NetSpotter

User's Guide

Summary

NetSpotter system (application) helps the network administrators, the protocol and network designers (developers), and the system administrators in their network management, tuning, monitoring and testing work at the application of the present and future IP protocols.

This guide discusses the application of the system in two parts. In the first part you can read about the building of the protocol sequence and their implementations (operation). You can examine additional techniques of instances in order to sequence and define their features. After defining the message, the communication between instances and the complete sequence is executable and after the running activity we can analyse the results of the operation. In the second part we deal with the discovery of topology, we can find there the detailed description of module operation and all possibilities and features.

Content

1. Introduction

Netspotter is based on a server-client model. The client's menu contains tasks both for Agents, both for Topology discovery. *Refresh* function serves for listing of active Agents. Sequences and messages are defined with using discovered network elements.

2. Management of Server tasks

„Server tasks" menu is for managing tasks. The left side of the screen contains running, scheduled tasks with their parameters (type, scheduled time, and period), the right side is for task modification buttons („Schedule", „Activities", „Delete task", „New Task").

Parameters of „Schedule" button are:

- one exact time
- first time + period time
- discrete, exact time list

After setting of time parameters the type selection is the next step. Under „Activities" button there are three possible types:

- Layer2DiscoverActivity
- Layer3DiscoverActivity
- SequenceExecutionActivity

All types have many parameters, some are compulsory and others are optional. Their modifications can be done with input field, with input box or with input panel („Edit" button). Description of optional parameters is formed HTML text visible on right side.

3. Sequence editor

3.1. Introduction

Nowadays Internet is becoming a public service. IP is a platform for the integration of communication services too. VoIP and IPTV are more and more popular. IP unicast communication paradigm is not efficient for real time and broadcasting services. The IP multicast paradigm is more efficient because broadcasting server sends data on one channel and multiplication happens on the data network. The IPv4 multicast solution does not have a widespread usage, although it is old enough. The HDTV may be a bigger stimulator to use IPv6 multicast facilities. User-friendly testing tools are needed for testing of new protocols. The main purpose of this system is to support and help the validation and other testing methods of protocols.

3.2. Editing modul

After starting NetSpotter an opening window appears on the screen:

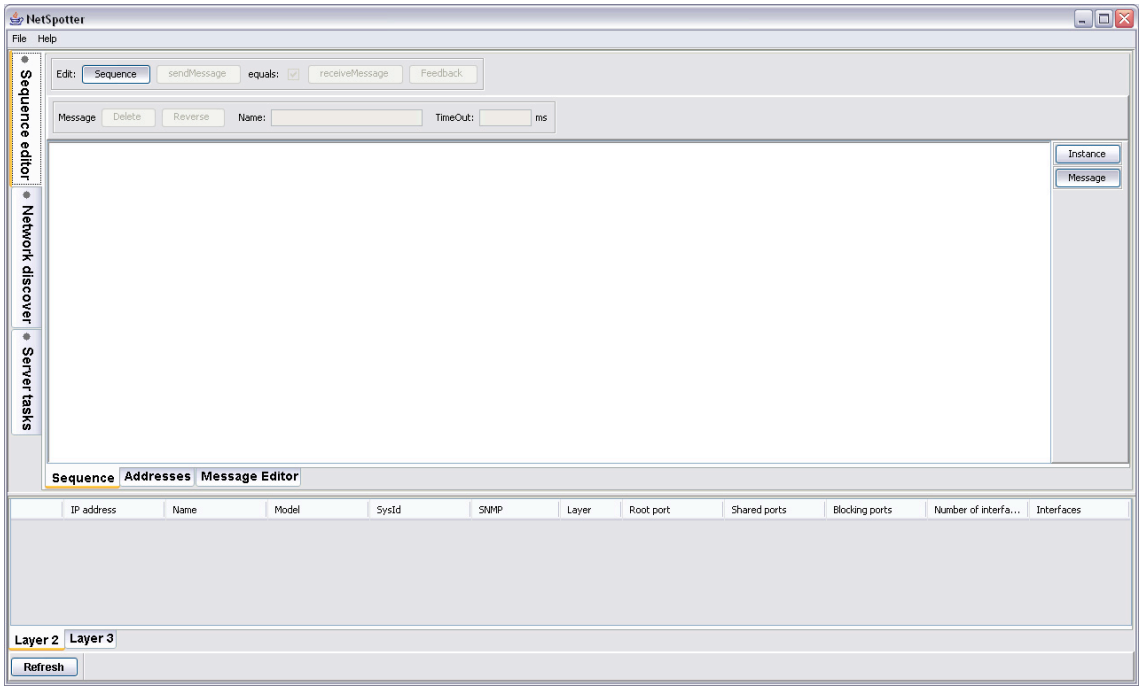


Figure 3-1 Starting window

After starting we step to *Sequence editor*. On the left side we can find other two components: *Network discover* and *Server tasks*. On the bottom of the screen there are *Agents* of discovered network equipments and logged users of servers.
The editor has a menu on the top with the tools and the buttons of the right side. On the bottom we can select submoduls: *Sequence, Addresses, Message Editor*. Next we go into details.

3.3. Building of Sequence

The Sequence editor describes the process of communication, the defining, the sending and the reading of *Messages* in time scale *Instances*.

The graphic MSC (ITU-T Z.120) is an example for flowcharts, the communication instances are special axes, the messages are arrows.

3.3.1. Instances

Every instance contains one ax, one header and one trailer. Every instance has a name. The axes are represented time passing. The management functions of instances are reachable with the *Instance* button or clicking on the instance. Double clicking serves for adding a new instance to the position indicated on drawing-tables. If the Agent is a new instance, then it will be dedicated with a new colour.

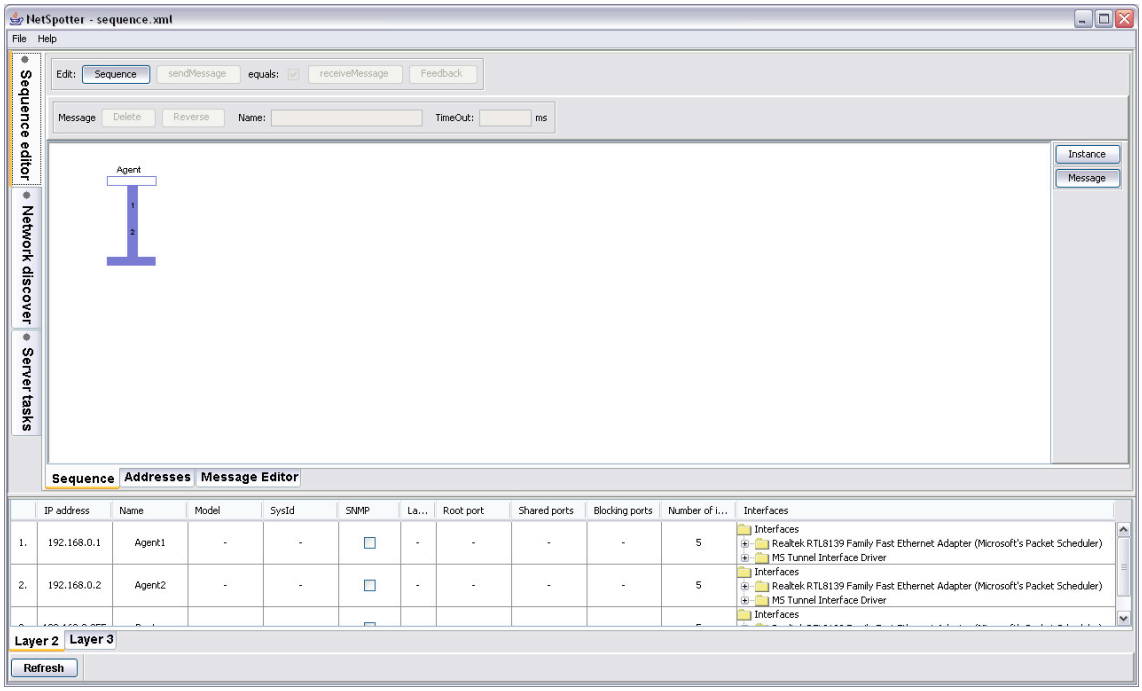


Figure 3-2 Adding of the Agent to the Sequence

Switch or Router is coloured as you have already seen on the figures.

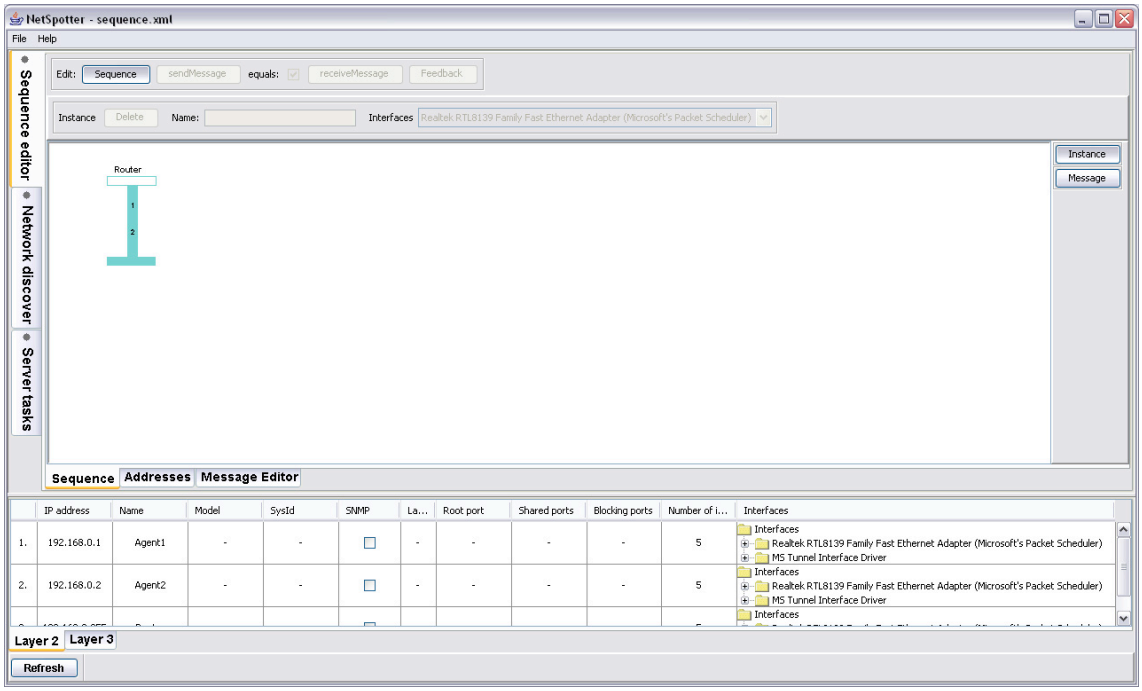


Figure 3-3 Adding of the Switch or Router to the Sequence

The Agent instance has message sequences, Feedback serves for monitoring the messages. The non Agent instance is a passive element. The distances between instances are set by the program automatically. The *Instance* menu tool is for managing instances. The selected instance is grey. When you delete it, all the related messages are deleted. Moving is with simple clicking, drag and drop. The instance remains at the original place indicated with other colours.

