

Neutral Access Network Implementation Based on Linux Policy Routing

Andrea Seraghiti and Alessandro Bogliolo
Information Science and Technology Institute
University of Urbino
Urbino, Italy 61029
Email: {andrea.seraghiti, alessandro.bogliolo}@uniurb.it

Abstract—Neutral access networks (NANs) have been proposed as a mean for overcoming the stagnation of broadband market by granting positive externality to access infrastructures, in order to trigger a virtuous circle among investors, network operators, users, and service providers. From a technical point of view, NANs prompt for the development of new solutions to design and implement a shared infrastructure which enables unregistered users to associate to the network, to take advantage of local services delivered within the access network, and to register seamlessly with the Internet service provider of choice to gain instantaneous access to the Internet.

This paper presents a NAN implementation, based on Linux policy routing, which has been developed and tested within the wireless campus of the University of Urbino.

I. INTRODUCTION

The concepts of *open access* [1] and *operator neutrality* [2] have been recently introduced as an alternative to vertically-integrated access networks (which are deployed, owned, maintained, and operated by a unique operator) in order to realize scope economies and enable a fair competition among operators on a common access infrastructure.

An *open access network* (OAN) acts as an intermediary between users and service providers, giving to the end user the freedom of choosing among different services and the perception of connecting directly to the provider of choice. From a service provider perspective, OANs provide the key advantage of sharing investments and maintenance costs.

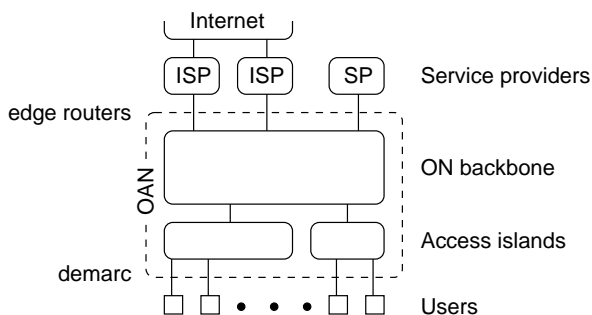


Fig. 1. OAN architecture.

The layered architecture of an OAN is represented in Figure 1. Users connect to local access infrastructures (called *access islands*), which are connected to a common *operator-neutral*

backbone which contains the edge routers of the *service providers* (SPs). According to Barcelò *et al.* [3], the access network is a truly shared infrastructure only if it contains independent edge routers for all the service providers that operate on it. *Internet service providers* (ISPs) are nothing but a special category of SPs.

Several technical solutions have been proposed to implement OANs [3], but they are mainly focused on the issue of allowing the end user to register with the service provider of choice, while keeping the access infrastructure as transparent as possible. Apart from the beneficial separation between network access and services, this vision doesn't change too much the business model of service providers, nor the motivation of the users for entering the network [4].

The idea behind *neutral access networks* (NANs) is that of granting visibility to the shared access network in order to induce a positive externality which might help to overcome market stagnation and to bridge digital divide [5].

A *positive externality* occurs whenever a new node/user/service is added to a network since it increases the networking opportunities for all other entities. While this beneficial effect is apparent in the Internet, it is usually unexploited in vertically-integrated access networks, where it is limited to low-cost internal communications offered by some ISPs to their subscribers.

In order for an access network to benefit from positive externality, it has to exhibit the features of a full-fledged network by itself. According to the NAN model [5], a sizeable set of services have to be delivered within the access network and made available to the users before they register with any ISP.

The entry of a new user into the NAN has a beneficial effect for all other users since it enhances their communication potentials and it helps reaching the critical mass of users required to incentivize the provisioning of new services. Similarly, the entry of a new SP has a spillover benefit for all other providers since it induces new users to enter the shared marketplace and it contributes covering the costs of the infrastructure. Finally, the extension of the infrastructure (possibly consisting in the deployment of new access islands owned and/or managed by new operators) has a positive effect on the usage of the existing infrastructure since it increases networking opportunities, ultimately affecting users'

motivations to become part of the network.

