

DS-Lite:

Implementation experience and views

NOKIA

CMCC workshop, Nov 2009

B. Patil/T. Savolainen/F. Junior



Overview of DS-Lite

Need for IPv4-over-IPv6 tunneling

- Earlier, tunneling discussions focused on IPv6-over-IPv4:

“How to reach IPv6 domain from IPv4-only networks?”

- However, due to the looming IPv4 address exhaustion, the bigger question now is:

“How to reach IPv4 domains from IPv6-only networks”?

- There are various methods, and one such is Dual-Stack Lite
- DS-Lite looks like Dual-Stack for devices/applications, but in reality it:

**Automatically tunnels IPv4 over IPv6 to
Address Family Transition Router (AFTR)**

AFTR *aka* Carrier Grade NAT *aka* Large Scale NAT *aka* DS-Lite tunnel concentrator

DS-Lite provides

- IPv4 connectivity to hosts and/or home routers (CPEs) that are provisioned with only IPv6 addresses
 - No need for private addresses in operator's network
 - Public IPv4 addresses, previously used in operators' intra, are harvested into more productive use
- Dual-Stack connectivity for hosts connected to IPv6-only access networks
 - Less need to maintain IPv4 or dual-stack access networks
 - A lightweight solution for providing IPv4 connectivity over IPv6 only access
- Single NAT – i.e. no need to have multiple layers of NATs
- Avoidance of protocol translation – and problems caused by it
- Multiplexing limited number of public IPv4 addresses for even larger number of customers than before
- Automatic tunnel establishment
 - Tunnel endpoint (AFTR) is discovered with Stateless DHCPv6 (new option defined)
- Port forwarding capability on the AFTR
 - With technologies such as: Web-UI, NAT-PMP, **UPnP**, A+P
- **In 3GPP**, would enable IPv4 connectivity over IPv6-only PDP context
 - Handy, **if number of simultaneously open PDP contexts must be limited**

DS-Lite costs

- Supporting host and/or CPE and/or other node must be updated
- Encapsulation effort, especially on the network side (large number of tunnels)
- MTU issues due to encapsulation (possibly causing fragmentation/reassembly)
- Legal requirements: Carrier Grade NAT (AFTR) has to store the binding it makes for legal traceability
- Challenges in deep packet inspection, e.g. for QoS purposes (differentiating tunneled traffic to dedicated bearers)
- Short NAT timeouts, - possibly only 3 minutes for idle TCP sessions!
 - However, latest draft revision states behave RFCs SHOULD be followed
- Big-and-Rather-Expensive-Piece-of-Hardware for AFTRs
- Application Layer Gateway functionality centralized in AFTR

DS-Lite standard architecture models



- Router (CPE) based architecture

- The home CPE, also called **B4** (Basic Bridging BroadBand Element) advertises RFC1918 addresses to LAN, and tunnels IPv4 packets over IPv6 to **AFTR** for NATting

=> Home users do not see much difference whether NAT is at CPE, or at AFTR

- Each home network can use the full RFC1918 space

B4 is pronounced as "Before"

