Optimised Routing in Hybrid Active Networks

Farouk Belghoul[1]
Institut EURECOM
Sophia Antipolis
France

Farouk.Belghoul@eurecom.fr

Olivier Marcé
Alcatel Research & Innovation
Route de Nozay
F-91461 Marcoussis
France
Olivier.Marce@alcatel.fr

Abstract

Nodes in active networks are allowed to execute some code in order to provide new services. This paper tackles the optimisation of the routing of packets transporting the code, with the objective of favouring the routes towards the active nodes able to execute the code. The proposed solutions is based on the extensions of classical routing protocols. Active network specific information is shared between the routers, and existing algorithms are modified to take these information into account. An implementation for OSPF is described.

1 Introduction

The impressive growth of IP networks during the last few years made IP technology the common network infrastructure for a wide range of applications and domains, from wired data networks to mobile Voice Over IP. In the meantime, new requirements in terms of dynamic configuration, flexibility and content adaptation have emerged, with the objective of providing a better service to the end-user at the lowest possible cost for the operators. One of the promising evolution of traditional packet based network is the Active Network concept which proposes to make the networks able to dynamically provide new services, on demand. After more than 5 years of intensive study, the concept has been fully proved as feasible and able to support the new requirements for flexibility. Nevertheless, several important issues are still opened, before the Active Network would be an industrial reality.

This paper addresses the issue of deployment of Active Network in a near future, from the routing protocol point of view. The considered case is the progressive integration of active nodes in a traditional network, and the objective is to optimize the active nodes utilization and the active IP packet routing. The proposed open solution is supported by an existing routing protocol and its implementation. It is based on the OSPF optional mechanisms defined in a previous IETF draft [22] to automatically identify Active Node capabilities in the network. It proposes two different algorithms to exploit those information in order to have an adaptive, flexible and optimized IP routing mechanism for the active IP packets.

The next section is a rapid overview of the Active Networks, and the context of the work is detailed. Section 3 describes the issues of routing packets in a network composed of both active and passive nodes. The section 4 proposes a solution for sharing information about activeness of router, and section 5 details two algorithms for improving the routing of packets in an active network. The implementation supported by the OSPF routing protocol is described in section 6. Section 7 concludes the paper and gives the future orientation of our work.

2 Active Network overview

An Active Network is composed of Active Nodes which are able to process the data traversing them [3]. This paper deals with the IP networks, then the nodes with active function are the Active Routers. Such routers provide the classical forwarding and routing functionality as well as advanced routing features like QoS management or Traffic Engineering. In addition, the Active Routers are able to retrieve executable code, which can specifically process the packets which are forwarded. The execution of the code takes place into an *Execution Environment* (EE) [19][3][1].

Several architectures have been proposed for Active Network. In one of these approaches, the code to be executed is entirely contained into an IP packet, and the scope of the execution is limited to the packet itself [5][21]. In this case, every packet transports the code which is used to forward it at each node. Another approach, the discrete one, is based on the concept of *capsules* [1]. Capsules contain code, or a reference to code, which is installed into the node in order to further process the packets following the initial capsule. The exact way of transporting the code

[1] This work has been done at Alcatel R&I Marcoussis

[17][18], the use of embedded or referenced code, as well as the code's language are not considered here.
From now on we will use the following definitions: A *Passive Node* (resp. Router) is a node (resp. router) providing the conventional IP network functionality. An Hybrid Active Network, *Hybrid Network* for short, is a network composed of both Active Nodes and Passive Nodes. An Heterogeneous Active Network, *Heterogeneous Network* for short, is an Active Network composed of Active Nodes with different EE. The code transported into the capsule or retrieved thanks to the transported reference is called *Active Code*. The *Active Application* is the result of the execution of one or more Active Codes on one or more Active Nodes.
The rest of the paper is focused on a Hybrid and Heterogeneous Active Network, and defines mechanisms to deal with Active Node detection and Active Code routing in this network.