Integrating the distinguishing features of an access network with those of a full-fledged network is the main challenge to be faced in the implementation of NANs. The technical solutions adopted so far for implementing OANs [3] are not suitable to be applied to NANs since they fail in granting visibility to the access network. Moreover, most of them are tailored to wireless access networks targeting only nomadic users. Among existing solutions [3], *virtual tunnelling* [6] seems to be the closest to the needs of NANs: unauthenticated users are allowed to associate with the access network and to take advantage of internal services, but they need to authenticate to establish a virtual tunnel towards the edge router of an ISP in order to gain access to the Internet [6]. Virtual tunneling is general enough to be applied to any kind of network, but it introduces some overhead and it is not *convenient* enough [3], [7]. In fact, tunneling clients are not pre-installed on most terminals and they are not sufficiently user-friendly to be widely accepted.

This paper outlines a convenient open-source implementation of NANs based on Linux policy routing [8].

II. NETWORK ARCHITECTURE

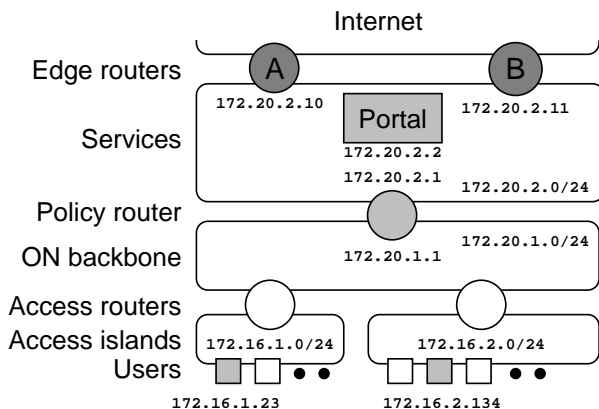


Fig. 2. Reference architecture of a NAN.

The proposed architecture, shown in Figure 2, is an IP network which is independent of the nature of the links between the end users and the access network. The network is open to unauthenticated users, who associate with an access island and are assigned with unique IP addresses. Depending on the network management policies and on user's typology (nomadic vs residential), IP addresses can be either statically or dynamically assigned. Moreover, they can be taken either from public or from private pools. Private IPs are used in the example provided in Figure 2.

Multiple access islands (possibly managed by different operators) can be available for guaranteeing scalability and technological diversity. Each access island is connected to the operator-neutral backbone by means of an access router. Depending on the technology, access routers can be, for instance, public WiFi hot spots, Hiperlan/WiMax base stations, mesh

```
#
# reserved values
#
255 local
254 main
253 default
0  unspec
#
# local
#
1   ISP_A
2   ISP_B
```

Fig. 3. Content of file `/etc/iproute2/rt_tables` installed on the policy router of Figure 2.

access controllers, digital subscriber line access multiplexers (DSLAMs), etc.

The ON backbone is the networked infrastructure which provides connectivity between access routers and services. The heart of the architecture is the router placed between the backbone and the service network, called *policy router*. For the sake of simplicity here we assume that the ON backbone is organized as a unique sub network and that there is only one policy router, which is the default gateway of the ON backbone. Generalization and scalability will be discussed in Section IV.

A subnet is used to publish the internal services directly available within the access network, while a captive portal (denoted by Portal in Figure 2) redirects the users to a predefined landing page whenever they attempt to access an external URL which is not white-listed. The landing page belongs to the internal web portal which grants access to all the local services available. Moreover, the landing page of the captive portal allows the end user to choose among several edge routers (managed by different SPs/ISPs) to gain access to the Internet or to external services.

As the user makes his/her choice, a source-based policy rule is dynamically created on the policy router in order to forward across the selected edge router all the external traffic originated from the IP address of the user.

III. IMPLEMENTATION

The network architecture described so far is based on the separation between access, backbone, and service subnetworks, which is common practice in networking. The only element which is worth a further discussion is the dynamic management of policy routing. The proposed implementation exploits the support for advanced routing provided by the Linux kernel [8].

