Wireless robot teleoperation via internet using IPv6 over a bluetooth personal area network

Teleoperación inalámbrica de un robot vía internet utilizando IPv6 sobre una red de área personal bluetooth

Carlos Araque Rodríguez[1], Fabio Guerrero[2]*

[1]Empresas Municipales de Cali, EMCALI E. S. P., Cra 75 N.º 15-06 Sede Limonar, Cali, Colombia

[2]Escuela de Ingeniería Eléctrica y Electrónica, Universidad del Valle, Calle 13 N.º100-00, Cali, Colombia

(Recibido el 20 de febrero de 2009. Aceptado el 23 de septiembre de 2009)

Abstract

This article presents the design, construction and testing of a system that allows the manipulation and visualization of the "Microbot Teachmover" robot using a Bluetooth wireless connection with an IPv6 address allowing the manipulation of the robot in three different scenarios: from a mobile device that is located on the robot's piconet, from a desktop computer that is in the same piconet as the robot, and from any computer that is connected to the Internet via IPv6.

----- *Keywords*: IPv6, bluetooth, wireless personal area networks, internet, teleoperation

Resumen

En este artículo se presenta el diseño, construcción y pruebas de un sistema que permite la manipulación y visualización del robot *Microbot Teachmover* usando una conexión inalámbrica *Bluetooth* con dirección *IPv6*, brindando la posibilidad de manejar el robot desde diferentes escenarios: desde un dispositivo móvil que se encuentra en la misma *piconet* del robot; desde un computador que se encuentre en la misma *piconet* del robot y desde un computador que se encuentre conectado a Internet con una dirección *IPv6*.

----- *Palabras clave*: IPv6, bluetooth, redes inalámbricas de área personal, internet, teleoperación

* Autor de correspondencia: teléfono: + 57 + 2 + 339 21 40 ext 109, correo electrónico: fabioguerrero@correounivalle.edu.co (F. Guerrero)

Introduction

Nowadays, there is a vast number of Internet-based applications being developed in a variety of scenarios which are related to remote interactive operation. For example, MIT's iLab project [1] is aimed at providing Internet accessible laboratories and promoting their sharing by academies in worldwide locations. Another example is that of the medical robots such as the RP-6 [2], which are being designed to allow physicians to control hospital surgical facilities from their homes or offices, etc. For all these kinds of applications, the Internet is being used as a universal information transport platform where, on a worldwide scale, IPv4 (Internet Protocol version 4) still dominates the network layer protocol infrastructure even now, at a time when most routers are prepared for the advent of IPv6 (Internet Protocol version 6). Despite the massive usage of IPv6 expected in the near future [3], there are several practical aspects of IPv6-based application development that need to be addressed such as, for instance, the complexity imposed by the IPv6-IPv4 mixed networking environment. In [4] a header compression method is proposed to allow voice over IPv6 over Bluetooth. The proposed technique is assessed using a Matlab software simulation. In [5] an architecture to guarantee continuous connection across several wireless access technologies (cellular, Wi-Fi, and Bluetooth) over IPv6-based networks is proposed. In [6] some experimental results on voice delivery to IPv6-enabled Bluetooth handheld devices connected to an IPv6 local are network are reported.

The main goal of this article is to report on the experience and results of the design and implementation of a system that allows the manipulation and visualization of a device, a "Microbot Teachmover" robot, via Internet using a Bluetooth wireless connection (physical and data link layers) and the IPv6 network protocol (network layer).

This article is structured as follows: First, the methodology of the design and implementation, together with a description of the system's final assembly is presented. Results for the tests carried out on the final system are then presented, followed by a brief discussion of results obtained including the advantages and limitations of using IPv6 over Bluetooth. Finally, the main conclusions of this work are summarized. It is assumed that the reader has a basic knowledge both of Bluetooth technology and IP basic network concepts. Any formal discussion of robotics is beyond the scope of this article because, this work is not oriented to discuss the design of a control technique or similar for the robot manipulator but, to report the results of the design of a system for the teleoperation of a Bluetooth-enabled device using IPv6 over the Internet. The results of this work may be useful to a canonical set of similar networking applications.

System description and design methodology

Figure 1 shows the total assembly, which is composed of: an embedded system that is responsible for managing the robot and receiving calls via the Bluetooth interface with an IPv6 address; a mobile device in which there is an application that allows the handling of the robot via Bluetooth connection; a fixed, local IPv6 router with functionalities that allow connectivity for IPv6 on the Bluetooth enabled network island, and an IPv6 global fixed-remote device in which there is an application that allows the manipulation and visualization of the robot.

The three main elements of this application are:

Embedded System - This is the server for the application that has been developed and it represents the fundamental part of this project. A web server is located in the embedded system (Cherokee) as well as a database (SQLite) against which users are authenticated. The site is dynamically generated by PHP (PHP Hypertext Pre-processor), and the latter also is responsible for sending commands to the serial port.

Router - This device is responsible for providing IPv6 connectivity to the embedded system. Since

native IPv6 connectivity to the Internet is not yet possible, the router uses a tunnel IPv6 overlaid by IPv4. The router will receive an IPv6 address /48 through the tunnel, which creates an island of IPv6 over Bluetooth. Further, this same equipment includes a video streaming server (VLC) that allows the display of the robot.