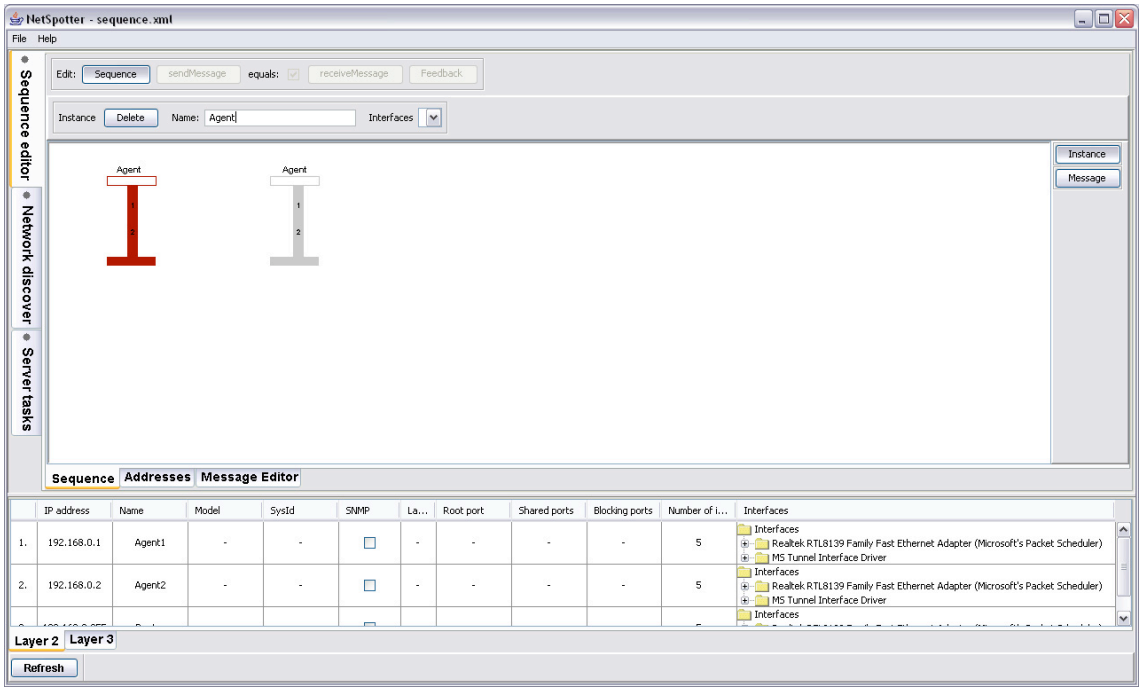


Figure 3-4 Moving the Instance

3.3.2. Messages

The messages are represented with arrows between instances. The time passing is indicated only with the order of the messages (sequences).

The management of the message is done with *Edit Sequence Edit tool Message* button or clicking to the message. New message is gained with empty message moving with the mouse from one instance to another.

In *Message* tool we can find the features of the message:

- *Name*
- *TimeOut (default* 3000 ms).

Furthermore the *Delete* button is for deleting and *Reverse* button is for changing the direction.

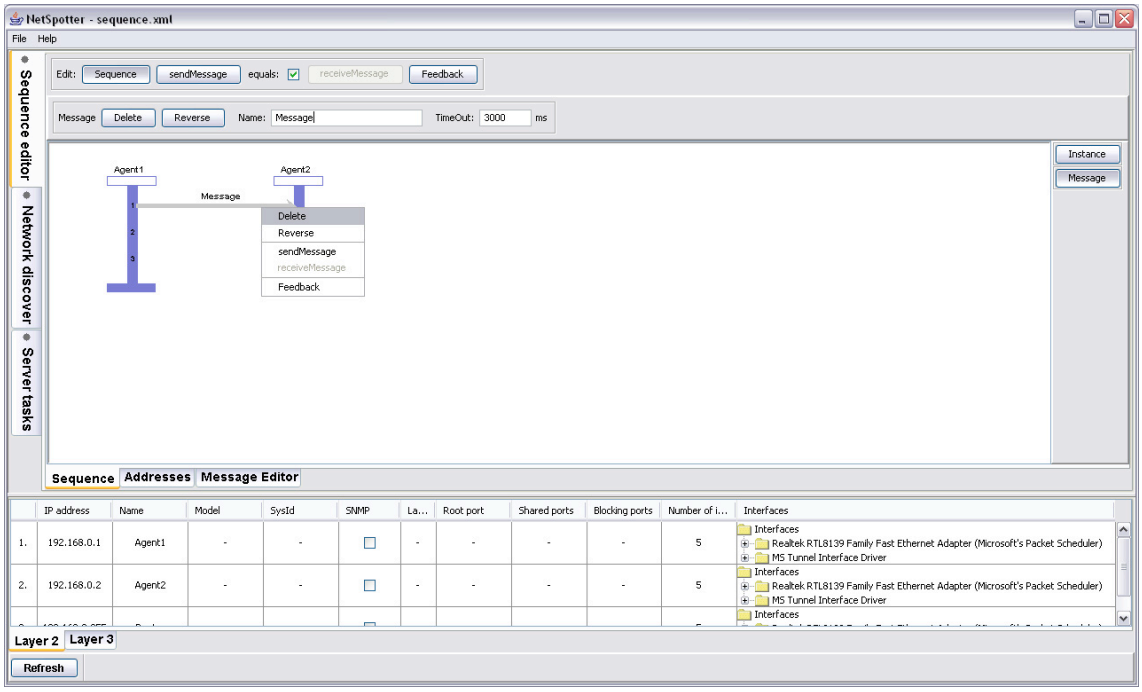


Figure 3-5 Properties of Messages

When the protocol has more messages, we can rearrange them with drag and drop. Old messages remain at their original place indicated with another colour, later it disappears.

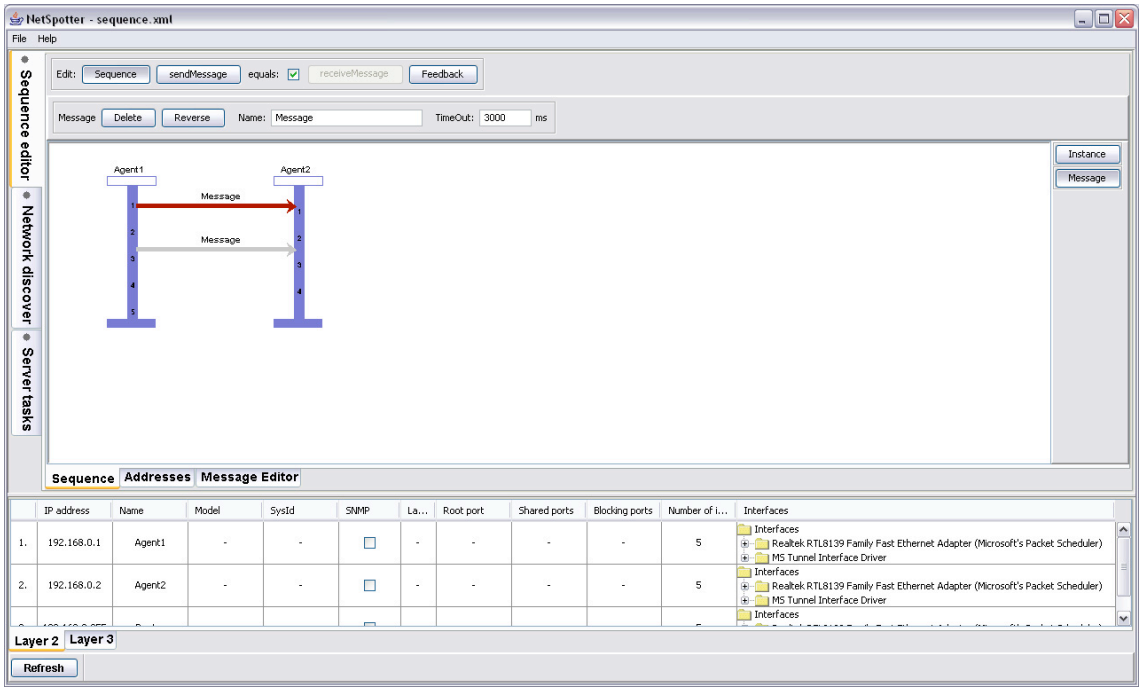


Figure 3-6 Moving the Messages

We can copy the message, if we push CTRL button before the release mouse button. + sign indicates copy, there is no movement. When the message is copied then the structure and parameters of the message (message panel) are also copied.

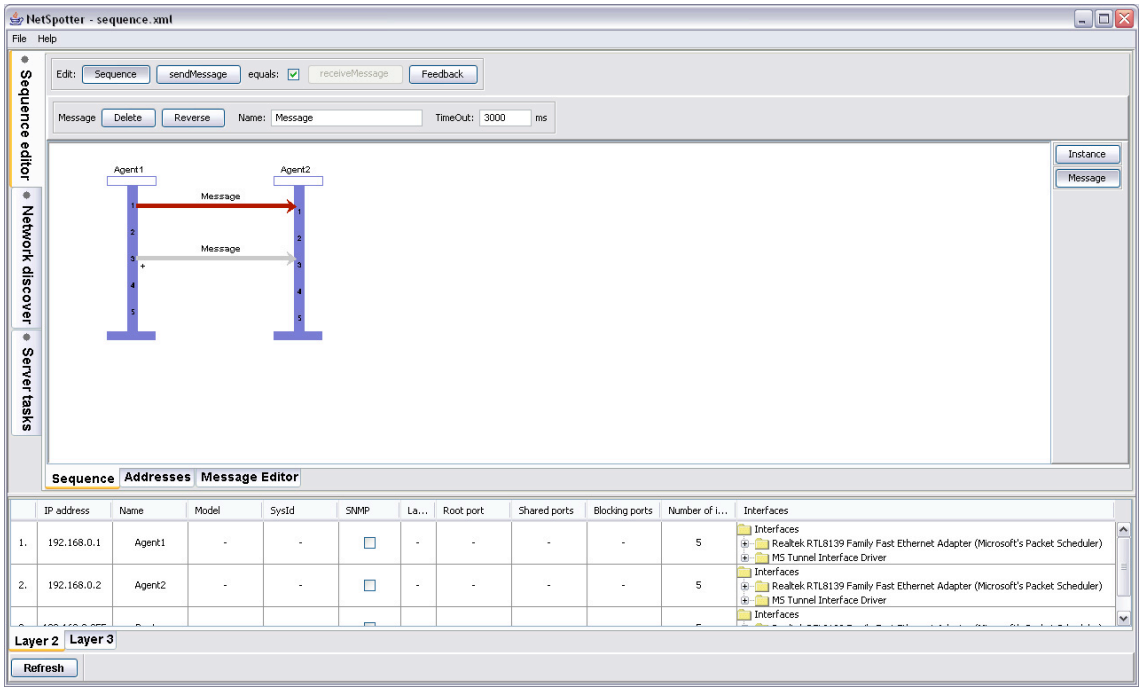


Figure 3-7 Copy the Messages

3.3.3. Delay

Delay is used by a special system, which helps the time sequences. It is possible to substitute the message which doesn't work in that time period. The colour is blue.

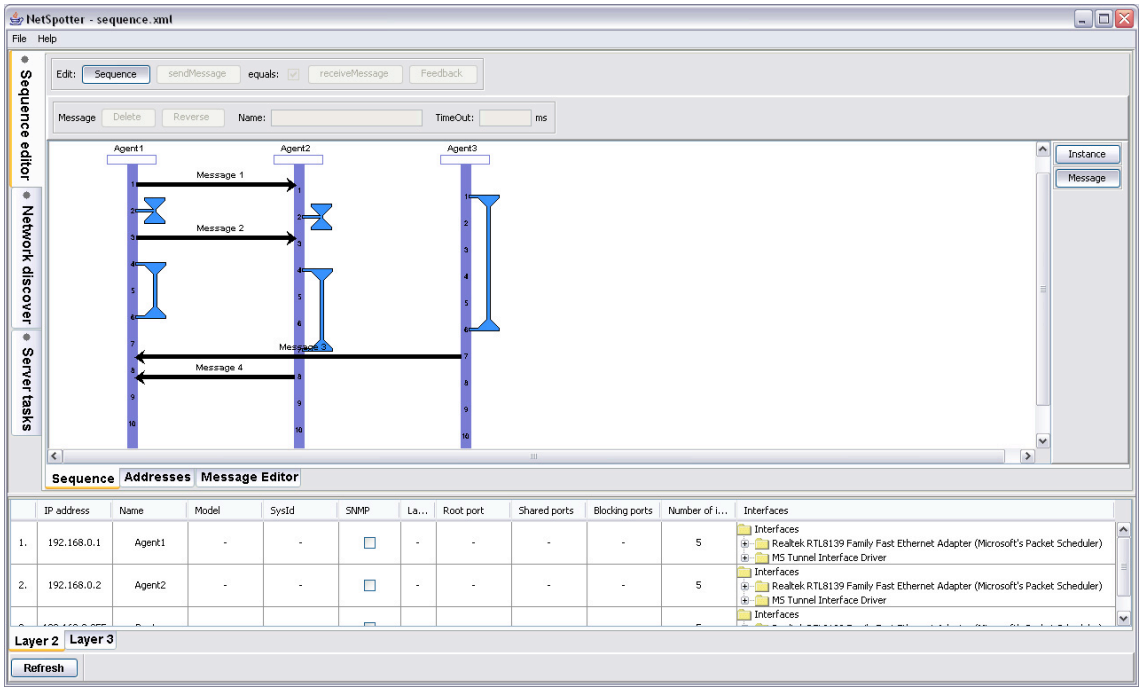


Figure 3-8 Delays

3.3.4. Message panel

Every arrow has one or more description of message content. Between two Agents we manage two messages content, one for sending and one for receiving, the default is that they have equal structures. For not Agents they both are really equal. *Edit* tool *sendMessage* button or the right button of the mouse serves for editing the content of the message.