Figure 3 shows the `rt_tables` file installed on the policy router, which contains a list of the available routing tables with the corresponding priorities (from 1 to 255). The first 4 rows specify the standard tables assigned with reserved priority values, while the last two rows specify the new tables inserted for handling the two ISPs of the example of Figure 2. In general, a specific routing table is required for each edge router.

```

# table ISP A
ip route add default via 172.20.2.10 dev eth1 table ISP_A
ip route add 172.20.2.0/24 via 172.20.2.1 table ISP_A

# table ISP B
ip route add default via 172.20.2.11 dev eth1 table ISP_B
ip route add 172.20.2.0/24 via 172.20.2.1 table ISP_B

```

Fig. 4. Tables to be added to the policy router of Figure 2 for managing the edge routers of ISPs A and B.

Once the tables have been declared, they require to be created by adding a rule at the time by issuing the proper *ip* commands. For our purposes, each table needs to contain at least one rule, which sets the corresponding edge router as the default gateway. The commands used to populate the tables for the two ISPs of our example are shown in Figure 4. The second rule added in each table is used only for optimization purposes, in that it makes the internal services directly reachable from the policy router.

Notice that the definition of a routing table does not imply its application. This allows us to define in advance the tables for all the edge routers and to enable the users to select among them at run time according to the edge router of choice.

In order to grant to the users the freedom of choosing the edge router they prefer, routing tables are dynamically associated with the IP address of the user. This is done by means of *source-based policy routing* [8], directly controlled by the end user by means of a web-based frontend implemented in the main portal.

The portal is the standard default gateway for the policy router. Hence, it acts as a captive portal for all the users who have not yet selected any edge router. When a user attempts to reach a web page which is not internal to the access network (or in the white list of the captive portal), it is redirected to a predefined landing page containing an active list of all the edge routers available. As soon as the user clicks on the name of an edge router, a record is added to a database containing the source IP of the user, the name of the edge router of choice, and a *pending flag* set to 1.

The database is periodically checked by a daemon running on the policy router. For each pending request, the daemon issues an *ip* command to specify that the route table of the selected edge router (e.g., *ISP_A*) has to be adopted for processing the packets coming from the IP of the user (e.g., 172.16.1.23):

```
ip rule add from 172.16.1.23 table ISP_A
```

Once the rule has been added, the pending flag in the corresponding entry of the database is reset.

Lookup rules are removed by the same daemon whenever a new rule has to be added for the same source IP (meaning that the corresponding user has decided to switch to a different edge router), or whenever no traffic is generated by the source IP for a period exceeding a pre-defined timeout. An example of active rules (as listed by the *ip rule show* command) is provided in Figure 5.

Notice that the policy routing mechanism does not entail user authentication. In fact, it has the only effect of forwarding

```

p:      from all lookup 255
32764:  from 172.16.2.134 lookup ISP_B
32765:  from 172.16.1.23 lookup ISP_A
32766:  from all lookup main
32767:  from all lookup default

```

Fig. 5. Example of active rules (as listed by the *ip rule show* command) in the policy router of Figure 2.

the external traffic generated by each user to the edge router he/she has chosen, which can implement its own captive portal to manage user authentication and access control. Traditional IP-based captive portals can be used to this purpose.

IV. SCALABILITY

The NAN architecture depicted in Figure 2 suffers from three scalability issues: the ON backbone is implemented as a single broadcast domain, all the services are published within the same subnetwork, and the policy router is a bottleneck for the traffic. All these issues are addressed in the generalized architecture shown in Figure 6.

No architectural constraints are imposed to the backbone, which can be organized in multiple subnetworks. Moreover, network-to-network tunnels can be established to make independent NANs interoperate. For instance, two separate NANs are depicted in Figure 6, deployed in two different metropolitan areas, denoted by **L** (i.e., left) and **R** (i.e., right).

Several service subnetworks can be used to publish internal services and edge routers. Subnetworks containing edge routers need to be connected to the backbone by means of policy routers, while standard routers (white circles in Figure 6) can be used to grant access to subnetworks publishing only internal services.

