Attacks Detection Based on IP and TCP Protocols Violations

Norma Rodrigues Gomes and Luiz Antonio da Frota Mattos

Abstract - One of the biggest challenges in the network intrusion detection field is the limitation imposed by the use of well-known attack signatures that disable the previous detection of new attacks. This work presents a packet analysis methodology for detecting anomalous behaviors, not based on attack signatures, but on verifying whether the network protocols are being violated, and on the content of the respective headers. The biggest benefit of this methodology is the possibility of detecting anomalies or inadequate behaviors that can correspond, totally or partially, to variations on well-known and unknown attacks.

Key Words: attack signatures, intrusion detection, protocol violation, unknown attacks.

I. INTRODUCTION

The need for network security is growing and evolving with the increasingly open and interconnected nature of current computer networks. The use of computer networks coupled with large technological advances has ushered in critical security-related problems. A security mechanism gaining in popularity is the Intrusion Detection System (IDS), which is a component-based structure that secures network environments against unskilled and sophisticated attacks. The majority of the IDS framework is based in attack signatures, which describe known behavior standards that are considered suspect and constitute security problems [2]. One of the biggest challenges in the area of network intrusion detection is the limited ability to detect attacks when using signature based systems, in which there is no possibility of new attacks detection that use unknown signatures [6].

Our proposal showcases a packet analysis approach with the purpose of detecting anomalous behaviors, based not on attack signatures but rather on verifying network protocol violations and on the content of the respective headers. Using this type of analysis allows a user to detect certain known attacks, or some variation therein, as well as new attacks with no previous violation history. Specifically, our packet analysis methodology for detecting attacks is based on the specifications of IP and TCP protocols. Section 2 presents information about the problem caused by packets that violate network protocols. Section 3 shows the consistency of the proposed packet analysis, defining the employed methods. Section 4 compares the method based on protocols violations with the one based on attack signatures. Section 5 synthetizes the obtained conclusions and presents future ways for improving the proposed work.
II. Problem Description

In computer networks, the interactions among the various hosts that constitute the environment is conducted with protocols, with mandatory specifications. However, it is possible to mount attack packets that violate the communication protocols, causing unexpected reactions on the packet’s receptor side. A computer network is vulnerable to suffering various types of attacks with multiple purposes, including network scanning to identify any services or operational systems used, and Denial of Service (DoS) to interrupt legitimate access to services, servers or other resources. The IDS was developed as a security mechanism to detect network attacks. Most of the IDS infrastructure is based on attack signatures, which detect intrusions, analyze their behaviors and generate attack-descriptive rules. Thus, whenever a particular behavior reoccurs, the respective attack is detected and the network is shielded.

Although valuable, signature-based IDSs are incapable of detecting attacks that have not been encountered previously. This limitation in IDS technology represents one of the greatest challenges in the area of intrusion detection. One approach to solving this matter, postulated by Allen in [1], that warrants further examination is to allow the system to understand normal behavior patterns and detect any divergences. Such an approach consists of determining what constitutes normal behavior inside the networks, hosts or applications, and flagging any suspicious activity that does not reflect what it is expected. When observing signature attacks, one can see that some of them represent violations to mandatory network protocols (for example, TCP/IP). Thus, it is important to understand the standard behavior of the involved protocols in the network data traffic in order to detect if this behavior attends to the specifications of the protocol in question. Such specifications are defined in the Request For Comments (RFC) documents, which describe expected standards for individual protocols [8]. Therefore, when verifying the protocol and its specifications, as defined in the given RFCs, some anomalous behaviors can be detected, rendering unnecessary the use of some attack signatures and enabling the detection of some variations of existing signatures.

III. Packet Analysis for Protocol Violation Detection

Our proposal consists of conducting a packet analysis for detecting behaviors that diverge from those outlined by the protocol. Toward this goal, we have defined the work scope, in terms of protocols and analysis types, methods to formalize the protocol rules of expected behavior and an implementation of this packet analyzer, at the archetype level, as detailed below.

