Leveraging on Digital Signage Networks to Bring Connectivity to IoT Devices

J. David de Hoz, Jose Saldana,
Julián Fernández-Navajas, José Ruiz-Mas
I3A, University of Zaragoza
Ada Byron Building, 50018, Zaragoza, Spain
e-mail: {dhoz, jsaldana, navajas, jruiz}@unizar.es

Rebeca Guerrero Rodríguez, Félix de Jesús Mar Luna, Raúl Iván Herrera González
Technological University of Durango
Carretera Durango – Mezquital, 34080 Durango, México
e-mail: {rebeca.guerrero, felix.mar, raul.herrera}@utd.edu.mx

***Abstract*—The number of Internet-connected devices exceeds the world's population by more than three times and this figure is expected to be doubled within the next five years. The Internet of Things is a concept that describes this trend and outlines certain aspects of design and functionality that new devices should incorporate for a successful integration into the Internet. In this respect, Digital Signage networks traditionally used for audiovisual media, accomplish many of the characteristics of the Internet of Things devices: interoperability, mobility, scalability and ubiquity, both in terms of access and control of devices and regarding the information they generate. This paper raises the power to employ a proposed Digital Signage network as a substrate to connect other types of devices that can benefit from the advantages of this kind of networks. For that aim, the main problems for this integration are discussed, mainly those related to the bidirectional tunneling scheme used in the proposed Digital Signage solution. The effects of this tunneling approach are analyzed in scenarios with bandwidth constraints, and different solutions are proposed. Tunneling performance in mobility is improved, to increase the amount of Internet of Things devices and applications that can benefit from this type of network.**

Keywords— Digital Signage, Internet of Things, port forwarding, network mobility, OpenSSH tunneling.

I. INTRODUCTION

The Internet of Things (IoT) is changing the traditional concept of the Net. In this new scenario universal connectivity arises, as the interconnection of networks with different devices and services becomes possible. This concept also comprises a change in the paradigm supporting the structure of the Internet. The IoT is composed by different networks that aim to provide communication systems, telephony, network security, control and operation of connected devices around the world, allowing the ability to incorporate, review and distribute information and knowledge across the network [1] [2].

By 2020 the number of devices connected to the network for each individual is expected to be 6.6 (Table I) [3]. This fact encourages the creation of applications with high potential to analyze data and process information, in order to identify trends and patterns from the data provided by ubiquitous sensors and control systems [4].

This work has been partly financed by CONACYT (PEI 682/2014); Servicios d TI de Durango S.A. de C.V.; Ateire S.A.C., and de ER H2020 Wi 5 project (Grant Agreement no: 644262).

TABLE I. CONNECTED DEVICES VS. WORLD POPULATION [3].

	2010	2015	2020
Connected devices[a]	12.5	25	50
Connected devices for each people	1.84	3.47	6.58
Wolrd population[a]	6.8	7.2	7.6

[a.] Billions

The increase of the number of connected devices influences the authorship of the information available online. According to recent studies [3] [5], a significant part of the information published on the Internet is automatically generated by connected devices (Fig. 1). Therefore, the nature of the information and its increasing amount makes it necessary to develop technology able to facilitate the processing of this information through systems that enable interoperability between devices [6]. Constant network deployment adds value to commodity, making it feasible the new conception of extending Internet to Everything [7].

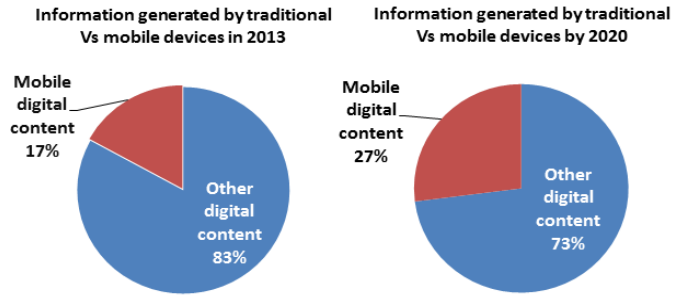

Fig. 1 Scheme Information generated by mobile devices [4].