3 Routing in Active Networks

Several routing mechanisms in an Active Network are possible and have been studied.
In the hypothesis that the network is entirely composed of Active Nodes, the routing issue is the same than for a passive network Although this assumption is viable, it would restrict the potential wide deployment of Active Nodes to fully active networks only. This would implicitly mean that the Active Network technology could be deployed into small size LAN only.
In this paper however we will consider *Hybrid Networks*, where both passive and active nodes coexist: this is the typical scenario that will happen in the MAN/WAN environment. There are two ways to consider transport of capsules toward the Active Nodes which would further process the packet.
In the *explicit addressing* approach [5], capsules are explicitly sent to the target Active Nodes. This implies that the sending application must know the Active Nodes addresses and, in the Hybrid Network context, have a topological knowledge of the network in order to locate the Active Nodes. If the destination of a capsule is the Active Node N1, every node routing the capsule, no matter if passive or active, will simply forward the packet toward N1: the Active Nodes on the route between the sender and N1 will act as Passive Nodes. This requires the application to control the way capsules are treated in the network, and to perform the routing task which is normally delegated to the network. This approach does not allow to process application-level flows: sender S that wants to send a flow to D to be processed by active code, would first need to find all the routes to D and then to explicitly sends capsules to the intermediate Active Nodes as described previously. The general issue of dynamic routing is left for further study as explained in section 7.
The second approach, which will be studied in this article, is the *implicit addressing* [18] of Active Nodes: capsules and data packets sent from host A to host B will be considered only by the Active Nodes traversed from A to B. In this approach there is no need for the sender to be aware of the topology of the active network, the location and addresses of active nodes, etc. The drawback is the lack of control concerning the deployment of Active Code within the network: Active Nodes which are not traversed by the capsules will not have the code installed.
In this article we propose a mechanism allowing the routing of packets into a Hybrid Network and taking into account the existence of capsules among the packets to be routed. The mechanism will allow the usual routing of passive packets, and will provide the ability to route the capsules, and their associated Active Code, toward the Active Nodes considered as the best suited to perform active application.
The routing criteria considered are the following.

A capsule must be routed in priority towards:
1. a node able to execute the Active Code (*execution criterion*)
2. among these nodes, the node where the active code will have the most important impact (*efficiency criterion*)

Noting it is worth that, in case of an *Heterogeneous and Hybrid Network* it is not sufficient to route the packets toward an Active Node. The EE supported by the Active Node (or one of the supported EE if the node is multiple EE capable) must be able to execute the Active Code.

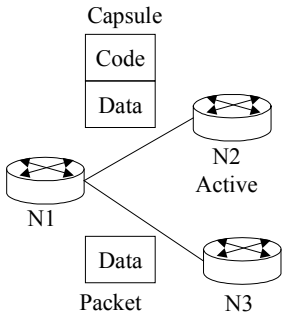

Figure 1 Different routes for packets and capsules

The efficiency criterion completely depends on the Active Application: for example, in the case of an Active Application providing on the fly transcoding of a multi-media flow, the Active Code would be better executed on Active Nodes located at the boundary of two networks with different transport or bandwidth characteristics. As a second example, an Active Application being designed to respond to a detected network congestion [2] or a network intrusion would not have specific placement requirements, its only

objective being to be executed on the Active Node *closest* to the point where the trouble occurs. Some other Active Applications could require the capsule to be executed on the highest possible number of Active Nodes, within the shortest path (or the lowest cost path). The policy of dynamic placement of the Active Code will therefore be different from one Active Application to another. An efficient routing mechanism in a Heterogeneous and Hybrid Network would need to meet the efficiency criterion for the different Active Applications. This would require to have as many routing mechanisms flavors as the number of possible Active Applications.
This point is left for further study and is listed in section 7.

4 Opaque LSA for sharing Active Network information

Among intra-area routing protocols, OSPF has been chosen to support the extension for sharing the Active Nodes capabilities information in an Hybrid network.

