Security considerations in a multihomed operating systems.

BLAZEJ ADAMCZYK
Silesian University of Technology
Institute of Computer Science
Akademicka 16, 44-100 Gliwice
POLAND
blazej.adamczyk@polsl.pl

Abstract: This short article presents an interesting behaviour of popular operating systems when multiple network interfaces are being used at the same time. Even if the IP layer forwarding (routing) is disabled some operating systems still deliver spoofed packets to the application not checking if they came from the proper physical interface. This paper verifies and compares the behaviour of three most popular operating systems, i.e. Microsoft Windows, Linux and Apple OS X. The behaviour was verified experimentally. The results show that all tested systems behave differently and there is no agreed way of processing IP traffic. This is an important problem which is not sufficiently documented nor described and may lead to security flaws when improperly interpreted by system administrators.

Key–Words: multihomed systems, firewall, packet spoofing, protocol stack, network security

1 Introduction

Multihomed means a host has more than one IP address. Usually different IP addresses are used on different physical interfaces. With the expansion of IPv6 protocol [1] and the Shim6 protocol [2] users start to understand better the advantages of using multihoming. Unfortunately, as the paper will present, multihoming brings also some security risks which need to be correctly understood by administrators. What is more, as it will be shown, there is no standard behaviour defined and operating systems behave differently while handling multihomed traffic.

The literature vastly covers networking and multihoming in context of different operating systems (e.g. [3]). It also covers the details of implementation of different multihoming helper protocols (e.g. the Shim6 protocol [4]). There are also some studies of the performance benefits while utilising multihoming functions of the operating system - [5]. Unfortunately current literature lacks a proper study of security of a multihomed hosts. This paper presents a not standardised and interesting behaviour while handling multihomed traffic. Administrators may not be aware of such behaviour what may allow for certain attack vectors.

2 Problem description

Lets consider a computer system with two network interfaces. Both interfaces have separate Internet Protocol (IP) configuration. This configuration is called "multihomed". Additionally the system administrator may decide to run different services at those different IP addresses. See Figure 1 for a system example.

The example in Figure 1 presents a multihomed machine which has a service running at the 10.0.0.1 IP address and port 80. This address is bound to the upper network interface. Upper subnet is 10.0.0.0/24 and bottom subnet is 10.0.1.0/24.

In such configuration the administrator could think that there is no way for computers from bottom subnet to access the service at the upper address unless there is an IP route configured which allows to reach the upper interface (i.e. 10.0.0.1). Unfortunately this is not true in some operating systems what will be explained in the following sections.

3 Networking stack internals

Operating system usually has a networking stack. This means that protocols at different layers of OSI model are separate kernel modules which work independently [6]. The interesting part of this process is presented in Figure 2.

After receiving and assembling a frame in memory (for further processing) it is forwarded to the second - data link layer. This layer processes all the required Ethernet header validations. If the frame is correct it is further passed to the upper layer according to the EtherType header field. If this field specifies IP protocol then the Ethernet payload (IP packet) is passed to the IP layer for further processing.

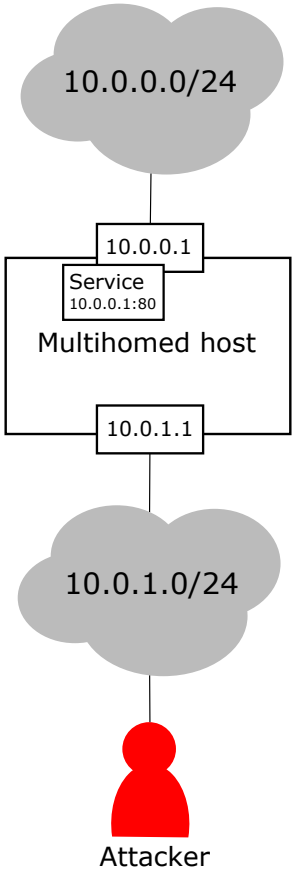

Figure 1: Attack model.

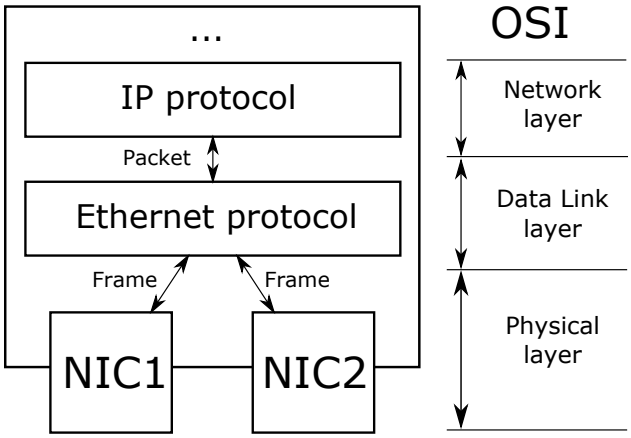

Figure 2: Lower layers of operating system networking stack.

Such design may lead to a situation where the IP layer is not actually aware where did the real frame come from (from which physical interface). Because the operating system is multihomed, some kernels may decide to not validate this information and simply treat all incoming packets (no matter from which interface) as valid.

4 Exploiting the problem

In order to exploit the described problem the attacker who has a direct connection to the bottom subnet has to send an IP packet destined to the upper multihomed host address. At the Ethernet layer the frame should be addressed to the MAC address of the bottom interface.