- Host based architecture

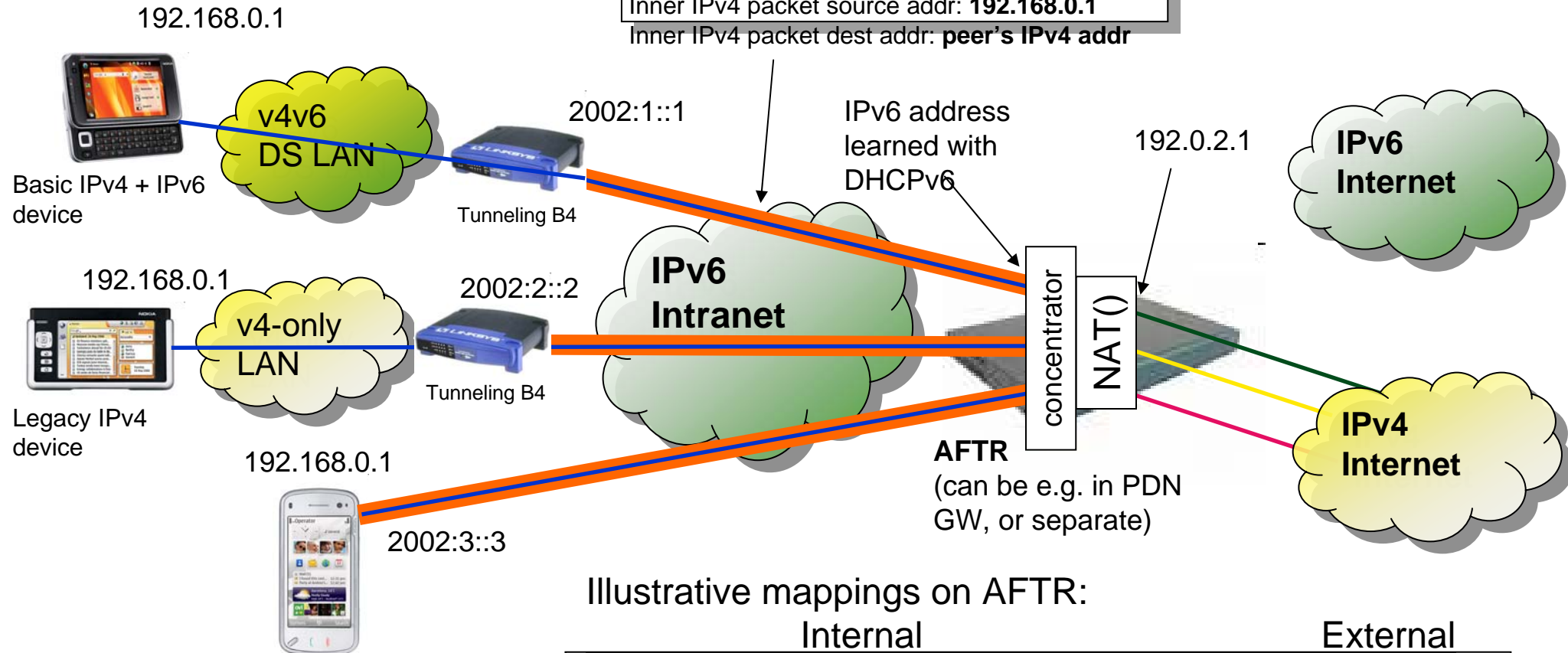
- A host, like mobile phone, configures virtual IPv4 interface with RFC1918 address (or a dedicated IANA assigned address from subnet 192.0.0.0/29), and tunnels IPv4 over IPv6 to AFTR for NATting
- AFTR can be implemented e.g. in GGSN, PDN-GW, or as separate network entity



Illustration of DS-Lite architectural models

point-to-point IPv4 over IPv6 tunnel

Outer IPv6 packet source addr: **2002:1::1**
 Outer IPv6 packet dest addr: **AFTR's IPv6 addr**
 Inner IPv4 packet source addr: **192.168.0.1**
 Inner IPv4 packet dest addr: **peer's IPv4 addr**



Updated B4 enabled device connected directly to intranet, or e.g. with IPv6 PDP context to PDN GW

Illustrative mappings on AFTR:

Internal	External
—	—
(2002:1::1 * 192.168.0.1:3001)	↔ 192.0.2.1:5001
(2002:2::2 * 192.168.0.1:3001)	↔ 192.0.2.1:5002
(2002:3::3 * 192.168.0.1:3001)	↔ 192.0.2.1:5003