A. Packet Analysis Scope

Given the popularity of the IP/TCP protocols, the scope defined for accomplishing the packet analysis are the IP and TCP headers. Thus, as we will work with TCP, a trustworthy protocol guided by connection, the packet analysis will be conducted in two stages:

- Stage 1 – Stateless Inspection
- Stage 2 – Stateful Inspection

In the Stateless Inspection (Stage 1) the examination of individual packets ensues. That is to say, the analysis is restricted to the content of the header’s fields of one specific packet, without concern for keeping track of the connection TCP/IP state.

In the Stateful Inspection (Stage 2) the examination of a sequence of packets is conducted, in which the analysis embraces the header’s data of more than one packet, allowing one to detect standards of behavior among packet sequences and to create correlations. This article emphasizes the methods and results related to Stage 2 of the proposed packet analysis.

B. Stateless Inspection

The first stage of the TCP/IP header’s analysis is conducted under two aspects:

- Syntactic analysis – in which the structure of the header is verified, and whose fields are the basic unit of formation; and
- Semantic analysis – in which the meaning of the header is verified in terms of the values of its fields.

Thus, for a header to be considered valid it is necessary that it is valid syntactically and semantically. In other words, the header must obey the syntactic rules that determine which field chains can form headers, and the semantic rules that determine if the header has meaning based
on the field values. From the study and analysis of
the content of the RFC documents (RFC 791, 793,
1323 and 2018), and from some considerations
regarding these documents [7], we defined 31
syntactic and semantic rules for the IP and TCP
headers. Once the rules of the Stateless Inspection
aspect were defined, the next step was to apply a
logical model to create a theory corresponding to
the set of mandatory rules that ensure that the IP
and TCP headers are considered valid.

1) Application of a Logical Model to Formalize
the Header Rules

In order to define the syntactic and semantic
rules established for the IP and TCP headers
logically, we constructed a formalism applying
first-class logic. The rules of the IP and TCP
headers are interpreted from the construction of
a first-class theory. This theory is formed by a set
of first-class logic formulas generated from the
rules defined for the headers.

After formalizing the syntactic and semantic
rules of the headers, the main formula was
constructed:

"∀x ∀y (packet_hd_val(x,y) ← rules_ok(x,y))".

Whereby "x and y are, respectively, valid IP
and TCP headers of a package if x and y obey
all rules". The expression "rules_ok(x,y)" will be
true if the conjunction of all the corresponding
formulas to the rules of the headers are true.
Thus, the predicated "packet_hd_val" will be used
to consult if a given set of values represents a
package with valid IP and TCP headers.

2) Archetype for Accomplishing the Stateless
Inspection (RECAB)

The implementation of the rules that define
the validity of IP and TCP headers, formalized
through a set of formulas of first-class logic, is
accomplished by programming in logic. One of
the great advantages of programming in logic,
as compared to conventional programming, is
that the task of the programmer is summarized,
practically, to the specification of the problem that
must be solved, since the logical languages can
be seen simultaneously as languages for formal
specification and languages for the computer
programming. Therefore, the created theory to
formalize the rules of the headers, which consists
of a set of formulas of the first-class logic
corresponds to a program in logic. Thus, the
archetype to test the referring rules to IP and TCP
headers, called RECAB, was developed in Prolog.
The essence of RECAB consists of the analysis of
individual packages in which it is verified if the
referring rules to IP and TCP protocols are obeyed.
In the case of a protocol violation, the package is
identified and the detected errors are listed. To
ensure that the packages, in windump format,
are processed by the archetype, a conversion of
the same packages in Prolog terms is conducted,
such that they are represented by atomic formulas
[3]. The original idea, referring to the processing
of the packages for RECAB, consisted of placing
the packages under analysis as part of the proper
Prolog program. Because of the great number
of packages that had been analyzed, there was
a danger of overloading the program. Thus, to
prevent compromising program execution, the
packages, defined as Prolog terms, were stored in
a text file. This file, whose content corresponds to
the net traffic collected for analysis, is passed as
in-parameter for RECAB, which makes a sequential
reading of this file, analyzing each read package,
and telling in the case of a protocol violation,
until the end of file.

