Annales UMCS Informatica AI 6 (2007) 185-194

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

TCP/IP traffic patterns: attacks, errors, steganography or normal behaviour?

Marta Rybczyńska*

Faculty of Electronics and Information Technology, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warszawa, Poland

Abstract

This paper presents the results of research on the way the network security is affected by the current state of TCP-IP protocol suite behaviour. A number of examples of possible issues are presented. When discussing classes of such issues, it is pointed out that security is affected by the existence of not only incorrect implementations, but also differences in implementations that can be used for various purposes.

In the main part of the paper an analysis of the traffic collected on an Internet backbone link from the year 1999 up to 2006 is presented. The results show that the predicted behaviour can be observed in the real-world traffic. The differences between the measurement results and the theory are analysed, with a more in-depth look into a number of patterns and the changes of the patterns between the traffic collected in different years. In addition, an operating system detection tool is used to estimate the operating systems used by the nodes. Then the estimation is compared with anomaly patterns and the conclusions are presented. After analysing the findings, the pros and cons to different possible explanations of the observed patterns are presented, including flaws, attacks, various kinds of errors and steganography.

1. Introduction

The TCP/IP protocol stack takes its name from two protocols of the suite, TCP (Transport Control Protocol) and IP (Internet Protocol). The set is one of the most popular, if not the most popular one, protocol stacks currently used. Basic protocols of the stack, including IP and TCP, but also UDP (User Datagram Protocol) and ICMP (Internet Control Message Protocol), appeared in their first versions more than 20 years ago. Due to the long time from and because of the stack popularity (and the need for new features), they have evolved. The newest implementations, however, are required to work together with the oldest ones. During the time of the protocols evolution a number of new

* *E-mail address*: marta@rybczynska.net

features were introduced, some older ones became obsolete. Additionally, some required ones are currently rarely used. The evolution also increased complexity.

The protocols have different implementations. They range for very simple ones (from approximately 100 kilobytes of C language source code) to those that are complex and include all or nearly all features (two megabytes and more of source code size, like in Linux or BSD systems). Simple implementation support only a limited range of features, not even all that are, in theory, required. More complex ones, on the other hand, require much testing before deployment, as they have a very large number of special cases to handle. There is also a bigger chance for a bug, like a not handled special case.

The RFC (Request for Comments) documents defining, among others, the TCP/IP protocols, do not take care over all possible scenarios. They leave much to the person implementing the protocol. It leads to two things. The first one is that a special case may pass unnoticed and, as one of the possible results, it may not be handled correctly. The second possibility is that it will be noticed and implemented, but the way it is done, and also the behaviour, may differ from one implementation to another.

The causes presented above lead to the fact that two implementations of the same protocol may present different behaviour in many situations. In practice, such differences are observed. They are also used for different purposes.

The fact that the differences can be used in practice moves the issue from the protocol validation and verification point of view to the security-related one. It also leads to many questions, including those about the number of cases or situations that can be used for malicious purposes and the types of attacks possible using those features. Another question is if efficient protection methods do exist, given the fact that the protocols cannot be easily changed, just as the systems already deployed.

2. Previous work

The problems caused by the TCP/IP implementation issues have been investigated since the early days of the Internet. An early overview of such problems can be found in RFC 2525 [1]. The more current state of the implementations and their common problems were presented by Bellovin in [2].

One of the first applications of an implementation problem was the „Ping of death". An error in most TCP/IP implementations at that time led to a system crash when specially crafted packet of length longer than the theoretical maximum was received [3].

The ICMP Echo request and the Echo reply messages, because of their long data field, were used in a number of tools to pass data through a firewall or any

other security tool without being noticed. Two examples are project Loki [4,5] and an attack tool "Stacheldraht" analysed in [6].

Another widely explored topic is the method of IP identification numbers and TCP initial sequence numbers generation. For security reasons, the values should be unpredictable, with certain constraints, but easily predictable generators do exist. The systems and their generators were evaluated by Zalewski in [7] and [8]. The OpenBSD generator is presented in detail in [9].

Handley et al show how TCP/IP features can be used against IDSes in [10].

The field of operating system detection is probably the one where differences in the network stack implementation are most widely used. There are different approaches to the problem: passive and active.

Passive detection uses the packets that travel across the network (so they can also work off-line using saved traces). The detection patterns include properties of the packets sent in certain states, for instance when starting and closing a TCP connection. Such tools are very hard, or impossible, to detect. [11] provides a description of the methods used. An example of a passive fingerprinting tool is p0f.

