2012 15th International IEEE Conference on Intelligent Transportation Systems
Anchorage, Alaska, USA, September 16-19, 2012

Continuous Broadband Communication System Based on Existing Open Source Network Tools for Vehicular Environments

Gorka Urquiola, Asier Perallos, *Member, IEEE*, Roberto Carballedo

***Abstract*—Due to the widespread adoption of Intelligent Transportation Systems, the number of applications used in vehicular environments is growing very quickly. This fact implies a greater consumption of network bandwidth and a competition for network use. Consequently an exhaustive control of the bandwidth consumption is needed to provide Quality of Service according to the demands of certain applications, giving priority to the most relevant data traffic. Moreover, to guarantee the continuous communication in mobile environments (between vehicles and ground control centers) is another target to be tackled. In this paper a software system to provide continuous broadband communication in vehicular environments, as well as other capabilities such as the management of the Quality of Service and priority of applications, transparent change of the active communication link and security of the data transmission, is presented. The remarkable challenge is that has been addressed a solution to this problem not based on a proprietary software development, but by reusing already implemented and tested software utilities, originally designed to be used in several non-mobile environments.**

I. Introduction

In surface transportation systems, it is usual to adopt vehicle to ground architectures [1], in which it is necessary to maintain the communication between the mobiles and control center nodes or even the communication between all the mobile nodes.

Unlike the common network configurations used in non-mobile environments, such as in an office (where static network links are used, continuous communication can be assured using wired and backup links [2], and the service failure probability depends on rare environmental factors and internet service provider quality), in mobile environments the continuous communication is not as easy to guarantee. The reason is that the communication could be affected by several dynamic factors, such as changing coverage according to the location of the vehicle, data packet loss and event cuts in the communication may occur.

Moreover, the number of applications used in this kind of mobile environment is growing in an exponential way due to the requirements of the Intelligent Transportation Systems (ITS) [3]. The mobile internet services offered by the internet service providers are not always capable of providing a suitable bandwidth that meets the needs of such applications. Consequently an exhaustive control of the bandwidth consumption is needed to provide quality of service according to the demands of certain applications, giving priority to the most relevant data traffic and leaving in background the not critical traffic [4].

In order to have a greater connectivity and coverage, we could use 3G modems for accessing to the Internet. Instead of developing software able to manage the different links, establishing the active channel to use based on factors such as coverage, availability and bandwidth, we decided to combine existing software tools to achieve a communication system of continuous connectivity in a mobile (vehicular) environment.

To do this, we designed a system based on a GNU/Linux distribution, using only free and open-source software with a fairly widespread use. Thus, we can develop a communication system with a minimum initial investment and whose robustness and fault tolerance is guaranteed by the maintenance of a community of developers.

The paper is organized as follows: initially the objectives and requirements of the developed communication system are described, then its design with the description of the used tools, explaining the reasons for choosing each of them; in the fourth section some scenarios that could be used as well as tests to check its behavior are described; and finally the conclusions as well as future work are included.

II. Technical Requirements

The number of applications used in vehicular environments is growing very quickly, which implies a greater consumption of network bandwidth. This can be a problem for the applications that consume lower bandwidth, but which transmission priority is higher. Thus the regulation of applications network traffic becomes important, as an excessive network bandwidth consumption by a secondary application may cause delays or even data packet loss of a priority application.

The best way to get a communication system with broadband coverage is to have multiple internet links of different ISPs and using the link that best suits the circumstances. This implies two problems: first, that the IP address of each link will be variable; and the second one is that applications will have an uncontrolled behavior when the system performs a link change because they may assume it as a break in the link. So it is required a transparent middleware for the applications, so when a change of the active channel occurs they can continue to operate without being aware of this change.

Gorka Urquiola, Asier Perallos and Roberto Carballedo are with the DeustoTech Mobility Research Group from the University of Deusto, Avda. Universidades 24, 48007 Bilbao, Spain (e-mails: {gurquiola; perallos roberto.carballedo}@deusto.es).