B. Stateful Inspection

In the same way as the Stateless Inspection,
the scope of the Stateful Inspection consists of
IP and TCP headers, having as emphasis the TCP
connection. According to a previous study [8], the
more generic behaviors can be analyzed when the
scope is the TCP protocol, referring to:

1) Connection Start;
2) Data Transfer; and
3) Connection End.

Following this baseline schematic, a project
for studying the behavior of a TCP/IP connection
during the Stateful Inspection stage is proposed
in this work. This project is known as the ESTCON
(Study of Connections), and represents the
location at which the behavior will be verified
if a connection was established with a three-
way handshake, if the data transfer occurred
after the establishment of the connection and
if the connection was completed with flag FIN
or RESET. The ESTCON studies the behavior of a
given sequence of packages, in order to identify
any violation of the expected behavior of the
protocol that represents some type of attack. The
implementation of the ESTCON supports itself in
the use of a database to correlate the packages.
The text file with the packages represented for
Prolog terms, used in the Stateless Inspection,
was imported to a database, originating with a table whose structure is composed for all fields of IP and TCP headers. The following describes what was verified in the proposed project for accomplishing the Stateful Inspection, the ESTCON.

1) Classification of the packages by Connection

As the Stateful Inspection has as its central focus the TCP connection, the first phase that composes the ESTCON consists of classifying the packages according to the connection to which they belong. This classification is conducted through the “pair socket” that consists of the source and destination addresses of the IP header, and of the source and destination doors of the TCP header. This identifies a unique connection in the net by grouping the packages. Once the classification of the packages by connection is concluded, a process of verification is initiated, consisting of four items for each set of packages identified as being a connection, which is described in the following section.

2) Items Verified in the ESTCON

The basis of the ESTCON consists of verifying four items referring to the most generic behaviors of TCP/IP connection, as defined previously [8], which are identified herein in Table 1. Item 1 it is verified if the establishment of the connection occurred through the process of a three-way handshake. Item 2 is verified if it had the data transferred in some direction with the respective confirmation of receiving inside the connection. Item 3 is verified if it requested the end of the connection through the FIN flag. Finally, item 4 is verified if the RST flag was used at some moment inside the connection. Thus, with the values obtained in the four items the system can identify what occurred inside each connection, according to Table 1, which presents the 16 possible combinations of the four items, corresponding to a table-truth of the ESTCON items. The content of the column “Meant” is inferred from the combination of the values of the four items, whose bigger objective is to summarize what happened inside the set of analyzed packages.

The term “Cold Start”, used in Table 1, indicates the situation in which a connection was established before the net traffic began to be monitored [5]. Analogously, the term “Cold End” will be used to describe a connection that was not finished when the log file was terminated.

3) Status Definition proposed in the ESTCON

The value of “STATUS”, as defined in Table 1, is based on the study of the standard behavior of the IP and TCP protocols, as well as in some attack techniques that reveal themselves by violating such standards. The “NORMAL” status implies that the set of analyzed packages presents an acceptable or expected behavior by TCP/IP protocols. The “SUSPECT” status means that the set of analyzed packages does not present a standard behavior, but is not considered suspicious. The “VERIFY” status means that only on the basis of the values of the four items, it is not possible to identify the true status of the analyzed packages. This requires the system to verify some questions (outlined below).

