ROUTING PROTOCOLS IN IPV4 / IPV6 USING LINUX

Tudor Mihai BLAGA, Student Member IEEE, Virgil DOBROTA, Member IEEE
Technical University of Cluj-Napoca, Department of Communications, 26-28 Baritiu Street, 400027
Cluj-Napoca, Romania, Phones: +40-264-401816, Fax: +40-264-597083, Web: http://www.com.utcluj.ro,
E-mail: Tudor.Blaga@com.utcluj.ro, Virgil.Dobrota@com.utcluj.ro

ABSTRACT. *This paper focuses on the study of routing protocols in IPv4 and IPv6. Their classification is based on type of updating (distance-vector versus link-state), working domains (intra- versus inter-domain), and number of paths (single or multi-path). In addition, some of the routing protocols could involve a flat scheme, whilst others follow mainly a hierarchical mechanism. The performances are highly dependent also on the unicast or multicast way of exchanging information within the autonomous system. The study is completed by practical experiments, using a software routing package under RedHat Linux/ Fedora called zebra. The paper covers the unicast routing protocols: RIPv1, RIPv2, RIPng, OSPFv2, and OSPFv3 for both IPv4 and IPv6.*

I. Introduction

One of the basic functions of the Internet Layer is to forward packets from the source to the destination. This process of moving packets across an inter-network is called routing and the device performing it is an IP router. Routing generally involves the (optimal) path determination and the packet switching [4]. The protocols involved use metrics to evaluate the best single or multiple paths to be followed by a packet in order to reach the destination. To perform the path determination process, the algorithms establish and maintain routing tables. According to this dynamically updated route information, the switching is able to move the packet from a router's interface to another, forwarding the datagram towards the next hop. The static and dynamic allocation of logical IP addresses is described in details in [1], [3]. During the current migration from IPv4 to IPv6, due to the lack of addresses and/or the exponential growth of the routing table size, additional mechanisms were implemented too. Among them, CIDR *(Classless InterDomain Routing),* VLSM *(Variable Length Subnet Mask)* and NAT *(Network Address Translation)* have a certain success. There are no doubts that IPv6 remains the complete solution, expecting to be generalized as soon as possible. For these reasons, the paper is focused on a major problem during the transition from one IP version to another: the routing protocols. Although the approach has taken into consideration the existing achievements in the field (including those from Cisco Systems, the world leader) the practical experiments were conducted in Linux, due to its availability for the new coming IPv6-based routing protocols.

II. Basics of Routing Protocols

The several existing routing algorithms may have different impacts on network but certain common properties are requested for all of them: correctness, simplicity, robustness, stability, fairness and optimality. There are two methods used to build a routing table: static and dynamic. *Static routing* does not involve an algorithm, as the network administrator manually configures the routes. On the other hand, the *dynamic* one utilizes algorithms that

automatically discover, calculate and maintain paths through the network. These are classified into three main categories: *distance-vector, link state* and *hybrid*. The multiplicity of routing protocols could be explained also by the domain they were designed to operate in. *Intra-domain protocols* work only within an AS (*Autonomous System),* whilst *inter-domain protocols* are applied between several routing domains. Note that an autonomous system is a collection of networks under a common administration. It might be possible to have an overlapping of routing domains within an AS, as the router may work with different routing protocols in the same time. However, an *administrative distance* (from 0 to 255) is a rating of the trustworthiness of a routing information source. For a given route, the protocol having the lowest administrative distance will be chosen. By default the distances for a connected interface is 0 and for a static route is 1. The need for balanced utilization of the network resources has lead to complex routing protocols that support multiple paths to the same destination. This evolution requested a different approach. Within the initial *flat routing systems*, all routers were on the same level, all of them exchanging routing information and all being peers of all others. *Hierarchical systems* divide the network into routing areas. Some routers in an area can communicate with routers in other areas, while others can communicate only with routers within their area. The use of hierarchical routing reduces the amount of routing update traffic and simplifies sometimes the algorithms. The process of bringing all routing tables to a state of consistency is called *convergence*. The time it takes a network to converge depends on the routing algorithm, the *convergence time* being one of the most important performances criteria. The continuous circling of traffic between two or more routers is referred to as a *routing loop*. The *count-to-infinity problem* is specific to distance vector protocols only and it consists on a continuous increment of the path cost up to infinity. Obviously, the routing loops and the count-to-infinity problem have to be carefully considered and eliminated. *Table 1* presents some of the most used routing protocols, classified according to the criteria previously discussed.

