Integrated System for Collecting, Processing and Storing Network Information

Catarina Monteiro[†], Paulo Salvador[¥], António Nogueira[¥], Eduardo Rocha[¥], Rui Valadas[¥]
[†]Portugal Telecom Inovação, Rua Eng. José Ferreira Pinto Basto, 3810-106 Aveiro, Portugal
Phone +351234403200, Fax +351234424723, email: est-c-monteiro@ptinovacao.pt
[¥]Instituto de Telecomunicações – University of Aveiro, Campo Universitário de Santiago, 3810-193 Aveiro, Portugal
Phone +351234377900, Fax +351234377901, email: {salvador, nogueira, eduardorocha, rv}@ua.pt

***Abstract* — The fast Internet growth and the huge increase in the number of services and in the heterogeneity of network technologies have been accompanied by an increasing number of security threats to networks and their users. Network administrators should have a complete view of the network in order to detect any possible network anomalies/attacks and take the right management decisions. However, the main problem for an efficient management strategy lies on the amount, diversity, dispersion and complexity of the network information, making any analysis and subsequent action over the network almost impossible on real (or at least useful) time. Thus, it is necessary to have tools that can autonomously collect information that is dispersed by different network equipments, correlate and filter it in such a way that only relevant information is presented to the network administrator in a structured form. This paper proposes a network monitoring system that is able to (i) collect relevant network information, such as log files from servers, routers and switches, traffic measurement data or traffic model parameters; (ii) filter, process and store the most important information on a database system and (iii) present the relevant information on a suitable web interface. This tool is intended to become part of a distributed platform for the real time detection of BotNet PCs.**

Keywords: network monitoring, distributed architecture, security, intrusion detection, BotNet.

I. INTRODUCTION

Network monitoring has been around as long as there have been networks. Most routers, switches, and intelligent hubs collect some level of network traffic statistics, while application servers store all service occurrences in their log files. This information is important to network administrators who have responsibilities over the network operation. Without network monitoring systems, it would be difficult to identify and solve many network problems.

Network monitoring can be seen as the ability to collect and analyze network traffic, trying to identify deviations from the *a priori* expected behavior. Most intelligent networking devices offer analysis of layer 1 traffic (physical network problems such as link status, CRC errors, bipolar violations, and framing errors). Moving up from layer 1, dedicated monitoring equipment is often used to analyze layer 2 and layer 3 traffic (layers 2 and 3 monitoring systems are commonly referred to as protocol analyzers). The latest generation of network monitoring products is designed to support very specific applications: for example, some monitoring products are designed to help network administrators identify security threats; some are designed to provide law enforcement officials with tools for real-time surveillance; some are designed to analyze the performance of specific applications; and some are designed to collect raw data for intensive out-of-band analysis. Each of these specializations can yield a focused solution that is designed to address the specific requirements of a vertical market.

The most recent network security threats, like zombie PCs and BotNets, are quite complex, and their detection requires a multi-functional framework composed by several modules and interactions: the detection tool should interact with a distributed traffic measurement and analysis system [1], should have a database with relevant network information and user's profiles, an updatable list of illegal traffic signatures and a system of alert and counter-measures. In fact, the detection of BotNet PCs must be based on several inputs, such as the historical traffic profile of network users, mathematical traffic models that can accurately describe network traffic or user profiles, traffic measurements that could be carried out on some specific probes and artificial intelligence systems that can take some combination of the inputs in order to produce a relevant output that can used by the decision support system [2, 3, 4].

This paper proposes a system for colleting and processing relevant network information that integrates several indispensable functionalities: (i) collect important network information, such as log files from servers, routers and switches, traffic measurement data or traffic model parameters; (ii) filter, process and store the most important information on a database system and (iii) present the most relevant information through a suitable web interface. This integrated monitoring system will be one of the building blocks of a platform, with a distributed architecture, for the real time detection of BotNet PCs.

The rest of the paper is organized as follows: section II briefly describes some of the most recently developed monitoring and intrusion detection systems; section III presents the architecture of the proposed data collection, processing and storage system; section IV describes the general ideas of the distributed framework for the detection of BotNet PCs that will incorporate the proposed monitoring system and, finally, section V presents the main conclusions.

This work was done under the scope of the Euro-NF Network of Excellence, funded by the European Union.