The security in communications is another aspect to consider as applications may be required to transmit sensitive information between the mobile node and ground centers, such as ticketing information using NFC or contact/contactless SmartCards.

Another target is not requiring that all applications run on a single device, allowing to use additional devices such as sensors or IP cameras, that act as additional nodes in a subnet. That is the reason because it would be desirable that the on board communication system had to be able to manage a subnet and serve as a gateway to the control application located on ground center.

III. SYSTEM DESIGN

The system follows a vehicle-to-ground architecture based on the existence of an Onboard Communication Module (OCM) and a Ground Communication Module (GCM). The onboard module has several different wireless communication links – 3G modems, WiFi antennas – and the ground module has a wired broadband link (Fig. 1). The onboard module will only use one of the available links (we refer it as the active channel), and in case that the active channel breaks, the active channel will be changed to another available link in order of priority. Similarly, if a link with higher priority recovers the system will re-establish the link as the active channel.

Using a host-to-host type VPN will cover two of the system requirements. On one hand, the communication will be encrypted with a cryptographic symmetric key, providing an additional security layer for data transmitted between onboard and ground modules. Furthermore, the use of a VPN involves the creation and use of a virtual interface whose IP address will be the same regardless of the physical link being used at any time, so the use of these virtual interfaces for the communication between the two extremes ensures that applications do not have to change their settings every time the active channel is changed.

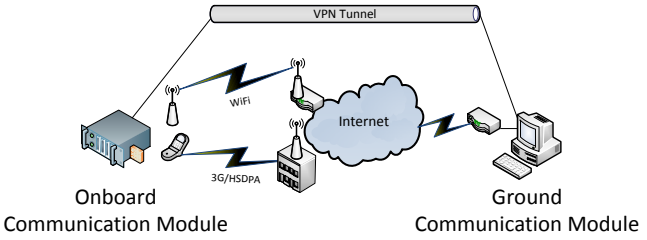

Fig. 1. System conceptual architecture

Since most of network traffic will be sent from the mobile node to ground center, a Quality of Service (QoS) management solution will be implemented in the mobile end, setting the network traffic rules on the virtual interface created by the VPN. Thus, no matter what is the currently active link, since all network traffic will be transmitted using the virtual interfaces.

Finally, the combination of the virtual interface and the software tool ip route [5], (available in most GNU/Linux distributions) allows changing the active link in a transparent way for the applications. This is because when the default outgoing route is changed in the routing table, the software responsible of managing the VPN connection automatically sets the new route and reestablishes the connection, so the applications using the virtual interface are not aware of this change and is not detected as a broken link.

Summing, the onboard module must behave like a kind of router able to: decide which the active channel is between the potentially available physical links, manage the host-to-host type VPN to ground module, manage the private subnet of the mobile node, run third party software and redirect the data traffic of the private subnet to the ground center.

A. Software utilities used

No software was developed in this solution, but it has tried to combine and configure already existing and available software tools to meet the initial requirements of the environment.

For the deployment of the system, the software used was: Ubuntu 11.10 [6] as GNU/Linux distribution, OpenVPN [7] for the host-to-host VPN management and various utilities from the Iproute2 utility collection and Netfilter framework [8], mainly iptables and traffic control for the QoS management.

1) Operating System – GNU/Linux

Although it can find equivalent tools on different operating systems like Microsoft Windows, it was decided to choose a GNU/Linux distribution for two reasons: first, that is free and open source, and second, that is easier to find and modify network management tools than in others.

2) VPN – OpenVPN

As VPN management software, OpenVPN was used due to its ease installation and free use.

A host-to-host type VPN must be configured for each onboard module to be managed from the ground node. The latter is the responsible of managing the communications between the different mobile nodes if they wanted to make a communication from a mobile node to another.

This type of VPN requires that one of the two nodes acts as a server and the other one as a client. Considering that the onboard physical links will have variable IP addresses depending on which the current active link is and the location of the mobile node, the ground module will be the VPN server and will be in charge of receiving request for connection establishment from each of the physical interfaces installed in the onboard modules.