Active tools, on the other hand, use carefully-crafted packets that lead to different responses from different systems. The popular program nmap is an example of such an approach [12]. The tools like nmap can be detected quite easily, as packets generated by them differ from those one expects to find in a network. It is also possible to protect from such a tool, as shown in [13].

Fisk et al [14] have looked into a number of traces in search for steganography in TCP/IP. Their results show that the predicted behaviour can be found. They present numbers and conclusions, but do not try to look into the patterns nor try to explain them using different possible reasons.

Medina et al in [15] have measured the adoption of new TCP/IP mechanisms. Apart from noticing that old implementations are widely used, they also mention a number of problems, like hosts dropping connections when uncommon or new features are tried. They raise a number of interesting points. However, their results cover in the main part only the biggest web servers (which are also likely to have sophisticated protection and load-balancing mechanisms), not end-user computers, and are limited to web browsing, so they may not be representative to the Internet as a whole.

3. Traffic and measurement

The measurement presented in this paper was made using IP version 4 traffic traces from MAWI (Measurement and Analysis on the WIDE Internet) [16], from an Internet backbone link between Japan and the USA. The trace files

came from the years 1999 up to 2006, every year from the same period of time (the last days of February and the first days of March), one file from each day.

A single trace file covers traffic from fifteen minutes to approximately one hour and consists of between two and more than eight million packets. The general statistics of the traces used in the research is presented in Table 1. As it is expected, the majority of packets use IP with TCP. UDP and ICMP are much less common. Other protocols (including IP version 4) are used by roughly 2% of the packets. The number increases over time, however.

Table 1. Number of packets, by protocol used, in the analysed trace files, by year

Year	IPv4	TCP	UDP	ICMP	Others-IPv4	Non-IPv4
1999	23 065 627	18 605 749	3 310 796	1 094 743	54 339	130
2000	23 054 869	16 173 181	5 190 319	1 623 698	67 671	188
2001	32 756 718	26 562 696	4 749 414	1 190 609	253 999	212 664
2002	54 103 070	44 791 521	7 079 641	1 379 663	852 245	610 596
2003	50 079 649	40 804 567	7 346 383	1 653 492	275 207	228 459
2004	42 920 859	38 999 609	2 899 761	1 040 368	64 865	105 921
2005	67 964 160	56 547 378	7 879 861	2 093 372	1 443 549	922 389
2006	73 618 505	59 382 628	11 021 736	1 242 883	1 971 258	1 794 437
Total	367 563 457	301 867 329	49 477 911	11 318 828	4 983 133	3 874 784

The traces have packet headers, but no contents. Also, the IP addresses were anonymized, which requires a separate analysis of each file. The lack of packet content also means that the data transferred in the packets of special interests cannot be investigated.

The traces were analysed using custom tools, which were used to calculate the number of packets having certain properties, like non-zero value in the reserved fields that should be zero in all packets. The parameters needed for analysis of the results were also calculated. An example of such value may be the number of packets with different TCP or IP options. The last versions of the tools used more than 100 different parameters.

All of the values were calculated without context. That speeds up the measurements, but leaves many possible issues not covered.

In the next step of the research, the passive operating system detection tool was used to look into the operating systems used by the systems generating anomalies and compare the results with the number of hosts using different OSes

in the complete trace file. The tool of choice was p0f [17]. Different factors may affect such tool's accuracy, but it may be a source of valuable information. Taking into consideration that there was no way to check if the detection was correct, the p0f results were considered as guidelines only.

4. Results

The results of the performed measurement differ from the theory in many ways. In this part of the paper the findings will be presented. While presenting the results, the separate traces will be explored in certain cases. Also the possible explanations of the observed behaviour will be shown.

The anomalies were found to be sent from approximately 3% of all IP addresses which seems to be a high number.

The rest of this section will consist of presentations of a number of chosen patterns found in the results. Due to the number of parameters and the number of dependencies between them it is not possible to present the complete results. Instead, a set of them has been chosen. Each pattern from that set will be discussed in detail, with possible causes and explanations.

4.1. Attacks

The first finding was that the number of events of the same type differs greatly between the trace files. When investigating that fact it was found out that such differences are usually caused by a small number of flows with a high number of anomalies.

One of such examples is connected with the Urgent Pointer field and URG bit of the TCP packet. The original purpose of the Urgent Pointer is to show the offset of priority (or urgent) data. That field is evaluated when URG bit is set to 1 [18]. The RFCs does not define the field's value when URG is 0, but it is customary to use the value of 0.