Implementation on Maemo

Technical solution done for N810

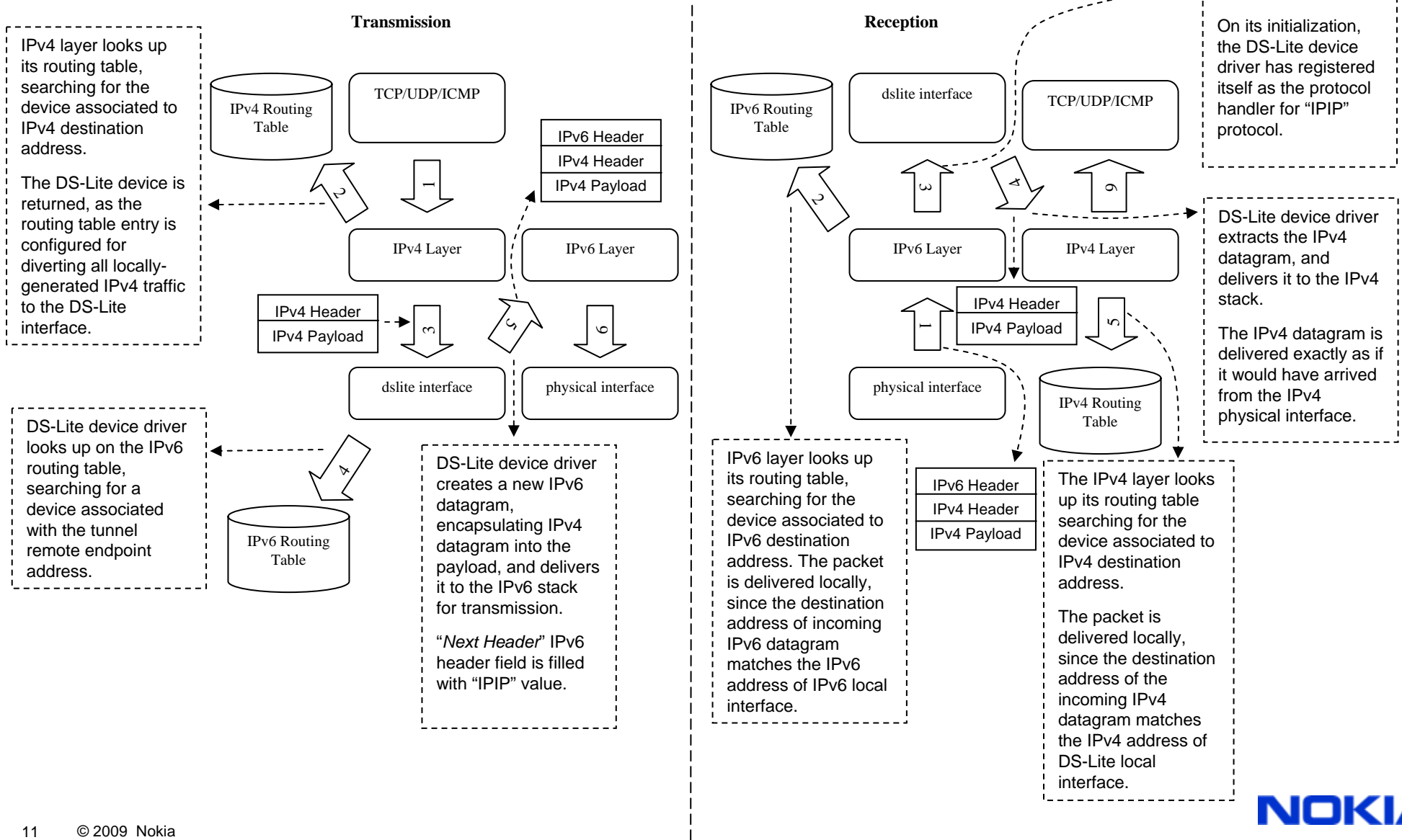
- Current DS-Lite implementation is based on the *host model* architecture for client side implementation
 - However, it is possible to adapt the solution for *CPE model* architecture
- Linux MAEMO platform
 - Customized *Debian* Linux distribution for mobile devices
 - Binaries are cross-compiled for ARM platform
- N810 WiMAX Edition terminals as test platform
 - MAEMO version 4.1.2 (*DIABLO* release), based on Linux kernel version 2.6.21