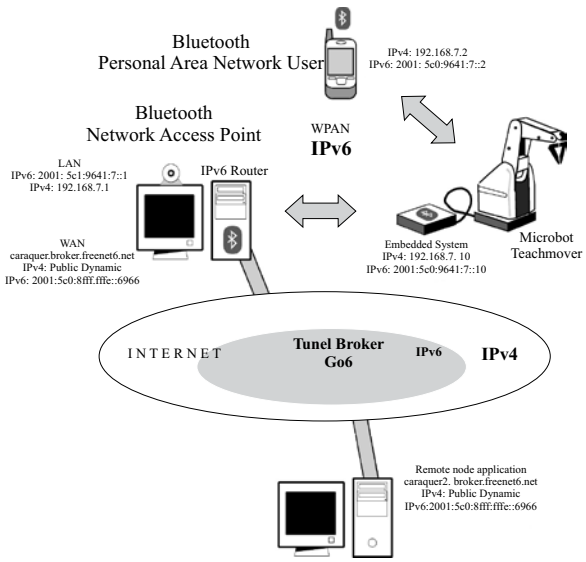

Figure 1 Description of the remote control system for the Microbot Teachmover robot

Client Application (fixed or mobile device)/ Application Client (fixed or mobile device) - As a result of using IPv6 and a web server, the application's only requirements are a remote web browser and IPv6, being as it is detached from the technology that is used for the connection (Ethernet, Wi-Fi, GSM, etc.). In order to see the video live a "VLC media player" is needed.

System hardware

Embedded system

For the hardware of the embedded system three options were considered: Blueboard_UV hardware board with BlueRaven software [7, 8], Btnode [9] and Gumstix [10]. The decision made for this project was to use Gumstix which runs the Linux 2.6 kernel, and has dimensions of 80 x 20mm. Apart from the technical advantages the cost of the system was very competitive and, furthermore, the only option to offer Bluetooth 2.0 +EDR.

Gateway Router

In order to have a connection to the Internet using IPv6 it is necessary to have a computer with a Bluetooth interface for connection to the embedded system coupled with an Ethernet interface for connection to the Internet. The minimum requirements are: a Pentium 4 or AMD K7, 256Mbytes of RAM, a hard disk, 5 Gbytes, one 10/100 ethernet interface and two USB ports. For the Bluetooth interface a USB Encore dongle was used (ENUBT-C1E). This is Class 1 and complies with the standard Bluetooth 2.0 + EDR. The video footage was captured using a webcam (genius Look 316) connected to a USB port.

Client application

For the application Client the minimum hardware requirement is a Bluetooth interface and support for a web browser. In this project the same computer was used as fixed client and as a router. The mobile device used was a PDA (Personal Digital Assistant) palm TX which has an Intel XScale PXA270 312-MHz CPU with Wi-Fi and Bluetooth v1.1 interfaces.

Microbot teachmover robot

The robot has six degrees of freedom; it is moved by six stepping motors that are mounted on the body of the robot and control each of the joints. The block diagram of electronic circuitry that drives the robot is shown in figure 2.

System software

Embedded system

The final configuration of the software that supports the implementation of the embedded system is shown in figure 3. The embedded

system uses two communication interfaces; one is the serial port that communicates with the robot and the other is the Bluetooth interface.

The PXA255 processor [11] has four UARTs for different purposes, in this project the STUART was used for the serial port.

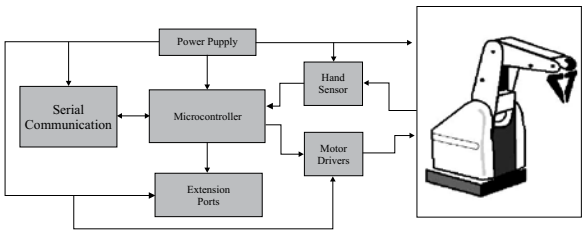

Figure 2 Block diagram of the electronic circuit of the robot

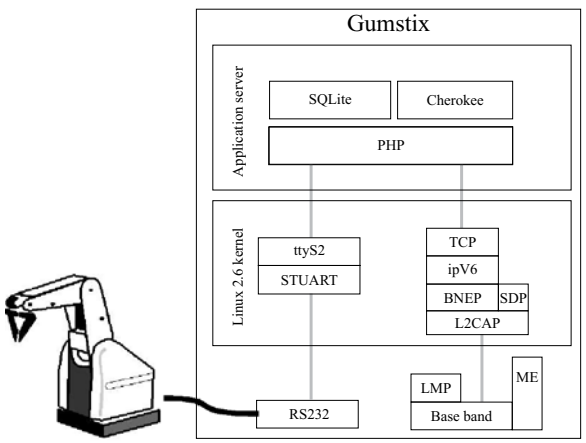

Figure 3 Embedded system software

The Bluetooth interface is managed by BlueZ, the official Linux Bluetooth protocol stack [12]. According to the specifications of the Bluetooth PAN profile [13], this device should be set up under the GN role.