Therefore, the design of the network architecture follows a star topology, where the ground module is the central node of the graph and the mobile modules are the leaf nodes.

The client-server connection establishment is made using the default route defined by the routing tables of the *iproute* command, from the Iproute2 utility collection, available in most of the GNU/Linux distributions. In case of modification of the default route, OpenVPN detects it and manages the reconnection to the server using the new route. This channel change is transparent to the applications and it is not considered as a change in the network or as a broken link, since the virtual interface for which it is communicating has been active the whole time.

3) QoS – iptables and Traffic Control

To ensure the quality of service of the active channel a combination of iptables and Traffic Control utilities has been used (Fig. 2).

Iptables belongs to the framework Netfilter and it is the default firewall used in GNU/Linux. For this solution its packet marking module will be used. With this module a mark will be added to each data packet redirected to the external network, either from the private subnet or the same system. This classification is based on the port used to transmit each of the packages, so the data packets of each application running in the mobile node can be classified.

The data packet marking rules are easily configurable and replaceable in case of making changes on the system.

Traffic Control, from the utility collection Iproute2, will manage the queue disciplines, prioritizing the outgoing data traffic. After iptables has marked the data packets according to established rules, Traffic Control associates each mark to a priority class and then classifies and manages the bandwidth usage limits.

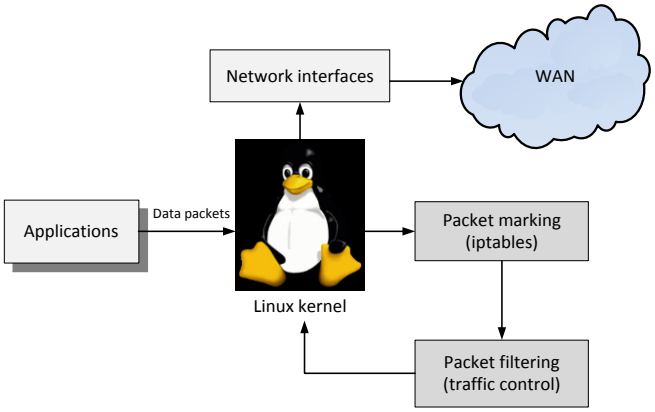

Fig. 2. QoS management in GNU/Linux using iptables and Traffic Control utilities.

Queue disciplines are used for the identification of classes of the different types of network traffic, using Hierarchical Token Bucket algorithm [10] to filter it. This algorithm allows dividing the available network bandwidth indicating a maximum and a minimum usage for each application, ensuring that the highest priority applications of the system may have the required bandwidth at any time.

B. Active channel change

It is necessary to have a system whereby the active link change is transparent to applications that are running in the mobile node. It implies not to consider the change as a network outage and be able to continue to communicate with ground node naturally and without to deal to redirect the data traffic to the new link. The same requirements applies to the traffic from ground to mobile node, which after a link change involves having to configure the applications each time the change takes place, requiring an additional system that alerts that this has happened.

This is because a virtual interface provided by a host-to-host type VPN has been used. This virtual interface provides to applications a layer of abstraction over the physical layers, as this virtual interface will not change at any time. OpenVPN, the VPN manager software, will be in charge of reconnecting the two ends when a physical link change occurs.

In the GNU/Linux distribution used, as in most of distributions, the active channel change occurs when the table of routing is changed. The routing table contains all the information about the routes of the available network interfaces. When a default route (including the default interface) wants to be modified, it must be done using with the ip route command.

By default, the priority of a link over another is managed by NetworkManager [9] software utility, which is the network manager used by default in Ubuntu 11.10. The priority order of the interfaces is defined in its compilation (wired, WiFi, 3G), so in case of requiring to change it could be done by modifying and recompiling the manager. Internally, NetworkManager uses the ip route command to set the active channel.

Summing, NetworkManager changes the routing tables when a link breaks or a more priority link sets up again and this can be also done manually using the ip route command.