Digital Signage (DS) is a technology focused on selective broadcasting of audiovisual media such as video, animation, sounds, images and interactive applications [8]. This broadcasting is usually made on screens distributed on wide areas on which contents are segmented according to its location. In parallel, DS networks [9] can be defined as those required to connect these audiovisual devices with the servers providing the contents to be displayed. DS requires a logical network allowing flexibility, reliability and control regardless of the geographic location of each device. The main purpose of

such networks is to communicate multimedia content, in some cases with interactive capabilities through recent digital technology devices.

The main structure of a DS system is composed of two elements: (1) the *players*, i.e. the devices that reproduce the content and (2) a *central server* or *core network*, as a DS cloud, which provides access to different services, control devices and content distribution.

When a DS system is deployed, a number of connected devices are spread in a geographical zone. Thus, if they are equipped with specific communication interfaces, the DS technology can be raised to interconnect other types of devices in a transparent way, providing scalability and mobility just as the IoT paradigm states. The DS element would then act as a relay for connecting other wireless devices to the network, sharing its backhaul link with them. So a third element should be added to the scenario, namely the IoT devices. We will call them *"alien devices"* in this document (Fig. 2). Obviously, security and privacy issues may arise if this scheme is followed, taking into account that the *player* will act as a relay for the information of other devices.

It can be said that nowadays DS technology lacks standardization, and a clear description of its functionality does not exist so far. In this paper a specific solution is presented, based on open technology applied to both the central server and the players composing the DS network. This design has been elaborated making an efficient use of the concept of IoT, to generalize interactivity and accessibility through the DS network. To achieve this goal, every player is controlled via a web interface, real-time control is permitted through servers and a bidirectional tunneling connection scheme is used between servers and players. In this paper the structure of a DS network is presented, which has been deployed in real scenarios. Different use cases including a number of wireless devices are also considered. In addition, different mechanisms for optimizing its communication system, based on bidirectional tunneling, are studied.

The remainder of the paper is organized as follows: In Section II (Related Work) common IoT features and DS classic systems are pooled. In Section III a proposed architecture for DS is explained. The section is focused on introducing all DS elements. Its communication scheme is also presented to justify its use with IoT devices. Also two ongoing projects are analyzed. Section IV describes some limitations of the tunneling described communication approach, and how to overcome them. Two test beds are proposed to measure and optimize communications performance and Section V ends this paper with the conclusions.

II. Related work

In [10], a decentralized architecture is proposed for a DS network integrating Radio Frequency IDentification (RFID) systems. Its implementation by some companies in the sector is described. The proposed design includes elements of great relevance when considering a more general use of the network following the IoT paradigm. Security in communications is implemented and its architecture allows a flexible network deployment. The presented architecture is based on the decentralization of services, applications and network functions. The network included in this scheme allows the integration of various elements related to RFID technology and also allows sending visual messages to users through DS devices.

In the present article, although this type of network is not intended to be used in a more widespread sense, the importance of decentralizing processing services, running applications and communication control is stressed, as this increases scalability and resilience to failure. These ideas are kept in proposals for generic decentralized networks that are postulated as suitable platforms for a diverse number of services and applications for Internet of Things [11].

Given these ideas, the DS system presented incorporates certain aspects of decentralized networks in order to develop a layered platform, flexible enough to adapt to any communication need or device requirements in terms of software or hardware. This network is intended to allow traditional DS players with other devices without requiring substantial hardware or software changes. Thereby, this DS network becomes valid for supporting applications and services for the IoT paradigm, in which a traditional DS player becomes a possible relay for other devices, applications or services. Two examples of ongoing projects are explained below.

All in all, the contribution of this paper is twofold: (1) an architecture for a DS network is presented based on open technology and mature protocols: Transmission Control Protocol (TCP) over Internet Protocol (IP), OpenSSH and Hyper Text Transfer Protocol (HTTP); (2) the performance of bidirectional secure tunnels sharing a connection is studied. Both questions have been implemented and studied in real scenarios.