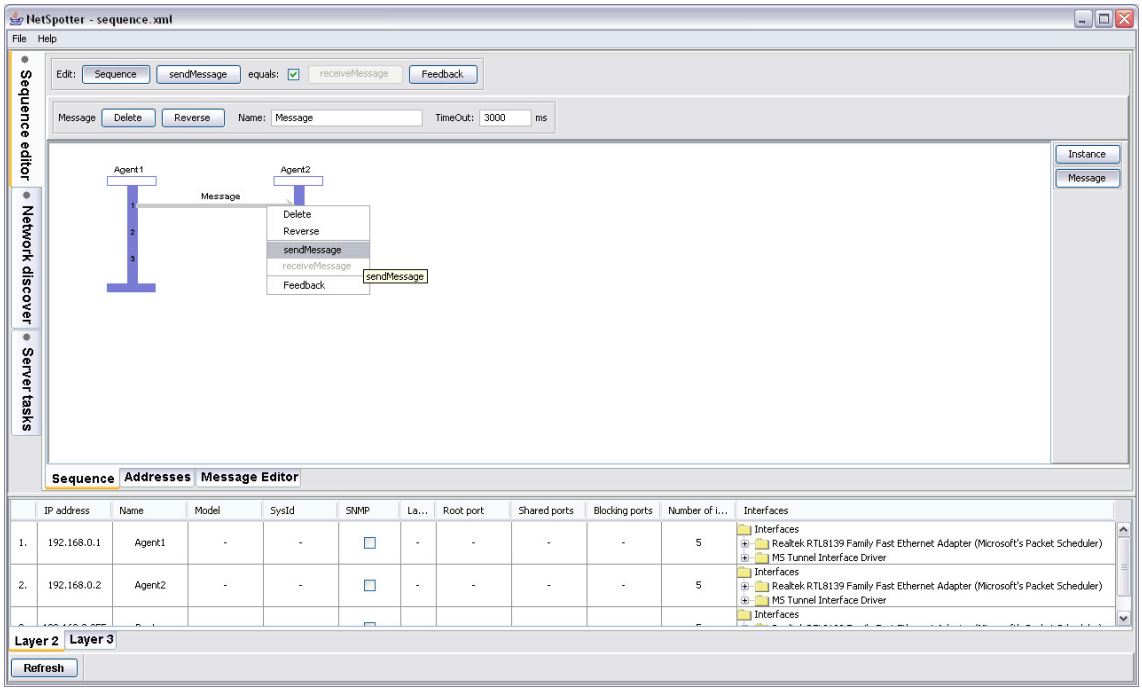


Figure 3-9 Contents of sendMessage

If the Agent waits for an input message that is different from the output message, then the editing of the message content can be done with *equals* box and *receiveMessage* button (or the right button of the mouse).

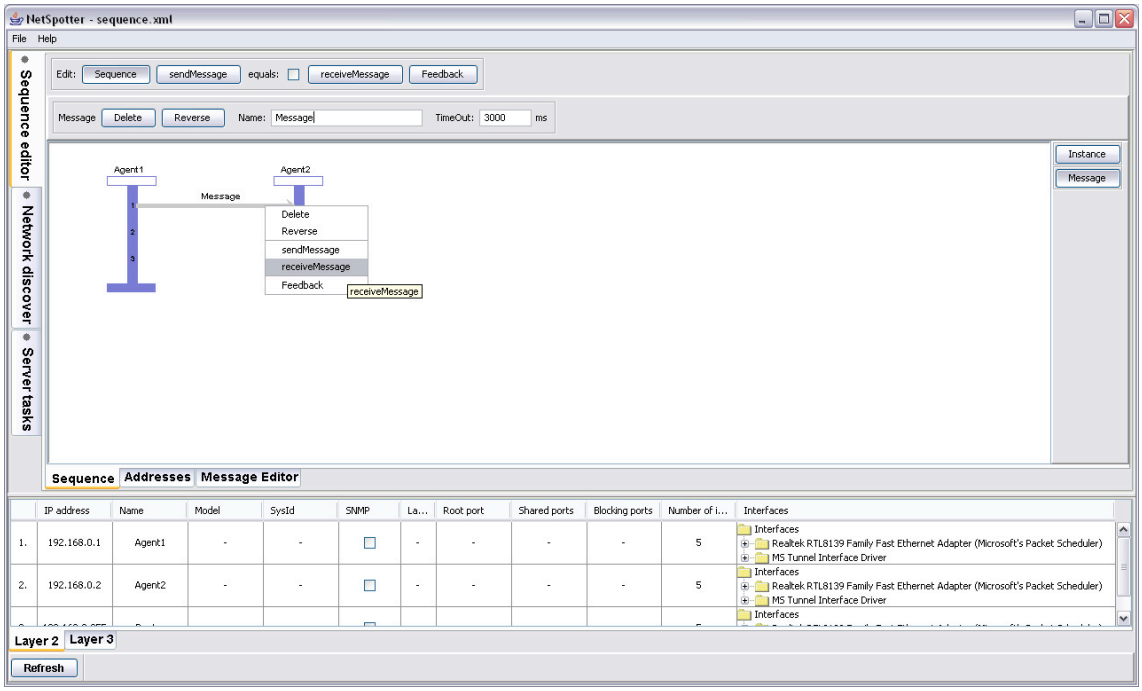


Figure 3-10 Contents of receiveMessage

The message skeletons are reachable with *Load* menu. After loading IPv4 or IPv6 skeletons we can check and set the right values of the fields. In embedding we define a new structure with *Body* menu *Edit* button and fix it with *Close Body* button.

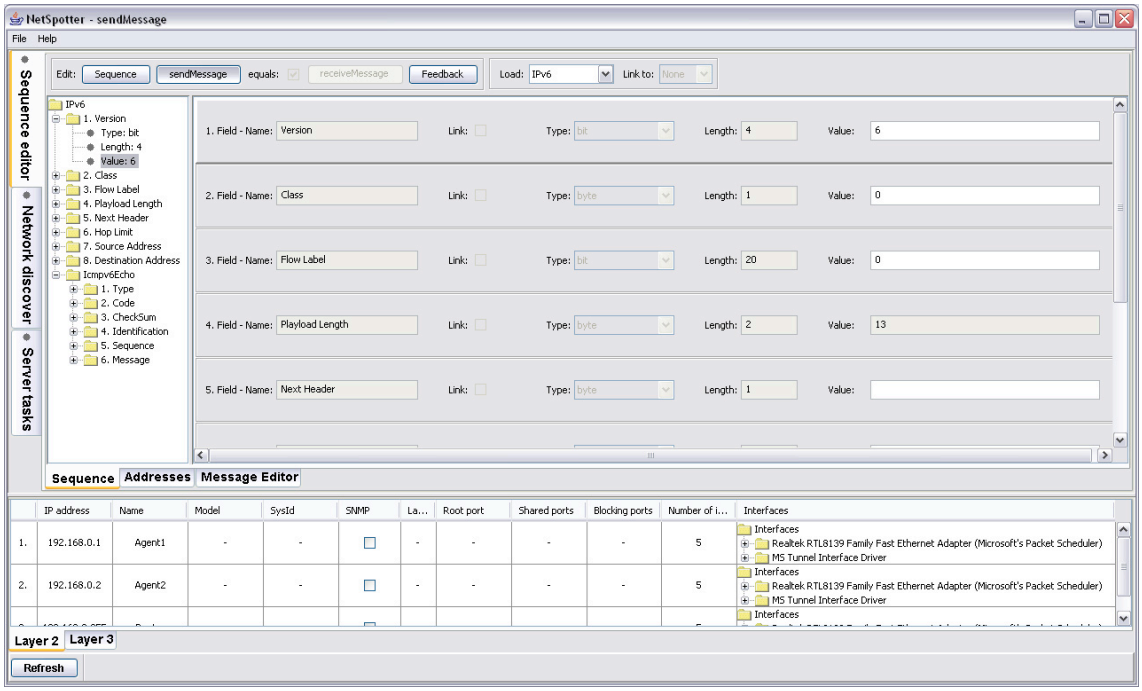


Figure 3-11 Editing of Message contents

Some fields are automatically counted: Total Length, Payload Length, Header Checksum, CheckSum. If they are not countable than their value is ANY.

The ready message is saved with *File* menu *Save, Save as* options. With *Edit* tool *Sequence* button we can return to edit.

3.3.5. Alias Addresses

IP and MAC addresses are in the lists, and we manage them with *Select address list* menu. *Delete* button clears all selected list.

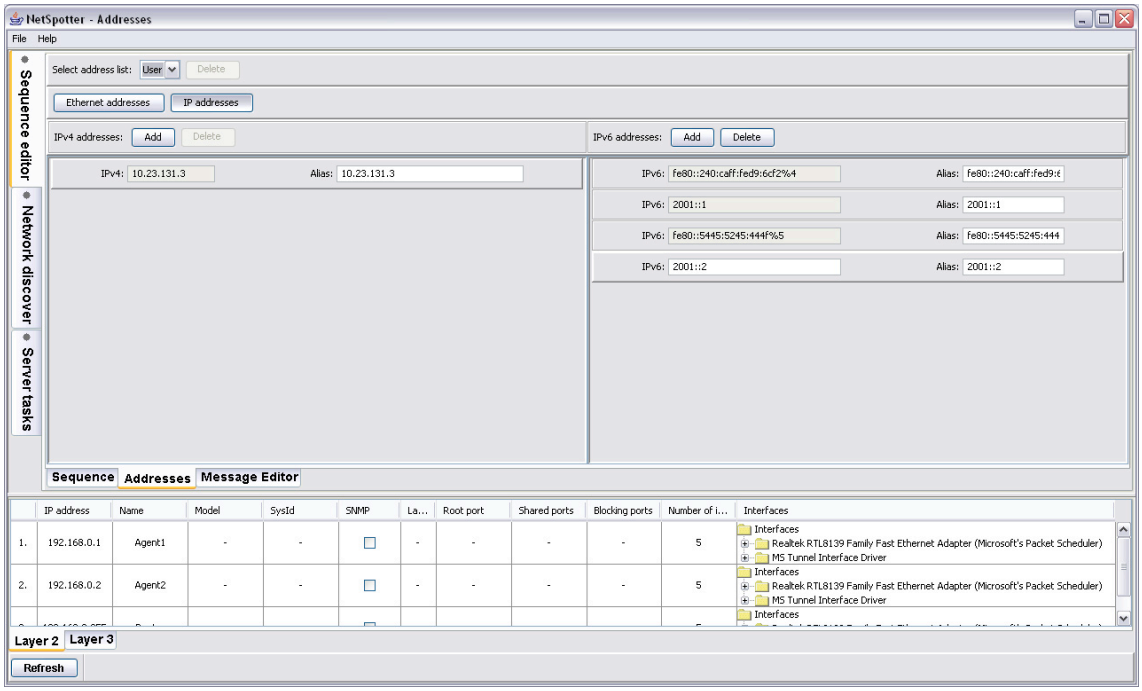


Figure 3-12 Alias-Addresses

IP addresses and *Ethernet addresses* buttons activate the list of addresses separately IPv4 and IPv6 types.