One of the flows is responsible for more than a half of TCP Urgent Pointer field being non-zero when URG bit is 0. When looking into that flow in more detail, it was recognized as a likely DoS (Denial of Service) attack or a very intensive port-scanning. At least three such flows were found in different trace files. As a side note, none of them comes from a day with a well-known, wide-range attack.

4.2. Probable implementation flaws

There are, however, more reasons for the Urgent Pointer being set when URG is 0. Such packets are distributed between nearly all trace files and start being more common from the year 2002. The total number of such packets is more

than 4 million which makes roughly 1.4% of all TCP packets observed. That can be compared with only 625 packets with the URG bit set.

There is a similar issue with Acknowledgement field and the corresponding ACK flag. The field points to the data offset received correctly by the receiver. The ACK is a set in more than 85% of the packets, but roughly 0.8% of packets have it zeroed and Acknowledgement field set to non-zero.

There may be different reasons for such behaviour, even if it is an intended one (taking into account that when it comes to network stack implementation, using random values is expensive [9]). Michal Zalewski in [19] throes some light into that issue. There seems to be a bug in certain Windows versions that causes Urgent Pointer to use the data from a different connection for that value. It is possible that the analysed data shows the results of that, or a similar problem.

4.3. Possible steganography

It has been already mentioned that the ICMP messages with the large data field can be used to transmit data without being noticed. The two messages with such data field, with a possible size of more than a kilobyte, are the Echo request and the Echo reply [20], which are also used by the popular diagnostic tool ping. The ICMP protocol specification states that the host should answer with the Echo reply after receiving the Echo request, and that is the only situation such a message should be sent.

The number of messages of both types in different years is shown in Table 2. The results clearly show that up to 2001 there were actually more Reply than Request messages. The traces examined do not provide the data field content, but it is likely that the higher number of Reply messages is caused by tools that use them to transfer data.

The observation that such tools prefer the Reply rather than Request messages is caused by the fact that firewalls tend to be configured in such a way, that they allow packets that are related to the actions of protected users. When the firewall (or IDS) works without ICMP flow tracking, it may let the incoming Echo reply message pass as a reaction to assumed earlier Echo request.

The lower number of the Reply messages in the later years may be connected to the fact of the growing popularity of stateful firewalls, which do not pass Reply messages if they have not recorded a Request.

It should be also noted that the high number of Echo reply messages in the traces from the year 2000 is caused by two files with approximately 1 million of such messages in total.

Table 2. Comparison of the number of ICMP Echo request and Echo reply messages, by year

Year	ICMP Echo request	ICMP Echo reply
1999	126 954	190 042
2000	95 364	1 342 600
2001	190 156	448 526
2002	448 172	314 280
2003	143 226	113 732
2004	533 844	19 353
2005	170 579	170 722
2006	303 459	165 080

4.4. Rare options

The TCP messages may use options. Some of them are widely used. Those shown in Table 3 are obsoleted, experimental or undefined.

Many of them, like Skeeter or Bubba, should not appear in the traces at all. The reason for their presence in traces is unclear. Those like Echo or Echo reply are obsoleted.

Their existence may be a result of software or hardware failure, but may also mean they have their use.

Table 3. The number of rare TCP options found in the traces

Option no	Option name	Number of occurrences
6	Echo	6
7	Echo reply	3
9	Partial Order Connection Permitted	1
10	Partial Order Service Profile	10
11	CC, Connection Count	459 061
12	CC, New	57 171
13	CC, Echo	3 131
14	TCP Alternate Checksum Request	1
15	TCP Alternate Checksum Data	4
16	Skeeter	1

Option no	Option name	Number of occurrences
17	Bubba	9
18	Trailer Checksum	15
19	MD5 Signature	3 551
20	SCPS Capabilities	1
21	Selective Negative Acknowledgement	0
22	Record Boundaries	0
23	Corruption Experienced	2
24	SNAP	4
25	Connection Filter	92
	Others	463

4.5. Operating system detection

The operating system detection tool p0f bases on the values of a number of fields (like packet length, the existence of options and their order etc.) in certain states. It means that, for a successful detection, the packets from one of such states must be present. Such packets were available for between 1.3% and 8.1% of the source IP addresses, depending on the year. That seems to be a small number, but the number of packets available for the majority of the source addresses is one or two.