The web server Cherokee [14] was installed as the application web server, this was chosen as a lightweight alternative to the popular Apache Web server. The language chosen for application was PHP version 5.2.0-r1 [15]. The database of users is handled through "SQLite" [16], which dispenses with the need for a "system of database management" because SQLite reads and writes directly to and from the files of the database that are on the disk. Since PHP5 has support for SQLite as an extension, there is no need to install any module for its use.

Web application design

The application was developed using the design pattern model-view-controller (MVC), where the model is the data base generated, the driver is the diagram of classes and the view is the site displayed to the user.

The main objective of having the database was to control the users who will have access to the robot "Microbot Teachmover", while at the same time recording the historic date and duration of sessions, as well as the details of the movements made during them.

As the application is accessed from mobile devices that have a small screen, as well as from fixed devices that have screens with greater capacities, the decision was made to have an authentication page demanding few requirements from the client device and where the users can then choose the type of interface they want.

Two types of interface were created; the first called Minimal (figure 4, left-hand pane), shows a page 240 pixels wide and 420 pixels high, with controls similar to those held by the Microbot Teachmover robot. This interface has four options: control, steps, administration and close. The option to control the robot is carried out by pressing the buttons for each motor and the speed of movement is changed by manipulating the field "VEL". In option *steps* the user must enter a number between -250 and 250 in each field to give the number of steps and the direction in which each motor will move. The possibilities given under the option "administration" depend on the user profile of the person who is handling the application. If the user has the profile "operator", the can only change their password and if the user is an administrator they can introduce new users to the system. Finally, in the option *close* the session is terminated and the robot is free for someone else to handle it.

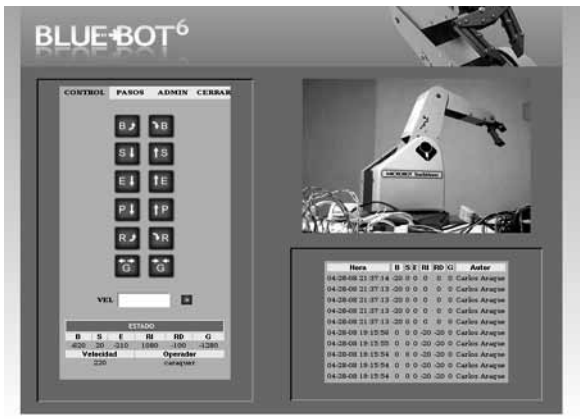

Figure 4 System Interface Snapshot

The second option called Full Interface (figure 4) has a screen size of 800x600. On the left side it has the same functionality and appearance as the minimal interface, but in the upper right quarter the video with the movement of the robot in real time is shown. The video shown on the interface is delivered by the webcam that is located on the computer that functions as a router. The bottom right quarter of the screen displays the most recent ten movements, the time they were made and their author.

Router software

For the implementation of the router the operating system Linux Debian 4 (Etch) was used, with kernel 2.6.18-6 which comes with IPv6 by default. The Bluetooth connection, the IPv6 tunnel, routing and server video streaming were set-up in this computer.

Bluetooth interface settings

The Bluetooth protocol stack used was also BlueZ, specifically the package "bluez-utilis 3.7-1 " which contains the tools and the demons to use Bluetooth. The package "Bluez-hcidump 1.32-1" was also used to analyze the Bluetooth packets. According to the specifications of Bluetooth PAN profile, this device should be set up under the NAP (Network Access Point) role.

Tunnel configuration

For the configuration of the tunnel a service called "tunnel broker" was used. The tunnel broker performs as a virtual ISP (Internet Service Provider) which provides IPv6 connectivity over an IPv4 infrastructure and a domain DNS (Domain Name Server). The Tunnel brokers are described in RFC 3053 [17].

The tunnel broker Go6 was chosen, because of its non-requirement for updating the current IPv4 address. This feature reduces the complexity of the configuration taking into account that in this project a dynamic IPv4 public address was being used.

Video streaming server

Remote teleoperation makes no sense if you cannot have any feedback of the movements through sensors or in visual form. The most interactive although less precise form of feedback is through graphics. This can be done using a streaming video server that can preferably be consulted from the same application thus avoiding inconvenience to the user.

It was decided to install the video streaming server on the router which should have good hardware resources to implement the video codec and send it in real time. The player and multimedia content server VLC (VideoLAN Client) [18] was selected as this offers IPv6 support. This software is free under the GPL.

Client application

The client application can be any Web browser with support for IPv6.

As mobile device a PDA Palm TX was used, which runs the Palm® Garnet 5.4 operating system [19]. Since Garnet 5.4, does not support the protocols bnep and IPV6, Linux Angstrom was installed on the PDA, allowing the installation of the IPv6 module, using the "BlueZ" Bluetooth protocol stack for Linux and the web browser *konqueror*.

It was also necessary to configure an IPv6 tunnel from the fixed remote device. The same gateway6 client was used for this part as used in the router, but in the appropriate version for Windows XP Service Pack 2.

Microbot teachmover robot

The firmware of the robot was written in assembly code. This is a state machine that receives commands by the serial port and executes it handling the motors.