Technical solution done for N810

- DS-Lite client specification requires IPv4-over-IPv6 tunneling capabilities
 - Linux has native support for such feature since version 2.6.22, using a *virtual* device driver concept (*ip6_tunnel*)
 - *iproute2* package (set of tools for configuring the Linux network) is also required for configuring the Linux tunnels on MAEMO
- *ip6_tunnel* intercepts IPv4/IPv6 packets on their path by kernel network stack:
 - On transmission, encapsulating IPv4 packets into IPv6 packets, according to specific user settings (tunnel parameters)
 - On reception, extracting the IPv4 packets from IPv6 packets, and delivering them to IPv4 network stack
- Linux kernel version 2.6.21 and above already support the dual-stack operation (IPv4/IPv6), necessary for IPv4-over-IPv6 tunneling
 - However, *ip6_tunnel* device driver supporting IPv4-in-IPv6 tunneling is just available from kernel version 2.6.22 and beyond

Technical solution done for N810

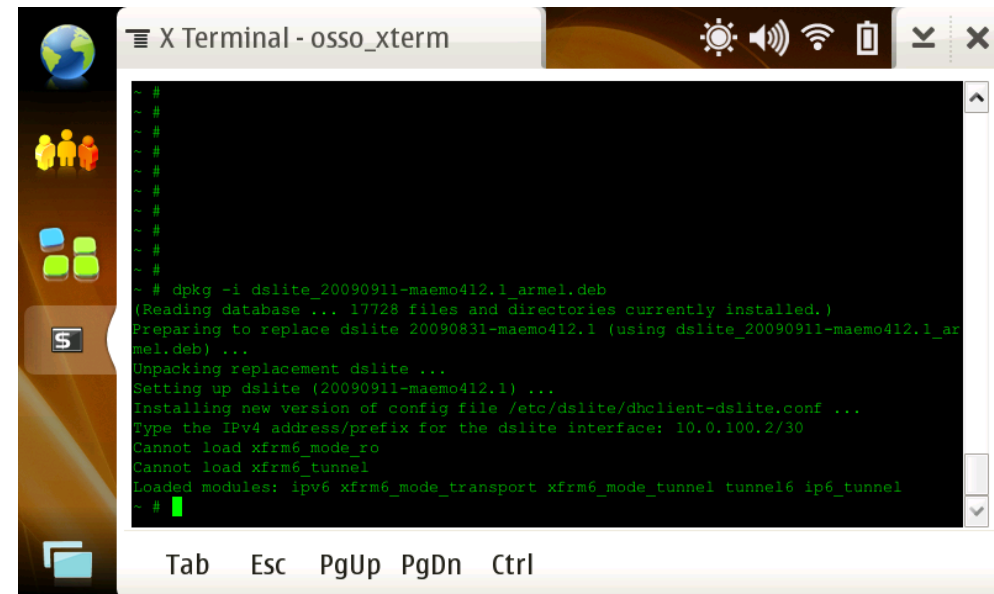


Installation and tunnel setup automation solutions

- Documentation was created on how to proceed with installation on MAEMO
 - Under review, to be released together with binaries in the next weeks
- Requirements for the MAEMO 4.1.2 device
 - Root access enabled (*rootsh 1.4 (gainroot)*)
 - IPv6 support enabled
 - BusyBox (*busybox_1.6.1.legal1osso15ipv6.1_armel.deb*)
 - Iproute2 (*iproute_20080725-maemo412.1_armel.deb*)
 - ISC DHCPv6 client (*dhcp-4.1.0p1.tar.gz*)
 - Bash2 (*bash2 2.05b-1maemo3*)
- A DS-Lite AFTR (CGN) must be available at the IPv6 network
- Current release expects automatic configuration of the AFTR tunnel endpoint using specific DHCPv6 options (<http://tools.ietf.org/html/draft-dhankins-softwire-tunnel-option>); therefore, a DHCPv6 server providing the address of the AFTR tunnel endpoint must be provided (we used ISC's DHCPv6 server)

