Sixth Int. Workshop on GNs, Sofia, 17 Dec. 2005, 43-46

Describing a Computer Network Using Generalized Nets

Milko Krachounov
milko@3mhz.net

1 Introduction

In the present article we will demonstrate the behaviour of a computer connected to a computer network with the aid of generalized nets [1]. The computer shown in the model is working as a workstation or a server and as a network gateway (i.e., it is doing packet forwarding) at the same time. We can assume the most simple case when there is basic packet filtering and Network Address Translation (NAT) done on the computer.

2 Behaviour of a single computer in a network

2.1 General computer behaviour

First, we are going to look at what happens with the network packets received by the computer or produced by a local process. On Figure 1 a model of what happens with the network traffic is shown. In the most general case, we can assume that a token corresponds to a single network packet, but we can also look at the network traffic as a flow, depending on the needs.

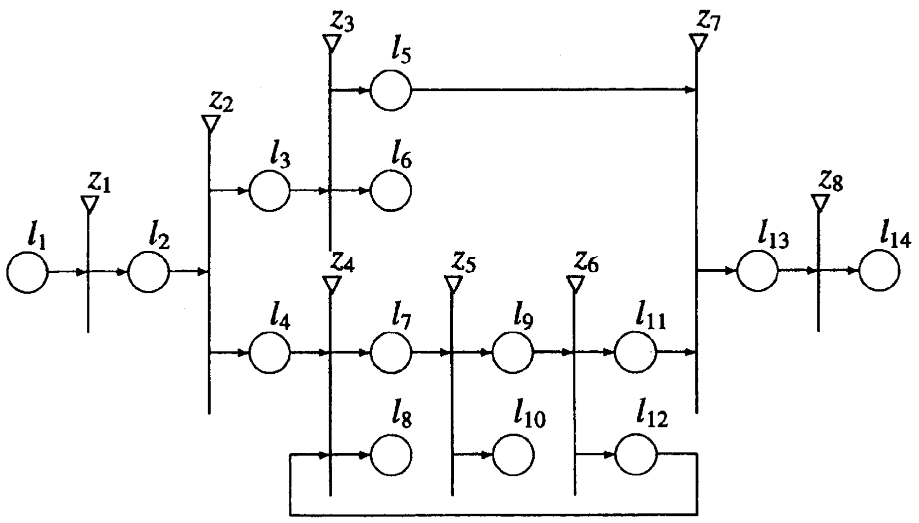

Figure 1: Net model of a single computer

The tokens corresponding to the packets received by the computer (or the incoming network flow) go to place l_1, the tokens corresponding to the sent packets leave through place l_{14}. Transition z_5 corresponds to the network-enabled processes running on the computer, i.e. it is where the main part of the processing of the information in the network packets is done. The routing decision, that is, whether the packets are destined to the computer, or should be forwarded to somewhere else, is taken in transitions z_2 (for received packets) and z_6 (for local packets). Unwanted network packets are discarded in transitions z_3 and z_4. The transitions z_1 and z_8 are where additional decisions are taken. For example, they can be used for NAT - changing the destination of the packet in z_1 or changing the source in z_8.

The model is similar to the behaviour of Linux Netfilter[1], although it is very simplified and is lacking a lot of the details, but the idea is the same and it can be used to explain how a computer works in a network and how packet filtering and routing are done.

2.1.1 Incoming packets

After the tokens have entered the net, they pass through transition z_1. The characterising function of place l_2 takes care to make the necessary modifications, like changing the destination of the network packet. In transition z_2 a decision is made whether the packet should be received by a local process or should be forwarded to another machine. The predicate $r_{2,4}$ is true when the destination of the packet matches with one of the addresses assigned to the computer, while $r_{2,3}$ is equal to its negation.

$$r_{2,4} = \neg r_{2,3} = \bigvee_i (dest = myaddress_i),$$

where *dest* is the destination of the packet, and *myadress*$_i$ are the addresses assigned to the computer.

Transitions z_3 and z_4 are identical in behaviour. The decision whether the packet should be accepted or dropped is taken there. The predicates $r_{3,5}$ and $r_{4,7}$ should be true for wanted packets, while $r_{3,6}$ and $r_{4,8}$ are their negations. They should be defined by a certain rule, which will be suitable for the situation in which it is used. For example, if *badsrc* is a list of the addresses of the computers that have tried to perform malicious activities on the network, and we want to ban them from accessing anything, then the predicates will be:

$$r_{3,5} = r_{4,7} = \neg r_{3,6} = \neg r_{4,8} = source \notin badsrc = \bigwedge_i (source \neq badsrc_i),$$

where *source* is the source address of the packet.