Routing Protocol	Working Domain		Type		Metrics		IP Version		Administrative Distance	Path		Unicast/ Multicast		Zebra Implementation
	Intra-	Inter-	Distance vector	Link-state	Single	Composite	IPv4	IPv6		Single	Multi	Uni-cast	Multi-cast	
RIP	X		X		X		X		120	X		X		X
RIP-2	X		X		X		X		120	X		X		X
RIPng	X		X		X			X	120	X		X		X
IGRP	X		X			X	X		100		X	X		-
EIGRP	X		X	X		X	X		90/170		X	X		-
OSPF-2	X			X	X		X		110		X	X		X
OSPF-3	X			X	X		X	X	110		X	X		X
MOSPF	X			X	X		X						X	-
IS-IS	X			X	X		X	X	115		X	X		-
DVMRP	X		X		X		X						X	-
PIM	X						X	X					X	-
EGP		X					X		140	X		X		-
BGP-4	X	X	X				X	X	20/200		X	X	X	X

Table 1. Classification of IP routing protocols

II.1. RIPv1, RIPv2 and RIPng

RIP *(Routing Information Protocol)* is a distance vector routing protocol that uses hop count as a metric to determine the direction and distance to any link in the internetwork. It

was designed for small domains, selecting the path with the lowest number of hops (not greater than 15). The routing tables are exchanged through routing updates that are broadcasted periodically every 30 seconds. Split horizon or split horizon with poison reverse are used to prevent the count-to-infinity problem. RIPv1 *(RIP Version 1)* requires all devices in the network to use the same subnet mask, because it cannot include subnet mask information in its routing updates. This is a typical characteristic of a *classful routing protocol*. RIPv2 *(RIP Version 2)* provides support for CIDR and VLSM, being a *classless routing protocol* [3]. It prefers multicasting instead of simple broadcasting of routing announcements. This reduces the processing load on hosts that are not listening for RIPv2 messages. Additional support for authentication and fully interoperability with RIPv1 are provided. RIPng *(RIP Next Generation)* was developed to allow routers within an IPv6-based network to exchange information. Although it uses the same algorithms, timers and logic as in previous version, there are two major differences. First, the RIPng does not include any native authentication support, relying on security features offered by IPv6. Second, the packets were updated to support the 128-bit IPv6 address format. RIPv1, RIPv2 and RIPng have the same limitation as any distance vector routing protocol, which is the slow convergence. An additional drawback such as the path cost restriction (maximum 15 hops) might be eliminated within a Cisco routers only, by choosing IGRP *(Interior Gateway Routing Protocol)* (up to 255 hops).

II.2. OSPFv2 and OSPFv3

OSPFv2 *(Open Shortest Path First Version 2)* is a link-state routing protocol developed to address the limitations of RIP. It provides immediate propagation of routing updates because they are events triggered, offering faster convergence and no cost limitations. OSPF was from the beginning designed as a classless routing protocol, by supporting CIDR and VLSM. Being very complex, it permits hierarchical routing. Thus OSPF networks are divided into a collection of areas (logical groups of networks and routers). Area 0 (known as backbone area) physically connects to all other areas. Each router has a view of the entire network (a routing loop-free approach) and it executes the SPF *(Shortest-Path First)* algorithm to determine the routes to the destination. Routing information is flooded by using LSA *(Link State Advertisement)* packets to the following IPv4 multicast addresses: `224.0.0.5` for OSPF routers and `224.0.0.6` for Designated/Backup Designated routers.

OSPFv3 *(OSPF Version 3)* was designed for both IPv4 and IPv6. Obviously, some modifications were needed due to the changes in protocol semantics or simply to handle the increased address size. New LSAs have been created to carry IPv6 addresses and prefixes. Authentication has been removed from the OSPF protocol itself, relying on IPv6's AH *(Authentication Header)* and ESP *(Encapsulating Security Payload)* [1]. Although the SPF algorithm was employed again, the multicasting is using to distinct IPv6 addresses: `FF02::5` for all OSPF routers and `FF02::6` for all Designated/Backup Designated routers. Note that the packets sent to these multicast addresses should never be forwarded because they are meant to travel a single hop only. This is the reason to have multicast addresses with link-local scope and packets sent to these addresses should have their IPv6 Hop Limit set to one.

III. Zebra: A Linux-Based Software Tool

Zebra is a Free Open Source package running under GNU/Linux, FreeBSD, NetBSD, OpenBSD, Solaris and providing (simultaneous) real routing services based on a collection of routing daemons, as in *Table 2*.