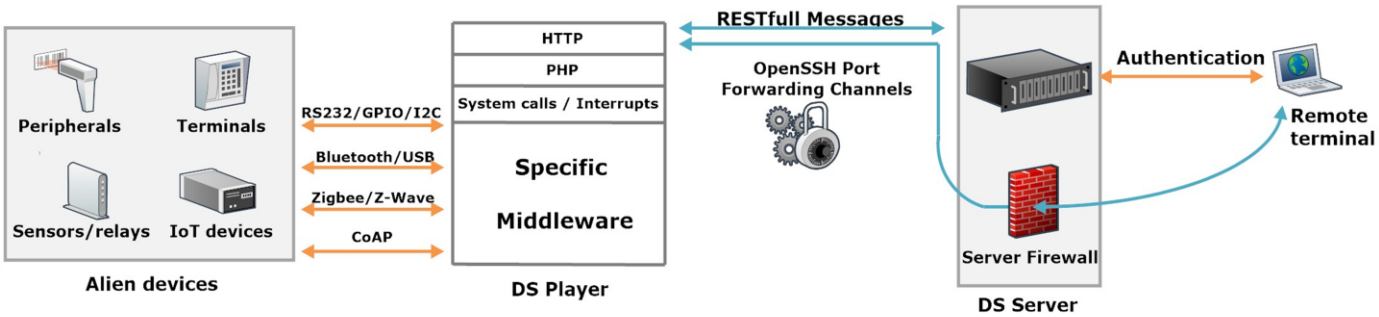

Fig. 2 Integration scheme for *alien* devices into DS network throught DS players.

III. Proposed architecture for Digital Signage

In the proposed design, several factors are considered: (1) The underlying Operating System (OS) of the players and the servers; (2) the web technology used to display and manage the content; (3) the structure of the content as web apps; and (4) the security in network communication and the flexibility in deploying this network in a way that it does not require complex configurations, allowing an overall management of device groups and content channels.

In DS, the reliability of the whole system is very important to reduce as much as possible the human interaction required for maintenance. This is an important feature in most DS devices whose required operating service is 24/7. For that reason it is necessary to address all the elements of the system as a whole. Although policy usually leads to closed solutions rather monolithic for certain scenarios, in our case, the design of the whole system is carried out in layers following several recommendations stated for "fog computing" [11]. This scheme can provide network services to other applications beyond web content prepared expressly for DS, thus promoting the interconnection and communication with other various devices. The DS network provides basic services, all of them incorporating encryption, and transparent management on communication establishment and monitoring:

- A bidirectional transmission channel for high and low priority content.
- A bidirectional transmission channel to manage content applications or configure the player, generally web based.
- A bidirectional transmission channel, with high priority, for internal network signaling. It is also used for notification of high priority events and alarms.
- Communication between devices on the same DS network. If necessary, the DS cloud can set authorized tunneled connections when requested by any application between two or more devices. These authorizations are based on policy rules following ownerships and granted rights from networking administrators.

A. Network scheme

The DS network previously presented has a logical hybrid connection scheme (Fig. 3). It mainly follows a logical star architecture, where each device is connected to the centralized cloud services through the Internet. However, in certain cases, DS devices can communicate with each other without using these centralized cloud services. This usually occurs when the devices are in the same local network and the distribution of the same contents to every device is required. However, the DS network can also provide tunneled communication with other devices if needed (Fig. 4). The communications between the players through the cloud also follow a bidirectional tunneling scheme.

This approach shares some similarities with some schemes outlined in IP Mobility protocols [12] [13], allowing real-time access to devices from any terminal connected to Internet.

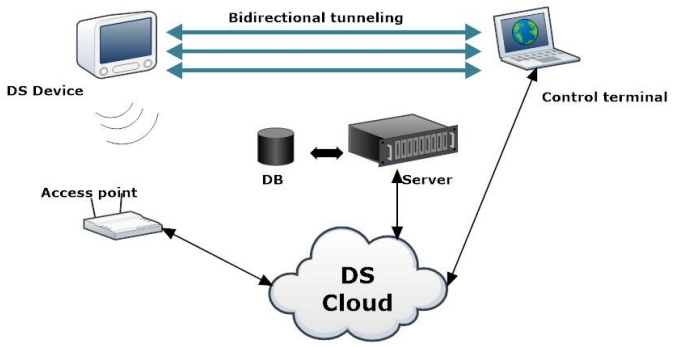

Fig. 3 Tunneled communications in DS network.

