International Journal of Cyber-Security and Digital Forensics (IJCSDF) 4(3): 427-441
The Society of Digital Information and Wireless Communications, 2015 (ISSN: 2305-0012)

Using Decision Trees to Extract IDS Rules from Honeypot Data

Pedro Henrique Matheus da Costa Ferreira, Leandro Nunes de Castro

Natural Computing Laboratory, Graduate Program in Electrical Engineering
Mackenzie Presbyterian University

Email: phmatheus@msn.com, lnunes@makenzie.br

ABSTRACT

It has been almost two decades since the first honeypot was proposed. Despite that, although there are several studies involving network traffic data, few are those dedicated to extract knowledge from honeypot data. The present paper uses data collected by honeypots to create rules and signatures for intrusion detection systems. The rules are extracted from decision trees constructed based on the data of real honeypots installed on internet connections without any filter. The results of the experiments showed that the extraction of rules for an intrusion detection system is possible using data mining techniques, in particular decision trees. The technique proposed allows the analyst to summarize the data into a tree, where he/she can identify problems and extract rules to help reducing or even mitigate the security problems pointed out by the honeypot.

KEYWORDS

Honeypot, Intrusion Detection System, Datamining, Decision Tree, Dionaea.

1 INTRODUCTION AND MOTIVATION

Over the past fifteen years there has been an exponential increase of devices connected to the Internet [1], which promoted the emergence of a new and fertile ground for cyber criminals. The system failures, the lack of technical training for network administrators and the lack of vision of the companies that information security is a vital area for the health of business [2] provide the perfect opportunity for criminals to take advantage.

One of the main difficulties of a network administrator is to keep the network safe from external attacks. According to [3] the attacks reported by companies in the last two years are divided as follows: 43% are of malicious code injection through SQL; 19.95% are targeted only at companies or services provided by companies (APT); 18.81% are *Botnet's*; and 18.24% are denial of service attacks (DoS).

Still according to this study, organizations face an average of 66 cyber attacks weekly, causing some sort of damage to businesses. Organizations in Germany and the United States experience the highest average weekly attacks, 82 and 79, respectively. Brazil and Hong Kong have the lowest average frequency, totaling 47 and 54 attacks per week, respectively.

This type of scenario brought to light studies, such as [4], which proposed the first intrusion detection system and the work of [5], which launched the first *honeypot*. The work in [6] proposed the creation of virtual honeypots. These works seek to create tools to assist the protection of computing assets by detecting intruders or creating traps to monitor malicious activities.

The present paper proposes the application of a data mining technique based on the C4.5 decision tree algorithm to datasets obtained from attacks targeting Dionaea honeypots. After the application of the technique it was possible to generate rules for the IDS. The method also reduced the volume of data to be analyzed allowing the network administrator to have an analytical overview of the information captured.

The paper is organized as follows. Section 2 provides a brief review of honeypots and Dionaea. Section 3 presents the datasets to be used in the experiments, their structure and a descriptive statistics of the data. Section 4 discusses intrusion detection systems based on decision trees for the four datasets, and Section 5 presents the extraction of decision rules from the decision trees. The paper is concluded in Section 6 with a general discussion

about the results obtained and some potential future works.

2 HONEYPOTS AND THE DIONAEA

The first intrusion detection model was designed by [4] to analyze real-time data in order to detect security breaches, invasions and other forms of abuse of computer access. This model was based on the assumption that security breaches could be identified through the system audit logs, allowing the detection of anomalies in usage patterns. Another design feature was the fact that it is independent of a system vulnerability or type of invasion. This model provided a general-purpose framework for intrusion detection systems.

Based on the assumptions used by Denning [4] to audit logs the first honeypots were proposed. The honeypot is a computer security system dedicated to being probed, attacked or compromised [7]. The first available honeypot was created in 1998 by Cohen in [5] and was designed to simulate a system with vulnerabilities. In the early 2000s worms began to proliferate, requiring the collection of these artifacts for examination and the creation of vaccines for antivirus systems. Having identified this need, in [5] virtual honeypots were proposed in which a single device can run multiple honeypots. This work led to the creation of the Honeyd project [7], which emulates in a single physical machine several different operating systems and multiple hosts on a network. In an attack the Honeyd tries to passively identify the remote host, collecting network traffic and TCP/IP stack information. This system has the capability to emulate all the TCP/IP stack enabling sophisticated network analysis tools such as nmap, to be deceived.

After the Provos [6] proposal, it began to emerge several honeypots to emulate complex operating systems, their network services and specific services independent of an operating system. With this new wave it became necessary to classify the types of honeypot and, therefore, it was proposed a classification into three categories: low, medium and high interaction.

2.1 Low Interaction Honeypots

Characterized by emulated computer systems through computer programs that contain the minimum operation standards of the service to be monitored [7]. This type of honeypot records the attack and its respective shellcode, offering little information about the attack to determine the cause or the mechanisms used. The information collected allows the administrator to identify whether your network is being targeted by attacks and scans.

2.2 Medium Interaction Honeypots

These types of honeypots are in between the high and the low interactivity ones. Its main feature is to provide the virtualization of the application layer where the operating system environment and the communication protocols are emulated in order to provide sufficient answers to deceive the attacker and get the payload [8].

One of the challenges of this system is its complexity of development and the remote possibility of the attacker to gain access to the host system, affecting all the equipment and the network in which the system is. This paper will address only medium interaction honeypots, specifically Dionaea.

2.3 High Interaction Honeypots

Characterized by real systems with known and purposely uncorrected failures. These are expected to be attacked and compromised [9]. In the high-interaction honeypot it is possible for the attacker to compromise and gain control of the system to install software artifacts and complete the malicious activity.

2.4 Dionaea

The Dionaea [10] was the honeypot chosen for this work due to its storage and data organization characteristics and its capability of capturing malicious artifacts. The collected data can be used to compare the techniques used in this work with works from the literature and the malicious artifacts captured will be used, along with data collected, in the feature extraction process.

The Dionaea is a honeypot of medium interactivity aimed at replacing its precursor, the Nepentes [7].

The great contributions that Dionaea brought to Nepentes were the separation of the core of the system developed in C++, the inclusion of support for Python [11] as scripting language, the use of the libemu library to shellcodes detection, and the native support for IPv6 [12] and TLS [13].

The Python programming language is used to develop the vulnerabilities and supported modules, together with the storage and transmission functions of the information collected. This inclusion brought some indirect benefits to the honeypot, such as the possibility of including other types of services not initially planned, for instance, the vulnerabilities in Microsoft SQL Server database [14] and Session Initiation Protocol [15], which is used for controlling multimedia communications sessions, among others.