IV. Usage Scenarios and Tests

To verify that the proposed system works a test plan has been developed and performed in laboratory settings, using a PC and an embedded system to simulate ground center and an onboard module. Both systems have Ubuntu 11.10 and OpenVPN installed.

Each test is divided as follows: a description of the presented scenario, the test actions being performed and the results obtained.

A. Mobile-to-Ground continuous connection

It is the basic test done on the communications system with the propose to prove that the active channel changes do not affect the normal behavior of the applications that use the virtual interfaces, since the VPN manages the reconnection to the other host when routing table is modified.

1) Scenario configuration

This scenario uses two machines: one simulates the OCM (Onboard Communications Module) and the other the GCM (Ground Communications Module).

The GCM has only one link: a broadband internet connection who serves to establish the VPN communication with the OCM. This connection must be configured so that a machine located outside the local network can access to the port assigned to the VPN. By default, this port is the UDP 1194 for OpenVPN.

The OCM has N links, either through Ethernet interface, WiFi or 3G modems, each independent links and access to Internet through different service providers (Fig 3). In a real scenario, all the links will be wireless.

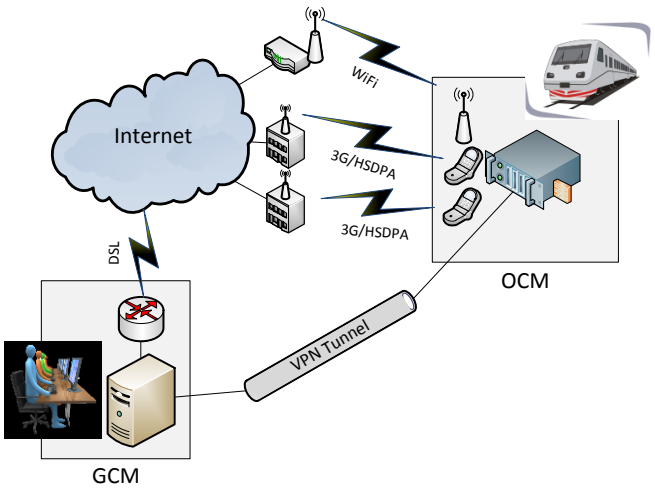

Fig. 3. Conceptual design of the Mobile-to-Ground communication system with one mobile unit and one ground control station.

2) Test actions to perform

First of all, we need to setup the OCM to make it behave as a router so may manage the private Ethernet network and redirect traffic to the ground node. To do this, it has been installed a software tool that allows to manage the system via a web interface easily. This software tool is called Webmin [11].

It is necessary a traffic monitoring tool to check if the appropriate physical interface is being used at all time and if the traffic is being encapsulated on the VPN. This can be made with different tools such as, for example, bwm-ng [11].

Once the systems are ready and monitoring tools have been prepared, the connection between OCM and GCM will be tested. Having in mind that the main idea is to test the continuous communication between the two virtual interface ends, one possible method is using the ping command. Following this method, we can both test the connection and check if any packet loss when active link changes.

The simulation of the active channel change will be done modifying the routing tables with ip route command while the OCM and ECM are communicating – with ping in this case.

The next lines are the trace of an ip route command result example, in which can be seen that the default route is via 10.64.64.64 using ppp0 named network interface.

```
admin@localhost:~$ ip route list
default via 10.64.64.64 dev ppp0 proto static
10.7.0.0/24 dev eth0 proto kernel scope link src 10.7.0.1
10.8.0.1 dev tun0 proto kernel scope link src 10.8.0.2
10.64.64.64 dev ppp0 proto kernel scope link src 75.80.149.142
169.254.0.0/16 dev eth0 scope link
```

3) Testing results

It can be observed how the active link changes are transparent to applications using the virtual interface when the default route is changed. Thus, this is a system with continuous connectivity in an environment where the active link change is an elemental key for optimal system performance.

B. Application server in the onboard Ethernet network

The monitoring of a fleet of trains or buses is one of the real scenario of use of this solution. It means that is the ground control station who wants to consume information about the vehicles, such as geo-data, reports or video streaming from onboard surveillance camera.