Results

Description of final assembly

Figure 1 shows the final assembly with the addresses used and the roles configured in each device; all devices were left with two stacks of IP. On the computers that are connected directly to the tunnel is a domain name given by the "Tunnel Broker" server.

The operational test of the three scenarios proposed was successful. These are:

Manipulation from a mobile device: The embedded system has the role GN, and the mobile device the role PANU, hence using the PAN profile as an ad hoc network group. Once it has configured IPv4 and IPv6 addresses in the devices, it is possible to access the application via the web browser using the IPv6 address as well as the IPv4 address.

Manipulation using a local fixed node: in this case the node has the fixed NAP role while the embedded system has the PANU, therefore using the PAN profile as a network access point. As in the above scenario once IPv4 and IPv6 addresses have been configured, it is possible to access the application via the web browser using the IPv6 address as well as the IPv4.

Handling from remote fixed node: for the remote node the use of Bluetooth is transparent, it only requires configuration of the IPv6 tunnel. For configuring the wireless personal area network (WPAN), the embedded system and the router have PANU and NAP roles respectively.

The embedded system may simultaneously be working with the GN role for connecting a mobile device and be connected to the router using the role PANU, in which case the mobile device has reached the router via the scatternet formed.

The analysis focused on the development of two types of connections that are handled. The first is the one used in the WPAN where it has Bluetooth and IPv6, and the second is the tunnel from fixed node and remote node.

Analysis of IPv6 over bluetooth

The analysis was performed in three stages: first, review the structure of the protocol BNEP and IPv6 from the captured frames generated handling the robot. Second, test the delay between the devices, and finally establish the data rate that the connection can deliver.

IPv6 – BNEP Connection

This test was performed, driving the robot from the application in the router and capturing the packets generated at the interface PAN0 of it, through the software hcidump and Wireshark. It was necessary to capture packets with the two tools that hcidump acknowledges as the protocols of Bluetooth, while Wireshark recognizes the protocols of the upper layers.

At the time of the creation of the connection only PAN frames that are Bluetooth are exchanged; these are shown below:

> ACL data: handle 42 flags 0x02 dlen 11

L2CAP(d): cid 0x0040 len 7 [psm 15]

BNEP: Control(0x01|0)

Setup Req(0x01) size 0x02 dst 0x1116(NAP) src 0x1115(PANU)

< ACL data: handle 42 flags 0x02 dlen 8

L2CAP(d): cid 0x0040 len 4 [psm 15]

BNEP: Control(0x01|0)

Setup Rsp(0x02) res 0x0000

First comes a 0x01-type BNEP packet with 0x01 control type; that is BNEP_SETUP_CONNECTION_RESPONSE_MSG where it is specified that UUID of 2 bytes are managed and that the destination is the UUID NAP (0x1116) and the source is the PANU (0x1115). To accept the previous packet, a 0x01 BNEP packet type with 0x02 control type is sent; that is BNEP_SETUP_CONNECTION_REQUEST_MSG and response 0x0000 indicating that the connection is successful.

After the connection the Linux operating system, which is used in the system, embedded in the router, bnep0 creates an interface through which continuous communication takes place. The packets captured by hcidump at the time of the manipulation of the robot have the following characteristics:

< ACL data: handle 42 flags 0x02 dlen 111

L2CAP(d): cid 0x0040 len 107 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

> HCI Event: Number of Completed Packets (0x13) plen 5

> ACL data: handle 42 flags 0x01 dlen 28

L2CAP(d): cid 0x0040 len 107 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

< ACL data: handle 42 flags 0x02 dlen 111

L2CAP(d): cid 0x0040 len 107 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

> HCI Event: Number of Completed Packets (0x13) plen 5

< ACL data: handle 42 flags 0x02 dlen 79

L2CAP(d): cid 0x0040 len 75 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

> ACL data: handle 42 flags 0x02 dlen 71

L2CAP(d): cid 0x0040 len 67 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

> ACL data: handle 42 flags 0x02 dlen 83

> ACL data: handle 42 flags 0x01 dlen 28

L2CAP(d): cid 0x0040 len 107 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

< ACL data: handle 42 flags 0x02 dlen 111

L2CAP(d): cid 0x0040 len 107 [psm 15]

BNEP: Compressed(0x02|0)

[proto 0x86dd]

> HCI Event: Number of Completed Packets (0x13) plen 5

In the capture it can be seen that ACL packets handled. BNEP 0x02-type packets are sent over the L2CAP protocol (BNEP_COMPRESSED_ETHERNET) because there are no more devices on the network segment. In addition the network layer protocol 0x86dd is being carried which corresponds to IPv6.

Below is one of the packets captured by Wireshark at the time of the manipulation. It is important to clarify here that this is not the complete packet; it only shows part of the header and payload, the latter will be cut as it is not important to show this content analysis:

0000 00 80 37 2e 46 06 00 11 f6 09 5e 9d 86 dd 60 00 ..7.F... ..^...`.

0010 00 00 02 82 06 40 20 01 05 c0 96 41 00 07 00 00 .....@ . ...A....

0020 00 00 00 00 00 01 20 01 05 c0 96 41 00 07 00 00 ....A....