Dionaea was one of the first honeypots to add support for IPv6 protocol, allowing the analysis of the vulnerabilities that are being exploited in this new communication protocol that will replace IPv4.

3 AN OVERVIEW OF THE DATASETS INVESTIGATED

Companies are reluctant to release databases with honeypots' data because they contain sensitive information about the structure of their network and the attacks they are facing. In addition to revealing the addresses of their honeypots, it also reveals its configuration. For this reason the creators of Dionaea made two datasets public for experimentation: Paris and Berlin. In addition to these two datasets, in the experiments performed here two new datasets are used: Campinas, and Jacarei. The first one (Campinas) was installed in the Renato Archer IT Research Center, in the city of Campinas, SP, Brazil; the second one, named Jacarei, was obtained by a honeypot installed in a private home in the city of Jacarei, SP, Brazil. Table 1 summarizes some features of the four datasets used in the present research.

Table 1: Summary of the honeypot data.

Dataset	Start Capture	End Capture	Number of attacks	Number of malwares
Berlin	05/11/2009	07/12/2009	604.201	2.726
Campinas	20/07/2011	15/10/2011	3.754.124	165.088
Jacarei	28/04/2010	20/05/2010	44.883	13.605
Paris	30/11/2009	07/12/2009	7.822.148	749.518

3.1 Structure

The information collected by the honeypots was stored in a SQLite database. SQLite provides a software library that implements an autonomous transactional database service, without the need of servers or setup, as it does not require separate servers or processes. The library reads and writes information directly into the disk [16].

The entity relationship model and the honeypot database can be viewed in Figure 1 and is divided into five areas:

- A central table containing the primary information of the attack (*connections* table). This table stores information such as the IP address of the attacker, the IP address of the Honeypot, local and remote ports, time of the attack, connection types, protocol types, etc.;
- On the left there are three other tables that are used to store the information about the attacks against the Microsoft SQL Server service (MSSQL) (tables *mssql_commands, mssql_fingerprints* and *logins*). The information stored in these tables consist of commands sent to the honeypot to compromise the service, users and passwords in brute force attacks, and information about the attackers, such as version connection library, customer signatures, etc.;
- In the right hand side there are four tables that refer to honeypot firewall logs (*p0fs*), the resolution of attackers names (*resolves*) and services emulated by the honeypot (*emu_profiles* and *emu_services*). The latter contains the information about the codes used to circumvent the security of the application and send commands so that the honeypot performs actions aimed at compromising their security and integrity;

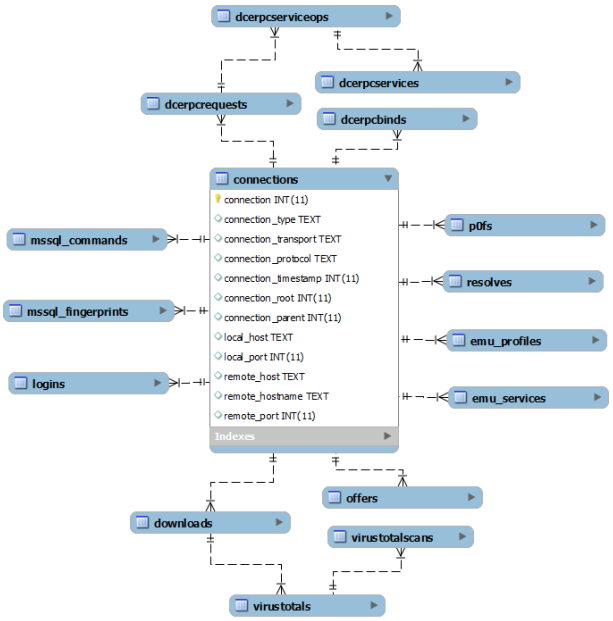

Figure 1: The honeypot database diagram.

- At the bottom there are two sets of interrelated tables. The first one refers to the collection of malicious artifacts, which stores information about where are these files and their MD5 hash (table *downloads*). This table is linked to two other *VirusTotal* and *virustotalscans* tables, which are used to store the information obtained by the *VirusTotal* tool. The second set relates to the offers table that stores the information concerning the provision of malicious artifacts to the honeypot. Often it is not possible to obtain the malicious artifact because where it was stored is no longer infected or it is off at the offer time;
- At the top there are the tables that refer to the Distributed Computing Environment (DCE) and Remote Procedure Calls (RPC) of the communication protocol of the systems based on the Service Message Block (SMB) (*dcerpcservices* tables, *dcerpcrequests*, *dcerpcserviceops*, *dcerpcbinds*).

The database has a total of seventy eight attributes divided into sixteen tables, and only forty-two useful attributes, because sixteen of them are connection attributes and relationships between the tables, and sixteen others are sequential indices of the attributes contained in the object table. The forty-two possible attributes to be analyzed are divided into two groups, thirty-seven nominal attributes and nine numeric ones.

The database is divided into five sets that store different information about the attacks that target specific services. This paper addresses three of the five sets of information that are defined by tables *connections*, *dcerpcbinds*, *dcerpcservices*, *dcerpcserviceops*, *downloads*, and *offers*.

This set of tables was chosen because it represents attacks on devices with the Microsoft Windows operating system. This type of attack is more than 90% of the attacks recorded by the Honeypot, as will be discussed below.

3.2 Descriptive Statistics

The descriptive statistics of the datasets is used to describe, simplify or summarize their main features, thus forming the basic steps of any data analysis process. In the present paper the classes and frequency of each class for the features selected (Table 2) are presented.

3.2.1 The Paris Dataset

The Paris database has the following characteristics:

- Number of attacks: 7,822,148 recorded in the connections table. After joining with tables *dcerpcbinds*, *dcerpcservices*, *downloads*, *dcerpcserviceops*, and *offers* the number of objects sum up to 19,755,323. This is due to the attacks that use a single connection to explore more than one vulnerability, allowing a record in the connections table to have more than one record in the other tables.
- Collection Period: between 30/11/2009 and 07/12/2009.
- Number and type of attributes: 11, being 1 numeric, one date, 1 attribute of type hour, and 8 nominal attributes.
- Additional information: the attributes are not standardized and those who have objects with missing values were filled with the word EMPTY.

Data were integrated eliminating index and interconnection attributes between the tables.

This dataset will be used to obtain the rules of the intrusion detection system (IDS).