1) Scenario configuration

There is an IP camera connected to the onboard Ethernet that provides a web interface for internal camera management and configuration as well as obtaining the video streaming from the camera (Fig. 4). The control station wants to access the IP camera video streaming for security reasons.

A continuous video streaming and without cuts, according to the bandwidth capacity that the active link can provide and without having to design and develop an application to manage the reconnection each time the active link changes.

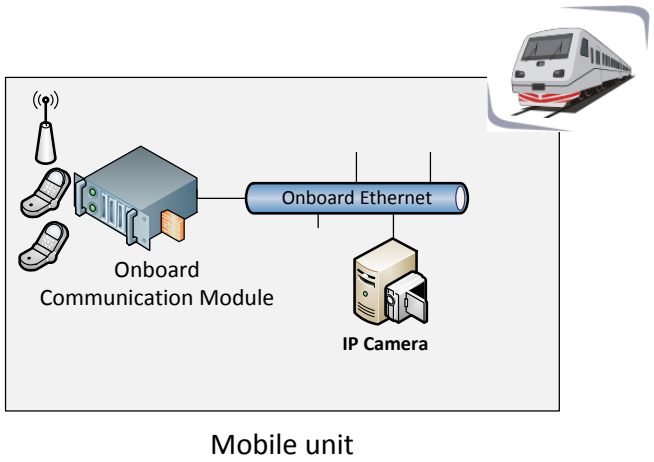

Fig. 4. Onboard Ethernet conceptual design with an attached IP Camera.

2) Test actions to perform

First, the OCM must be configured to do the correct port forwarding from the virtual interface to the IP camera, which acts as an independent node of the private onboard Ethernet network.

From the GCM the video streaming provided by the surveillance camera using the virtual interface IP is displayed. Thus, it can be checked if there are any cuts when the link changes occur. As in the previous scenario, the simulation of the active link change will be done modifying the routing tables with the ip route command.

3) Testing results

The video stream is continuous even when the active link change occurs. So, the applications and therefore the end user of the GCM are not aware of the active link changes because the OCM is reconnected to the VPN every time this happens.

C. Application server in the ground node

In addition to behaving as a router, the OCM has to be able to run applications that need to communicate with the GCM or with other mobile nodes connected to this GCM.

1) Scenario configuration

In a scenario of N mobile nodes connected to one GCM using the system proposed in the first scenario, it will be

used a VoIP software, for example, Mumble, a free to use and compatible with GNU/Linux utility.

Mumble [13] is a client-server application, so in this case the GCM will act as server and each of the mobile nodes will act as clients.

2) Test actions to perform

Mumble client must be run on each mobile node and configure it to connect to the ground node using the virtual interface provided by the host-to-host VPN.

In addition, as in previous cases, the routing tables will be modified to simulate the active physical link changes.

3) Testing results

The voice reception and transmission is continuous and Mumble clients and the server are not aware of the possible link changes that occur in all the mobile-to-ground communications. Therefore, all the link changes are transparent to applications.

Although the mobile nodes are not interconnected directly via VPN, using the GCM as communication hub, they can achieve the communication from one mobile node to another.

D. QoS management on multiple interfaces

In this system it is expected that applications have different priorities and consumption of bandwidth, thus the QoS management becomes a key aspect to be tackled. In that way, critical data, like the state of the vehicle motors or its geolocation, could be priorized above the transmission of video from a security camera.

1) Scenario configuration

In a scenario similar to the previous ones, in which there are N mobile nodes and one ground node, is going to be applied a QoS management system using the VPN interface on each mobile node.

2) Test actions to perform

As already noted before, the software tools used are very common and available in most of GNU/Linux distributions. For the QoS management it is used the combination of iptables and Traffic Control, in which iptables marks the data packets and Traffic Control filters the data packets based on the configuration of the rules.

After an analysis of the requirements of the applications running both on the OCM and in the onboard Ethernet, the traffic filtering rules will be designed and implemented. All rules will be applied to the VPN virtual interface and thus will apply no matter of the physical link being used at all times.