4.1 OSPF Overview

The routing protocol OSPF (Open Shortest Path First) [11] defines routing entity called *area*, which allows routing information isolation, hence a reduction of amount of exchanged signaling.
Sharing information between OSPF daemon is mainly done thanks to the LSA packets (Link State Advertisement). Such a packet contains data with a scope of either the link on which the LSA is sent (such a LSA is of type 9), either with a scope of the area (type 10) or network wide (type 11).
Some extensions have been defined for OSPF, with the objective of carrying more information than those initially defined. The Opaque LSA [12] has been designed for distributing any type of information. The basic mechanisms of the protocol see such LSA as bytes, hence the Opaque denomination. The Opaque LSA packets have a standard LSA header, followed by a specific 32 bits length field.

4.2 The solution

The proposed solution is based on the Opaque LSA which populates the activeness status of the Active Nodes. The Opaque-Type field is used to indicate that this Opaque LSA has been sent by an Active Node, and that it contains information about its activeness into the Opaque information field. The Opaque Type value would be registered to IANA [4] in order to be standardized.
Two type of Active Opaque LSA are defined. Type 1 packets inform the other routers in the area of the capacity of the Active Node, whereas type 2 packets are sent simultaneously to the Network LSA and list all the active routers in the area.
The Opaque LSA transported data is composed of the following fields:

- Active Packet type
- EE Id: this is the unique ID of the Execution Environment type. This information is used to determine the route of a capsule containing an Active Code, needing to be executed by a specific EE (e.g. a JVM [7], or a PLAN [6] interpreter)
- EE Info Length: size of the additional information concerning the EE.
- EE info: this field contains supplementary information about the EE (if needed) like its configuration or the list of available libraries. With such information, a router is able to chose the route for a capsule requiring specific EE and library.

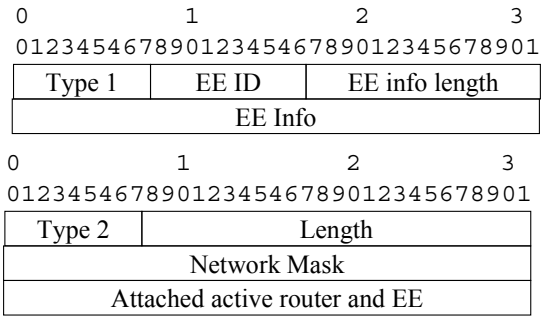

Figure 2 Active information formats

5 Adaptive Routing Algorithms for hybrid Active Networks

The routing operation consists in finding the best route to establish a communication between two nodes, with respect to a given criterion. In most of the case, the criterion to be satisfied is a cost function, with the objective to consume the minimum of (network) resources.
The routing protocols are used to let the nodes communicating with each other, and exchanging data about their links (cost, status, etc.). The routing daemons compute a representation of the network in the form of a weighted graph where the vertices represent the nodes and the edges symbolize the links between nodes. The weight of the edges are dependent on the link and on the criterion to be satisfied when the packet is routed. The routing table is built from this graph, by using a shortest path first algorithm.
Two types of algorithms are used. The Vector Distance Algorithm, based on the Bellman-Ford algorithm [8], is the one used in RIP [9][10], whereas the Link State Algorithm [8], based on Dijkstra's Short Path First algorithm is used in OSPF.

RIP and OSPF, as well as IS-IS, are the most widely deployed routing protocols for intra-area routing.
In order to be able to route capsules with respect to the execution and efficiency criteria, the routing protocols must be extended in the following manners.

1. Communication between daemons must include information related to the active capacity of the nodes. At least the following information must be shared:
   - the *activeness* of the node (i.e. if is it able to recognize a capsule)
   - and if active, the set of EEs it supports. In this paper, the EE is considered as self-sufficient, i.e. it is able to access all the resources that it needs by itself (mainly, libraries). It would be necessary to also consider the available configuration of the EE as an information to share.
2. The routing daemon must be able to compute a specific routing table, using the traditional graph of distances, and of the information about active nodes.

In addition, the forwarding plane (e.g. the Linux kernel, see section 6) must be able to route in a different way packets and capsules. When dealing with flows of data, the differentiation should be done not only between capsules and packets, but also between packets of passive flows, and packets from flows needing active processing. This issue is more detailed in the section 7.
The extension of the routing protocols for supporting the specific data exchanged for routing in an Active Network is addressed in the next section, with the support of OSPF.
The following proposals of routing algorithm modifications aim to satisfy the different criteria in an Active Network (execution and efficiency criteria).