A descriptive analysis of the resulting data set was performed, identifying that 93.99% of the attacks were directed towards the SMDB service on port 445. This analysis showed, for example, that the attribute *dcerpcbind_uuid* presented 95 different classes, being the class *4b324fc8-1670-01d3-1278-5a47bf6ee188* the most frequent one with a percentage of 60.71%. The second most frequent class was the EMPTY one, with a frequency of 27.93%; the other classes had a frequency equal to or less than 0.37%. This class is the DCERPC interface used to connect to the honeypot and is extremely important to the creation of the IDS rules.

Table 2: Attribute characteristics and descriptions.

Nº	Attribute	Type	Description
1	Connection	Integer	Table Connections Index
2	Connection_Transport	Nominal	Transport Kind (TCP, UDP, TLS)
3	Connection_Protocol	Nominal	Connection Protocol
4	Local_Port	Integer	Honeypot local port (de 0 a 65535)
5	Remote_Host	Nominal	Attacker IP Address
6	DceRpcBind_UUID	Nominal	RPC Bind Interface
7	DceRpcBind_TransferSyntax	Nominal	RPC Bind Interface Transfer Syntax
8	DceRpcService_Name	Nominal	RPC Accessed Service Name
9	DceRpcServiceop_name	Nominal	RPC Operation Service Name
10	DceRpcServiceop_Vuln	Nominal	Microsoft Security Report Name
11	Connection_Date	Date	Date of Attack in format YYYY-MM-DD retrieved from TimeStamp
12	Connection_Time	Time	Time of Attack in format HH:MM:SS retrieved from TimeStamp

The analysis gave rise to several interesting facts about the data set. For instance, when analyzing the attribute *dcerpcserviceop_name* it can be observed that vulnerabilities were exploited in three DCERPC services with an identical frequency of 22.57%, which together represent 67.71% of the data set: *NetPathCanonicalize*, *NetPathCompare* and *NetShareEnum*. The rest is divided into five classes: EMPTY with 31.99%, which is related to attacks that had not explored DCERPC services and therefore are with *EMPTY* value; class *RemoteCreateInstance* with 0.23%; class *RemoteActivation* with 0.06%; *DsRolerUpgradeDownlevelServer* class with 0.01%; and *NetAddAlternateComputerName* class with 0%.

Along with these data it could be observed that 45.14% of the exploited vulnerabilities refer to the Microsoft Security Bulletin MS08-67, in which it is informed that the vulnerability could allow remote code execution on the server. Another vulnerability pointed out by the analysis is provided by the Microsoft Security Bulletin MS04-12 with a frequency of 0.23%, followed by the MS03-26 security bulletin with 0.06% and, finally, the bulletin MS04-11 security with 0.01% frequency.

It is noteworthy that the honeypot was unable to relate 22.57% of the attacks reported to a known security bulletin. This can occur because the honeypot is not targeted at the vulnerabilities captured for they are new vulnerabilities or variations of vulnerabilities documented in Microsoft security bulletins. These records have been named as *Not_Identified*. The rest of the analyzed objects (31.99%) did not explore a DCERPC failure and, thus, were identified by the *EMPTY* class.

After the descriptive analysis it was found that the attacks on the honeypot were directed at Windows services and, therefore, IDS rules were designed to identify attacks on equipment with Microsoft OS. Table 3 shows the domain of each attribute.

Table 3: Summary of the attributes domain for the Paris dataset.

Nº	Attribute	Type	Domain
1	Connection	Discrete (Integer)	[0,∞]
2	Connection_Protocol	Discrete	smbd, httpd, epmapper, TftpClient, TftpServer-Handler, emulation, ftpd, remoteshell, ftpctrl, ftpdata
3	Date		30/11/2009 a 07/12/2009
4	Time		00:00:00 Times a 23:59:59 Times
5	Local_Port	Discrete (Integer)	[0, 65535]
6	Remote_Host	Nominal	Attacker IP address, from 001.000.000.000 to 254.255.255.255.
7	DCERPCBind_UUID	Nominal	94 distinct values in the range 00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff plus the EMPTY class.
8	DCERPCBind_TransferSyntax	Nominal	43 distinct values in the range 00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff plus the EMPTY class.
9	DcerpcService_Name	Nominal	SRVSVC, EMPTY, ISystemActivator, DCOM, samr, DSSETUP, IOXIDResolver, WKSSVC, SVCCTL, epmp
10	DcerpcServiceop_name	Nominal	EMPTY, NetCompare, NetPathCanonicalize, NetShareEnumAll, RemoteCreateInstance, RemoteActivation, DsRolerUpgradeDownlevelServer, NetAddAlternateComputerName
11	DcerpcServiceop_vuln	Nominal	MS08-67, EMPTY, Not_Identified, MS04-12, MS03-26, MS04-11

3.2.2 The Berlin Dataset

The Berlin database has the following characteristics:

- Number of attacks: 604,201 recorded in the connections table. After joining with tables *dcerpcbinds*, *dcerpcservices*, *downloads*, *dcerpcserviceops*, and *offers* the number of objects sum up to 635,836. This is due to the attacks that use a single connection to explore more than one vulnerability, allowing a record in the connections table to have more than one record in the other tables.
- Collection Period: between 05/11/2009 and 07/12/2009.
- Number and types of attributes: 11, being 1 numeric, one Date, 1 Hour, and 8 nominal.
- Additional information: the attributes are not standardized and those who have objects with missing values were filled with the word *EMPTY*.

Data were integrated eliminating index and interconnection attributes between the tables.

The Berlin dataset has different characteristics from the other data sets studied. Whilst in the other datasets the most frequently attacked protocol is SMBD, in Berlin it is the EPMAPPER with 62.20% of all attacks. This protocol is used to map the services RPC to TCP ports: in a normal situation the client makes a connection on port 135 to perform a query on which port the RPC service is allocated. To end this connection the client opens a new connection in the port informed by the EPMAPPER service [17].

This characteristic is confirmed when we analyze the attribute *local_port* which attack primarily port 135 with a frequency 62.21% and then port 445 with 21.11%. Another feature identified in the analysis is that 90.88% of the attacks did not use an RPC service. It can be assumed that the honeypot that recorded the Berlin data was not targeted by attacks that exploit vulnerabilities in RPC services or there was some kind of block between the honeypot and the attacker not enabling the attacker to continue exploiting existing vulnerabilities. This hypothesis is confirmed by checking that the *EMPTY* class is predominant in the attributes *dcerpcbind_uuid*,

with a frequency 88.86%, 88.86% on *dcerpcbind_TransferSyntax*, *dcerpcserviceop_name with* 97.29%, and 97.29% on *dcerpserviceop_vuln*.