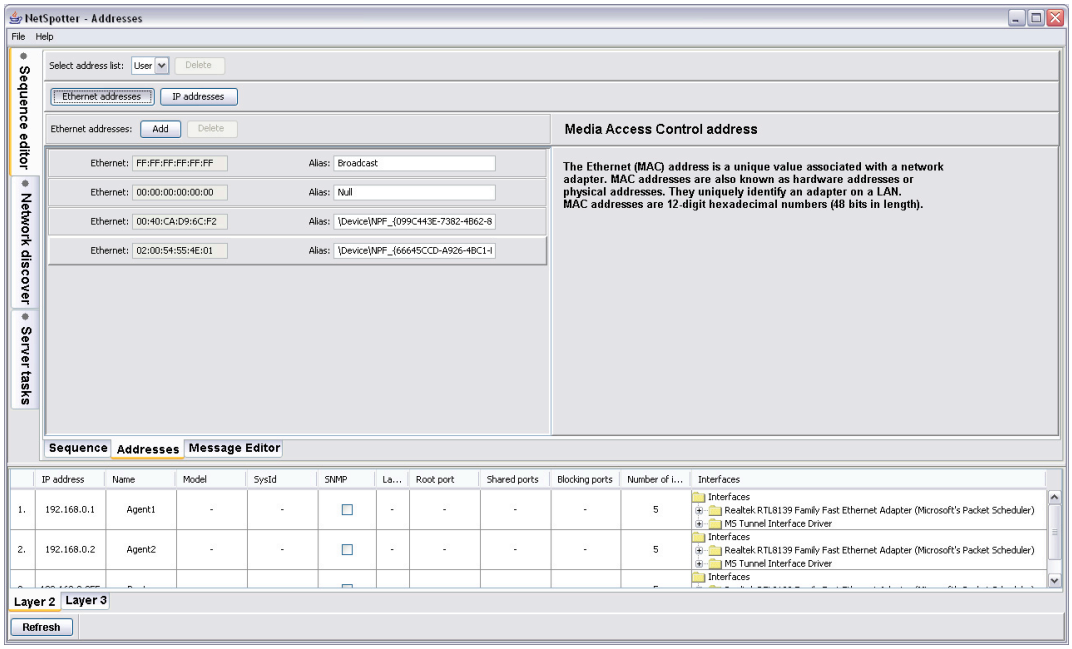


Figure 3-13 Alias-Addresses in MAC case

Add button is for adding new addresses, Delete button is for deleting. If we add the new instance (Agent, Router or Switch), then the addresses of the instance are

automatically added to the address lists. Error box is indicating the result of value checking.

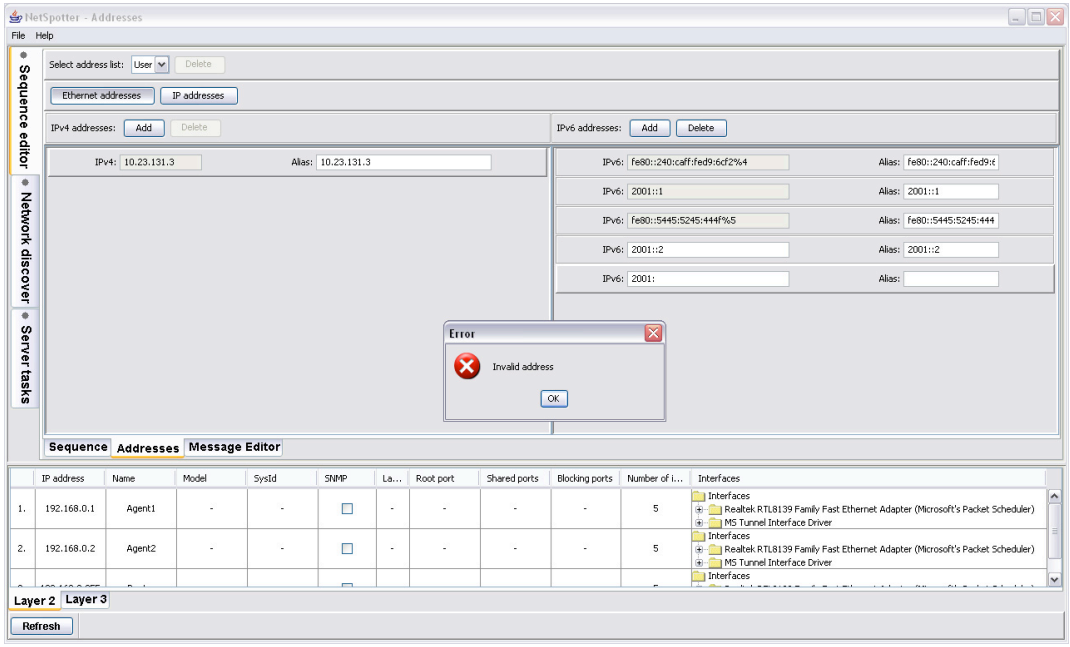


Figure 3-14 Error about invalid Alias-Addressing

3.3.6. Message Editor

Message Editor is for constructing new skeletons using as earlier defined. New Message button starts the activity; we give the name of the skeleton to it.

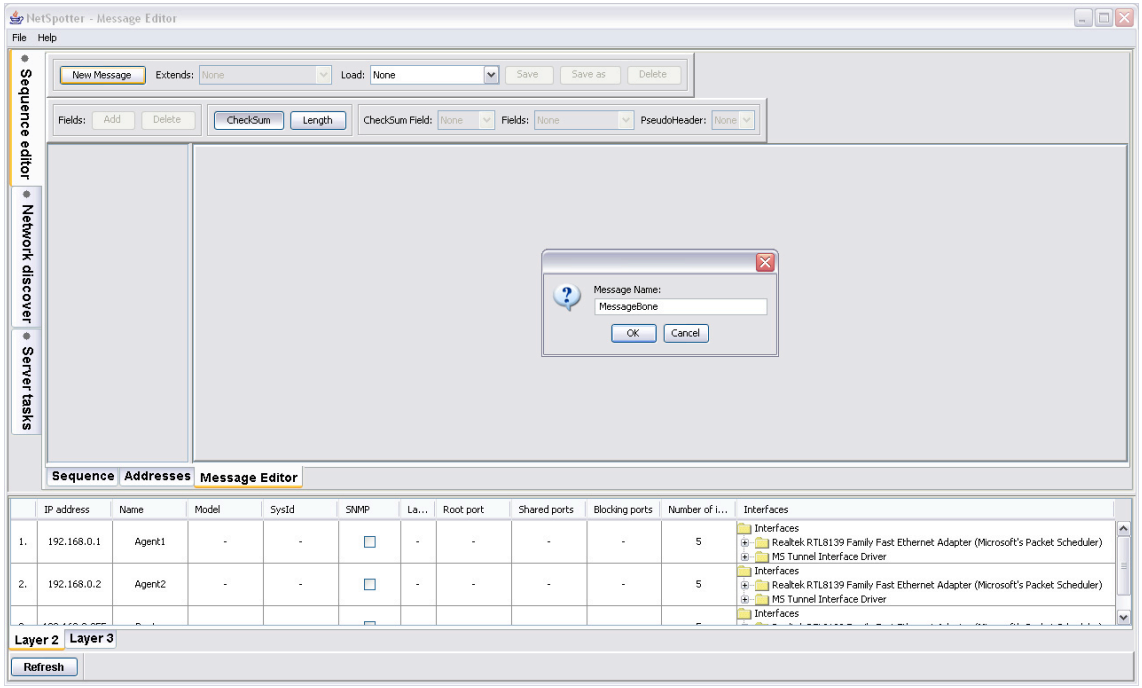


Figure 3-15 New Message Skeletons

In Body menu with Fields tool Add button we add: Name, Type, Length and Value of filed. Default names (Text, MACaddress, ip6address and ip4address) have fix length.

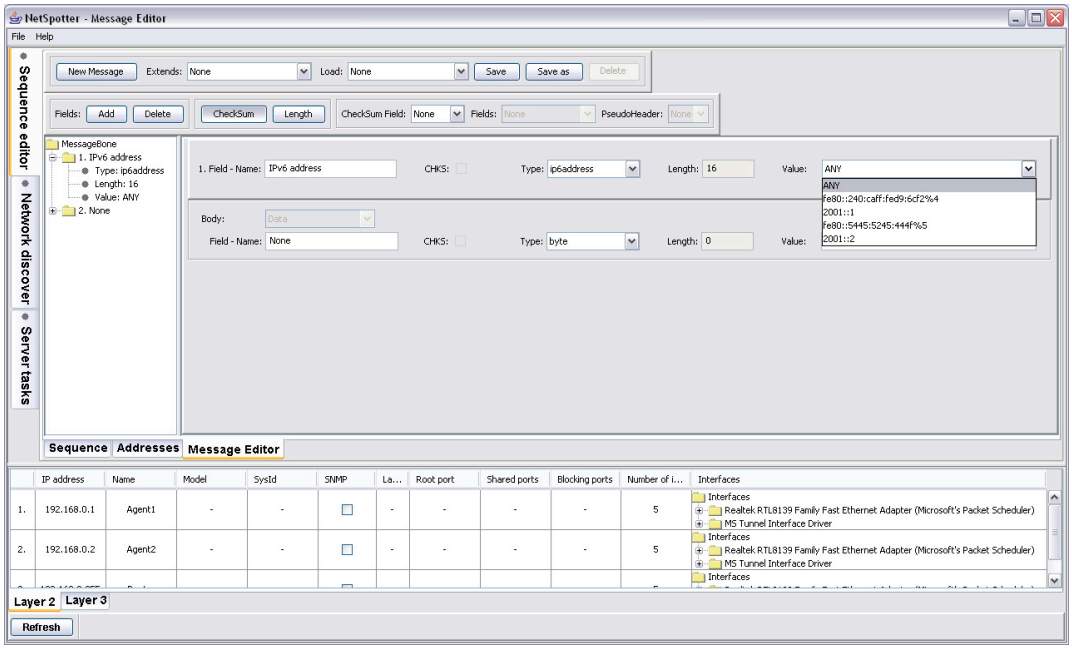


Figure 3-16 Adding of Filed to Message

Inherit feature works with *Extends* list with common rules in changing values of the ancestor.

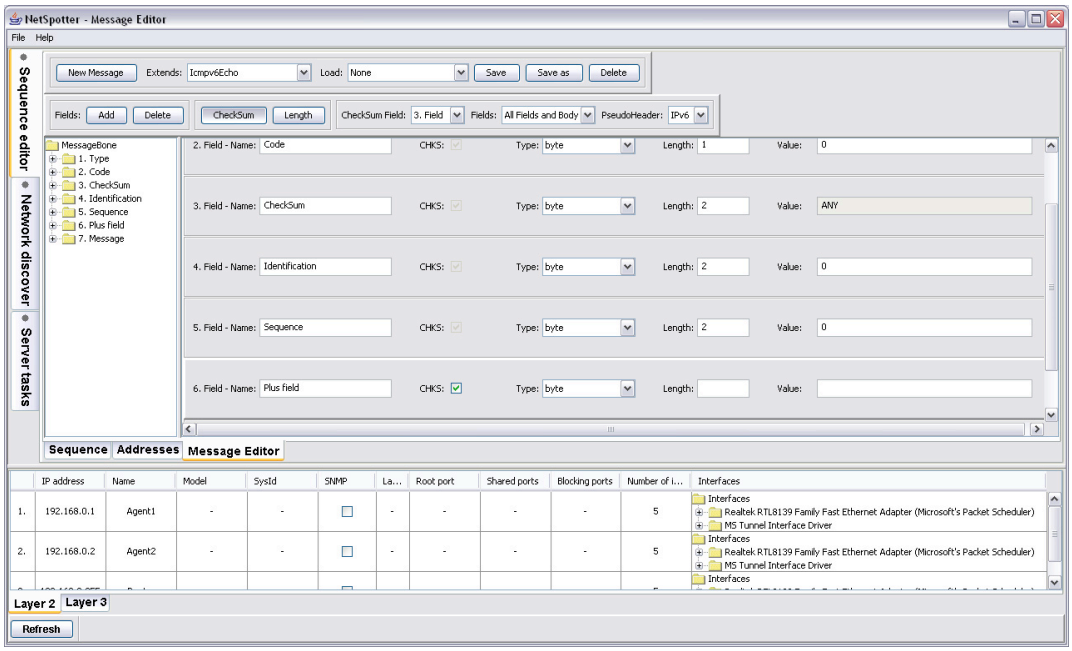


Figure 3-17 Inheriting of Message Skeleton

The *CheckSum* button activates the *CheckSum* tool, we select the field or fields:

- *Fields*: All Fields, All Fields and Body or Custom with CHKS flag.
- *PseudoHeader:* some fields from IP headers.

After fields selecting it takes the automatic checksum counting.