5.1 Attractiveness Algorithm

This approach allows to choose for a capsule a route with Active Node rather than a route with Passive Nodes only, in a configurable mmaner. This can be exploited by solutions needing a different routing table for different applications (see section 7).
For promoting the routes toward the Active Nodes, it is proposed to use the Dijkstra's algorithm on a graph on which the weight of the link toward an Active Node (with the required EE) is divided by a numerical non null value X. This modification provides a very simple way to route capsules toward the Active Nodes. But is does not ensure that the capsule will always be routed to the best suited Active Nodes, nor to any Active Node. This lack of guaranty of routing to the Active Nodes is the counterpart of the fact that, even if there is no suited Active Node, the capsule will still be routed toward its destination. One could object that in some case, this lack of execution would result in a complete failure in delivering the correct data to the destination. This would mean that the Active Applications is designed to make a replacement of the transport function of the network. If this can be a possible usage of the Active Networking concepts, it is expected that the Active Applications are always providing an *additional* service to the transport service, and that the failure of the value-added service would never result in the degradation or failure of the basic service.
In order to increase the attractiveness of Active Nodes, it is possible to improve the algorithm by applying the X divider to the links which are indirectly connecting some Active Nodes (i.e. a link L1 to a node N1 which in turn is sink of a link L2 toward an Active Node N2).
The following is the pseudo-code for the algorithm:

```
Parameter X : attractiveness factor
Parameter P : indirect attractiveness steps number
For each active node N
        Attract(N,X,P,Null)

Function Attract
Parameter N : node
Parameter X : attractiveness factor
Parameter P : recursion number
Parameter I : initial node
If P>1 Then
For each link L from N' to N
        Weight(L)=Weight(L)/X
        If N' is different than I
                Attract(N',X/2,P-1, I)
```

The following figure depicts the execution of the Attractiveness Algorithm in a network with two Active Nodes (N2 and N6). Not all the links have been drawn for presentation purpose. The weights W5, W2 and W6 are divided by X, because the corresponding links are going to Active Nodes. The link valued by W3 is indirectly going to the Active Node N2 after two steps, then its weight is divided by X/2. The weight W7 is divided by $(X/2)^2$ due to the fact that there are two links from N3 to an Active Node. The weight W4 is kept unchanged.

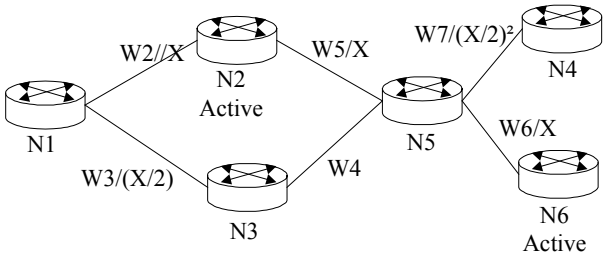

Figure 3 Attractiveness Algorithm result

5.2 Slice Based Routing

This algorithm is better suited to Active Applications that need that the capsules are executed on some identified Active Nodes, with regards to the destination host. An example of this is the intruder tracking, where the Active Code aims to be deployed on the Active Node the closest to a suspicious host. This solution is a kind of explicit addressing.

In this solution, if the destination host is known by the sending host (i.e. they exchanged routing information, they are in the same routing area) the capsule can be explicitly routed toward the Active Node which is the best suited from the sending host point of view. If the sending host does not have enough information, which is the normal case in the Internet, it sends the capsule to the closest Active Node in direction of the destination. This Active Node would then decide if it executes the Active Code, or if it forwards it toward another Active Node which is known to be closer to the destination host. This second solution would require that Active Nodes have additional information concerning the topology of the active part of the network. This could be provided, for example, by an extension of a protocol like BGP.

6 Implementation Issues

The proposed Active network Opaque-LSA extension of OSPF, has been implemented over the routing stack Zebra [13]. This is a modular and open source routing daemon for Linux. Its architecture allows to manage several routing protocols, which communicate with a Zebra daemon which in turn updates the kernel forwarding table.