Routing daemon	TCP port	Routing protocols
`ripd`	2602	RIPv1, RIPv2
`ripngd`	2603	RIPng
`ospfd`	2604	OSPFv2
`ospf6d`	2606	OSPFv3
`bgpd`	2605	BGP-4, BGP-4+

Table 2. Routing daemons in Zebra under RedHat/Fedora

As a major advantage, Zebra is also ready to support IPv6-based routing protocols (which is not the case of the Cisco routers available in our academic network). At least two Linux boxes with Zebra are needed for tests. The first one starts an integrated shell (`zebra vty`) at TCP port 2601, allowing to change the configuration and to display the routing table. The second Linux box starts a dedicated routing daemon. The machine exchanges routing information with other routers using the previously mentioned protocols and updates the kernel routing table. Note that the default commands for a Linux-based router configuration are `ifconfig` and `route`, whilst the status of routing table is displayed by `netstat`. These commands work only if the user has root privileges. On the other hand, Zebra is administrated in a different way. Actually, there are two modes: *normal* and *enable*. Within the first one, the user can only view system status but within the *enable mode*, he/she can change system configuration (does not matter his/her rights in Linux). Another option for testing would have been `gated`, but unfortunately, this routing daemon originally coordinated by Cornell University is not free, currently being developed by Merit GateD Consortium.

IV. Experimental Results

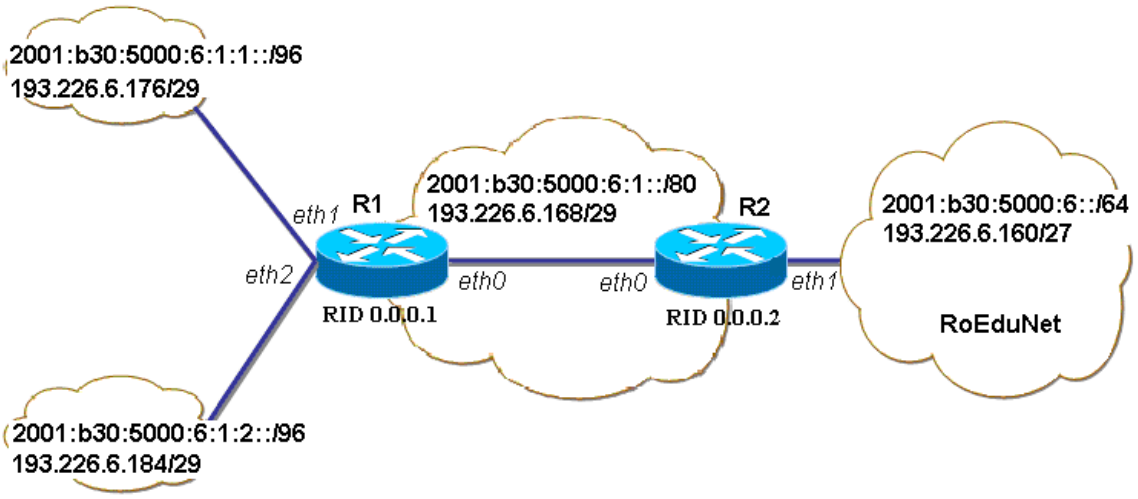

Figure 1. Testbed

The practical experiments regarding the performance evaluations of the routing protocols are under progress. This paper is discussing only the unicast tests using Zebra 0.94 under Red Hat 9.0/Fedora Core 1, for both IPv6 and IPv4. Apparently, the network topology described in *Figure 1* is not very complex but it is an excellent testbed for the study proposed. There are two routers, R1 and R2, actually two Linux machines, each running Zebra in two Linux boxes (as it was explained in section III). The following subsections present the configuration files of for each routing daemon (`ripd`, `ripngd`, `ospfd` and `ospf6d`), as well

some preliminary results. IPv6-based experiments are of a greater interest than those related to “classical” IPv4, so the comments are concentrated on the new coming routing protocols.