II. State of the Art on Monitoring and Intrusion Detection Systems

In the last few years, companies and organizations have shown an increasing concern about the development and usage of monitoring tools that can cover different network elements, network services and operational parameters and behaviours: databases, log files, network traffic, protocols operation, security issues, SNMP-based functional elements and network services. Several monitoring solutions were proposed in the last few years: AlertBot, a monitoring software for Web pages and Internet servers that alerts the network managers through email or SMS whenever a problem occurs; Activeworx, a security information management system that allows the monitoring of log files and includes a system for alerts, audits and reports on forensics analysis; GFI Network Server Monitor, that monitors the network and its servers, sending appropriate alerts and allowing the implementation (using scripts) of several actions that should be taken as a response to specific detected problems; ActiveXperts, one of the leaders on LAN and WAN monitoring solutions, being able to manage servers, printers, network equipments, databases, etc; HealthMonitor, one of the most detailed tools for systems management, enabling to control all network users and stations; Nagios, a UNIX compatible open-source tool that has the advantage of supporting the addition of new functionalities through plugins and is composed by a central module and several plugins written in C, Perl and Shell scripts; HP OpenView, one of the most widespread and used management platforms, mainly due to its very simple and complete interface and the ability to support the inclusion of additional modules that can be developed by different manufacturers.

Intrusions Detection Systems (IDSs) analyze activities in a network or host looking for abnormal or undesirable actions. When IDSs detect an intrusion, an alarm can be generated or a counter-action can be taken. Currently, IDSs are classified in two major groups: network-based and host-based.

A network-based IDS (NIDS) monitors network traffic of a particular network segment and analyzes the network, transport and application protocols in order to identify suspicious activities. NIDSs are commonly deployed at network boundaries, such as border firewalls or routers, VPN servers and remote access servers. Regarding the detection method used, NIDSs can use behaviour-based detection, in which the IDS looks for network activity that differs for known behaviour, and knowledge-based or pattern-matched detection, in which the IDS compares the collected data with a set of signatures (patterns) of known attacks. In the first approach, the IDS must be trained for a certain period of time so that it gains the knowledge of what is normal activity: the achieved accuracy strongly depends on the quality of this training phase. The main advantage of this technique is that the IDS does not only look for known patterns of intrusions but to all abnormal activity which may lead to the identification of new intrusions. In the second approach, the IDS compares the collected data with a set of signatures (patterns) of known attacks. Therefore, this technique leads to a low number of false-positives. However, the IDS using this technique can only detect known attacks and therefore new attacks or intrusions will not be identified. These systems also need to be constantly updated as they strongly depend on the quality of their signature's databases. There are some freely available NIDS, such as Snort, PortSentry and Scanlogd.

Host based IDSs (HIDSs) run on hosts that provide services and their source data are the different log files generated by the computational system and the services it runs. These files can be checked continuously or periodically. When an application or an OS generates a log, the corresponding action has already occurred and the IDS can only have a nearly real-time detection capability. The most known HIDSs are Tripwire, a public HIDS that detects file changes in UNIX systems; Advanced Intrusion Detection Environment, which it is also a file integrity checker that constructs a database of files specified in its configuration file and stores various attributes, like permissions, file size, mtime and ctime; OSSEC, an IDS that performs file integrity checking and can also detect rootkits.

The rapid network growth and the appearance of Gbps links bring new challenges to IDSs. The complexity and the amount of generated data make traditional IDSs, that were designed for individual or small-scale networks, ineffective. Therefore, new trends in intrusion detection will focus on scalability and heterogeneity. IDSs need to be scalable in order to accommodate the amount of data generated by current networks and must be able to process heterogeneous information from different components. The initial approaches started by creating IDSs that gather data in a distributed manner, although the analysis was still done in a centralized place (DIDS [5] and ASAX [6]). Although data was reduced before being sent to the central analysis module, the system scalability was still limited. On some recent systems, such as EMERALD [7] and GRIDS [8], various components are placed in different locations and each component receives audit data from a limited number of sources in order to avoid the system to become overwhelmed. The components can also be organized in a hierarchical manner and lower level components send their detection results to higher level modules. In this way, the information from different locations can be correlated, which leads to a more accurate detection of attacks and also solves the issues that were brought by high-speed links.

III. Proposed System Architecture

Since none of the existing monitoring solutions provides all the functionalities that we need to incorporate in the integrated BotNet detection tool, a new platform for data collection, processing and storage was developed. The architecture of the proposed system is illustrated in Figure 1 and includes the following functional tasks: (i) collect information; (ii) filter and process collected information; (iii) store data on a database system; (iv) present network resources and network information in a unified way, using a Web interface.