In case of a backbone with multiple subnetworks, more than one policy router can be used to enable load balancing and path optimization strategies to be implemented in the backbone. Consider, for instance the situation shown in Figure 6 for NAN **L**, where the backbone can use policy routers 1 and 2 to route the traffic originated from access islands a and b, respectively.

If no tunneling/synchronization mechanisms are implemented, NANs **L** and **R** in Figure 6 are independent from each other and they do not interoperate. This means that users connected to access islands a and b can only register with ISPs A and B or gain access to internal services *S_x* and *S_y*. Similarly, users connected from access islands c and d can only register with ISPs B and C or access internal service *S_z*. The only ISP covering both areas is B, which has edge routers in both NANs.

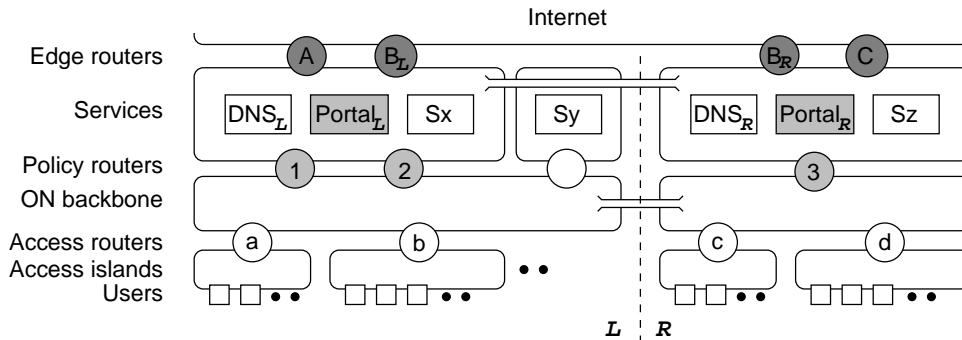


Fig. 6. Generalized architecture of two interoperating NANs.

All the internal services can be easily made accessible from both NANs by means of tunnels established between the two backbones, assuming that the IP addresses used within the two networks are compatible or suitably translated (by means of NAT).

An additional effort is required to allow the users of a NAN to choose the edge routers of the other one. Two issues have to be addressed to this purpose: creating the routes from the policy routers of a network to the edge routers of the other one, and managing their temporary association with the source IP of the users. We envision three different scenarios.

In the first scenario, dedicated tunnels are created to expose the edge router of a network within the service subnetwork of the other one. Consider, for instance, a tunnel established between the service subnetwork containing edge router A and the service subnetwork managed by policy router 3 in Figure 6. If edge router A is the default gateway for the tunnel, then the R-side of the tunnel can be treated as an edge router by policy router 3. Hence, it can be added to the list of ISPs available from NAN R by adding a specific table (namely, ISP_A) in policy router 3 and a corresponding entry in the landing page of the captive portal. The same approach can be adopted to make edge router C available to the users connected from NAN L. To this purpose a dedicated tunnel (having C as its default gateway) has to be established between the same subnetworks, a specific table (namely, ISP_C) has to be added in both policy routers 1 and 2, and a corresponding item has to be added in the landing page of $Portal_L$.

In the second scenario there are no ISP-specific tunnels, but there are tunnels between the backbones of two different NANs, having policy routers as default gateways. Assume, for instance, that a tunnel exists between the backbones L and R in Figure 6, with default gateway 3. Such a tunnel cannot be used by policy routers 1 and 2 to directly forward IP packets towards the edge routers of NAN R, but it can be used to forward packets towards policy router 3. To this purpose, a specific table (called NAN_R) has to be added in policy routers 1 and 2, and a corresponding link has to be added in $Portal_L$. When a user connected to NAN L selects NAN_R , all the external traffic it generates is forwarded towards the tunnel to the captive portal of NAN R, where edge routers B and C can

be selected to gain access to the Internet. If the user selects edge router C, then a policy route is created in 3 for that user. From the point of view of the user, the selection of an edge router belonging to a NAN which is not the one it is connected to is a two-step process. This approach has the advantage of minimizing the synchronization effort among different NANs. In fact, once the tunnels and the forwarding rules have been created, the edge routers possibly added to one of the NANs become automatically available, with no additional effort, to the users connected from the other ones.