This is similar to IP routing where a the host sends an IP destined to a far address to its closest gateway. Thus the attacker could simply add a route to her's routing table which tells her operating system to send all packets destined to the upper IP address to the bottom interface gateway. According to the examined example a Linux based command for adding such a route would be:

```
route add -host 10.0.0.1 gw 10.0.1.1
```

From now on the attacker can simply try to init a connection to the upper service, e.g. using ncat:

```
ncat 10.0.0.1 80
```

As very first tests showed the attack is successful in Linux operating system for both TCP and UDP traffic. These results lead the author to perform tests on other popular operating systems and compare their behaviour. The experiment and results are presented in the following section.

5 Experiment

In order to compare how different operating systems behave it is necessary to create a proper test-bed. The author used virtualization in order to connect two machines just like presented in Figure 1. The multihomed machine had two virtual interface and the attacker machine had only one. Author tested the following operating systems:

1. Microsoft Windows 7 Professional SP1
2. Linux Debian, kernel version 4.0.4
3. OS X version 10.11.5

Author repeated also the test for older variants of the mentioned operating systems and the behaviour was repeatable among single operating system.

Next, for each examined operating system the multihomed virtual machine had to be properly configured. Both addresses had to be setup properly. The author tested several different configurations in all tested operating systems: router and non-router mode,

Routing	Localhost	Transport protocol	Linux	OS X	Windows
YES	YES	TCP	X	X	X
		UDP	X	X	X
NO	YES	TCP	X	X	X
		UDP	X	X	X
	NO	TCP	V	X*	X
		UDP	V	V	X

Table 1: Experiment results
X* - attack potentially possible using TCP sequence number inference method

TCP and UDP communication. Additionally just to make sure the author also verified if the attack is possible when the service listens on a localhost interface instead of a real one. All the results are presented in Table 1.

The column "Routing" defines if the system was configure to act as an IP router. Of course in such situation the routing between external IP addresses is automatically enabled. This means that in all systems such situation should allow for connection. Further, the column "Localhost" specifies if the service was listening on a localhost interface (value "YES") or a standard external one (value "NO"). Finally all test scenarios were performed for both TCP and UDP transport protocols. Finally the last three columns in the table give the output of the test - value "V" means the system is vulnerable and the communication was successful, "X" means the connection did not succeed.

Looking at the results one can easily note that the attack succeeds in a Linux operating system but only when the service listens on external interfaces (not localhost). Apple OS X behaves differently - it passes the packet to the upper layers but the responses are being sent to the interface which contains the appropriate addressing (in presented example OS X receives the packet on the bottom interface but replies are being sent to the upper interface). This limits the attack surface a bit - the attacker could send a UDP message to such service successfully. Fortunately TCP connection cannot be successfully initiated as the SYN-ACK packet doesn't reach the attacker (this is why the result is marked in the table with a star). In such situation the attacker could also try to use some type of a TCP sequence number inference attack like [7, 8] or some type of side channel in order to find the appropriate TCP sequence number and initiate the connection. Finally, the Microsoft Windows system seems secure in all types of attacks - supposedly there is an additional check for the incoming interface in the IP layer which rejects packets coming from wrong interfaces.

Finally, the tests also proved that the localhost network is secure in all operating systems. This is very important as many critical services are being run on localhost interface only so that no one from outside could reach them. Thus it is good that the localhost services cannot be accessed this way. A sidenote is that Linux has a special kernel parameter called "route_localnet" which allows to route the traffic to localhost services. This setting is obviously by default disabled but the behaviour can be changed if needed.

6 Conclusion

This paper focuses on a problem of packet handling in popular operating systems. Multihomed systems can have several IP addresses assigned to different physical interfaces. Intuitively packets coming from one interface should not reach IP address set on the other. However, as the presented tests showed, it is not always the case. Linux for example allows for such communication. Of course this could be eliminated by additional firewall rules in order to drop such packets but the administrators have to be aware of such problem to address it properly. The purpose of this paper was twofold: firstly to spread the knowledge about the problem, secondly to examine the problem fully and test different popular operating systems in order to create a full picture of current state of the art.

As of future work regarding this problem, author would like verify other popular operating systems focusing on network equipment systems like Cisco IOS and similar.

Acknowledgements: The research was supported by Silesian University of Technology grant No. BKM/515/RAU-2/2015.

References:

[1] Deering Stephen E. Internet Protocol, Version 6 (IPv6) Specification. `https://tools.ietf.org/html/rfc2460`, 1998.

[2] Marcelo Bagnulo and Erik Nordmark. Shim6: Level 3 Multihoming Shim Protocol for IPv6. `https://tools.ietf.org/html/rfc5533`, 2009.

[3] Mike McCune. *Integrating Linux and Windows*. Prentice Hall Professional, January 2001. Google-Books-ID: 0SM3PEH9gagC.

[4] Sebastien Barre, Olivier Bonaventure, and others. Implementing SHIM6 using the Linux XFRM framework. In *Routing In Next Generation Workshop, Madrid, Spain*, 2007.

[5] Aditya Akella, Bruce Maggs, Srinivasan Seshan, and Anees Shaikh. On the performance benefits of multihoming route control. *IEEE/ACM Transactions on Networking*, 16(1):91–104, 2008.

[6] Rami Rosen. *Linux Kernel Networking: Implementation and Theory*. Apress, February 2014. Google-Books-ID: RpsQAwAAQBAJ.

[7] Zhiyun Qian and Z. Morley Mao. Off-path TCP sequence number inference attack-how firewall middleboxes reduce security. In *2012 IEEE Symposium on Security and Privacy*, pages 347–361. IEEE, 2012.

[8] Zhiyun Qian, Z. Morley Mao, and Yinglian Xie. Collaborative TCP sequence number inference attack: How to crack sequence number under a second. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, pages 593–604. ACM, 2012.