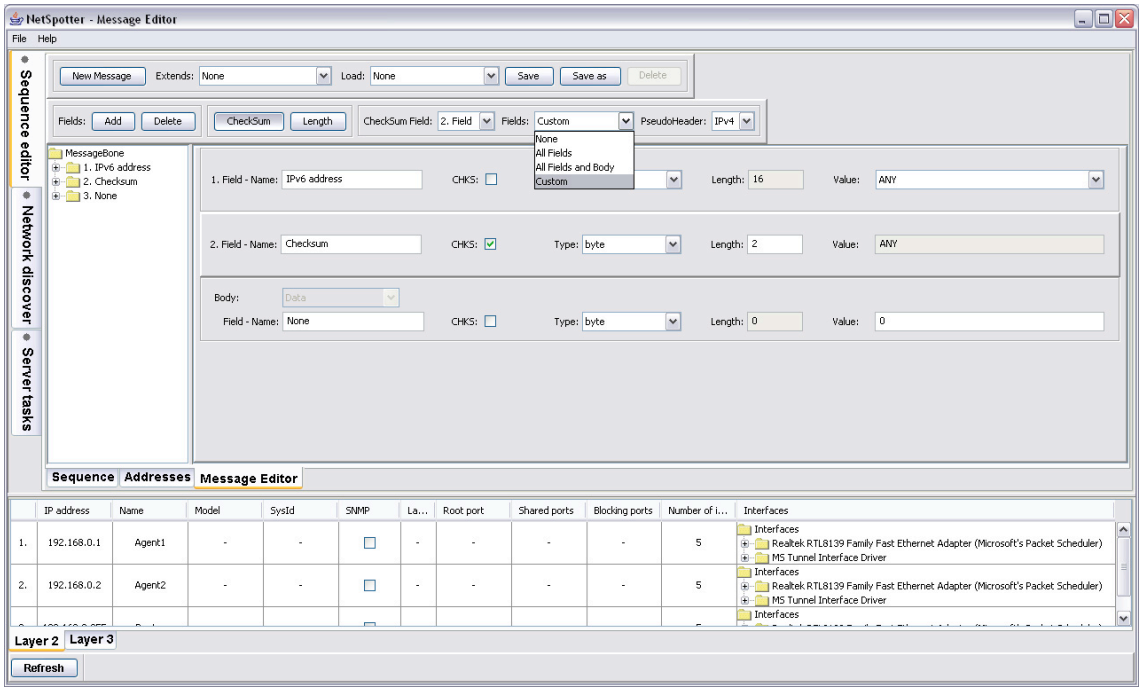


Figure 3-18 Checksum field

Length button works the same way as checksum (*Field* or *Fields* lists).

After fields selecting it takes the automatic length counting.

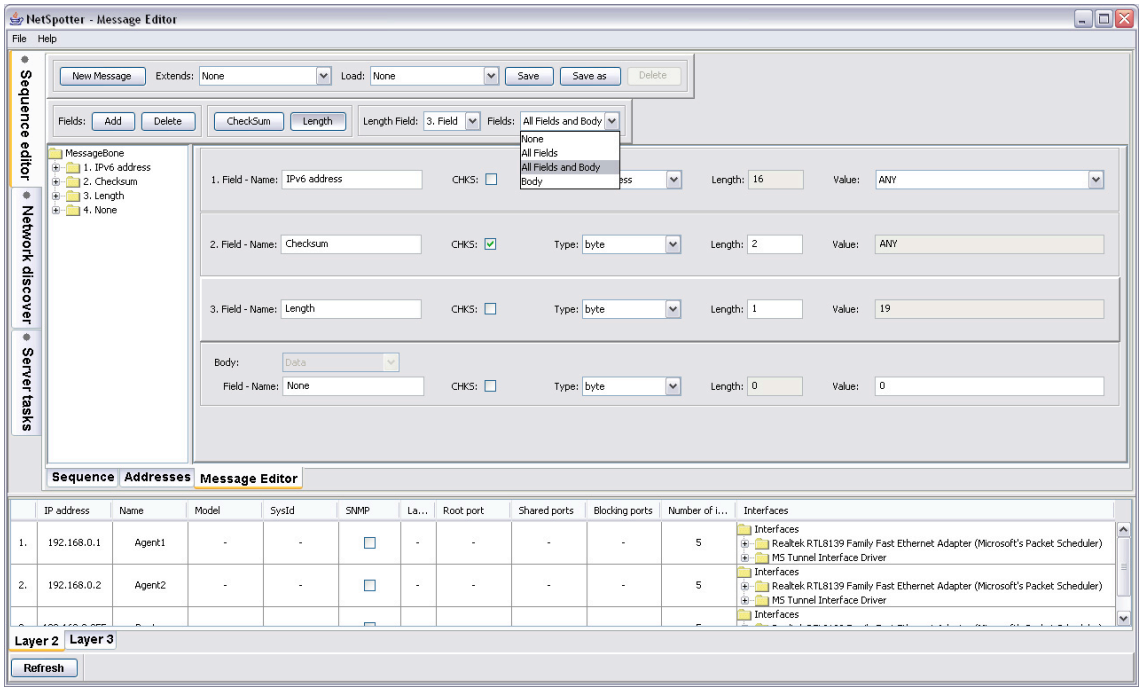


Figure 3-19 Setting of Length

The managing of skeletons is done in the usual way, *Load* list, *Save* or *Save as* and *Delete* buttons.

The program doesn't allow modifying „saved skeletons" (IP packets). This mechanism serves to use the standard communication. Naturally it is possible to edit any „free skeletons".

3.4. Two examples of Sequences

Two examples demonstrate the features of editor.

3.4.1. ICMPv6 Echo Request – ICMPv6 Echo Reply

Ping protocol contains two messages: ICMPv6 Echo Request and ICMPv6 Echo Reply with the same structure only Type field is different (129).

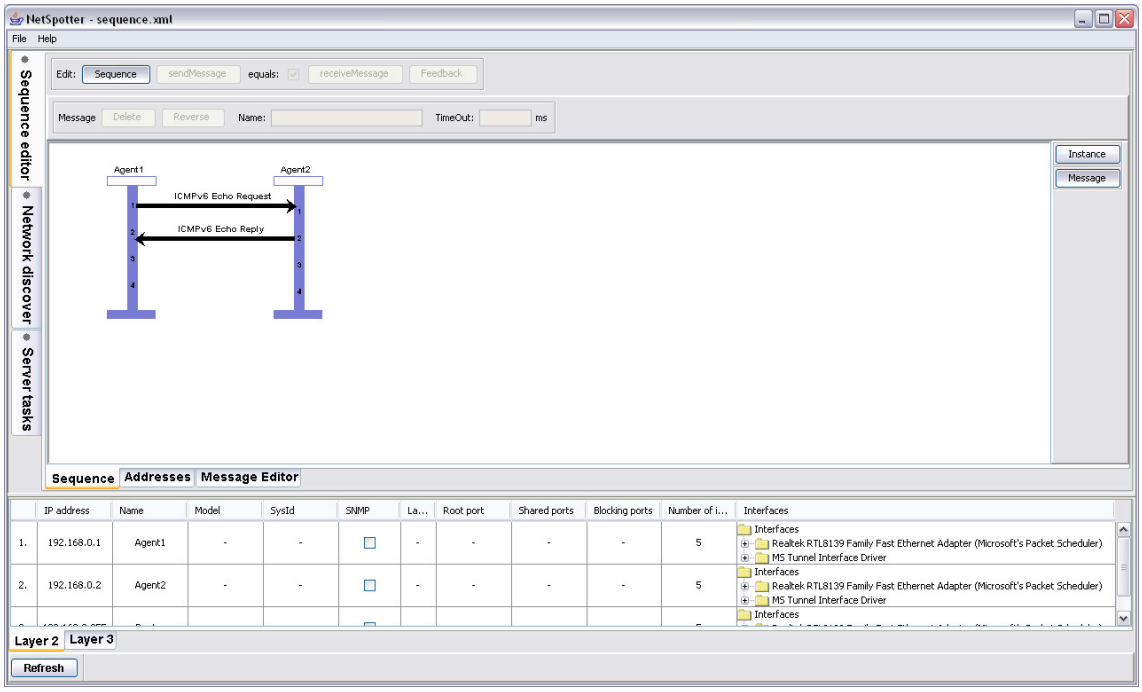


Figure 3-20 Ping6 messages

To construct a sequence: we choose two Agents, between them Echo Request message, with *sendMessage* button we start editing of message, from Load list we select IPv6 skeleton, Next Header field sets to 0x3a, Source Address is Agent1 IPV6 address, Destination Address is Agent2 and Body is set to Message type, indicated to embed the ICMPv6 segment to IPv6 packet.

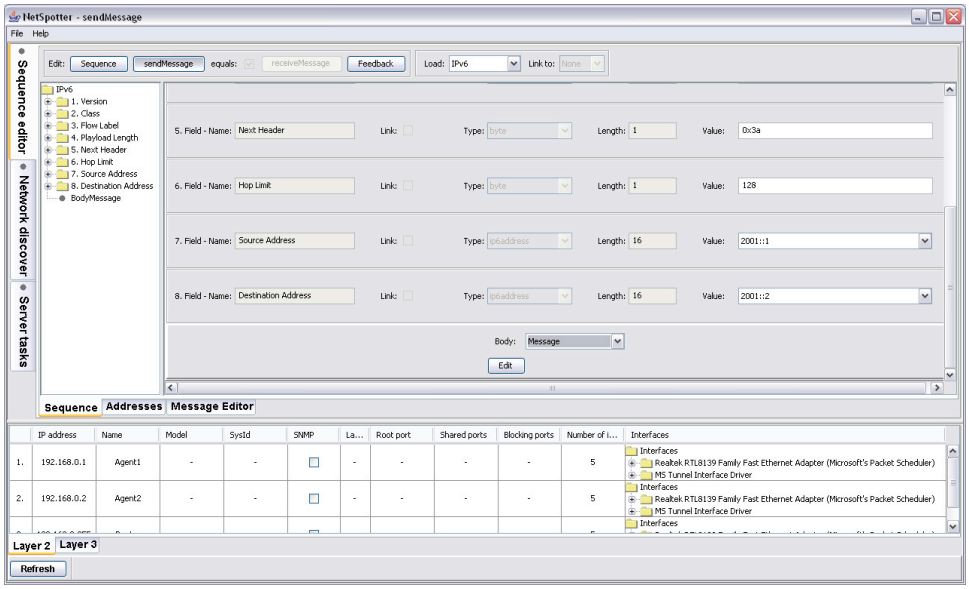


Figure 3-21 IPv6 packet

With *Edit* button we can construct Echo Request message: from *Load* list we use the Icmpv6Echo skeleton, with *Sequence* button we return to sequencing, with CTRL button is ready a copy of message, with *Reverse* button we change the direction of message, we change the name to Reply, with *sendMessage* button we start editing. After swapping of addresses of IPv6 packets and we set the Type field to 129 with *Edit* button of *Body* field it is ready the ping6 traffic.

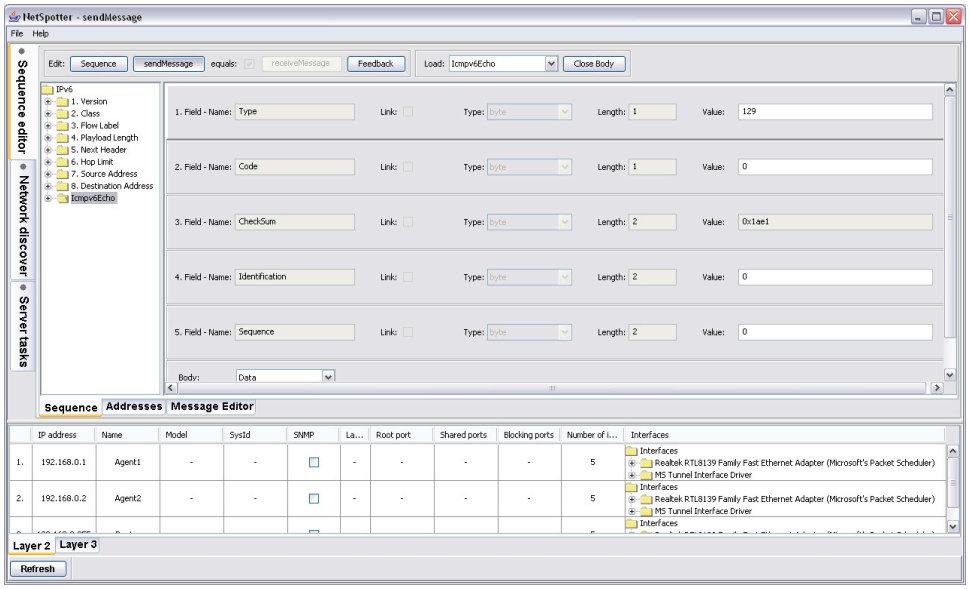


Figure 3-22 ICMPv6 Echo Reply

With *Sequence* button we return back to sequencing, next we point to the Agent2. To select the interface we choose it using the list of the *Interfaces* function.