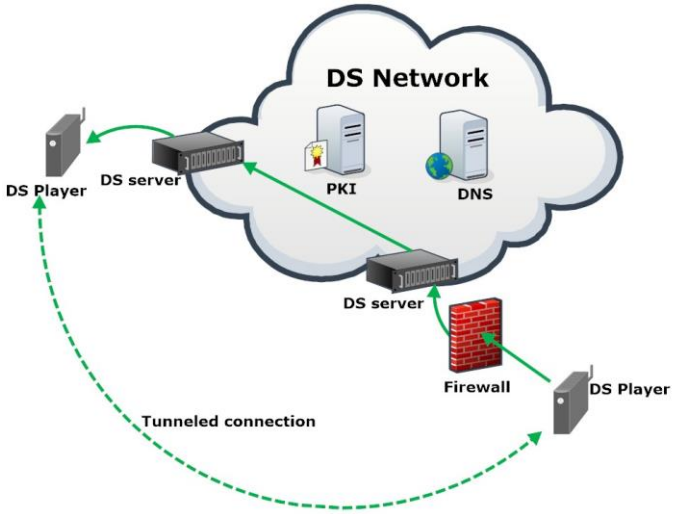

Fig. 4 Tunneled connection between players on different networks.

B. Communications security

This aspect of the DS network is addressed by introducing tunneled communications encryption via SSL (Secure Socket Layer). Encryption is performed using pairs of public-private RSA keys of 2048 bits (RSA-2048) that are assigned at the factory. The device registration and allocation to end users in the cloud is performed at the time of purchase or renting. This type of security helps to protect the integrity of the data and to certify the source, thereof avoiding potential phishing attacks when updating contents [14]. In addition, this communication permits the tunneling of different traffic between a local process and a remote service. This is set thanks to the Secure SHell (SSH) port forwarding feature. Securing communications following this scheme does not require significant modifications on existing software and services.

C. Quality of Service (QoS)

Communications can be classified according to their level of priority into three groups: control, management and content distribution. In DS networks, most QoS degradation scenarios are often linked to a high degree of congestion in the local

Internet access nodes. For that reason, a QoS system based on 802.11e devices [15] [16] [17] and including standard queuing disciplines when available [18] has been implemented in our DS system, thus improving Internet access of high priority communication.

D. Elements of the System

1) Screens Network for public transport in Mexico City

In Mexico City, we are collaborating with a local content provider [1] to equip buses with DS players using 3.5G+ connectivity. These devices also include a Global Positioning System (GPS) for geo positioning the bus with which to offer other future services under development. These players are in a mobility environment with moderate bandwidth and connectivity constraints, which our system has to overcome.

The network will consist of 300 units (screens in buses, see Fig. 5 and 6) operating at the same time. The geo position of all buses is centralized through the DS network and this allows us to develop tailored applications and services for travelers (better controlling the bus service times). At the same time, advertisers can restrict the display spots in certain geographical areas of interest. The system is however partly decentralized, as players can communicate each other directly when downloading information at bus stations. This feature reduces downloading times as information is downloaded from the cloud only once, and then transmitted to other players. These devices can also execute contents and applications locally, thus avoiding occasional connectivity loss to be noticed by passengers.

2) Network sensors for air quality

AirPi platform [19] is a clear example of an *alien* device to be incorporated into the DS network already described. This element has several environmental sensors and functions that can be added as expansion shields for RaspberryPi platform [20], a System on Chip (SoC) device with a Reduced Instruction Set Computing (RISC) processor based on Acorn RISC Machine (ARM) technology. Its integration into our DS network was simple and allowed the real-time monitoring of the status of all its sensors and devices. A DS player together with an AirPi (Fig. 7) can provide audiovisual information and weather statistics alongside helps government institutions to measure air quality parameters on different parts of the city.