IV.1. Unicast Routing Protocols in IPv6

The configuration file for the Zebra daemon is called `zebra.conf`, containing the router setup (hostname, password, enable password, IPv4/IPv6 interfaces addresses). *Figure 2* presents an example for router R1 that has three interfaces (eth0, eth1, eth2).

```
hostname R1
password zebra
enable password zebra
!
interface eth0
 ip address 193.226.6.169/29
 ipv6 address 2001:b30:5000:6:1::169/80
!
interface eth1
 ip address 193.226.6.177/29
 ipv6 address 2001:b30:5000:6:1:1::177/96
!
interface eth2
 ip address 193.226.6.185/29
 ipv6 address 2001:b30:5000:6:1:2::185/96
```

Figure 2. Configuration file for router R1

The routing daemons involved were `ripngd` and `ospf6d`. Each daemon had its own configuration file called `*.conf`. Supposing the case of a single area OSPF (area 0.0.0.0), with the router-id for R1 being 0.0.0.1 (R2 has a router-id equal to 0.0.0.2), the configuration files applied to router R1 are presented in *Figure 3.*

```
hostname ripngd
password zebra
!
interface eth0
!
interface eth1
!
interface eth2
!
router ripng
 network eth0
 network eth1
 network eth2
 redistribute connected
```

(a)

```
hostname ospf6d
password zebra
!
interface eth0
 ipv6 ospf6 cost 1
 ipv6 ospf6 hello-interval 10
 ipv6 ospf6 retransmit-interval 5
 ipv6 ospf6 priority 1
!
router ospf6
 router-id 0.0.0.1
 interface eth0 area 0.0.0.0
 interface eth1 area 0.0.0.0
 interface eth2 area 0.0.0.0
 redistribute kernel
```

(b)

Figure 3. Configuration files for routing daemons in IPv6: (a) `ripngd` (b) `ospf6d`

The proper operation within zebra could be seen by analyzing the entries from the routing table or the results of the command `ping6`. For instance, the information got from R2 must contain the routes learned from R1 (either by RIPng, either by OSPFv3). No static routes have been previously configured, so if the routing protocol does not work properly the networks connected to *eth1* and *eth2* at R1 cannot communicate with the network connected to *eth1* at R2. There are two ways of analyzing the routing table: a) from the zebra daemon with the command: `show ip route` b) from the Linux shell with the commands: `netstat`

`-r` or `route –A inet6`. For a better understanding, we can capture the packets with the software packet analyzer called Ethereal. For further testing of our IPv6 testbed, a failure situation must be provoked. Suppose one of the networks connected to router R1 will be disconnected either by unplugging the network cable or either by shutting down the interface. The time it takes for the new routing information to reach router R2 can give us the convergence time for the given routing protocol.

IV.2. Unicast Routing Protocols in IPv4

The same procedures were applied for IPv4, except the types of daemons started, which were in this case `ripd` and `ospfd`. *Figure 4* shows the requested configuration files.

```
hostname R1
password zebra
!
interface eth0
 ip rip send version 1 (2)
 ip rip receive version 1 (2)
!
[…]
!
router rip
 version 1 (2)
 redistribute connected
 network eth0
 network eth1
 network eth2
```

(a)

```
hostname ospfd
password zebra
!
interface eth0
 ip ospf cost 10
 ip ospf priority 10
!
[…]
!
router ospf
 ospf router-id 0.0.0.1
 redistribute connected
 network 193.226.6.160/29 area 0
 network 193.226.6.176/29 area 0
 network 193.226.6.184/29 area 0
```

(b)

Figure 4. Configuration files for routing daemons in IPv4: (a) `ripd` (b) `ospfd`

Note that the two version of RIP (RIPv1 and RIPv2) can be configured thus resulting two routing scenarios. The differences presented in section II.1 are easy to be notices from the packets captured. First, the destination address is `255.255.255.255` (local broadcast) for RIPv1 and `224.0.0.9` (multicast) for RIPv2. Second, the version field and the routing information carried are different.

V. Conclusions and further work

The paper was focused mainly on the study and configuration of the routers working with IPv4/IPv6-based routing protocols. As the experiments are under progress, the networks reachability was the preliminary criterion for successful tests. Regarding the convergence time OSPFv2/OSPFv3 needed up to 3 seconds to learn about the changes, whilst for RIPv1/RIPv2/RIPng it took more than half a minute.

References

[1] V. Dobrota, *Retele digitale in telecomunicatii. Volumul 3: OSI si TCP/IP*, Editia a II-a, Editura Mediamira, Cluj-Napoca, 2003

[2] A. Rodriguez, J.Gatrell, J.Karas and R. Peschke, *TCP/IP Tutorial and Technical Overview*, IBM, August 2001

[3] A. Tanenbaum, *Computer Networks*, Fourth Edition, Prentice Hall, 2003

[4] ***, *Internetworking Technologies Handbook*, *Cisco Systems*, 1-58705-001-3, 2002

[5] ***. http://www.zebra.org

[6] ***, http://www.redhat.com