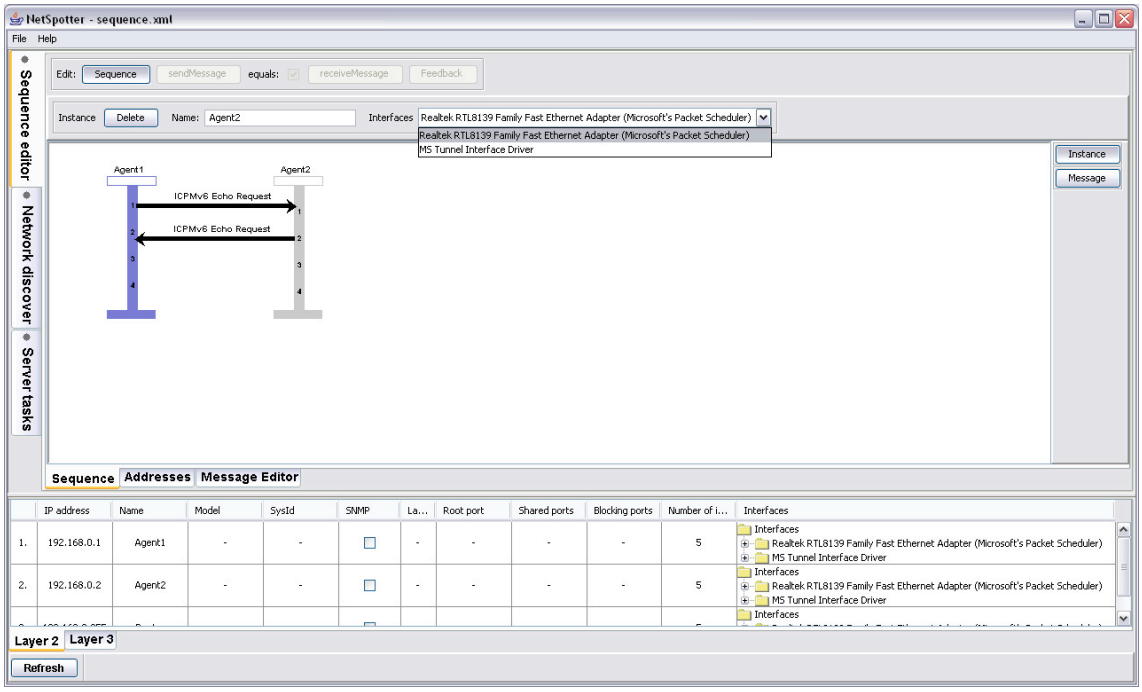


Figure 3-23 Interface selecting

For executing we construct a new task with *Server tasks* tool and under *Activities* tool the *SequenceExecutionActivity* menu with *Browse* button starts the operation between Agents.

With *Refresh* button we can see the result of communication, normal operation indicated with green colour, errors with orange and yellow.

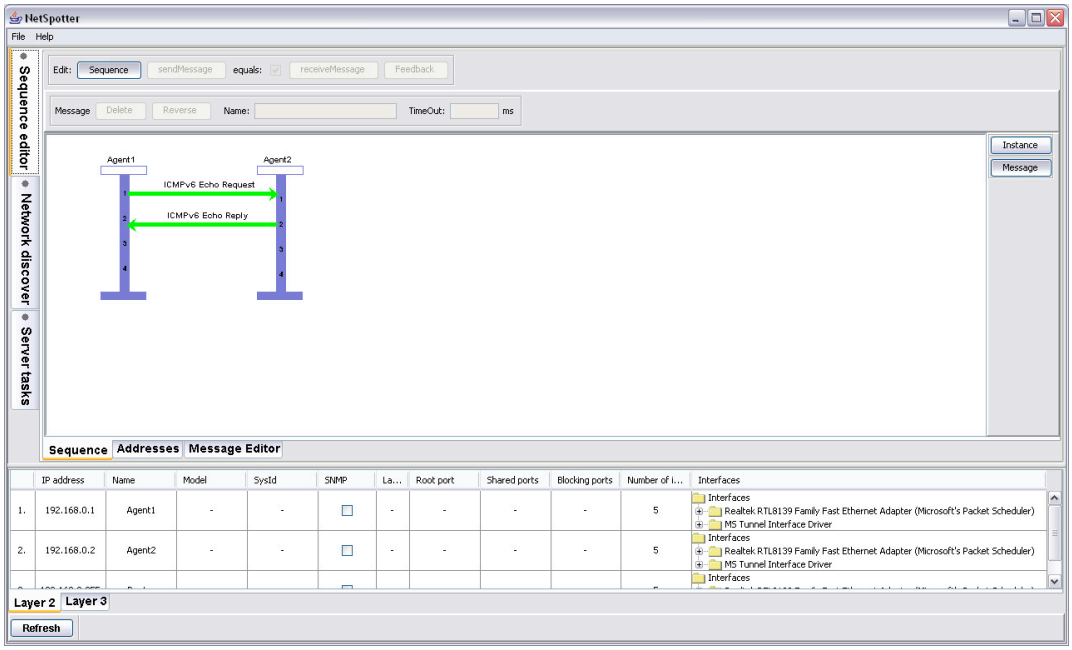


Figure 3-24 Result of Sequence

In *Edit* tool a *Feedback* button shows the real message with time parameters and with read colour those fields are indicated which differ from earlier defined values.

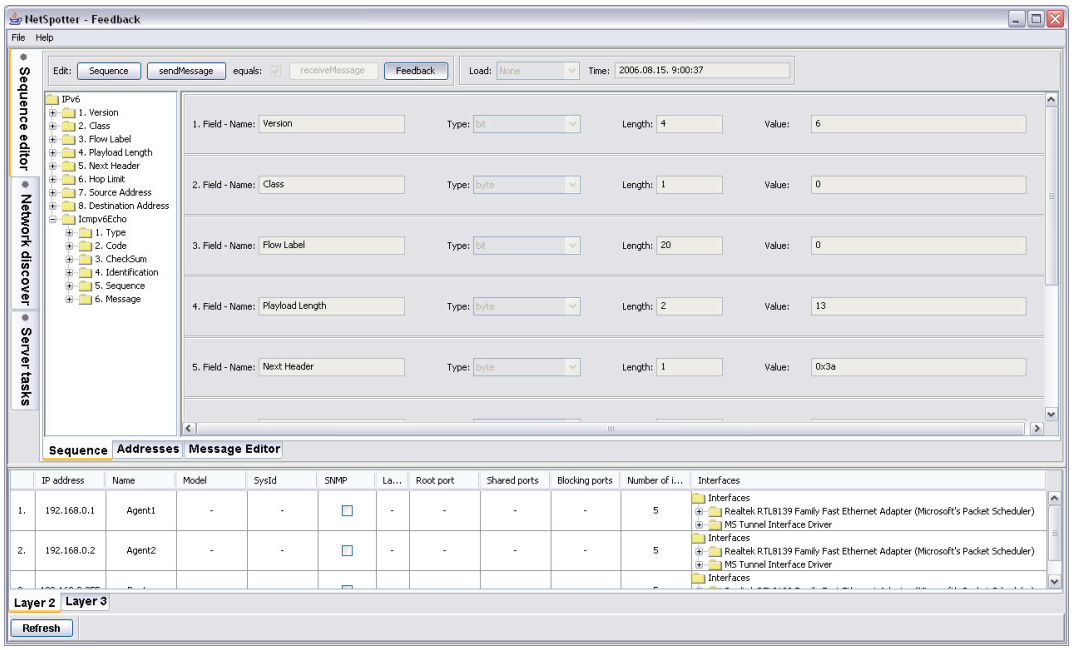


Figure 3-25 Content of Feedback

3.4.2. MLDv2Report Include in multicast channel

Another example is creating MLDv2 Report message for joining to multicast group.

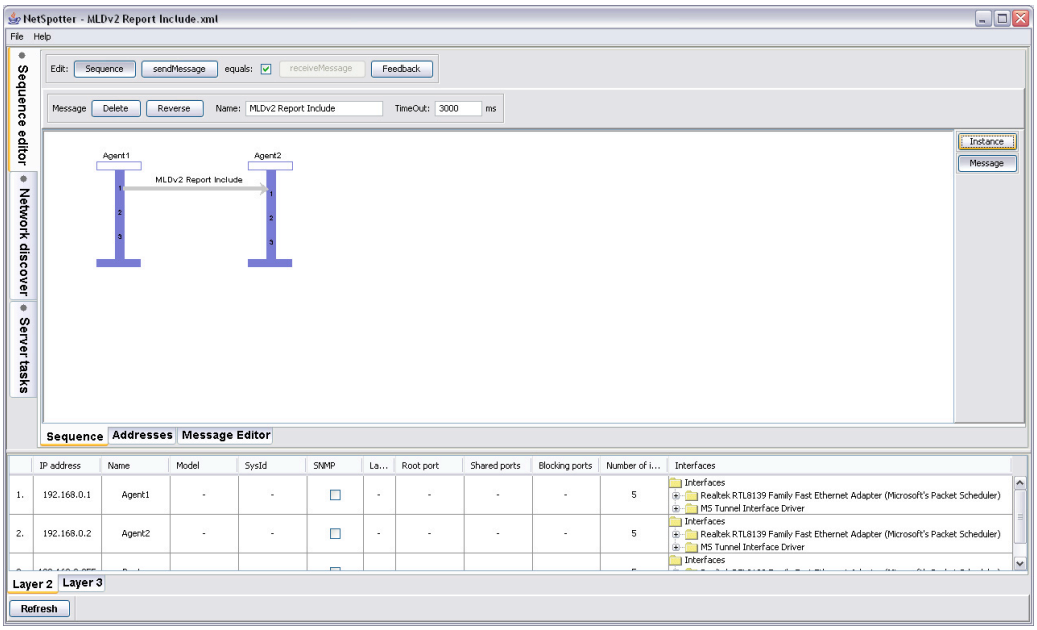


Figure 3-26 MLDv2 message

We enter to edit function with pushing the *sendMessage* button and select from Load list the IPv6 skeleton. Value of Next Header field is set 0x00a. Body is a HopbyHop structure embedded of IP packet.

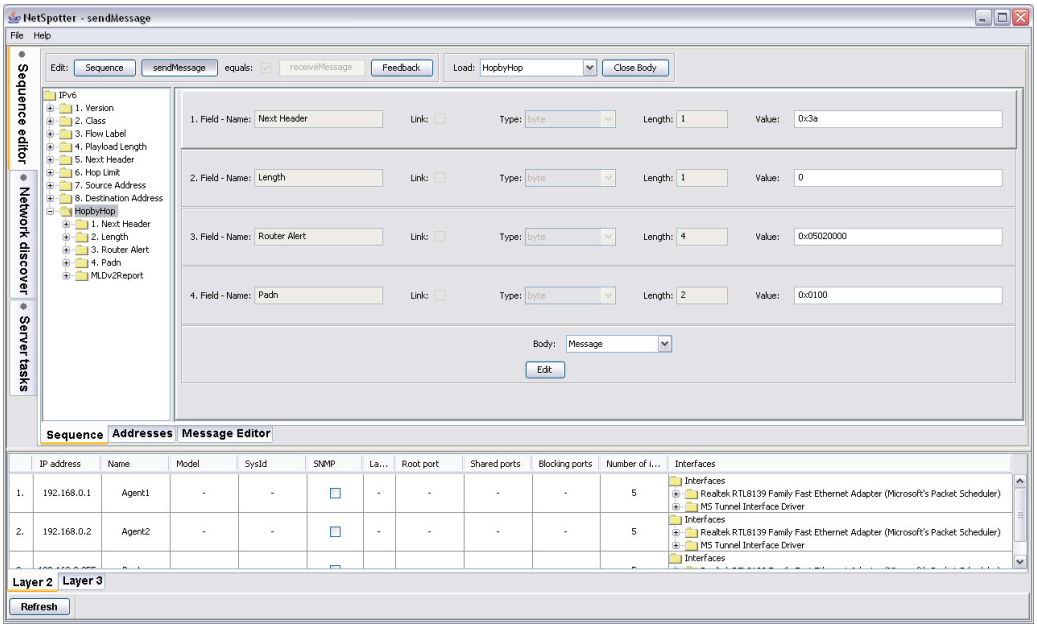


Figure 3-27 HopbyHop IPv6 heading supplement

With *Edit* button we step to embed message, from *Load* list we can choose the HopbyHop message skeleton, the Next Header is set to 0x3a value (ICMPv6 MLD header type). After setting the type of Body to Message we embed MLDv2Report message to HopbyHop message. To join to one multicast channel we push *Edit* button, we chose form *Load* list the MLDv2Report skeleton, we set the Nr of MACddress Records field to 1.