Fig. 5 Public transport bus in Mexico City.

[1] Tele Urban, http://www.teleurban.tv/

Fig. 6 Indoor layout of the screens on buses.

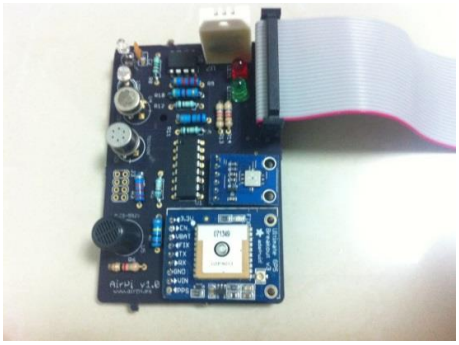

Fig. 7 AirPI device for measuring environmental parameters.

Fig. 8 Totem STI installed in Durango Fair 2014.

This device has been successfully installed inside audiovisual outdoor cabinets called "*totems*" (Fig. 8) and operates in parallel to their DS player which manages the digital screen inside the totem. Thanks to DS flexibility, it can be controlled just as another DS device in the network.

In this scenario, AirPI can get its running applications updated through a DS content channel. These updates can be done specifically to one device, some of them or all, exploiting content channel services in DS network. The environmental parameters measured by this device can also be monitored in

real time and use the network event log to also review available historical data and statistics. Finally, an AirPi can interact with a Digital Signage Player directly to report data required for displaying it on screen, or any content in the player can query information directly to AirPi.

IV. TESTS AND RESULTS

As shown in Fig. 3, the proposed architecture considers a number of bidirectional SSH tunnels between the DS device and the control terminal. It has been observed that this system presents problems when the information is transmitted through port forwarding. In this scenario, SSH works as an extendable proxy: half of the proxy is local and the other half is on a remote machine. Both halves communicate with each other through a *forwarder-tcpip* channel [21]. However, all the recent OpenSSH implementations include an incoming buffer with a fixed size of 2 Mbytes [22], set on the local *forwarder-tcpip* channel side. As a result, when there is an intensive throughput application sharing a session with other flows, the overall latency increases because the size of that incoming buffer degrades the overall performance.

These problems are well-known in the literature [23], and optimizations of OpenSSH implementation have been proposed, but mainly intended to improve throughput performance, not latency. To overcome this problem, we propose not to multiplex tunneled connections through the same SSH session. Instead, the connections of each player with the server should be generated in different SSH sessions. This allows each SSH session to have its own incoming buffer, thus preventing high latency values to propagate from one flow to another. It also allows SSH to apply DiffServ using the type of service (ToS) field of the packets from each session, allowing Linux default queuing discipline (*pfifo_fast*) to coordinate datagrams sending, based on their priority [18]. This approach does not require kernel modifications on most devices and facilitates future implementation on embedded android systems.

In order to study and optimize this approach, we next separately analyze two aspects: performance evaluation of the main TCP variants' congestion control, and latency degradation at application layer. According to the results, the validation of this communication scheme will be discussed.

A. Performance of TCP congestion control variants

To study the impact of buffering in port-forwarding connections using OpenSSH 6.6, a series of measurements have been made in communications between DS players and servers. The proposed testbed is as follows: four players are connected to the Internet with a 3.5G+ Huawei E173 modem each. This device allows High Speed Downlink Packet Access (HSDPA) 7.2 Mbps for the downlink channel and High Speed Uplink Packet Access (HSUPA) 2.1 Mbps for the uplink channel. All the players are running OpenSSH 6.6 in a Linux OS based on kernel 3.10.48. Each player runs a different TCP congestion control algorithm to test the upload channel. Connections with the server are made synchronously to prevent random effects to affect only one single connection. 3G signal level is -75 dBm in all devices, and all measurements were taken with the vehicle at rest.