The tokens corresponding to the unwanted packets go to places l_6 and l_8 and leave the net. If $r_{3,5}$ is always false, then the computer is acting as a simple workstation (or a server) and is not forwarding any packets. Instead of using predicates, we can also create generalized nets that can be used in the place of the z_3 and z_4 transitions to decide what should be done with the packets, so we can do a more complex packet selection.

[1]Linux packet filtering framework, http://www.netfilter.org/

The characteristics the packet has received after the z_1 transition can also be used in the packet filtering in z_3 and z_4. For example, in Linux Netfilter you can mark the packets in a certain manner when they are received and do the further filtering based on their mark.

2.1.2 Packets sent to the computer.

Transition z_5 corresponds to the running local processes. We we will look further into this in the next section. Each packet is received by the program it was send to, it is processed and then the token either goes to l_{10} and leaves the net, or the process generates a packet as an answer, whose token continues to the z_6 transition, where a decision similar to the one in z_2 is taken. By analogy, predicate $r_{9,11}$ is true for each packet that should be send to another machine, while $r_{9,12}$ is true for each packet that is destined for another process running on the computer.

2.1.3 Leaving packets

All packets that should be sent to another computer in the network go through transition z_8 and the characterising function of the place l_{14} alters different kind of information about the packet, for example, if source NAT is performed, it can change the source address of the packet.

2.2 Local processes

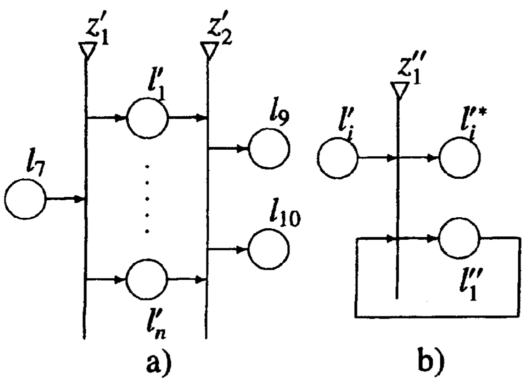

Figure 2: Local processes

A generalized net, corresponding to the z_5 transition, is shown on Figure 2. Positions $l'_1, .., l'_n$ correspond to each of the n network-enabled programs running on the computer. If the packet is for the process j then the predicate r_{l_7,l'_i} is true when $i = j$ and is false for each $i \neq j$. If the network is using the Internet Protocol [2], and if we assume that there will be only TCP [3] communication, the predicates would look like this:

$$r_{l_7,l'_i} = \bigvee_j ((destport = port_j) \wedge (port_j \in ports_i) = (destport \in ports_i),$$

where *destport* is the port to which the packet was send, $port_j$ $(1 \leq j \leq 65535)$ are the TCP ports on which processes can listen, and $ports_i$ is the set of the ports, which process i is listening on.

When the packet was processed and it does not require an answer the token goes to l_{10}, otherwise the answer goes to l_9.

On Figure 2b the generalized net representing a single process is shown. It corresponds to place l'_i in the net given on Figure 2a. There are tokens waiting in place l''_1 so that the process can try to contact something on the network as a result of some internal decision and not as an answer to a received packet. The characterising function of place l'^*_i decides whether the token should leave the net or it carries an answer that should be transmitted to the network. This decision is then honoured by transition z'_2.

3 Behaviour of the network as a whole

We can easily generalize the given model, so that we can describe the whole network, as it is shown on Figure 3. Tokens corresponding to packets sent from one computer to another go to place l_0. Places l'''_1, ..,l'''_n are for the computers in the network, and each one of them corresponds to a generalized net like the one shown on Figure 1. The predicate from l_0 to l'''_i is true when i is the destination of the packet.

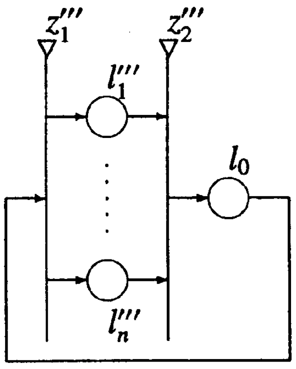

Figure 3: Net model of the whole network

References

[1] Atanassov, K. *"Generalized Nets"*. World Scientific, Singapore, 1991

[2] Postel, J. (ed.). *"Internet Protocol - DARPA Internet Program Protocol Specification"*, RFC 791. USC/Information Sciences Institute, September 1981

[3] Postel, J. (ed.). *"Transmission Control Protocol - DARPA Internet Program Protocol Specification"*, RFC 793. USC/Information Sciences Institute, September 1981