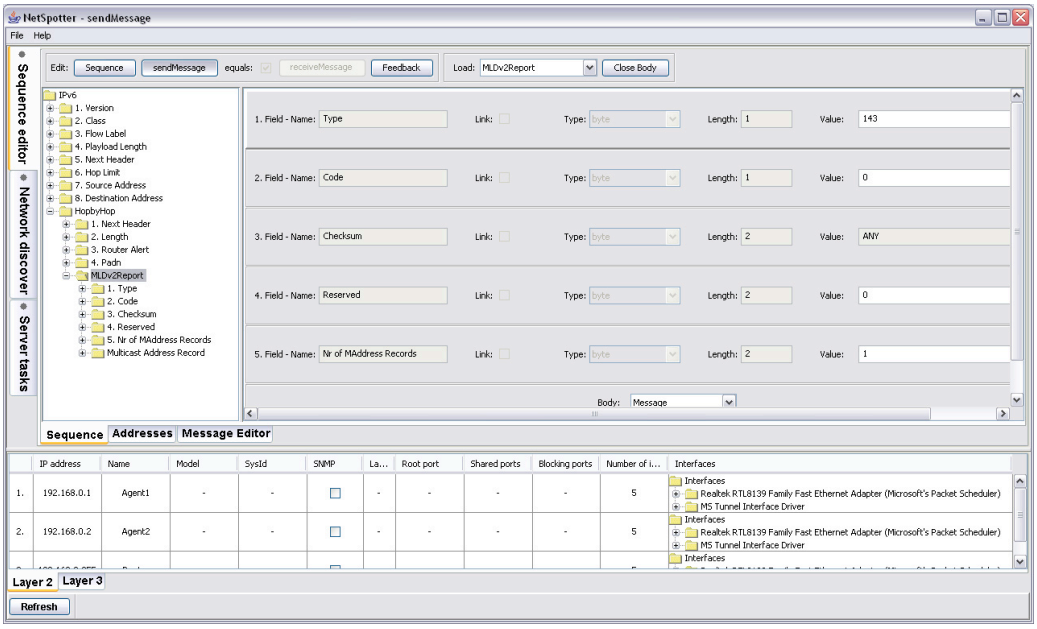


Figure 3-28 MLDv2Report message

Next embedding is to construct MLDv2Report Include message. Steps are: to set Body type to Message (MLDv2Report contains Multicast Address Record), with *Edit* button we are in embedding, from *Load* list we choose the Multicast Address Record skeleton and last we choose the IPv6 address of the Multicast Address field.

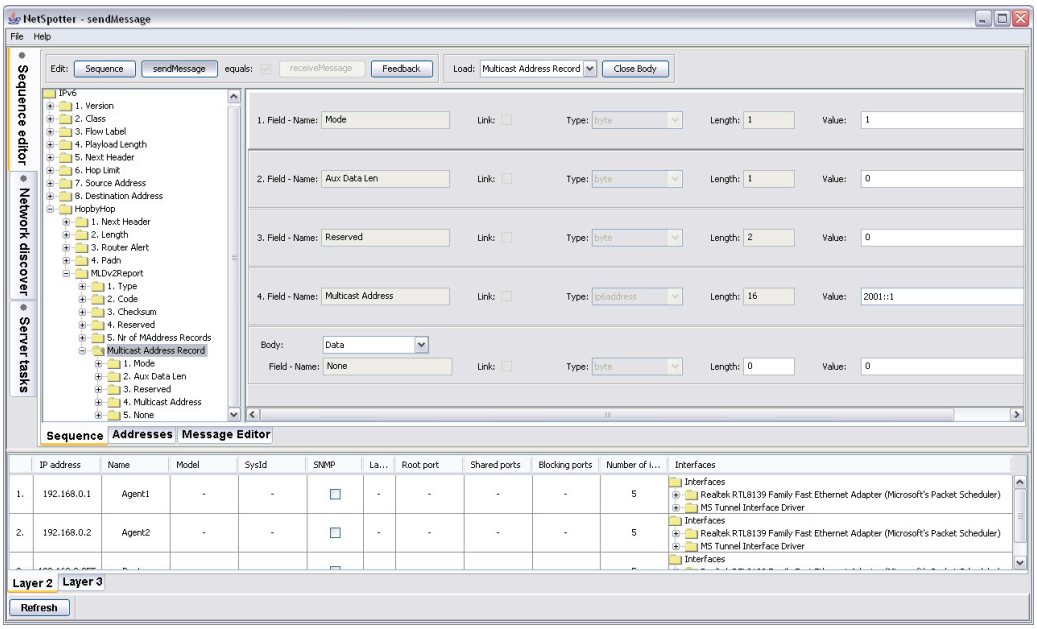


Figure 3-29 MLDv2Report contains a Multicast Address Record message

One interface of Agent2 is assigned for readings the messages of Agent1. Agent1 sends MLDv2Report Include message to Agent2. Feedback shows earlier communication.

4. Discovery of Topology

4.1. Introduction

The topology of networks is very important for informatics managers. Discovery subsystem of Netspotter is planned to do this in the second and the third layer (OSI).
Motivations:

1./ Ethernet does not have OAM (Operation, Administration, and Maintenance) support like SDH, ATM, MPLS. Also it does not have such error control like in IP the ICMP control packets. In future it may change because of the influence of IEEE 802.1ag [1][2] OAM standard.

2./ In the third layer (WAN) we can find effective tools for IPv4 but not for IPv6 protocol. IPv6 data network is coming up and it is useful to prepare for monitoring the huge systems with Agents as main actors in this story.

Discovery subsystem tries to fulfil the above mentioned tasks, all discovered information is saved on discovery server. Discovery is a scheduled task and not a necessary client activity. Earlier discovery information is reachable form archive.

4.2. Discovery workflow

Discovery of topology is based on information of nodes defined by user with IPv4 or IPv6 addresses and SNMP Community address. Client defines (requests) the server activities and represents the result on its GUI.

In this workflow the server plays the main role in scheduled manner. The activity of discovery is unambiguous defined by:

- SNMP list,
- config parameters of discovery,
- type of discovery,
- scheduling of discovery tasks for server.

The discovery may be cyclic. Collected information and discovered topology are stored in the database system by the server. The server tasks defined by the client run parallel according to the time scheduling. The latest topology is on the client screen as a default. The explicit request is necessary to analyse earlier information from archive.

The process is the same in Layer 2 and Layer 3. Two processes can be done independently from each other.

4.3. Controlling of discovery (graphics)

Control of representation is managed with „Network discovers" and discovery with „Server tasks" tools.

4.3.1. Flow control

The „Server tasks" tool is more general than the simple starting-stopping system.
We explain the details of the special parameters, general management specified in 2. Section is valid here also.
Layer2DiscoverActivity and Layer3DiscoverActivity have the same config parameters:

- access list (Ipv4/Ipv6 address – community peer)
- maximal number of concurrent threads
- SNMP UDP port number
- SNMP UDP timeout (ms).

Access list is for SNMP usage. SNMP inactive equipment are missed from actual topology, in the second layer all equipments are visible, in the third layer only routers. We suppose that Agents are not SNMP capable equipments. „Edit" button is for add/modify/delete functions of the list. The „Close" button fixes the list.

The measure of parallelism is set by the number of the threads. The performance of discovery is more sensitive to the number of parallel tasks. The user is responsible for appropriate tuning this parameter.

SNMP UDP port number may be given explicitly by user increasing the effectiveness.

SNMP UDP timeout indicates what is the reasonable waiting time for SNMP reply message is. The older equipments are slower in responding, it is better to give big number to reach them; the timescale is millisecond (ms).

4.3.2. Monitoring

„Network discover” tool has two parts. In upper windows we can see the topology, in lower windows the list of the discovered equipments grouped by layers.

The layer selection is done by „Layer 2” and „Layer 3” menu. Display control is done with „Refresh Display”, „Freeze nodes” and „Release nodes” buttons.

The upper list of nodes (name, model, addresses, interfaces,) serves both for topology presentation and both sequence editor.

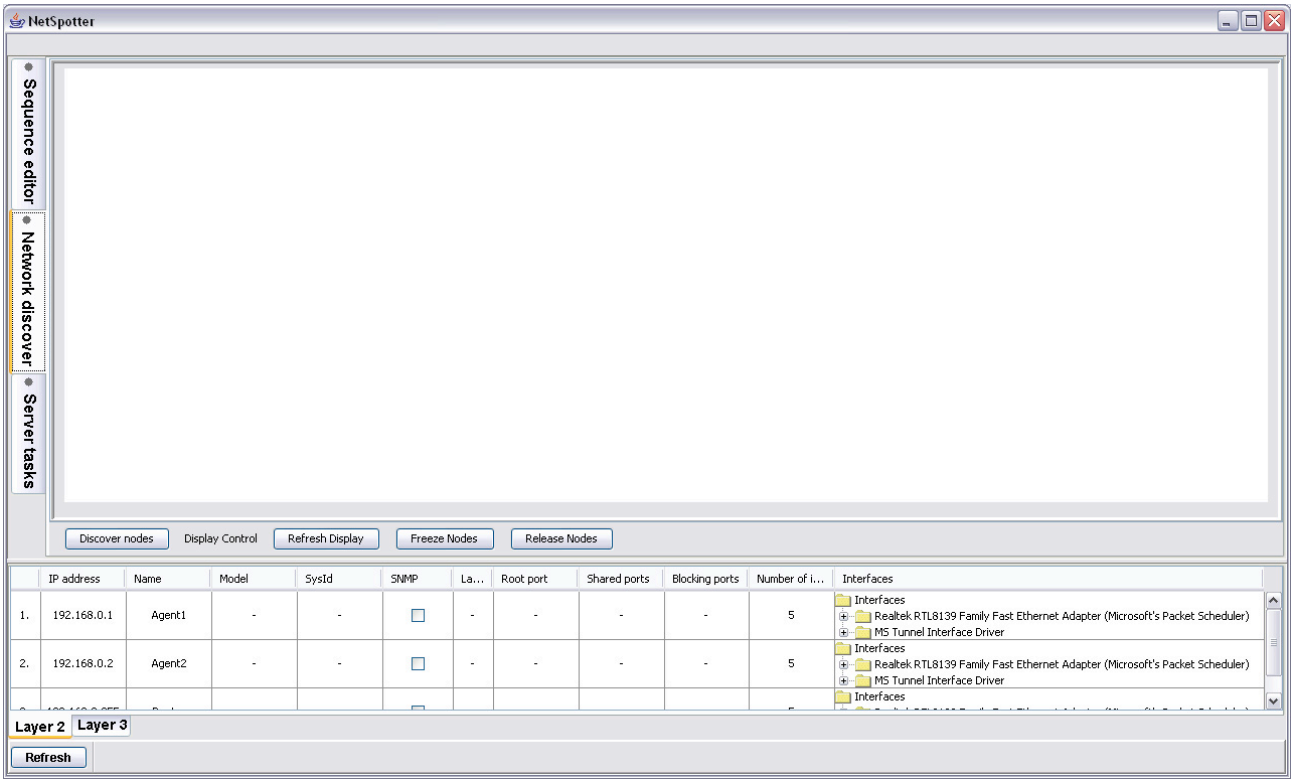


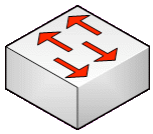
Figure 4-1 Discovery

The menu-line on the top of NetSpotter window depends on the right function selection. „Network discover” tool switches off the top menu-line (no commands in this discovery function).

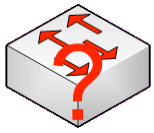
4.4. Topology on screen

Presentation of topology starts after discovery with pushing the „Refresh nodes” button. The topology is actual until the result of next discovery process. If the client has a valid topology, than the client program tries to arrange it well. The User has a possibility to rearrange the icons with „drag and drop” by its own style.