Each tests series consists on a set of 15 transmissions run in different hours of the day. On each test two tunnels are established through independent SSH port-forwarded connections from all players to a private testing server at Montreal, Canada, in the same datacenter where our DS cloud is hosted. This private server is linked to the cloud but during experiments its services were set offline. On the first connection, Iperf tool [24] is used to generate traffic for 120s and to measure statistics, and NetEm [25] is configured on the Point to Point Protocol (PPP) network interface of each player to model bursty packet losses. A 2% and 5% packet loss scenarios are modeled in order to study the worst cases on High-Speed Packet Access (HSPA) while the terminal is moving [26]. On the second tunneled connection of each player, a Python script is used to sample Round Trip delay Times (RTT) using echo server port through the second port-forwarded channel for interactive connections. The scheme is shown in Fig. 9. The tested TCP variants are Reno, Bic, Cubic and Westwood.

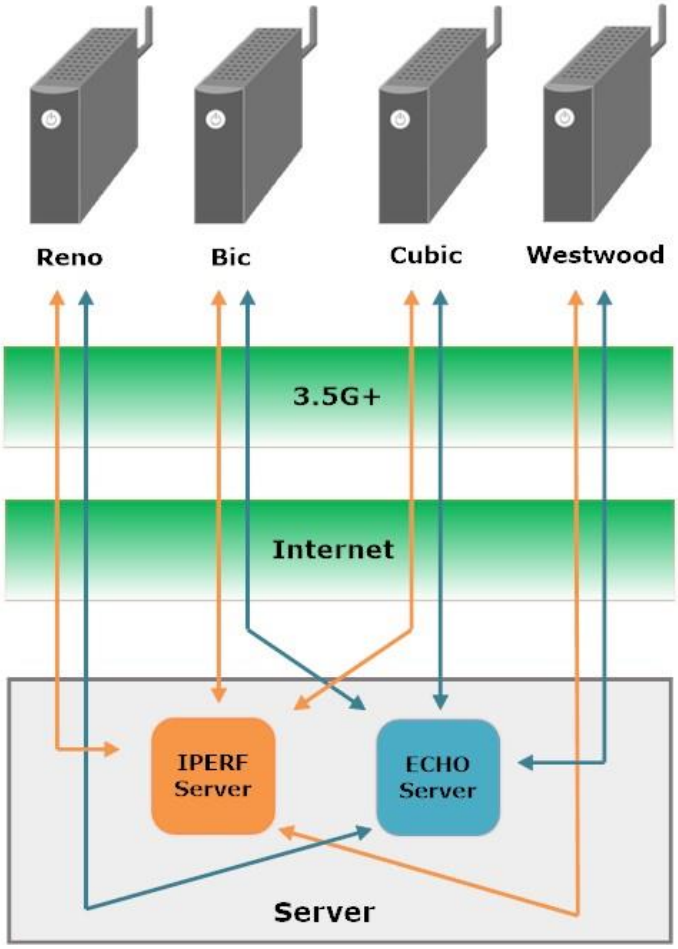

Fig. 9 Experimental testbed to study congestion control aglorythms through independent port-forwarded OpenSSH channels.

The results presented in Fig. 10 and Fig. 11 represent the Cumulative Distribution Function (CDF) of Round Trip delay Time (RTT) values which indicates that the policies implemented to prevent high latencies at the interactive tunnel are working correctly. In all the tests performed, 85% echo packets are under 350 ms latency while Iperf is transmitting. In

these circumstances, if an interactive application with low bandwidth requirements needs to transmit, its RTT will not increase due to parallel data flows, allowing the interactive application to work correctly.

TABLE II. THROUGHPUT MEASURED AT PPP INTERFACE.

	Westwood	BIC	Cubic	Reno
2% packet loss	397 Kbps	340 Kbps	422 Kbps	355 Kbps
5% packet loss	193 Kbps	222 Kbps	240 Kbps	210 Kbps

The bandwidth usage is summarized in Table II, where Cubic obtains the best performance mark. This result is similar to the one obtained in experiments run on non-tunneled connections [27].