In the case of code 8 in Table 1, the system has verified that the last package, inside the set under analysis, is situated in the end of log. In the positive case, the status of the connection is considered “NORMAL” assuming the closing of log occurred before the register of the posterior phases (transfer and end). Otherwise, the status is set to “ABNORMAL”. For the effect of the ESTCON, the positioning of the package inside of log is given by the total number (N) of packages stored, dividing the log file into three equal parts (beginning, middle and end). Each one contends (N*33,33%) packages. The cases of codes 13 or 14 are verified if the first package, inside the set under analysis, is situated at the beginning of log. In the positive situation, the status of the connection is considered “NORMAL” assuming “Cold Start” occurred. Otherwise, the status is set to “ABNORMAL”. In the case of code 15, the only example of an RST flag, the system verified which packages had provided an origin to the RST packages. Based on these packages, the system verified if their <End_Origem> fields repeated in other connections also classified with code 15. In cases where the number of repetitions exceeds a pre-defined threshold (default value 10), the status of the connection is considered as “SUSPECT” because this excess of repetitions means that the same <End_Origem> sent packages to diverse destinations, characterizing a SCAN coming from the <End_Origem> address. In case a SCAN is not configured, the number of packages that originated RST packages is verified by analyzing groups of packages. In the event that this number
is superior to a threshold (default value 300), the status of the connection is considered to be "SUSPECT", configuring an attack of denial of service (DoS) through FLOODING of the destination door with innumerable requests [8]. In the event that a FLOODING is not configured, the system verifies which flag was used to generate the RST package. If the flag=SYN, the status is considered as "NORMAL", assuming that it was an attempt of a non-accepted connection. If the flag=ACK, the system verifies whether the package is situated in the beginning of log, in which the status is considered as "NORMAL", assuming that it occurred as a "Cold Start" with an abrupt ending. In the negative case, the status is said to be "SUSPECT", assuming that there was an attempt of recognition attack with an ACK flag without a pre-established connection. Finally, if the flag=FIN or any another unknown value, the status is considered as "SUSPECT" by the same reasoning of probable attempt of a recognition attack.

IV. Obtained Results

Various tests with the RECAB and the ESTCON were made involving real net trafficking, and mounted packages with anomalies simulating various forms of attack. For the purpose of comparison, the same tests were made with Snort, an open-source IDS for public-based attack signatures. All tests were accomplished in the Windows 2000 environment, with an Ethernet interface of 100 Mbits/sec. This was used for running the Snort for Windows version 1.9.1 (http://www.silicondefense.com/support/window). To follow, the accomplished tests are presented.

A. Tests with RECAB

As this article emphasizes the Stateful Inspection, the results of the tests with the RECAB will be described in a resumed form, emphasizing only the most significant points. The RECAB was

<table>
<thead>
<tr>
<th>Connection Establishment</th>
<th>Data Transfer</th>
<th>FIN Flag</th>
<th>RST Flag</th>
<th>MEANT</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item 1</td>
<td>Item 2</td>
<td>Item 3</td>
<td>Item 4</td>
<td>(summary of what happened inside each connection)</td>
<td>(valuation)</td>
</tr>
<tr>
<td>1</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Three complete phases</td>
</tr>
<tr>
<td>2</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Three complete phases</td>
</tr>
<tr>
<td>3</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Abrupt Interruption (possible fail of operation)</td>
</tr>
<tr>
<td>4</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Data Transfer not finished (Cold End)</td>
</tr>
<tr>
<td>5</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Starts and Ends the connection, without data transfer</td>
</tr>
<tr>
<td>6</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Starts and Ends the connection, without data transfer</td>
</tr>
<tr>
<td>7</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Abrupt Interruption before beginning of data transfer (possible fail of operation)</td>
</tr>
<tr>
<td>8</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>Connection not used yet</td>
</tr>
<tr>
<td>9</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>No connection establishment (Cold Start)</td>
</tr>
<tr>
<td>10</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>No connection establishment (Cold Start)</td>
</tr>
<tr>
<td>11</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Cold Start and abrupt end</td>
</tr>
<tr>
<td>12</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Cold Start and Cold End</td>
</tr>
<tr>
<td>13</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>It is only showed the end of connection</td>
</tr>
<tr>
<td>14</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>It is only showed the end of connection</td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Answer to a not accepted request</td>
</tr>
<tr>
<td>16</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>Unknown</td>
</tr>
</tbody>
</table>
skillful at identifying violations of protocol rules by verifying the rules in each package. This characteristic allowed the detection of attacks that combined more than one signature, referring to protocol violation, in one unique package, a fact that was not observed in the analysis accomplished by Snort. Moreover, for some protocol violations, Snort did not generate any alert. However, it must be stated that the archetype presented overhead in the treatment of large numbers of packages. Thus, the programming of our system must be adjusted to handle large data sets in order to be useful as a practical application. This may require using a different programming language.