0030 00 00 00 00 00 10 8b 58 00 50 b2 5d 57 b9 ae c9 .....X .P.]W...

0040 fb d6 80 18 01 9e 66 37 00 00 01 01 08 0a 00 1e ......f7 ....

0050 3d ee 00 03 c5 9e 47 45 54 20 2f 63 6f 6e 74 72 =.....GE T /contr

0060 6f 6c 5f 4d 69 6e 69 6d 61 2e 70 68 70 3f 63 6f ol_Minim a.php?co

0070 6d 61 6e 64 6f 3d 64 6f 4d 6f 74 6f 72 42 26 73 mando=do MotorB&s

0080 65 6e 74 69 64 6f 3d 6d 61 20 48 54 54 50 2f 31 entido=m a HTTP/1

The first three fields correspond to the BNEP protocol: the first is the destination MAC (00 80 37 2e 46 06), the second is the MAC origin (00 11 f6 09 5e 9d), and the third is the type of network layer protocol (86 dd), which corresponds to IPv6. Here it is interesting to note that the Wireshark recognizes BNEP as Ethernet II, which allows the conclusion that the emulation of Ethernet with BNEP is completely fulfilled.

With regard to IPv6 protocol, you can see the first field indicating the version (6). What follows are types of traffic (0) and flow label (00 00 00), which as you can see are not used because their value is zero. Then comes the size of the load (02 82), the identifier of the protocol of the header that follows in this case is TCP (06), which breaks the limit which is set at 64 (40), the source address (20 01 05 c0 96 41 00 07 00 00 00 00 00 00 00 01) and the destination address (20 01 05 c0 96 41 00 07 00 00 00 00 00 00 00 10) which correspond to those shown in figure 1.

The remainder of the information captured belongs to TCP and HTTP, and is not a priority for this study.

IP-Bluetooth Delay

To measure the delay in IP-Bluetooth connection 500 ping commands were issued from the router to the embedded system at different distances (1m, 4m and 7m) using both IPv4 and IPv6. Table 1 shows the mean and standard deviation of the data obtained in each of the tests in order to provide far more detailed information on the behavior of the delay in the connection.

Table 1 Mean and standard deviation delay for IPv4 and IPv6 at different distances using Bluetooth

Distance (m)	*IPv4 Delay (ms)*	*IPv6 Delay (ms)*
1	59.72 ± 8.21	59.26 ± 8.88
4	59.27 ± 8.14	58.69 ± 9.7
7	58.51± 8.14	58.54 ± 7.82

The results show that the connection within seven meters has no variations in the delay as the distance increases, since both the average and the standard deviations keep similar values and, most importantly for the study, there is no difference whether it uses IPv4 or IPv6.

IP-Bluetooth Throughput

To measure the data rate that can be achieved between the connection of the embedded system and the router a 1.5 MB file was sent from the router into the embedded system using scp (ssh). A larger file was not sent because there was no more available memory in the embedded system.

Table 2 shows the average data rate at which the file was able to be sent. These averages were obtained from ten samples, and taken from information provided by the scp after sending the file.

Table 2 Average data rate at different distances using IPv4 and IPv6

Distance (m)	*IPv4 data rate (kB/s)*	*IPv6 data rate (kB/s)*
1	71.27	67.63
4	74.21	71.04
7	69.33	70.99

As in the analysis of the delay, this shows that the data rate does not present any significant amendments when changing the distance between 0 and seven meters, and neither does it depend on the version of IP protocol.

IPv6 tunnel analysis

As in the preceeding analysis of IPv6 over Bluetooth, the structure of the plot, the delay and the data rate of the packets sent through the tunnel from the remote node to the router for WPAN were analyzed.

IPv6 – IPv4 tunnel structure

The capture took place in the packet interface connected to the WAN (Wide Area Network) with the Wireshark software. Below a packet of embedded system response to the remote node is shown and discussed. This covers where it sends the html code that is displayed on the user's web browser. It is important to clarify here that this is not the complete packet, it only shows part of the header and payload, and it has been cut as it is not important to show the contents for analysis. In general the packet has protocols Ethernet, PPPoE, PPP, IPv4, IPv6 and TCP.

0000 00 19 c6 07 ca b1 00 16 17 20 57 fd 88 64 11 00 ..... W..d..

0010 13 14 05 16 00 21 45 00 05 14 00 00 40 00 40 29 .....!E.....@.@)

0020 f0 ba be 63 ed d8 40 56 58 74 60 00 00 00 04 d8 ...c..@VXt`.....

0030 06 3f 20 01 05 c0 96 41 00 07 00 00 00 00 00 00 .? ....A.....

0040 00 10 20 01 05 c0 8f ff ff fe 00 00 00 00 00 00 .. ..

0050 98 51 00 50 04 c1 c6 6f df b7 33 50 b2 e4 50 10 .Q.P...o..3P..P.

0060 25 b0 57 22 00 00 3c 2f 74 64 3e 0a 20 20 20 20 %.W"..</td>.