When analyzing the *connection_date* and *connection_time* attributes it was identified that attacks occur mostly (75.11%) between Thursday and Saturday, between noon and midnight. The analysis allowed to study the domain of each selected attribute, which can be seen in Table 4.

Table 4: Summary of the attributes domain for the Berlin dataset.

Nº	Attribute	Type	Domain
1	Connection	Discrete (Integer)	[0, ∞]
2	Connection_Protocol	Discrete	epmapper, smbd, pcap, httpd, TftpClient, remoteshell, ftpdata, ftpctrl, ftpd, emulation, TftpServerHandler, ftpdatalisten, mirrorc, mirrord.
3	Date		05/11/2009 a 07/12/2009
4	Time		00:00:00 Times a 23:59:59 Times
5	Local_Port	Discrete (Integer)	[0, 65535]
6	Remote_Host	Nominal	Attacker IP address, from 001.000.000.000 to 254.255.255.255.
7	DCERPCBind_UUID	Nominal	57 distinct values in the range 00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff plus the class EMPTY.
8	DCERPCBind_TransferSyntax	Nominal	42 distinct values in the range 00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff plus the class EMPTY.
9	DcerpcService_Name	Nominal	EMPTY, samr, DSSETUP, SVCCTL, IOXIDResolver, SRVSVC, ISystemActivator, DCOM, PNP, epmp, lsarpc
10	DcerpcServiceop_name	Nominal	EMPTY, DsRolerUpgradeDownlevelServer, RemoteCreateInstance, NetCompare, NetPathCanonicalize, NetShareEnumAll, RemoteActivation, PNP_QueryResConfList
11	DcerpcServiceop_vuln	Nominal	EMPTY, MS04-11, MS08-67, MS04-12, Not Identified, MS03-26, MS05-39

3.2.3 The Jacarei Dataset

The Jacarei database has the following characteristics:

- Number of attacks: 44,883 recorded in the connections table. After joining with tables *dcerpcbinds*, *dcerpcservices*, *downloads*, *dcerpcserviceops*, and *offers* the number of objects sum up to 191,764. This is due to the attacks that use a single connection to explore more than one vulnerability, allowing a record in the connections table to have more than one record in the other tables.
- Collection Period: between 28/04/2010 and 20/05/2010.
- Number and type of attributes: 11, being 1 numeric, one Date, 1 Hour, and 8 nominal.
- Additional information: the attributes are not standardized and those who have objects with missing values were filled with the word EMPTY.

Data were integrated eliminating the index and interconnection attributes between the tables.

The Jacarei has 99.96% of the attacks targeted at the SMBD protocol and 90.42% of the attacks using the RPC SRVSVC service. As in the Paris dataset it is possible to observe the same behavior for the *dcerpcserviceop_name* attribute, where the sum of the three classes *NetPathCanonicalize* (30.14%), *NetPathCompare* (30.14%) and *NetShareEnum* (30.14%) has the same value of SRVSVC class of the *dcerpcservice_name attribute*. This behavior is repeated in the frequency of the *dcerpcbind_uuid* attribute, where the class *4b324fc8-1670-01d3-*

1278-5a47bf6ee188 represents 90.42% of the registered attacks. The analysis of *dcerpcbind_uuid* shows another peculiar feature: most classes have the same frequency (205 and 2), suggesting that the attacker was seeking vulnerabilities with other DCERPC interfaces, most likely in a variation of the original attack that explored the MS08-68 vulnerability, which represented 60.28% of the registered attacks, followed by the *Not_Identified* attacks

When analyzing the attributes *connection_date* and *connection_time* it seems that most of the attacks took place in the afternoon and evening (between 1:00 p.m. and 12 p.m.) for every day of the week, but with less intensity on Tuesdays. The analysis allowed to study the domain of each selected attribute, which can be seen in Table 5.

Table 5: Summary of the attributes domain for the Jacarei dataset.

Nº	Attribute	Type	Domain
1	Connection	Discrete (Integer)	[0, ∞]
2	Connection_Protocol	Discrete	epmapper, smbd, pcap, httpd, TftpClient, remoteshell, ftpdata, ftpctrl, ftpd, emulation, TftpServerHandler, ftpdatalisten, mirrorc, mirrord
3	Date		05/11/2009 a 07/12/2009
4	Time		00:00:00 Times a 23:59:59 Times
5	Local_Port	Discrete (Integer)	[0, 65535]
6	Remote_Host	Nominal	Attacker IP address, from 001.000.000.000 to 254.255.255.255.
7	DCERPCBind_UUID	Nominal	57 distinct values in the range 00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff plus the class EMPTY.
8	DCERPCBind_TransferSyntax	Nominal	42 distinct values in the range 00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff plus the class EMPTY.
9	DcerpcService_Name	Nominal	EMPTY, samr, DSSETUP, SVCCTL, IOXIDResolver, SRVSVC, ISystemActivator, DCOM, PNP, epmp, lsarpc
10	DcerpcServiceop_name	Nominal	EMPTY, DsRolerUpgradeDownlevelServer, RemoteCreateInstance, NetCompare, NetPathCanonicalize, NetShareEnumAll, RemoteActivation, PNP_QueryResConfList
11	DcerpcServiceop_vuln	Nominal	EMPTY, MS04-11, MS08-67, MS04-12, Not Identified, MS03-26, MS05-39

3.2.4 The Campinas Dataset

The Campinas database has the following characteristics:

- Number of attacks: 3,754,124 recorded in the connections table. After joining with tables *dcerpcbinds*, *dcerpcservices*, *downloads*, *dcerpcserviceops*, and *offers* the number of objects sum up to 11,162,100. This is due to the attacks that use a single connection to explore more than one vulnerability, allowing a record in the connections table to have more than one record in the other tables.
- Collection Period: between 20/07/2011 and 15/10/2011.
- Number and types of attributes: 11, being 1 numeric, one Date, 1 Hour, and 8 nominal.
- Additional information: the attributes are not standardized and those who have objects with missing values were filled with the word EMPTY.

Data were integrated eliminating index and interconnection attributes between the tables.

The Campinas data set contains the greatest diversity of recorded attacks, not in quantity but in variety. The honeypot recorded attacks not only on the

SMBD service (94.40%) but also on different services, such as MSSQLD (3.69%), HTTPD (0.96%), XMPPCLIENT (0.23%) and SIPSESSION (0.10%). This honeypot attacks were recorded in the third quarter of 2011, while the other sets were recorded in late 2009 and early 2010. It is clear in this study a change in the behavior of the recorded attacks. The previous attacks were directed exclusively to exploit vulnerabilities of remote procedure calls of Windows machines, but the current ones begin to explore other attacks, such as the Microsoft database service and IP telephony systems.

