Test of inter-working and translation mechanisms between IPv4 and IPv6 [12]

G. Somlai, L. Sógor, L. Martonossy, M. Fidrich, G. Dikán,
T. Tarjányi, P. Hendlein and M. Bohus

The Internet Protocol version 4 (IPv4) has been available since 1981. As it can be seen today, its success is indiscutable. However, the rapid growth of the Internet has created a number of problems for the administration and operation of the global network. Some new challenges that must be handled have also appeared, such as security, mobility and real-time support. The new version 6 of the Internet Protocol (IPv6) includes expanded addressing capabilities, header format simplification, improved support for extensions and options, plug and play services, authentication and privacy capabilities, native mobility support, and also real-time and quality services.

Maintaining compatibility with the large installed base of IPv4 hosts and routers while deploying IPv6 will streamline the task of transitioning the Internet to IPv6. There are two main cases of interoperation between IPv4 and IPv6, which should be solved. The first is that of running an IPv6 network that communicates with other IPv6 networks using the existing IPv4 infrastructure. The solutions for this case (configured and automatic tunneling, 6over4, 6to4) have already been cleared out, most of the mechanisms has been defined in RFCs. The second case is that of IPv4 and IPv6 hosts that want to communicate with each other directly. This communication is very complicated mainly because of the different addressing schemes and it can be solved using translation techniques at one of three different protocol levels. At the IP level, Network Address Translation - Protocol Translation (NAT-PT) employing the Stateless IP/ICMP Translator (SIIT) mechanism can be used. Relaying of UDP and TCP sessions can be done at the transport level, while application proxies such as SOCKS run at the application level.

This article presents the methods used and results obtained in testing the communication among: i) IPv6 hosts in different IPv6 networks, which are interconnected through IPv4 infrastructure and ii) an IPv6 host on an IPv6 network and an IPv4 host on an IPv4 network. For the first the *6to4* method and *configured tunneling* was used, while for the second one the *SIIT* and a *simplified version of NAT-PT*. For 6to4 and SIIT we had to develop our own pieces of software. Different configurations were used, consisting of IPv6, IPv4 and IPv4/IPv6 hosts, interconnected in various ways.

Due to its free source code and good documentation, the Linux operating system was chosen as a test platform. The nature of the tasks to be resolved and the solutions chosen required implementation at the kernel level. The tunneling procedures in the case of 6to4 and header conversion procedures necessary for NAT-PT were implemented as kernel modules, which can be loaded/unloaded dynamically with the modutils and configured by the ifconfig utilities. These modules create pseudo network devices, which receive and send the packets they need to process using standard routing procedures. The necessary routing rules can be given with the route utility. All utilities used are part of standard Linux distributions.

Due to the lack of implementation of some quite basic features of IPv6 (which we rely on) at this stage of the development of the Linux kernel, some small patches to the kernel were also needed. Hopefully - maybe based on our suggestions - later versions of the kernel will include these changes.

Several tests were performed to find out the performance of the methods, and, additionally, the conformance and robustness of our implementation. Results and the conclusions we drew are presented.

[12]This article presents the results of a research work initiated and financed by Nokia Hungary as a joint project between the University of Szeged and Nokia Hungary.