B. Snort Signatures versus IP/TCP Protocol Violations

In addition to the RECAB tests, a survey of the signatures used by Snort was conducted. This survey verified that few signatures constituted a protocol violation, whereas the majority performed searches for suspect strings in data in the packages. However, even for a minority, it was verified that some rules treated in RECAB could represent a given set of signatures of Snort, arriving in a ratio of 1:12 (one rule for twelve signatures). Thus, through a pre-analysis, using RECAB, it is possible to reduce the number of signatures in an IDS, that, depending on the ratio, would increase the efficiency of the detection mechanism.

C. Tests with ESTCON

Initially, tests with 100,000 collected packages of real TCP/IP net traffic were conducted, whereby a situation with a “SUSPECT” status, code 15 of Table 1, was identified. It was verified that more than 600 SYN packages had been sent from the same origin address (host-a), to 66 different destination addresses in diverse doors, which replied with RST packages. Such a situation was characterized by ESTCON as a SCAN from “host-a”, where the intention would be to identify which doors were active or not in diverse hosts. However, when the net administrator was contacted, they verified that the host-a corresponded to a Network Address Translation (NAT) server whose IP address is used by all hosts of the internal net for accessing the Internet. Thus, we concluded that it was not a scan leaving from the same origin, but rather from several internal hosts trying to access diverse destinations, generating, therefore, a FALSE-POSITIVE.

To duly test ESTCON and compare it to Snort, some sequences of the packages simulating the attacks had been generated, using the tools NMapWin (www.insecure.org) and Engage Packet builder (www.engagesecurity.com). The results are presented in Table 2.

When using the NMapWin tool, we observed that it always sent in its “scans” some package with a type of anomaly (flag NULL or flag ACK with <Num_Ack>= 0 or flag URG-PSH-FIN), not restricting to send packages with flags in accordance with the scan respective denomination. These “scans” were sent from one origin for diverse doors in one host with the objective to identify the operational system used in the destination host (fingerprinting). The SYN-flood attack consisted of sending 800 SYN packages to one door in the same host, with the objective of generating a DoS in the destination door.

Thus, it can be verified that Snort, although having generated alerts for three scans, generated

<table>
<thead>
<tr>
<th>Attack</th>
<th>ESTCON</th>
<th>Snort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scan SYN Stealth (NMapWin)</td>
<td>Scan from &quot;host-y&quot; - Flags SYN, ACK, NULL, URG-PSH-FIN (Attempts: 1616)</td>
<td>NMAP FINGERPRINT (stateful) STEALTH ACTIVITY (XMAS scan) STEALTH ACTIVITY (NULL scan)</td>
</tr>
<tr>
<td>Null Scan (NMapWin)</td>
<td>Scan from &quot;host-y&quot; - Flags NULL, ACK, URG-PSH-FIN (Attempts: 1881)</td>
<td>NMAP FINGERPRINT (stateful) STEALTH ACTIVITY (XMAS scan) It generated more than a thousand of alerts: STEALTH ACTIVITY (NULL scan)</td>
</tr>
<tr>
<td>ACK Scan (NMapWin)</td>
<td>Scan from &quot;host-y&quot; - Flags ACK, URG-PSH-FIN (Attempts: 1876)</td>
<td>It generated three alerts: STEALTH ACTIVITY (XMAS scan)</td>
</tr>
<tr>
<td>SYN-flood (Engage Packet builder)</td>
<td>Denial of Service (flooding) to “host-x: door-x” from &quot;host-y&quot; - Flags SYN (Attempts: 800)</td>
<td>&lt;no-alert&gt;</td>
</tr>
</tbody>
</table>
them as a result of the anomalous packages sent by NmapWin. Therefore, with the exception of the Null Scan, the conclusion that we can draw is that Snort did not detect the others scans and neither did the SYN-flood.