When analyzing the attributes *dcerpcservice_name*, *dcerpcserviceop_name*, *dcerpcbind_uuid*, *dcerpcbind_transfersyntax* and *dcerpcserviceop_vuln*, it is observed that the most frequent class is *EMPTY*, where the honeypot was unable to treat requests generated by the attacker. This is confirmed when studying the frequency of *dcerpcserviceop_name* attribute, which recorded 37 different calls from those observed in the other datasets (*NetPathCanonicalize*, *NetPathCompare* and *NetShareEnum*). One factor to be noted is that in 91.41% of the cases the honeypot was unable to treat and or associate an attack to a Microsoft security bulletin and, differently from the other datasets, the vulnerability exploited in the MS08-67 security bulletin represents only 5.92% of the attacks. One hypothesis for the low number of attacks exploiting the vulnerability MS08-67 is that it was posted by Microsoft on October 23, 2008 and, as the honeypot collected data in the third quarter of 2011, the attackers no longer explored this vulnerability because it had already been corrected.

4 IDS BASED ON DECISION TREES

The use of Decision Trees (DT) as a model to classify malicious activity is interesting because of both their classification performance and the possibility to extract rules that identify each type of attack. Moreover, once generated the decision tree it can be used to identify anomalous malicious activity [18].

According to Markey [18], decision trees are techniques that help in the analysis of large sets of data for intrusion detection, being able to answer questions like: "What rules should be used to distinguish malicious traffic from legitimate one?", or "What are the most common features of a scanning activity when compared to other data traffic?". In the experiments performed in this paper, it was chosen the software RapidMiner to implement the DT [19]. To evaluate the selected attributes and the decision tree it was defined the *DceRPCServiceop_Name* as the class attribute, because Microsoft releases the Remote Procedure Call Protocol Extension [20], which indicates which calls and subscriptions lead to remote procedures (attribute *DceRPCService_name*).

A *k*-fold cross validation, with $k = 10$, was used to estimate the DT classification performance. The first difficulty to run the algorithm was the number of existing objects in the database (nearly 20 million). Even running experiments on a computer with 32GB RAM and 128GB swap, the machine could not handle all these data. Given this difficulty, it was decided to sample the data based on time periods.

Thus, the Paris data set was the one with the lowest number of samples, totaling eight subsets of data sampled from the whole set. The Berlin group was active for five weeks and, therefore, 5 samples were obtained, one for each week of honeypot activity. The Jacarei honeypot was also active for five weeks and, thus, had 5 samples. The Campinas dataset corresponded to a thirteen-week time period, and thus resulted in 13 samples.

After obtaining all the samples, it was created 31 samples to be studied and submitted to the C4.5 decision tree algorithm [21].

4.1 Paris

Three different trees were created, the first one corresponds to the sample set of the first day, the second one represents days 2,4,5,6,7,8 and, finally, the third tree corresponds to day 3. To analyze a decision tree one must follow the path between the root and the leaf nodes of the tree. Each path between the root and a leaf generates one decision rule.

For the first tree, if we start from the dcerpcserviceop_vuln attribute with value MS04-11, the *connection_protocol* attribute is SMBD, *connection_transport* is TCP, dcerpcbind_uuid has a value *Not_Identified* and the *dcerpcbind_transfersyntax* attribute is divided into two, one with

Not_Identified and another with *8a885d04-1ceb-11c9-9fe8-08002b104860*.

By analyzing the left hand side of the tree, the leaf is *NetAddAlternateComputerName*. This rule can be interpreted as follows: a connection that exploits the vulnerability described in MS04-11 report used the SMBD protocol on a TCP connection did not have a DCERPC interface and a transfer syntax identified, and tried to add an alternative computer name.

The resulting rule on the right can be interpreted as follows: a connection that exploits the vulnerability described in MS04-11 report used the SMBD protocol on a TCP connection, did not have a DCERPC interface identified, the identified transfer syntax was *8a885d04-1ceb-11c9-9fe8-08002b104860*, and tried to run a call to change a permission of a domain server. In both cases the Microsoft report says it is a Buffer Overflow vulnerability, allowing the remote execution of arbitrary commands.

The right hand side branches can be interpreted as follows: an attacker exploiting the vulnerabilities described in MS08-67 used the SMBD protocol on a TCP connection with a DCERPC interface *4b324fc8-1670-01d3-1278-5a47bf6ee188* and a transfer syntax *8a885d04-1ceb-11c9-9fe8-08002b104860* using the SRVSVC service attempted a *NetPathCanonicalize* call to convert a path into a canonical name.

Each of the decision trees generated for each of the eight data subsets was evaluated using a *k*-fold cross validation, with $k = 10$. After evaluation, the accuracy, the false positive rate (FPR) and the false negative rate (FNR) were calculated for each subset. The results of each subset showed average accuracy values greater than 75%, average FPR around 3% and average FNR around 14%. Only one set had a lower result with an accuracy of 45%, FPR = 9.74% FNR = 71.42%.

When the *k*-fold results are individually analyzed, it is observed that subset 3 had the worst performance. To understand this behavior the confusion matrix generated by the model was investigated. It was noted a great confusion between classes and some classes missing a mapping. For example, classes *EMPTY*, *RemoteCreateInstance*, *RemoteActivation* and *NetAddAlternateComputerName* were not covered by the rules. Besides the confusion matrix, it was also made an investigation into the distribution of objects from subset 3. This analysis showed that: 1) *NetCompare*, *NetPathCanonicalize* and *NetShareEnumAll* classes all have the same number of objects; 2) the model was unable to properly separate the objects in their classes; and 3) when individually analyzed objects from different classes sometimes have identical features.

When browsing the tree nodes it can be seen that the algorithm was able to identify that the attacks to the SMBD protocol occurred in non-standard ports (ports ≥ 290). This raises the hypothesis that the attackers were seeking to compromise other systems or a system configured to not use the standard ports of the SMB service. This may indicate that the attackers have a knowledge of the network structure in which the honeypot was installed. One thing to stress is that in all confusion matrices the model has difficulty in properly separating the objects of the *NetPathCanonicalize* and *NetCompare* classes. When analyzing the objects marked in these two classes it can be seen that they have the same values defined in different classes, which makes it impossible to adequately separate them. New attributes should be added to increase the classifier accuracy.