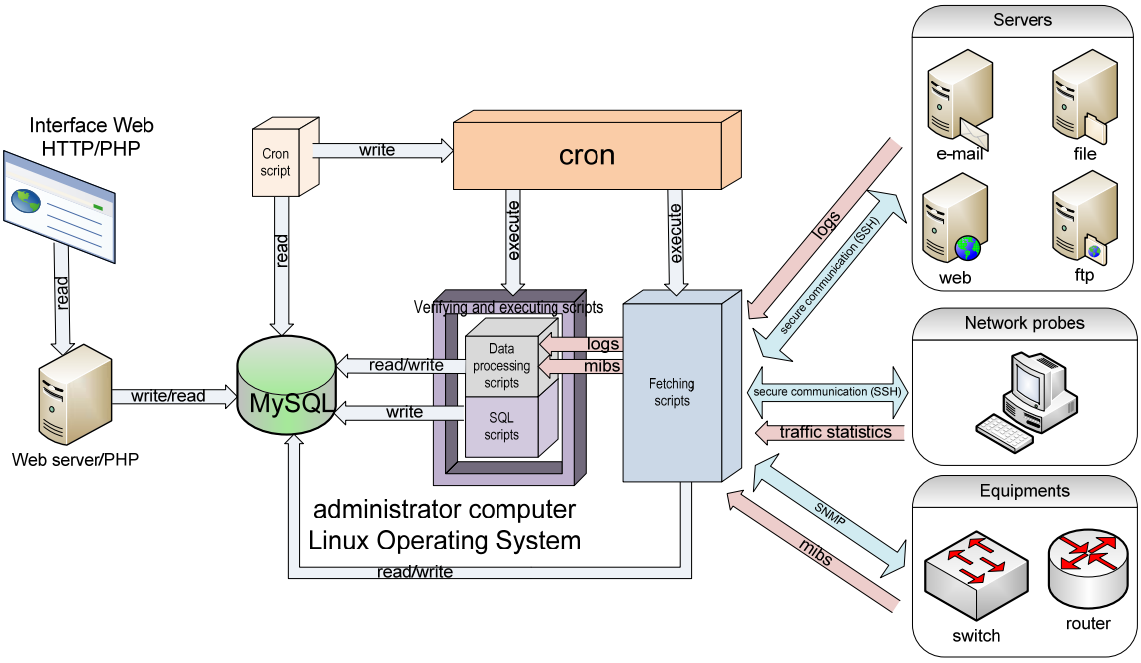

Fig. 1. Architecture of the proposed integrated monitoring system.

In the proposed system architecture, Linux was selected as the operating system running on the administrator machine and Cron was used to schedule system processes. Cron "wakes up" every minute and examines all the stored configuration files, called crontabs, to check each one of them for commands that may be scheduled to be executed at the current time. Shell scripts are used as the basis for programming tasks since they have several advantages that can be exploited: they are an interpreted language that is natively executed; they are very fast, have a quite simple syntax and, together with the use of Cron, allow a complete autonomy of the tasks. The database server will be based on MySQL, an open-source software that has the capacity to handle a lot of logs, has an excellent performance, is easy to use and can be a multi-task and multi-user software. The Web interface will be developed using PHP, a scripting language originally designed for producing dynamic web pages that has evolved to include a command line interface capability and can be used in standalone graphical applications.

As depicted in Figure 1, the fetching scripts will remotely collect the relevant monitoring information. The following scripts were developed: (i) log fetching scripts, used to collect log files from servers (this information transfer is conveniently secured by using SSH); (ii) probe fetching scripts, used to collect traffic statistical information that was measured at the network probes (this transfer is also secured through SSH); (iii) SNMP fetching scripts, to collect network equipment (routers, switches, access points, etc) information that is available on their MIB objects.

After collecting the relevant data, the data processing scripts will process and extract the most relevant information from log files, statistics and network equipment MIBs using shell script commands.

Regarding log files, a lot of scripts were created in order to cover information related to some of the most important network services: (i) for the Linux OS, log files from the Apache, ProFTPD, DHCPd, OpenSSH, Samba, NFS and Sendmail servers were considered; (ii) for the Windows OS, log files from the IISWeb, IISFTP, DHCP, IISSMTP, Terminal and File servers were included in our platform.

Regarding network equipments, several scripts were created to deal with relevant information about switches and routers that is contained into several MIB objects: system, interface, IP, datagram, ICMP, TCP and UDP statistics, UDP ports, and network.

Different scripts were also written in order to create the database, create the necessary tables inside the database and write on the appropriate table fields. SQL scripts are used for dealing with server's information, while shell scripts are used to deal with information related to network equipments, since SQL scripts do not accept input and output parameters.

Finally, a Web interface was developed in order to allow the network administrator to perform several tasks, like inserting the network servers and equipments that should be monitored or making the intended data collection and processing actions, and provide the network administrator a unified vision of the network data and resources.

IV. PLATFORM FOR THE DETECTION OF BOTNET PCS

The proposed data collection system will be one of the building blocks of a distributed platform for the real time detection of BotNet PCs.