Installation and tunnel setup automation solutions

- Installation of the DS-Lite support itself is quite automatic
 - Need to install manually the Debian package (provided by INdT), by typing the command below on X Terminal:
`dpkg -i dslite_20090630-maemo412.1_armel.deb`
- During the installation process, the user is requested to type the IPv4 address/prefix for DS-Lite interface
 - Once defined by newer versions of the DS-Lite I-D, this step will be no longer required (*current assumption is that 192.0.0.0/29 range is allocated, so the IP for client could be e.g. 192.0.0.2, as the 192.0.0.1 is reserved for the AFTR element*)
- After that, the DS-Lite support is available the next time the device connects to an IPv6-only Wi-Fi connection



```
X Terminal - osso_xterm
~ # dpkg -i dslite_20090911-maemo412.1_armel.deb
(Reading database ... 17728 files and directories currently installed.)
Preparing to replace dslite 20090831-maemo412.1 (using dslite_20090911-maemo412.1_armel.deb) ...
Unpacking replacement dslite ...
Setting up dslite (20090911-maemo412.1) ...
Installing new version of config file /etc/dslite/dhclient-dslite.conf ...
Type the IPv4 address/prefix for the dslite interface: 10.0.100.2/30
Cannot load xfrm6_mode_ro
Cannot load xfrm6_tunnel
Loaded modules: ipv6 xfrm6_mode_transport xfrm6_mode_tunnel tunnel6 ip6_tunnel
~ #
```

AFTR solutions available

- Two sources for the AFTR node:
 1. A10 Networks has an AFTR product – This requires a licence and runs on an A10 networks platform
 2. Open source implementation from ISC – Was publicly announced at IETF75
- Initial testing was done with the A10 networks' AFTR located in the Comcast lab*