In the third scenario the network architecture is exactly the same described in the second one, but there is a tighter synchronization between the policy routers of the two networks in order to make the two-step selection process transparent to the end user. Although there is only one tunnel from the backbone of NAN L to policy router 3, a link to edge router C is directly added to the landing page of the captive portal of L, together with the links to edge routers A and B. However, no tables to edge router C are added to policy routers 1 and 2. If an user connected to access island b selects edge router C, a source-based routing policy is added both in policy router 2 and in policy router 3. The rule added in 2 has the effect of forwarding the external traffic towards 3, while the rule added in 3 has the effect of routing external packets towards edge router C.

V. CONCLUSIONS

According to their definition [5], NANs have to support: unauthenticated connection to a shared infrastructure, direct access to internal services, and authenticated access to external services. In addition, they should guarantee independence from access technology, minimize the technical commitment among the entities involved, and guarantee scalability.

New technical solutions need to be developed in order to implement NANs which meet all these challenging requirements. This paper has presented a viable solution which makes use of source-based policy routing, as made available by the Linux support for advanced networking.

Although the proposed solution meets the main requirements of the NAN model, there are several open issues that need further attention. First, wireless open access points provide no support for wireless link protection. Although secure-

socket layers or VPNs could be used to this purpose, new MAC-layer mechanisms should be implemented to combine the advantages of protected wireless networks with the accessibility of open access points. Second, source-based policy routing requires the users to be uniquely associated with IP addresses. The mechanism fails in case of *IP masquerading* NAT possibly performed within the access islands (all the users hidden behind the same IP would be routed across the same edge router) and it requires specific measures to be adopted to contrast *IP spoofing*. Third, the access infrastructure described in this paper is conceived as a best-effort IP network which provides no support for implementing end-to-end service level agreements.

All the solutions outlined in this paper have been implemented within the *Urbino wireless campus* testbed [6], which is a shared multi-gateway access infrastructure deployed and maintained by the University of Urbino together with many public and private partners. Current reserach efforts are mainly focused on the solution of the three issues mentioned above, and on the deployment of large-scale NANs.

ACKNOWLEDGMENT

The authors would like to thank Nicola Deboni, Luca Lazzaletti, Federico Serafini, and Matteo Valentini for their contribution to the development of the testbed.

REFERENCES

- [1] R. Battiti, R. Lo Cigno, M. Sabel, F. Orava, and B. Pehrson, "Wireless LANs: From WarChalking to Open Access Networks," *Mobile Networks and Applications*, vol. 10, pp. 175–287, 2005.
- [2] V. Kordas, E. Frankenberg, S. Grozev, B. L. N. Zhou, and B. Pehrson, "Who should own, operate and maintain an opeartor neutral access network?" in *LANMAN2002: IEEE Workshop on Local and Metropolitan Area Networks*, 2002.
- [3] J. Barceló, A. Sfairopoulou, and B. Bellalta, "Wireless open metropolitan area networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 12, no. 3, pp. 34–44, 2008.
- [4] Broadband Working Group, "The broadband incentive problem," *MIT Communications Futures Program White Paper*, September 2005.
- [5] A. Bogliolo, "Introducing neutral access networks," in *Proceedings of Int.l Conference on Next Generation Internet Networks (NGI 2009)*, 2009.
- [6] —, "Urbino wireless campus: A wide-area university wireless network to bridge digital divide," in *Proceedings of AccessNets-07*, 2007, pp. 1–6.
- [7] S. Khanvilkar and A. Khokhar, "Virtual private networks: an overview with performance evaluation," *IEEE Communications Magazine*, vol. 42, no. 10, 2004.
- [8] M. Marsh, *Policy Routing Using Linux*. SAMS, 2006.