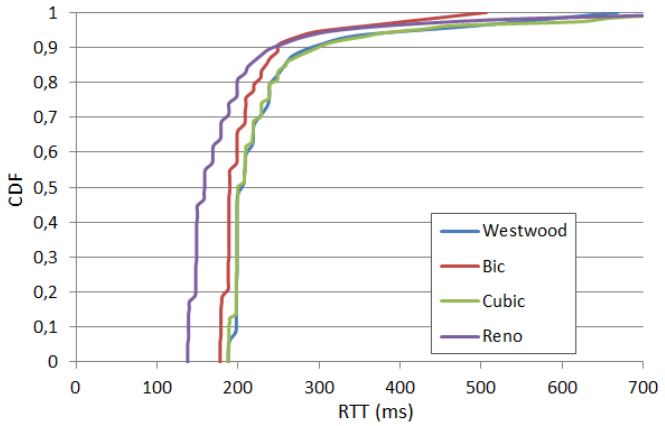

Fig. 10 Comparison of the latency with 2% packet loss.

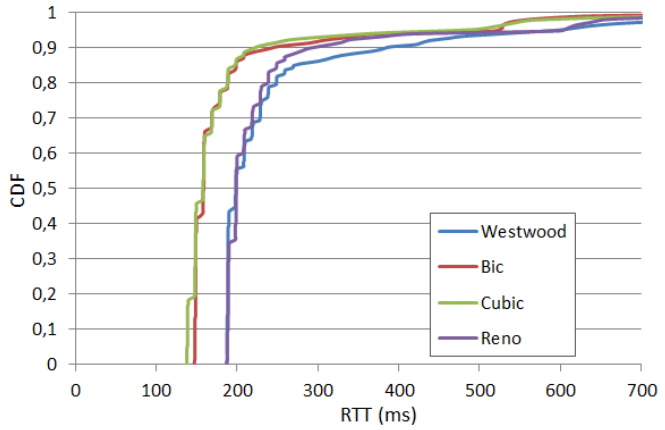

Fig. 11 Comparison of the latency with 5% packet loss.

In Fig. 11, a slightly poor Westwood performance is detected. This may be caused by the interaction of OpenSSH with RTT traffic flow values: Westwood congestion window depends on RTT traffic flow values and these figures are altered in port-forwarding connections due to the proxy behavior of OpenSSH. This is clearly shown only on Fig. 10, but this slight degradation is always present.

B. *Latency degradation at application layer*

It is necessary to analyze the impact of tunneling at application level and to measure the RTT performance penalty that port-forwarded communications may imply. To measure this degradation, a test in a scenario including mobility (Fig. 12) is performed as follows: One player is set in a car and it is connected to the Internet using the same 3.5G+ modem introduced before. The car follows a 20 min. bus city trip at Durango, Mexico (Fig. 13). There is an Internet Group Management Protocol (IGMP) ping sample each second besides the two port forwarded connections. The RTT measured through echo pings sent by the Python script through the tunneled connection is compared to the RTT presented by plain IGMP pings. This information helps to measure RTT penalties regarding to port forwarding connection. All these measures are performed in presence of background traffic generated through Iperf to measure bandwidth over time. This scheme is presented is Fig. 12.

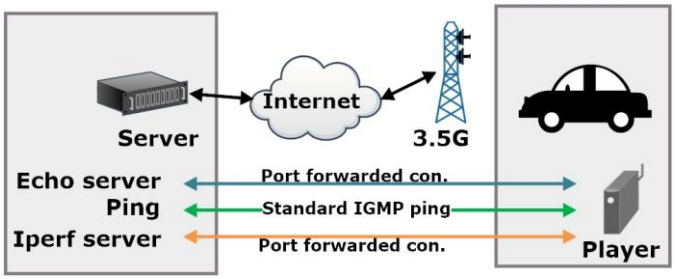

Fig. 12 Experimental motion testbed to measure latency degradation in port forwarded communications in presence of background traffic (Iperf).