4.2 Berlin

When analyzing the samples it was found that the target attribute, and the attributes *dcerpcbind_uuid*, *dcerpbind_transfersyntax*, *dcerpcservice_name*, *dcerpcserviceop_name*, and *dcerpcserviceop_vuln* contained a single class (*EMPTY*). The sample related to the first and second weeks have a total of 71.53% and 54.14% of the attacks targeted at the SMBD protocol. This reinforces the idea that there was a blockage between the attacker and the honeypot, preventing the continuation of the attack, making the honeypot a target only of port scans. Therefore, it was only possible to generate the decision trees for the third, fourth and fifth week. In this case three different trees were generated, one for each sample.

When browsing the tree for the third week it is noticeable that most information is targeted at the EMPTY class, where the honeypot was unable to obtain information about the attack. Unfortunately this sample did not bring any information that could

help the network administrator to protect the network. Class EMPTY was predominant in attributes *dcerpcbind_uuid* (96.80%), *dcerpbind_transfersyntax* (96.80%), *dcerpcservice_name* (97.40%), *dcerpcserviceop_name* (99.15%) and *dcerpcserviceop_vuln* (99.15%).

For the fourth week an interesting characteristic is observed: the *dcerpcbind_transfersyntax* attribute has value *8a885d04-1ceb-11c9-9fe8-08002b104860* and then the model divides the attribute into EPMAPPER and SMBD. From this point the honeypot is no longer capable of identifying the methods used by the attackers, but it notes that when using the SMDB protocol it is used the *NetShareEnumAll* as the call to the service. There are two hypotheses to such behavior: the first one is related to a variation or a new type of attack; and the second, and most probable one, that there was some type of communication block between the honeypot and the attacker.

The tree generated by the fifth week sample leads to the observation of a cascade of events that result in the exploration of vulnerability MS08-67. Eight DCERPC interfaces called some type of RPC service, but only *4b324fc8-1670-01d3-1278-5a47bf6ee188* led to a branching in the tree. This DCERPC interface represents 1.76% of the sample, and the largest class (EMPTY) occur in 58.28% of the cases. This tree lead to two different attack profiles. First, the attacker used the SMDB protocol in a *4b324fc8-1670-01d3-1278-5a47bf6ee188* DECRPC interface, using the SRVSVC service of the *8a885d04-1ceb-11c9-9fe8-08002b104860* transfer interface with the *NetPathCanonicalize* call to explore vulnerability MS08-67. The second profile has the same features, changing only the call to *NetShareEnumAll*, where the honeypot was incapable of associating with a Microsoft security bulletin.

The quantitative analysis showed average accuracy values greater than 99.53%, average FPR around 0.06%, and average FNR around 16.28%. Similar to the Paris data, an analysis of the confusion matrix showed that the model has difficulties in separating the *NetPathCanonicalize* and *NetCompare* classes. Again it was found identical objects in different classes.

4.3 Jacarei

The Jacarei dataset resulted in three different trees: the first one corresponding to the first week subset, the second one corresponding to the second week, and the third one corresponding to the third, fourth and fifth weeks. The first and third trees are quite simple, with one node and three leaves or one node and four leaves, respectively. These trees did not give us any conclusion about the dataset.

The tree for the second week was very similar to the tree for the fifth week of the Berlin dataset, with a branching from the *dcerpcbind_uuid* node until the *NetPathCompare* and *NetShareEnumAll* leaves.

The quantitative analysis showed average accuracy values greater than 69%, average FPR around 9.95%, and average FNR around 19.01%. Similar to the Paris data, an analysis of the confusion matrix showed that the model has difficulties in separating the *NetPathCanonicalize* and *NetCompare* classes. Again it was found identical objects in different classes.

4.4 Campinas

The 13 subsets sampled from the Campinas data generated three different types of trees: the first one for the weeks 1,2,3,4,6,8,9,11; the second one for the weeks 5 and 7; and the third one for the weeks 12 and 13.

The first tree has nine leaves from the dcerpcbind_uuid node and one *connection_protocol* node. The branches found inform the network administrator that the *4b324fc8-1670-01d3-1278-5a47bf6ee188* DCERPC interface is used by two different transfer syntaxes aimed at exploring the same vulnerabilities. In the other datasets single transfer syntax was used. In such case it is clear that the attacker is trying to explore the same vulnerability in different ways.

A similar behavior is found in the other trees. Differently from the other datasets that were collected in the same time period, here it can be clearly observed that the attackers are trying to explore means for which the honeypot is not ready to deal with.

The quantitative analysis showed average accuracy values greater than 60%, average FPR around 1.32%, and average FNR around 67.05%. Similar

to the Paris data, an analysis of the confusion matrix showed that the model has difficulties in separating the *NetPathCanonicalize* and *NetCompare* classes. Again it was found identical objects in different classes. It was also noted that the attacks in the tenth week were 100% targeted at the communication protocol XMPPCLIENT, but the honeypot was incapable of dealing with such attack.

5 EXTRACTING RULES FROM THE DECISION TREES

There are several intrusion detection systems in the market. The present paper uses the Snort [22] to illustrate the generation of rules for an IDS. Snort was chosen because it is capable of processing DCERPC information, the main type of information available in the datasets.

The analysis of the four datasets revealed similarities in the decision trees branching, sometimes changing only the target class or the leaf of a tree and, in other cases, the DCERPC connection interface. Thus, this branching was chosen to illustrate the extraction of rules for the IDS.

The sample tree can be observed in Figure 2. The analysis of the tree results in the following:

- Vulnerability explored (*dcerpcserviceop_vul attribute*): MS08-67 and *Not_Identified*;
- Protocol used: SMBD;
- DCERPC interface (attribute *dcerpcbind_uuid*): *4b324fc8-1670-01d3-1278-5a47bf6ee188*;
- Transfer syntax (attribute *dcerpcbind_transfer-syntax*): *8a885d04-1ceb-11c9-9fe8-08002b104860*;
- Service used (attribute *dcerpcservice_name*): SRVSVC;
- Service call used (attribute *dcerpcserviceop_name*): *NetPathCanonicalize e NetShareEnum*.

The rule generated, presented in Rule 1, has the following information obtained from the decision tree:

- **alert tcp**: The SMBD protocol Works with the TCP protocol.
- **[135,139,445,593,1024:]**: These are the doors used by the SMBD protocol.
- **(msg:"Vulnerability MS08-67 Attack")**: Message to be registered in the IDS logs based on the vulnerability identified by the honeypot.
- **dce_iface: 4b324fc8-1670-01d3-1278-5a47bf6ee188**: Information obtained from the *dcerpcbind_uuid* attribute.
- **dce_opnum: 32,15**: Information obtained after a research about the DCERPC interface, the transfer syntax, the service used, and the call to the service used [23].
- **reference:bugtraq,20081026**: Information obtained from the MS08-67 register of BugTraq.
- **reference: CVE, 2008-4250**: Information obtained from the MS08-67 register of CVE.

