

To summarize: Parts that are semi-complete are the automated VMWare image creation script, a base for the web-framework – which will be additionally built – and a few visualisation modules. Concerning the modules, the topology viewer with Google Maps integration, as well as various chart displaying, are complete. For the future, we plan to finish and extend the web interface in such a way that it will allow administrators and other users to interact with any part of the system to view activity of connected clients, to manage access points, block out attackers, and warn administrators of any suspicious activity. To achieve this goal, we must design parts that have not been implemented in the previous stages of the project, i.e. user management, remote control, etc.

REFERENCES

[1] Blahák, P.: Reputační systémy ve WiFi sítích, Master's thesis, 2006.

[2] Hirš, P.: User interface for wifi network security system, Master's thesis, 2007.

[3] Django, [Online] Available www.djangoproject.com.