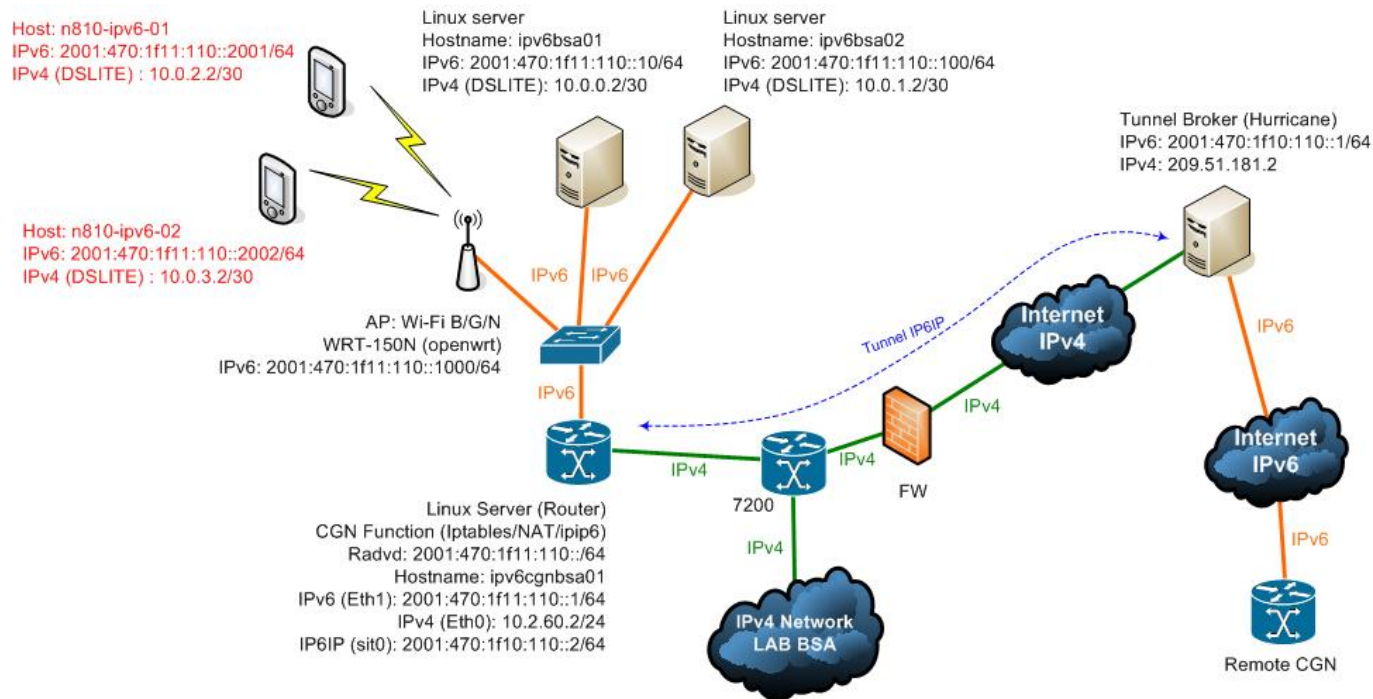
*Thanks to Alain Durand and Yiu Lee

Experiences on implementation and testing

- Back-porting *ip6_tunnel* from 2.6.22 to 2.6.21 was required in order to provide DS-Lite client capabilities for MAEMO
 - Functions inside the *ip6_tunnel* 2.6.22 code not supported by *DIABLO* release were replaced by semantically equivalent code (e.g. *ip_hdr*, *ip6_hdr*, *skb_reset_network_header*, *mac_header*, *transport_header* and *network_header*)
 - *Bug-fix of tunnel6 code on DIABLO kernel*: IPv4 tunneled packets with overall size (header size + payload size) lesser than the size of the IPv6 header (40 bytes) were dropped
 - Used the GPL toolkit *scratchbox* (sponsored by NOKIA) for cross-Compiling the resulting code (*ip6_tunnel* and *tunnel6*) for the ARM architecture
- The pre-installed MAEMO Diablo release of *iproute2* version doesn't support IPv4-in-IPv6 tunnels
 - We needed to update this package in order to allow the setup of IPv4-in-IPv6 tunnels on Nokia N810 tablet
 - *iproute2* source code was downloaded (version 2.6.26) from its web site, and it was also cross-compiled using *scratchbox* toolkit

Experiences on implementation and testing

- We were able to run *legacy* applications (i.e. *not aware of DS-Lite*) like Web browsing and Skype without any problems
- IPv6 tunnel connectivity was through tunnel broker (Hurricane.net)



Experiences on implementation and testing

- As can be observed, the test scenario was not necessarily compliant with the deployment model
 - However, we were able to test the implementation for ICMP, TCP and UDP protocols
- Issue arose for packets larger than the IPv6 tunnel with Hurricane.net's MTU of 1280 bytes, as we end up with *black-hole* TCP connections, and UDP drops
- We adjusted the MTU on the interfaces accordingly, in order to avoid fragmentation for TCP connections
 - Needed to update IPv4-over-IPv6 tunnel implementation on *ip6_tunnel.c* in order to allow MTUs smaller than 1280 bytes (due to tunnel broker overhead)
 - We can avoid such *hacks* by avoiding IPv4 fragmentation, and implementing IPv6 fragmentation on the IPv6 tunnel endpoints between the device and the AFTR
 - Hack is only possible for host-model, as for the CPE-model the CPE must be prepared to handle MTUs up to 1500 bytes

Experiences on implementation and testing

- Compilation of ISC AFTR itself was fairly easy
 - Tarball file containing source code for installation
- Problems arose during ISC AFTR setup phase
 - Lack of documentation for the released version
 - Sample configurations were not clear, sometimes confusing
 - Tricky automatic tunnel setup available after updating the Access Control Lists for the specific networks (only informed after e-mail communication with the creators)
- ISC AFTR stability issues
 - After some period of activity, the AFTR process stops responding to the client tunnel messages; terminal commands to the AFTR get an “ALIVE” response, but there is no further actions to incoming packets from the client tunnel, so we need to restart the AFTR process again

Feedback received on demo @ IETF#75

- The two DS-lite profiles (home and CPE) were made available for test during IETF #75
 - CPE profile was available through a specific IETF's IPv6-only Wi-Fi network
 - Host profile was available on some N810 terminals used by Nokia people
- Several N810s with DS-Lite implementations operated on the IETF's IPv6-network without any problems
- Host profile on N810 demonstrated during IETF #75

DS-Lite implementation open sourced

- The DS-Lite implementation for Maemo has been open sourced recently
- Details are at:
 - <http://ds-lite.garage.maemo.org/>

Summary

- DS-Lite client implementation on Maemo is fairly straightforward
- Creates a new virtual interface which captures and tunnels IPv4 packets when attached via an IPv6 only access
- Implementation has been tested with AFTRs from A10 Networks and ISC
 - Demonstrated at IETF75
- DS-Lite implementation for the N810 is now open sourced