Due to the inherent complexity of this task, the detection of BotNet PCs will rely on a diversity of network information: the historical traffic profile of network users, mathematical traffic models that can accurately describe network traffic or user profiles, traffic measurements that will be carried out on some specific probes and artificial intelligence systems that can take some combination of inputs in order to produce a relevant output that can be used by the decision support system.

In a first step, the monitoring and detection platform should collect and store diverse network information that is dispersed over several network components: (i) state variables and statistics that are constructed and maintained by routers, switches and access points; (ii) log files that are stored on different types of network servers; (iii) traffic captures or statistics that are obtained and stored on network probes that are distributed over the network infrastructure.

Then, the developed system should process and correlate the collected information, obviously having the possible final goals in mind: help on the detection of network anomalies/security flaws, propose appropriate counter measures whenever necessary and help in other network management tasks. This phase of the detection platform construction must rely on the research of new methodologies and strategies for correlating relevant network information: is fact, this part of the proposed management system must have the "intelligence" to select important information from the huge amount of data stored in the system database, process and relate it in such a way that it can become useful for network managers to take important decisions.

The different system nodes will compile data searching for deviations from the normal behavior and identifying the location and class of any possible security threat. After a potential threat detection, the system should be able to automatically perform a deeper analysis over archived data or more detailed (and localized) monitoring actions.

Following an anomaly detection, the more relevant data should be passed to the alarm processing modules. These modules should, first of all, perform automatic validation of the alarm, correlating all data with known/authorized causes of abnormalities (new users, new legit applications, protocol updating, etc...). After validation, the anomaly information should be passed to the monitoring modules for updating the corresponding models. If the alarm processing module can not automatically validate the anomaly the human manager should be notified and asked for action. For severe threats, the alarm processing modules can deploy automatic protection within the network (i.e. new firewall rules, new access list rules at the routers, isolation of network areas, traffic rerouting, etc...). Moreover, the alarm module should filter all data, presenting only the most relevant information to the human network manager and suggesting him an appropriate action. At last, the human decision should be transferred to the monitoring modules for model updating.

Finally, the developed system must propose/suggest and, possibly, deploy appropriate counter-measures for specific security problems or flaws that were identified. As examples of these actions, we can mention the establishment of optimal routing strategies, change current routing approaches, change/optimize configurations on different network elements, like servers, switches, routers, etc. In a sentence, the proposed system must deploy appropriate counter measures and effectively aid in taking the appropriate human-based counter-actions.

V. CONCLUSIONS

This paper proposed a network monitoring system with the ability to perform several tasks (i) collect relevant network information, such as log files from servers, routers and switches, traffic measurement data or traffic model parameters; (ii) filter, process and store the most important information on a database system and (iii) present the relevant information on a suitable web interface. This data collection, processing and storing system will be part of an integrated platform for the real time detection of BotNet PCs: the framework will interact with a distributed traffic measurement and analysis system, have a database with relevant network information and user's profiles, an updatable list of illegal traffic signatures and a system of alert and counter-measures.

REFERENCES

[1] P. Salvador, R. Valadas, "A Network Monitoring System with a Peer-to-Peer Architecture", 3rd International Workshop on Internet Performance, Simulation, Monitoring and Measurement (IPS-MoMe 2005), 15-16 March, 2005, Warsow, Poland.

[2] A. Nogueira, P. Salvador, R. Valadas, "A Framework for Detecting Internet Applications", International Conference on Information Networking (ICOIN'2007), January 23-25, 2007, Estoril, Portugal. Lecture Notes in Computer Science, Springer-Verlag Berlin, 2007.

[3] P. Salvador, A. Nogueira, U. França and R. Valadas, "Detection of Illicit Traffic using Neural Networks", International Conference on Security and Cryptography – part of The International Joint Conference on e-Business and Telecommunications, July 26-29, 2008, OPorto, Portugal.

[4] R. Nogueira , A. Nogueira , P. Salvador , R. Valadas, "Identifying Differentiating Characteristics of Internet Applications using Principal Component Analysis", 6th Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP08), 23-25 July, 2008, Graz, Austria.

[5] S. Snapp, R., et al., "DIDS (distributed intrusion detection system)-Motivation, architecture, and an early prototype", Proceedings of 14th national computer security conference, pp. 167--176.

[6] A. Mounji, B. Charlier, D. Zampuniéris and N. Habra, "Distributed audit trail analysis", Proceedings of the ISOC'95 Symposium on Network and Distributed System Security, pp. 102--112, 1991.

[7] U. Lindqvist and P. Porras, "Detecting computer and network misuse through the production-based expert system toolset (PBEST)", Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 146--161, 1999.

[8] S. Axelsson, "Research in intrusion-detection systems: A survey", Technical report TR 98-17, Chalmers University of Technology, 1999.