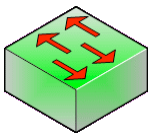
In graphic representation the nodes have icons. These are:



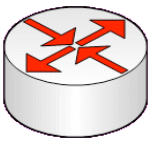
switch (bridge): managed equipment, reply for SNMP request, it is relevant information for discovery of topology



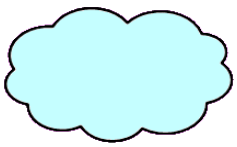
unknown equipment in second layer: indication of some equipment(s) derived from discovery information, not exact discovery, undiscovered area



unmanaged equipment: addressing was successful, but SNMP was unsupported



router: in second layer it is endpoint of segmented network with switches, in third layer it is a traffic node and segments the network



subnets: in third layer they are on interfaces of routers, two types: subnet with global addresses and „Private” with any detailed information

agent: a source/destination of communication sequences, in second layer it is endpoint of segmented network with switches, in third layer it is important entity of container subnetwork(s) indicated by line connection.

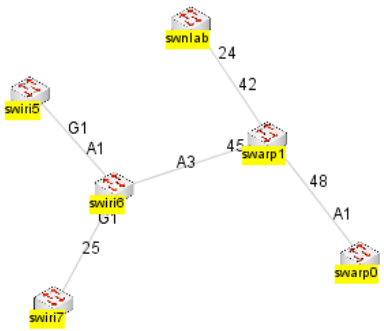
If they have names, than names are put under icon.

The next step is the drawing the graph (links) with text information.

4.4.1. Second layer

In the second layer two nodes are connected, if does not exist between them any discoverable equipment. One exception is when the system finds unknown equipment. On links we can find the number of ports connecting the equipments.

Here is a topology in the second layer, all equipments are managed.

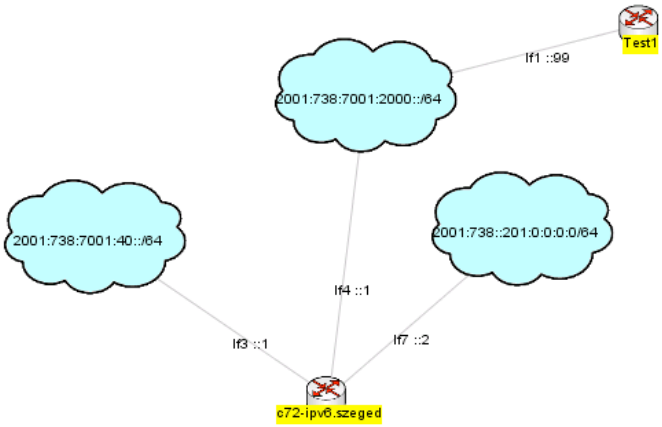


4.4.2. Third layer:

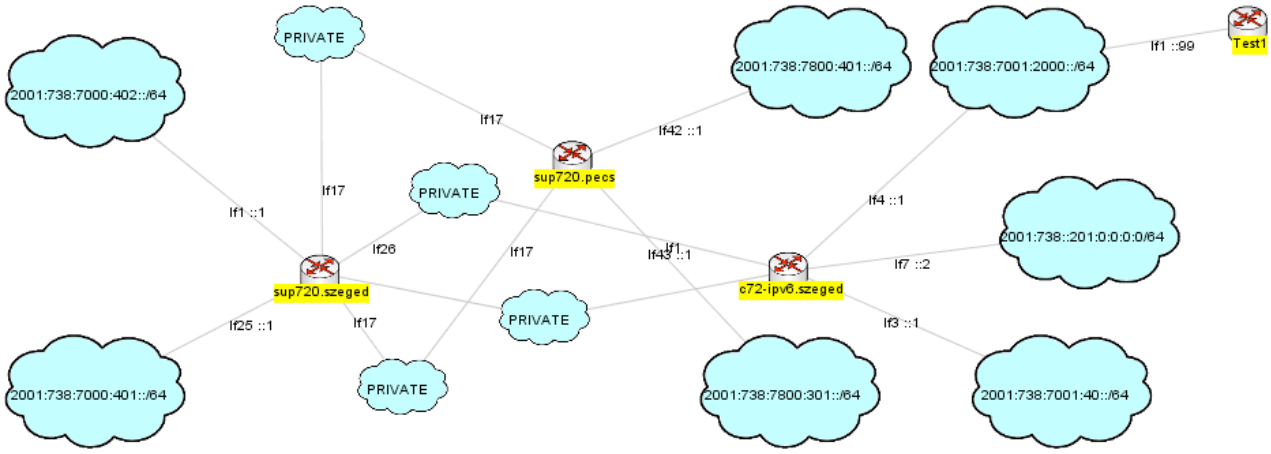
In the third layer two nodes are connected, if the direct traffic is defined between them. Router has subnets and subnets have routers. IPv6 subnet may get a „Private” indication. The subnets with global address - the program puts them to the icons. Agents are in relation with those subnets which they belong to.

The links have special features, that one endpoint is subnet; the information is connected with another node. We use an abbreviation to indicate only the information which is enough to differentiate. For example: if interface of router has IPV6 address: 2001:738:7001:2000::1, and this interface belongs to subnet 2001:738:7001:2000::/64, than on the we can find indication ::1.

In the next picture we can see two routers with common subnet.
c72-ipv6.szeged router has three links and unique (abbreviated) addresses.

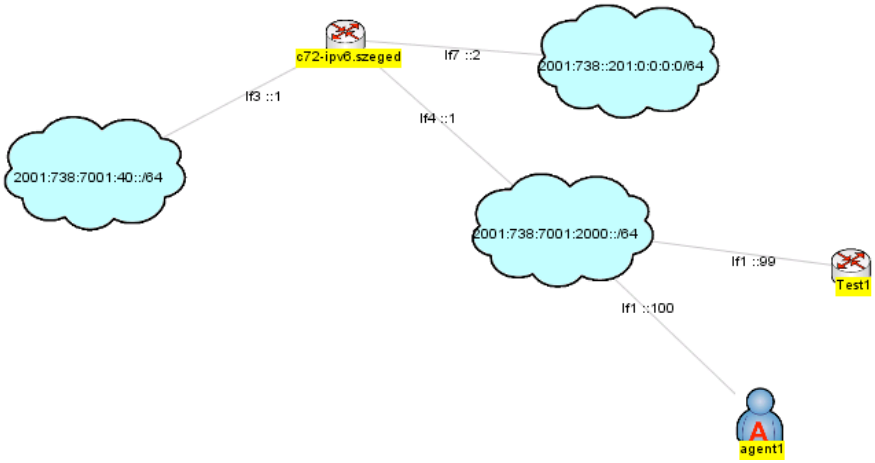


Adding the two new routers, this topology shows the barriers of discovery, not all subnets are discovered, and some are indicated with „Private" indication. The discovery information doesn't guarantee that in „Private" subnet only IPv6 traffic happens, but it is guarantied, that IPv6 communication may operate.

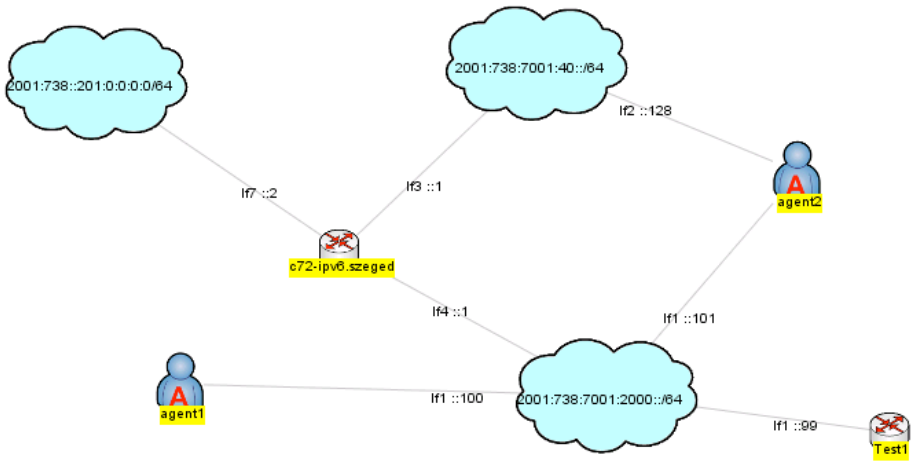


In the next two pictures we can find the agents in the third layer.

On the first only one agent with „1" interface connected to 2001:738:7001:2000::/64 subnet. On link the IPv6 suffix serves to count the global address: 2001:738:7001:2000::100.



The second picture is gained with adding the agent that belong to two different subnets. The agent2 has on „1” interface the 2001:738:7001:2000::101 address and on „2” the 2001:738:7001:40::128 address.



We can analyze more topologies or more representations of the same topology.

4.5. List of discovered network agents

We can find some more details about discovery tables joined to the discovery screen.

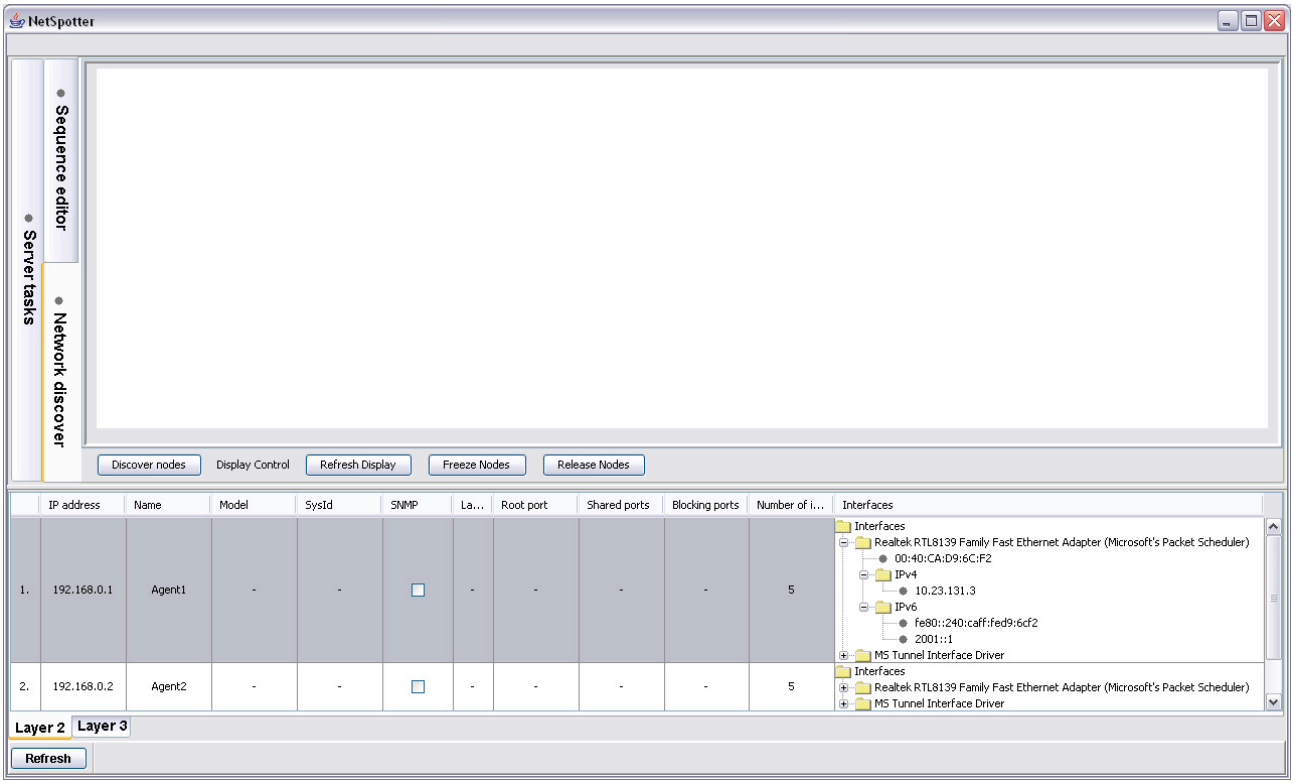


Figure 4-2 List of Agents

This table gives more information about discovered equipments ordered by layers.

4.6. Reusing of earlier discovery results

The users can freely use the earlier gained information for future design, testing and monitoring purposes. If it necessary, it is possible to see on screen the earlier discovery with the tables of the related instruments. The menu of GUI on client side gives the technical aids.

END