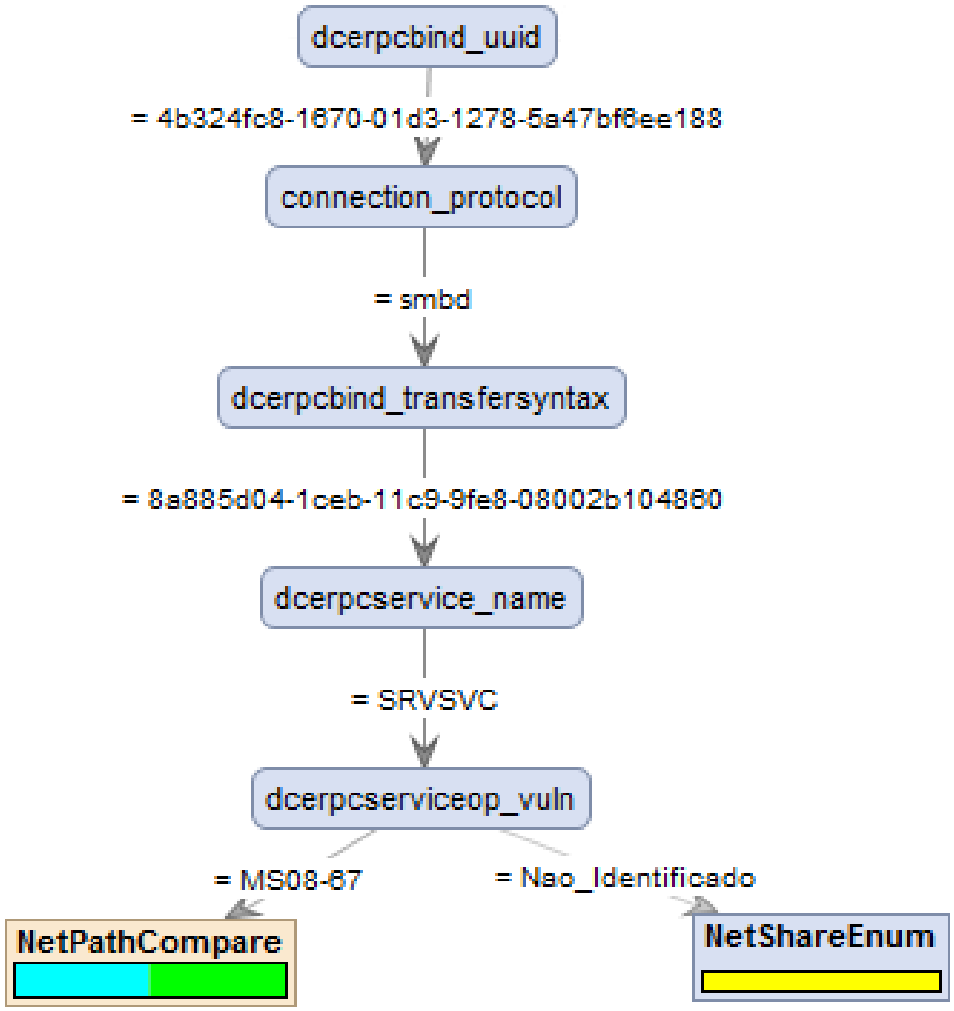

Figure 2: Sample tree obtained from the four datasets.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET [135,139,445,593,1024:] \
(msg:"Vulnerability MS08-67 attacked"; flow:established,to_server; \
dce_iface: 4b324fc8-1670-01d3-1278-5a47bf6ee188; dce_opnum:32,15; dce_stub_data; \
byte_jump:4,-,relative,align,dce;byte_test:4,>,256,4,relative,dce; reference:\
bugtraq,20081026;reference: CVE,2008-4250; classtype:attempted-admin; sid:1000068;)
```

Rule 1: Sample rule that can be generated by the Snort.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET [:290] \
(msg:" Not identified vulnerability "; "; flow:established,to_server; \
dce_iface: 4b324fc8-1670-01d3-1278-5a47bf6ee188; dce_opnum:15; dce_stub_data; \
byte_jump:4,-,relative,align,dce;byte_test:4,>,256,4,relative,dce;; classtype:at-
tempted-admin; sid:1000068;)
```