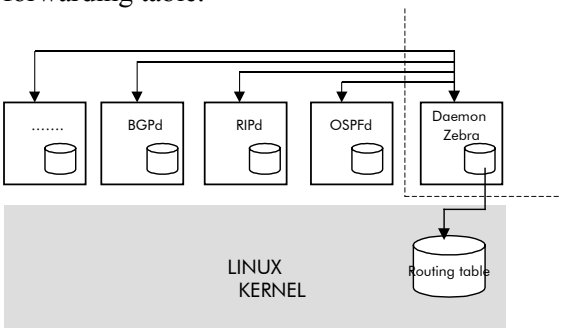

Figure 4 : Zebra global architecture

The work has been done on the version 0.91a, with a specific patch for the Opaque LSA [14].

The main modifications are the following.

- A new register type for the Opaque LSA has been defined : `OPAQUE_TYPE_ACTIVE_NETWORK_LSA`
- The functions called when an event occurs have been implemented (e.g. ospf_active_network_lsa_originate). They create the Opaque LSA containing the information described in section 4.2.

These modifications allowe to exchange information about active capability of routers, and to store this information in the Link State Database.

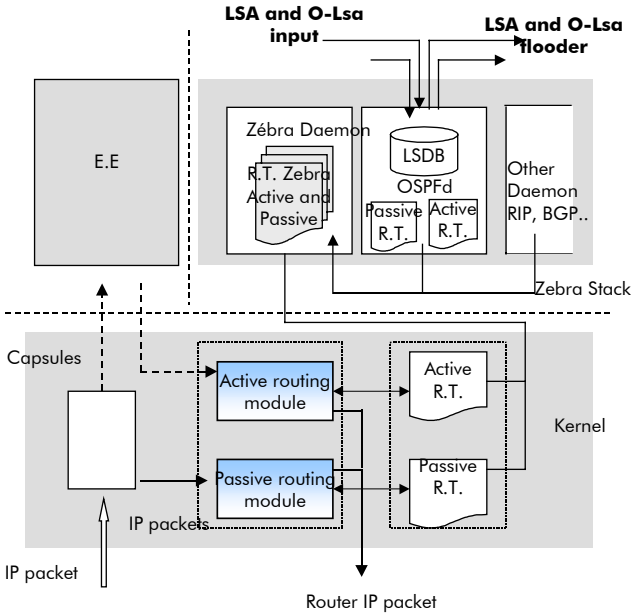

Figure 5 : Complete architecture for capsule routing

7 Conclusion and further works

The Active Network concept is a promising candidate for addressing the new issues of IP network in terms of flexibility and content adaptation. One of the key issues for a success story of the Active Networks is the smooth and consistent integration with the existing passive networks. In this context, it is needed to develop mechanisms allowing the capsules containing, or referencing, the Active Code to be routed accordingly to their specific constraints. The routing mechanism must be aware of the existence of two different kind of packets, and two different kinds of nodes.

This paper shows how to extend the existing routing protocols for supporting the routing of capsules in an Hybrid and Heterogeneous Network. The issues to be solved are addressed, and algorithms for routing in an Active Network are proposed. An implementation of the activeness information distribution has been done on the top of an open and modular routing stack. This proves that the needs for specific routing mechanisms can be easily met at low cost. The global performances are not deeply impacted, the modification resulting in a separation of passive packets and active packets. The worst case is when all the routers are active and they all receive active and passive packets: the tables are then duplicated in each planes.

Further works are of two kinds. The first one is to allow different routing of different types of capsules, depending of the targeted Active Application. In a Linux context, this could be easily done thanks to kernel specific filtering, based on the Netfilter framework, as well as distinct routing of capsules and packets.