0070 3c 74 64 20 63 6c 61 73 73 3d 22 73 74 79 6c 65 <td class="style

In the first field of the Ethernet header is the destination MAC (00 19 c6 07 ca b1). In the second field is the source MAC (00 16 17 20 57 fd) that corresponds to the interface of the router and the third field is the type of network layer protocol (88 64) which in this case is another link layer protocol called PPPoE (Point to Point Protocol over Ethernet).

PPPoE is used for connection to the ISP. The first field refers to the version (1), then type (1), which always has the value 1 in the version 1 of PPPoE. The third field is The CODE (00) which indicates that the PPP session has already started. Finally is the session identifier field (13 14) and size of the content (05 16). In the PPP there is only one field (00 21) indicating the encapsulation of IP. In the IPv4 header in the first place is (4) indicating the version of the protocol, then the size of the header (5), type of service (00), size of the content (05 14), datagram ID (00 00) and flags indicator (4). Then, to avoid packet fragmentation, fragment offset (00), time to live (40), protocol sent (29) indicating the encapsulation of IPv6, check sum of the header (f0 ba), source address (be 63 ed D8) which corresponds to the IPv4 publishing the router and finally the destination address (40 56 58 74), which, in this case, is the server's IPv4 tunnel broker.

In the IPv6 protocol is in the first field version (6), then kind of traffic (0) and flow label (00 00 00) which, as for the frames for IPv6 over Bluetooth displayed above, are not used because their value is zero. Then the size of the content (04 D8), the higher layer protocol TCP (06), the hop limit (3f), the source address (20 01 05 c0 96 41 00 07 00 00 00 00 00 00 00 10) which is embedded in the system and ultimately the destination address (20 01 05 c0 ff ff 8f fe 00 00 00 00 00 00 98 51) are shown.

One can see that the head of IPv6 is the largest of all those tested due to the large size of the source and destination addresses. The analysis of the protocols above is beyond the scope of this work.

IPv4-IPv6 tunnel delay

To measure the delay in the IPv6 tunnel-IPv4 500 ping commands were issued from the router to the remote node tunneled using IPv6 and IPv4. The results obtained using IPv6 are shown in figure 5, while the cast using IPv4 is shown in figure 6.

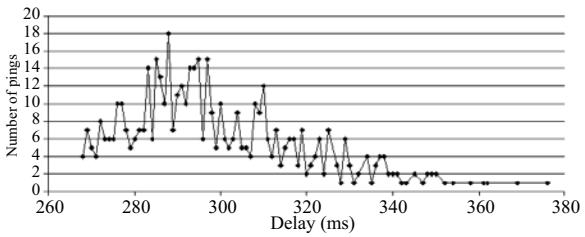

Figure 5 Ping delay from router to remote node using IPv6 tunneling

In figure 5 the mean and standard deviations for the ping command executed 500 times, using tunneled IPv6 were 299.39 ms and 20.24 ms respectively.

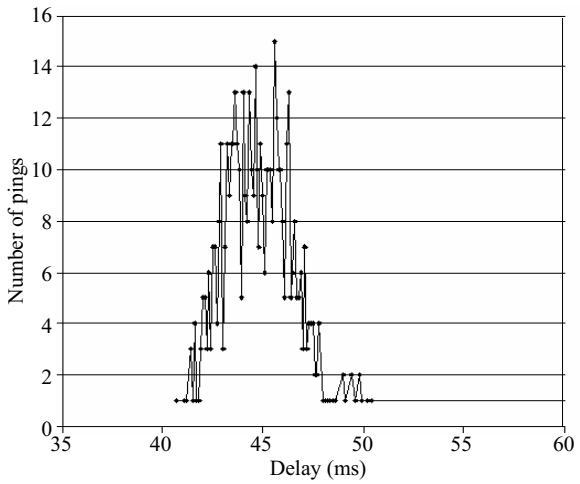

Figure 6 Ping delay from router to remote node using using IPv4

In figure 6 the mean and standard deviations from 500 ping commands, using IPv4 were 46.07 ms and 10.7 ms respectively.

From the above results it can be seen that there is a great difference between connecting to a host using IPv4 only or doing it using IPv6 over IPv4. This difference is because the tunnel broker server used is located in Canada while the testing equipment was located in Cali (Colombia). Data packets go to the server in Canada as the first hop of the IPv6 connection, but there are many intermediate IPv4 hops. Doing a tracert6 from the remote node to the router the following hops were displayed:

C:\Documents and Settings\admin>tracert6 caraquer.broker.freenet6.net

Trace to address caraquer.broker.freenet6.net

[2001:5c0:8fff:fffe::6967]

from 2001:5c0:8fff:fffe::9851 over a limit of 30 hops:

1 126 ms 124 ms 125 ms 2001:5c0:8fff:fffe::9850

2 281 ms 281 ms 281 ms caraquer.broker.freenet6.net

[2001:5c0:8fff:fffe::6967]

Trace complete.

It is important to note that the results of these tests are highly dependent on the ISP network used.