In ESTCON, the “SUSPECT” status was attributed to the four situations (Table 2). In the three first cases, the scan from “host-y” was detected, identifying the flags used in the sweepings and the number of packages sent. In the case of SYN-flood, the system identified the destination and the origin of flooding, the flag used and the number of packages sent.

D. Comparisons between ESTCON and a System of Reconstruction of TCP/IP Sessions (RECON)

RECON [4] is a system that allows one to reconstruct and to track the state of TCP/IP sessions using packet headers. Moreover, RECON implements some routines to carry through applications in intrusion detection. The objective of one of these routines is to accomplish a Detection of Host Scan based on sessions reconstructed for RECON. In our present study, this routine was compared with the ESTCON scan detection. RECON verifies the number of accessed hosts from the same IP using a list of TCP sessions created from the traffic analysis, where the destination door must be the same for all accessed hosts. In ESTCON, this same verification of the number of hosts is completed only where there is no established TCP session (code 15 of Table 1). In this scenario, it is not necessary that the destination door is the same, as only the “connection” (pair socket) is different. Thus, in the case of sweepings in a unique host in all doors (1 to 65535), RECON would not detect this malicious behavior. In ESTCON, as the doors vary, the packages are grouped in different “connections”. In this scenario, ESTCON would detect a scan leaving from one host with different destinations.

A disadvantage of this application is that RECON also analyzes TCP sessions established in the scan detention, but this does not occur in ESTCON. An example of real data, described previously [4], presents an activity in which one host, “host-x”, initiated sessions with several other hosts in the 25/tcp door that continued for over 40 consecutive hours, with peaks of up to 807 accesses to distinct hosts. Such a situation would not be detected by ESTCON, but was detected by RECON, providing further evidence that host-x was being explored and used to generate SPAM.

V. Conclusions

The proposed analysis of packages allowed us to render a formal definition of the IP and TCP specification protocols and create a correlation of packages that could efficiently conduct detection of some types of attacks and study the behavior of the TCP/IP network traffic. The biggest advantage observed in verifying the violation of protocols herein was that attacks generated by protocol anomalies would be avoided. Thus, variations of attacks and unknown attacks, that involve protocol violation, could be detected without the pre-existence of a signature.

In future studies, the rules of RECAB could be implemented as one plugin of Snort [9]. ESTCON could be turned into a study on the false-negative tax by applying the inferences presented in Table 1. A last suggestion is to increase the scope of the analysis of packages by including other protocols, such as ICMP and UDP.

References


Norma Rodrigues Gomes is Bachelor in Computer Science by the Federal University of Goiás - UFG, Brazil (1995) and Master in Computer Science by the University of Brasília – UnB, Brazil (2002), in the field of Intrusion Detection. She works as Federal Criminal Expert at the Criminalistic National Institute of the Brazilian Federal Police Department (www.dpf.gov.br), since 1999, with computer crimes. Currently, the major fields of study are logic and information science. Her address is:
National Institute of Criminalistics– Federal Police Department (DPF)
Setor Policial Sul – 70.610-200 – Brasília – DF – Brazil
Contact: norma.nrg@dpf.gov.br

Luiz Antonio da Frota Mattos is Bachelor in Mathematics by the University of Brasília – UnB, Brazil (1966), Master (1973) and Doctor (1975) in Mathematics by Rutgers - The State University of New Jersey - R.U., United States, and Pos-Doctor in Computational Security by Rutgers - The State University of New Jersey - R.U., United States (1988) and by University of Nebraska - U.N., United States (1995).
He works as professor at the Computer Science Departament of the University of Brasilia – UnB (www.cic.unb.br). His major field of study is Computational Security. His address is:
Computer Science Departament – University of Brasilia (UnB)
Campus Universitário Darcy Ribeiro – 70.910-900 – Brasilia – DF – Brazil
Contact: frota@unb.br