The second main direction of work is to consider not only the routing of individual capsules and packets, but the routing of flows. When a capsule reaches an Active Node, the Active Code is executed at this node. Then, the packets that this code must process must be routed toward this node also. This means that the scope of routing must be flows, rather than packets/capsules. In IP networks, this can be partially achieved by the use of resource reservation mechanisms only (like RSVP) [16][15]. Even if this allows to find a reliable route across the network, this does not insure that all the packets of a flow will follow the same path, and then be processed by the same Active Code executions. If in the context of Quality of Service, for which RSVP has been designed, the probability of failure of transport due to a route change is low, in the case of the Active Network where the whole flow can be processed and transformed on the fly, it is mandatory that, once determined, the same route is used for the whole flow. Such strong requirements on the routing system could be achieved thanks to very flexible kernel like the Linux 2.4 one, or in Network Processor, allowing a fast and flexible packet forwarding.

They key point for the adoption of such a differentiated routing is the balance between its expected benefits and its scalability. Even deployed in the access networks only, this mechanism needs more resources in the edge and access nodes than a traditional, passive, routing. But this allows to optimize the active resources usage, then to give activeness to a network, even with a few number of active nodes. The last point is that this solution is based on existing extension mechanism and that it relies on existing protocols. Its deployment can be done in an incremental way, every node supporting the mechanism adding value to the global system.

8 References

[1] D. J. Wetherall, J. Guttag, and D. L. Tennenhouse. ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols. In Proceedings of IEEE Openarch'98, April 1998.

[2] T. Faber, ACC: Using Active Networking to Enhance Feedback Congestion Control Mechanisms", IEEE Network (Special Issue on Active and Programmable Networks) , May/June 1998, p.61-65.

[3] D. L. Tennenhouse and D. J. Wetherall. Towards an Active Network Architecture. Computer Communications Review, 26(2), Apr. 1996.

[4] Internet Assigned Numbers Authority (IANA), www.iana.org

[5] B. Schwartz, et al. Smart Packets for Active Networks. In Proceeding of IEEE OPENARCH'99, March 1999.

[6] M. Hicks, P. Kakkar, J. T. Moore, C. A. Gunter, and S. Nettles. PLAN: A Programming Language for Active Networks. Technical Report MS-CIS-9825, Department of Computer and Information Science, University of Pennsylvania, February 1998.

[7] Patrick Tullmann, Mike Hibler, and Jay Lepreau. Janos: A Java-oriented OS for Active Networks . IEEE Journal on Selected Areas of Communication. Volume 19, Number 3, March 2001.

[8] Bellman-Ford T.H.Cormen, C. E. Leiserson, R. L. Rivest. Introduction to algorithms. MIT Press. 1990.

[9] Routing Information Protocol (RIP), RFC 1058

[10] Routing Information Protocol version 2 (RIP2), RFC 1723

[11] Open Shortest Path First (OSPF), RFC 2328

[12] The OSPF Opaque LSA Option, RFC2370

[13] GNU Zebra, www.zebra.org

[14] Masahiko Endo, Patch opaque LSA, http://marc.theaimsgroup.com/?l=zebra&m=98576206523511&w=2

[15] Prashant Chandra, Allan Fisher, and Peter Steenkiste. Beagle: A resource allocation protocol for an application-aware internet. Technical Report CMU-CS-98-150, Carnegie Mellon University, August 1998.

[16] Resource ReSerVation Protocol (RSVP), RFC 2205

[17] D. S. Alexander, et al. Active Network Encapsulation Protocol (ANEP). In RFC Draft, July 1997.

[18] D. J. Wetherall and D. L. Tennenhouse. The Active IP Option. In Proceeding of the 7th ACM SIGOPS European Workshop, September 1996.

[19] D. Decasper, G. Parulkar, S. Choi, J. DeHart, T. Wolf, B. Plattner, "A Scalable, High Performance Active Network Node," IEEE Network, Vol. 13(1), 1999.

[20] J[erome] H. Saltzer, D[avid]. P. Reed, and D[avid]. D. Clark. End-to-end arguments in system design. ACM Transactions on Computer Systems 2, 4 (November 1984) pages 277-288. An earlier version appeared in the Second International Conference on Distributed Computing Systems (April, 1981) pages 509-512.

[21] Jonathan T. Moore, "Safe and efficient active packets," Tech. Rep. MS-CIS99 -24, Computer and Information Science, The University of Pennsylvania, 1999.

[22] D. Galand, O. Marcé, Active Router Information in Routing Protocols, IETF Draft, November 2000