IPv6 – IPv4 tunnel throughput

In order to determine whether the connection IPv6 with IPv4 tunnel had reduced the data rate of the connection with respect to that obtained with IPv4 alone, a client FTP (File Transfer Protocol) was installed in the router. From this router a 1.5 MB file was downloaded using the remote node on ten occasions using IPv4 and again using IPv6 a further ten times. The average download data rates from the remote node for IPv4 and IPv6 were 54.06 kB/s y 13.48 kB/s respectively.

Although the data rate of download depends directly on the Internet connection, it was expected that at best the data rate with the two versions would be the same, but in fact it showed a pretty big decline in the use of the tunnel, which

corresponds to 24.93% of the data rate using IPv4.

As in the analysis of delay this decline is due to the long journey the packets have to make to go up the tunnel server broker.

Discussion

In the following, both the advantages and disadvantages encountered in this work while using IPv6 over Bluetooth are presented.

Benefits

Bluetooth provides wireless connectivity under a standard widely deployed, allowing the use of various communication modes through a single interface, along with low energy consumption, reasonable data rates. *Ad hoc piconets* can connect up to seven devices at a maximum distance of 100 m when class 1 devices are used, and from different piconets have the capability of forming *scatternets*. Without IP connectivity the Bluetooth PANs would be isolated from the Internet, losing the ability to access and provide services to other users or devices connected to the Internet anywhere in the world. Thanks to Bluetooth's good support for IPv6 the advantages of using Bluetooth with IPv6 connectivity is really the sum of the advantages of each of these technologies.

Although the main advantage of IPv6 is the large number of IP addresses available, it also has other important features such as auto configuration, no need for NAT (Network Address Translation) retrieving the end-to-end communication concept, offering a potential for any host device to exploit advanced services such as, labeling flow, security and very importantly for further study: mobility.

The above mentioned advantages make IPv6 over Bluetooth an interesting alternative for work in wireless sensor networks, ubiquitous computing and new types of applications where mobile embedded devices provide services and have M2M (Machine to machine) communication via the Internet. All these applications can take advantage of the IP convergence, allowing devices to communicate using different access technologies such as Bluetooth, GSM, xDSL, Ethernet or any other technology.

Disadvantages

Bluetooth is not the best access technology in all cases. For instance it is not the best option for sending small data packets in a large network where there is no need for a very high data rate, such as simple sensors or switches; for these cases a technology such as ZigBee [20] would be more appropriate, requiring minimal energy consumption. However, at present IPv6 over ZigBee is not yet mature. The RFC4944 which defines the transmission of IPv6 packets over IEEE 802.15.4 networks (physical layer and data link layer) was only issued in September 2007.

A problem detected with the Bluetooth technology is that the pairing of the devices is a bit complicated for an average user although this disadvantage has been addressed in version 2.1 EDR+.

Currently it is difficult to exploit all the features of IPv6 on the Internet because many local ISPs do not yet support it. One advantage of IPv6 which could feasibly be exploited on the Internet at present is the large number of addresses available, but a decrease in data rate due to the IPv4-IPv6 tunnels needed should be expected.

Likewise it is hard to find mobile devices that can exploit the advantages of IPv6 over Bluetooth because not all mobile devices have implemented the Bluetooth PAN profile and very few more support IPv6.

Conclusions

In this work, the design and implementation of a Bluetooth wireless personal area network using the IPv6 protocol was successfully achieved. The Bluetooth WPAN can be accessed from any IPv6 host on the Internet. The connection to the global IPv6 network was accomplished through an IPv6 over IPv4 tunnel. On the Bluetooth wireless

network is the Microbot Teachmover robot connected to an embedded system which has a web server along with a database. It runs a web application on the embedded system that allows the manipulation both locally and remotely of a robot. Additionally, in order to see the execution of sent commands, a video streaming server on a computer that acts as IPv6 router (gateway) for the PAN network was implemented.

The analysis of the structure of the packets sent with BNEP and IPv6 confirmed that BNEP makes a good Ethernet emulation because the Linux operating system creates a network interface that allows handling it as if it were an Ethernet connection, as the Wireshark protocol analyzer was not aware of the existence a Bluetooth connection. The results obtained showed that the result of using BNEP is completely satisfactory.

The delay of packets and the data rate of the connection achieved by using IPv6 and IPv4 at distances of 1 m, 4 m and 7 m was compared, and from this it is concluded that the IP version and the distance within a range of seven meters does not affect the delay or the data rate of the connection, since for most of the cases delays of around 59 ms and data rates of around 72 kB/s were observed.

For the IPv6 tunnel over IPv4 a similar analysis was performed. For a connection between two hosts using IPv6 through an IPv4 tunnel to the global IPv6 network, the delay had an average increase of 253 ms, while the data rate with IPv6 decreased to 24.93% of that obtained with IPv4. This difference in results is a consequence of the need for packets to go to the tunnel broker server which was, in this case, located in Canada.

Bluetooth is a communication interface suitable for devices with low power consumption and intermediate data rate needs, i.e. it is a solution situated somewhere halfway between ZigBee and Wi-Fi. Bluetooth makes it possible to bring IP convergence to medium capacity devices, allowing the fulfillment of one of the IPv6 goals which is to provide IP connectivity with a lot of devices.