Rule 2: IDS sample rule generated by the Snort for the tree obtained by the subset of day 3 of the Paris dataset with ports less than 290 and opnum = 15.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET [290:] \
(msg:"Not identified vulnerability "; flow:established,to_server; \
dce_iface: 4b324fc8-1670-01d3-1278-5a47bf6ee188; dce_opnum:4; dce_stub_data; \
byte_jump:4,-,relative,align,dce;byte_test:4,>,256,4,relative,dce;; classtype:at-
tempted-admin; sid:1000068;)
```

Rule 3: IDS sample rule generated by the Snort for the tree obtained by the subset of day 3 of the Paris dataset with ports greater than 290 and opnum = 4.

The other parameters used to generate the rule are standard and can be altered based on the administrators needs. In the example provided, the only parameters that could be altered when generating the rules are *dce_opnum* and *dce_iface*. This modification is because the attacks use another DCERPC connection interface (attribute *dcerpcbind_uuid*) or the modification of the target class parameter *dce_opnum* (attribute *dcerpcserviceop_name*).

The tree for the third and fourth weeks of Berlin cannot be used to extract decision rules because the values in the nodes are EMPTY. The trees for the first, third, fourth and fifth weeks of Jacarei cannot be used because they have a single node and the leaves.

Rule 2 presents a possible rule using the information obtained from the third day of the Paris dataset, whilst Rule 3 presents a potential rule for the branching greater than or equal to 290. The data that were altered are detached in bold.

6 DISCUSSION AND FUTURE WORK

The honeypot is a strategic tool for the network administrator to identify potential treats to the network assets. They can generate a vast amount of data, turning the analyses slow and complex. Data mining techniques can be used to automatically process these data and extract useful information for network security.

The present paper provided a statistical analysis of four honeypot datasets, collected in different locations and time periods, and showed that, despite that, there are several similarities among the data collected. For instance, it was noted that more than 90% of the attacks are directed at Windows communication protocols, more specifically at the RPC protocol, where the vulnerability MS08-67 was explored.

It was also noted that Berlin and Campinas data have unique features. Most Berlin attacks do not use protocol RPC, raising the hypotheses that these attacks did not use RPC calls or there was some type of blockage between the honeypot and the attacker. Campinas presented a migration of attacks to services like XMPP and SIP. Thus, the administrator must worry not only about its Windows assets, but also others. Furthermore, for the Campinas data other DCERPC interfaces were explored and, with them, new forms of compromising the network.

After this standard descriptive statistics of the data, decision trees were built from the data and used to extract IDS rules. The following conclusions could be taken: even though the data were collected at different locations and time periods, some attacks are similar and explored the same failures; some anomalous behaviors were identified with the access to a RPC service in standard ports (showing that the attacker had a previous knowledge of the network); and the trees presented specific services of the RPC transfer syntaxes. Quantitatively, the average accuracy of the classifier rules generated was always greater than 60% and in one case reached 99.53%.

As future works it is possible to detach a detailed study of the *bitstreams*, together with the *emuprofiles* table; the development of an automatic system to generate the decision trees and extract the IDS rules; and the comparison with other classifiers and DT extraction algorithms.

7 ACKNOWLEDGEMENTS

The authors thank Fapesp, CNPq, Capes and Mackpesquisa for the financial support.

REFERENCES

1. Cisco System. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update. [Online] Fev de 2013. [Citado em: 20 de 11 de 2014.] http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html.
2. Kaspersky Lab. Informe de Kaspersky Lab: Evaluacion del nivel de amenaza de las vulnerabilidades en programas. *Viruslist.com*. [Online] Fev. de 2013. [Citado em: 20 de 11 de 2014.] http://www.viruslist.com/sp/analysis?pubid=207271202.
3. Ponemon Institute. The Impact of Cybercrime on Business: Studies of IT practitioners in the United States, United Kingdom, Germany,. *Ponemon*. [Online] May de 2012. [Citado em: 20 de 11 de

2014.] http://www.ponemon.org/local/upload/file/Impact_of_Cybercrime_on_Business_FINAL.pdf.

4. Denning, Dorothy. *An Intrusion-Detection Model.* 2, s.l. : IEEE, Feb. de 1987, IEEE Transactions on Software Engineering, Vols. SE-13, pp. 222-232. ISSN: 0098-5589 DOI: 10.1109/TSE.1987.232894.
5. Cohen, Fred. The Deception ToolKit. *Risks Digest.* 19, March de 1998.
6. Provos, Niels. *A Virtual Honeypot Framework.* s.l. : USENIX Security Symposium, 2004.
7. Provos, Niels e Holz, Thorsten. *Virtual Honeypots: From Botnet Tracking to Intrusion Detection.* [ed.] Addison-Wesley. s.l. : Pearson Education, Inc., 2007. p. 440. Vol. 1.
8. Wicherski, Georg. Medium interaction honeypots. *German Honeynet Project.* 2006.
   The Honeynet Project. *Know Your Enemy: Learning About Security Threats - The Honeynet Project.* Second. s.l. : Pearson Education, Inc, 2004.
9. Dionaea Catch Bugs, Dionaea Catch Bugs. [Online] April de 2013. [Citado em: 20 de 11 de 2014.] http://dionaea.carnivore.it/.
10. Van Rossum, Guido. "Python Programming Language." *USENIX Annual Technical Conference*. Vol. 41. 2007.
11. Deering, Stephen. Internet protocol, version 6 (IPv6) specification. *Request for Comments.* [Online] IETF.org, 1998. [Citado em: 20 de 11 de 2014.] https://tools.ietf.org/html/rfc2460.
12. Dierks, Tim. The transport layer security (TLS) protocol version 1.2. *Request For Comments.* [Online] IETF.org, 2008. [Citado em: 20 de 11 de 2014.] https://tools.ietf.org/html/rfc5246.
13. Buffington, Jason. Microsoft SQL Server. *Data Protection for Virtual Data Centers.* s.l. : Wiley Publishing, Inc., 2010, pp. 267-315.
14. Rosenberg, Jonathan; Schulzrinne, Henning; Camarillo, Gonzalo; Johnston, Alan; Peterson, Jon; Sparks, Robert; Handley, Mark; Schooler, Eve. SIP: session initiation protocol. *RFC 3261, Internet Engineering Task Force.* [Online] HJP.at, 2002. [Citado em: 20 de 11 de 2014.] http://www.hjp.at/doc/rfc/rfc3261.html.
15. SQLite. SQLite. *SQLite.* may de 2013.
16. Microsoft Corporation. How RPC Works. *Microsoft TechNet.* [Online] 28 de 03 de 2003. [Citado em: 15 de 12 de 2014.] http://technet.microsoft.com/en-us/library/cc738291(v=WS.10).aspx.
17. Markey, Jeff e Atlasis, Dr. Antonios. SANS Intitute Infosec Reading Room. *SANS Institute Reading Room.* [Online] 05 de 06 de 2011. [Citado em: 20 de 11 de 2014.] http://www.sans.org/reading-room/whitepapers/detection/decision-tree-analysis-intrusion-detection-how-to-guide-33678.
18. Mierswa, Ingo; Wurst, Michael; Klinkenberg, Ralf; Scholz, Martin; Euler, Timm. *YALE: Rapid Prototyping for Complex Data Mining Tasks.* [ed.] Lyle Ungar, New York, NY, USA : ACM, 2006. KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 935-940. ISBN: 1-59593-339-5 DOI: http://doi.acm.org/10.1145/1150402.1150531.
19. Microsoft Corporation. [MS-RPCE]: Remote Procedure Call Protocol Extensions. *Microsoft Developer Network.* [Online] Microsoft Corporation, january de 2013. [Citado em: 20 de Novembro de 2014.] http://msdn.microsoft.com/en-us/library/cc243560.aspx.
20. Quinlan, John Ross. *C4.5: Programs for Machine Learning.* s.l. : Morgan Kaufmann Publishers, Inc, 1993.
21. Roesch, Martin. *Snort: Lightweight Intrusion Detection for Networks.* 1999. LISA. Vol. 99, pp. 229-238.
22. Microsoft Corporation. 3.1.4 Message Processing Events And Sequencing Rules. *Microsoft Developer Network.* [Online] 2014. [Citado em: 15 de 12 de 2014.] http://msdn.microsoft.com/en-us/library/cc247234.aspx.
23. Post it yourself. *Carnivore News.* [Online] 08 de 12 de 2009. [Citado em: 20 de 11 de 2014.] http://carnivore.it/2009/12/08/post_it_yourself.