Of those, the most popular systems are Windows with approximately 70%, FreeBSD with 6.7% and Linux with 5.7%. Detection was not possible for 20%. Detection of more than one system was possible.

Of the hosts with anomalous behaviour the ratio of those with attempted detection was higher, 88%. Of those, roughly 86% were detected as Windows, 3.2% as FreeBSD and 1.8% as Linux. For 23% the detection was unsuccessful.

The higher number of hosts with detected systems in the second case is probably due to their longer flows with enough data for the tool.

5. Conclusions and future work

The results show a number of issues that can be found in the real Internet traffic. The first finding is that not all hosts confirm the standards. The second one is that the obsoleted and never finished features are present. Hosts seem to use certain features in their own way.

Certainly, an analysis using the whole connections, not just single packets, should show more similar issues. Another area of possible future work is a similar analysis using protocols not covered in this paper, like SCTP or IP version 6.

The results also show that new implementators of the TCP/IP stack should be very careful about the features not supported and should react reasonably to unexpected events, as they can be found in the traffic. A similar approach should be taken when implementing and deploying network security mechanisms, like IDSes (Intrusion Detection Systems). It seems that false-positives, when using the packet-based approach, are likely. The issue of deciding if an event observed is an attack or just a result of a normal behaviour seems to be, in many cases, a non-trivial problem.

Acknowledgements

This work would not be possible without the traces from MAWI.

References

[1] Paxson V., Allman M., Dawson S., Fenner W., Griner J., Heavens I., Lahey K., Semke J., Voltz B., *Known TCP Implementation Problems*. March (1999), RFC 2525.

[2] Bellovin S., *A Look Back at "Security Problems in the TCP/IP Protocol Suite"*. In ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04), Washington, DC, USA, IEEE Computer Society, (2004) 229.

[3] Kenney M., *Ping of Death*. (1997)
URL: http://www.insecure.org/sploits/ping-o-death.html

[4] daemon9, *LOKI2 (the implementation)*. Phrack 7(51), September (1997).

[5] daemon9, alhambra, *Project Loki*. Phrack 7(49), August (1996).

[6] Dittrich D., *The "stacheldraht" Distributed Denial of Service Attack Tool*. (1999)
URL: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis

[7] Zalewski M., *Razor Paper: Strange Attractors and TCP/IP Sequence Number Analysis*. April (2001).

[8] Zalewski M., *Strange Attractors and TCP/IP Sequence Number Analysis – One Year Later*. September (2002).

[9] de Raadt T., Hallqvist N., Grabowski A., Keromytis A., Provos N., *Cryptography in OpenBSD: An Overview*. In Proceedings of the 1999 USENIX Annual Technical Conference, Monterey, California, USA, June (1999) 93.

[10] Handley M., Paxson V., Kreibich C., *Network Intrusion Detection: Evasion, Traffic Normalization and End-to-End Protocol Semantics*. In Proceedings of the USENIX Security Symposium, (2001).

[11] Honeynet Project. *Known Your Enemy: Passive Fingerprinting – Identifying remote hosts, without them knowing*.
URL: http://project.honeynet.org/papers/finger/

[12] Fyodor, *Remote OS detection via TCP/IP Stack FingerPrinting*.
URL: http://www.insecure.org/nmap/nmap-fingerprinting-article.html

[13] Smart M., Malan R., Jahanian F., *Defeating TCP/IP Stack Fingerprinting*. In Proc. 9th USENIX Security Symposium, pages 229-240, 2000.

[14] Fisk G., Fisk M., Papadopoulos C., Neil J., *Eliminating Steganography in Internet Traffic with Active Wardens*. In IH '02: Revised Papers from the 5th International Workshop on Information Hiding, London, UK, Springer-Verlag, (2003) 18.
[15] Medina A., Allman M., Floyd S., *Measuring the Evolution of Transport Protocols of the Internet*. SIGCOMM Comput. Commun. Rev., 35(2) (2005) 37.
[16] MAWI website. URL: http://www.wide.ad.jp/wg/mawi/
[17] p0f website. URL: http://lcamtuf.coredump.cx/p0f.shtml
[18] *Transmission Control Protocol*. RFC 793, September (1981).
[19] Zalewski M., *Windows setting URG value on certain SYN and RST packets*. URL: http://lcamtuf.coredump.cx/p0f-help/p0f/doc/win-memleak.txt
[20] Postel J., *Internet Control Message Protocol*. RFC 792, September (1981).