IPv6 has many advantages over IPv4, but in this work the end-to-end connectivity was the primary goal to verify. It is concluded that using the transition mechanism provided by the tunnel broker not all the possibilities announced by IPv6 can be exploited. Other methods of transition should be explored or, simply, it could be that native IPv6 connections are needed to fully exploit the improvements of the version 6 of Internet protocol.

It was verified that Bluetooth technology is fully capable of carrying the IPv6 protocol, but it will not be possible to exploit this until enough mobile devices have the PAN profile and IPv6 activated.

At first view, the increase in the size of the IPv6 header could lead one to think that the efficiency of communication can be seen as being diminished, but when looking at packet payload size it can be concluded that the 20 byte overhead is worth sacrificing compared to the benefits, especially bearing in mind that the data rate communications grow higher by the day and that the time spent in transmitting the overhead bytes is negligible.

During this work the convenience of Linux in several respects became evident. Its portability allowed us to have the Linux operating system running in the three types of device used in this work (embedded systems, mobile devices and personal computers). Linux's open software helped us avoid, at some points, that the project development should become unfeasible because of not having full access to the operating system code. Finally, the availability of tools and support for IPv6 and the Bluetooth PAN profile allowed us to meet the requirements of this project.

References

1. V. J. Harward, P. H. Bailey, K. DeLong, C. Felknor, J. Hardison, B. Harrison, C. Varadharajan, J. A. del Alamo, S. R. Lerman, J. Carpenter, I. Jabbour, S. Wang, P. D. Long, T. Mao, L. Naamani, J. Nothridge. M. Schulz, D. Talavera, K. Yehia, R. Zbib, D. Zych. "The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet

Accessible Laboratories". *Proceedings of the IEEE*. Vol. 96. 2008. pp. 931-950.

2. Y. Wang, S. Butner, A. Darzi. "The Developing Market for Medical Robotics". *Proceedings of the IEEE*. Vol. 94. 2006. pp. 1763-1771.

3. A. Vallejo, J. Ruiz, J. Abella, A. Zaballos, J. M. Selga. "State of the Art of IPv6 Conformance and Interoperability Testing". *Communications Magazine. IEEE*. Vol. 45. 2007. pp. 140-146.

4. R. Ali, J. K. Pollard. "Real-time Voice Over IP Over Bluetooth". *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*. IDAACS. Sept. 2005. IEEE. pp. 580-583.

5. S.Cheng, J. Lin, "IPv6-based dynamic coordinated call admission control mechanism over integrated wireless networks", *IEEE Journal on Selected Areas in Communications*. Vol. 23. 2005. pp 2093-2103.

6. S. Zeadally, A. Kumar, A. Banda, "Seamless Bluetooth Connectivity to IPv6 Networks". *Proceedings of International Conference on Embedded and Ubiquitous Computing* (EUC 2004*). Lecture Notes in Computer Science* (LNCS). Springer Verlag. Aizu (Japan). Vol. 3207. 2004.

7. O. Rodríguez, R. Maya. *Implementación de una red inalámbrica Bluetooth*. Thesis Universidad del Valle. 2003. pp. 55-83.

8. J. Puentes Caicedo, A. Montaño Sarria. *Desarrollo de una aplicación de redes inalámbricas para el monitoreo de eventos y transmisión de datos utilizando tecnología Bluetooth*. Thesis Universidad del Valle. 2006. pp. 27-46.

9. Btnode. *A Distributed Environment for Prototyping Ad Hoc Networks*. http://www.btnode.ethz.ch. Consultada el 15 de febrero de 2008.

10. Gumstix Inc. Embedded Systems. Connex 40xm motherboard. http://gumstix.com/store/catalog/product_info.php?products_id=136. Consultada el 3 de marzo de 2008.

11. Intel's PXA255 Processor Datasheet. http://www.alldatasheet.com/datasheet-pdf/pdf/77314/INTEL/PXA255.html. Consultada el 10 de febrero de 2008.

12. BlueZ Official Bluetooth Protocol Stack for Linux. http://www.bluez.org. Consultada el 5 de abril de 2008.

13. Personal Area Networking Profile, version 1.0, February 14, 2003. http://www.bluetooth.com/Bluetooth/Technology/Building/Specifications/Default.htm. Consultada el 23 de octubre de 2007.

14. Cherokee light web server, official site. http://www.cherokee-project.com/. Consultada el 5 de abril de 2008.

15. Official site of the PHP programming language. http://www.php.net/. Consultada el 10 de abril de 2008.

16. Official site manager of SQLite database manager. http://www.sqlite.org/. Consultada el 7 de abril de 2008.

17. A. Durand, P. Fasano, I. Guardini, D. Lento. "IPv6 Tunnel Broker". *RFC 3053, Informational*. 2001. pp. 1-10.

18. Official site of the VLC streaming server. http://www.videolan.org/. Consultada el 7 de abril de 2008.

19. Access company, "Garnet OS brochure", 2007. http://www.access-company.com/PDF/garnetos_brochure.pdf . Consultada el 7 de octubre de 2007.

20. ZigBee Specification, version 1. December 2004. http://www.zigbee.org/en/spec_download/download. Consultada el 2 de agosto de 2008.