The system check will be performed using a traffic monitoring tool while the applications are running. They are consuming as much bandwidth as possible and making sure that in any case it is appropriate prioritization of data traffic according to the rules implemented.

3) Testing results

Applying QoS rules on the virtual interface, it is possible to implement the same rules for the overall onboard system. Instead of having to apply a single set of rules on each of the available physical interfaces.

Thus, QoS has been managed in the communication system and the highest priority applications can transmit even if a non-priority application consumes a high bandwidth rate.

V. CONCLUSION AND FUTURE WORK

In this paper we have presented the results of twelve months of research relatives to the deployment of a software solution to provide continuous broadband communication in mobile environments. Other related capabilities are the management of QoS and the priority of applications, the transparent change of the active channel and the security of the data transmission. The remarkable challenge is that has been addressed a solution to this problem not based on a proprietary software development, but by reusing already implemented and tested software utilities, originally designed to be used in several non-mobile environments. We have property them to fulfill a specific need for broadband communications between vehicles and their ground control center.

Although the system has been designed for transportation communications, it is also applicable to other scenarios, in which is required a continuous broadband connection in a mobile environment.

All tests have been performed in a laboratory setting. The results indicate that it is a feasible solution which could be used in real transportation environments with a wide range of applications.

Despite not having tested the system in a real environment (some test are scheduled for the next three month in a fleet of buses owned by a transportation operator of the north of Spain), it can be concluded that the basis of this communication system works properly.

The future work will be focused on the development of a tool able to dynamically monitor the bandwidth of each of the available links and choose the best one at every moment. The integration of this tool in the system will improve its performance enabling an intelligent change of the active channel due to the fact that it will seek the link with the highest bandwidth. Actually, in the test done all link changes have been simulated by executing commands manually on a terminal. In the future the integration of this wool in the actual system should be very easy because this tool will execute the same commands that now are executed manually on the terminal.

ACKNOWLEDGMENT

This work has been funded by the Basque Government of Spain under GAITEK funding program (GEINFEVI project, IG-2011/00472). Special thanks to DATIK - Irizar Group for their support.

REFERENCES

[1] I. Salaberria. U. Gutiérrez, R. Carballedo, A. Perallos. "Wireless Communications Architecture for "Train-to-Earth" Communication in the Field of Railways", 2nd International Symposium on Distributed Computing and Artificial Intelligende (DCAI), 2009.

[2] D. Staessens, D. Colle, M. Pickavet, P. Demeester, "Computation of high availability connections in multidomain IP-over-WDM networks", presented at the International Conference on Ultra Modern Telecommunications & Workshops, (ICUMT '09), 2009.

[3] J.K.-S. Lau, Chen-Khong Tham, Tie Luo "Participatory Cyber Physical System in Public Transport Application", presented at the Fourth IEEE International Conference on Utility and Cloud Computing (UCC), 2011.
[4] U. Gutiérrez, I. Salaberria, A. Perallos, R. Carballedo, "Towards a Broadband Communications Manager to regulate train-to-earth communications", presented at the 15th IEEE Mediterranean Electrotechnical Conference, 2010.
[5] Ip route, from Iproute2 collection utility: http://www.linuxfoundation.org/
[6] Ubuntu GNU/Linux: http://www.ubuntu.com/
[7] OpenVPN, VPN management software: http://www.openvpn.net/
[8] Netfilter, packet filtering framework: http://www.netfilter.org/
[9] NetworkManager: http://projects.gnome.org/NetworkManager/
[10] J.L. Valenzuela, A. Monleon, I.San Esteban, "A hierarchical token bucket algorithm to enhance QoS in IEEE 802.11: proposal, implementation and evaluation", presented at the IEEE 60th Vehicular Technology Conference (VTC), 2004.
[11] Webmin: http://www.webmin.com/
[12] Bwm-ng: http://www.gropp.org/
[13] Mumble, VoIP software: http://mumble.sourceforge.net/