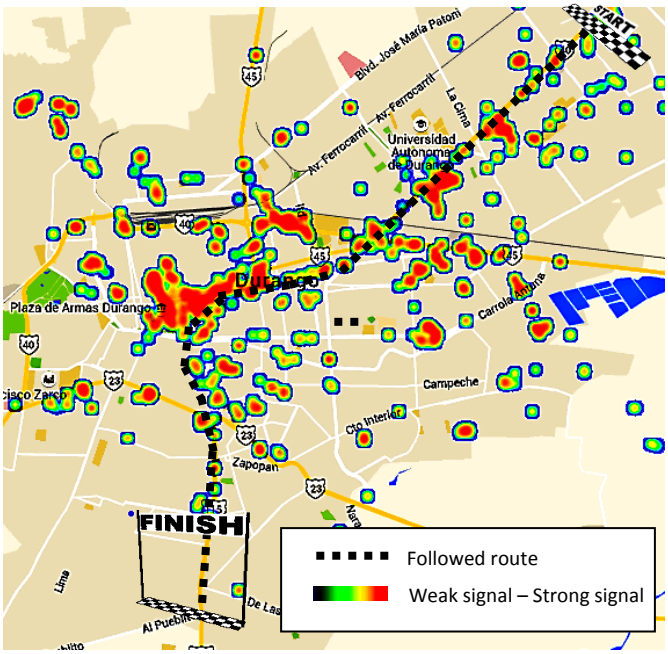

Fig. 13 City trip followed in Durango superimposed on 3G Movistar signal level provided by OpenSignal[2]

[2] OpenSignal: http://opensignal.com/

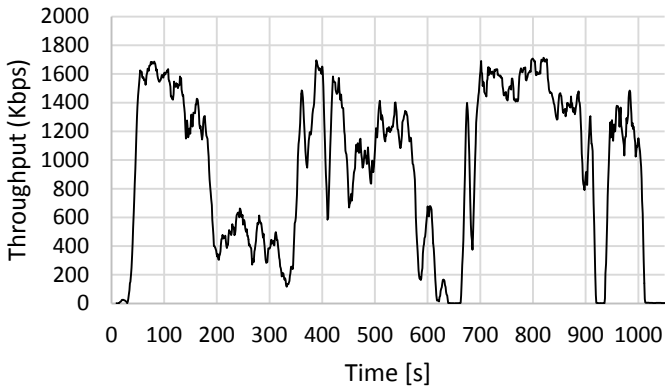

Fig. 14 Upload throughput measured at field test on a car.

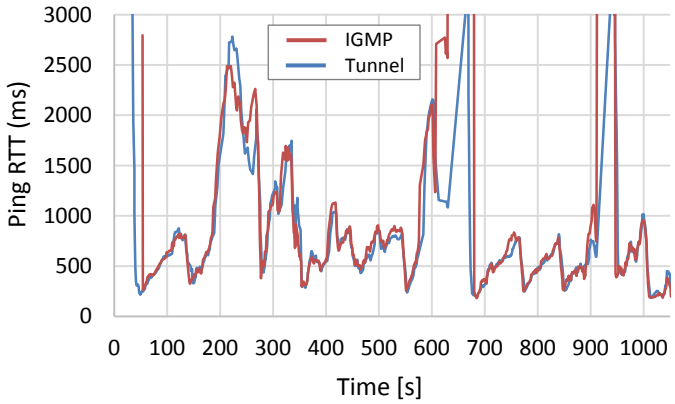

Fig. 15 Comparison between Latency on second tunnel using echo messages and IGMP when Iperf is transmitting. Values above 3 seconds have been cut, since we consider this value as the threshold of a disconnection.

TABLE III. PORT-FORWARDED AND IGMP COMMUNICATION RESULTS

	IGMP	**Port Forwarded echo**
Conectivity	90%	89%
RTT average	622 ms	632 ms

According to the results shown on Fig. 14, Fig. 15 and summarized on Table III, the degradation of the tunneled communications is less than a 1.7% in terms of RTT, barely degrading connectivity.

C. Validation of DS communication scheme

Communications do not seem to be affected when using tunneled connections through OpenSSH port-forwarding as detailed above. The average latency in presence of background traffic sharing PPP connection is high despite using DiffServ due to the mobility and 3G coverage fluctuations. However, this approach may still be valid for communications between IoT devices. For example, this communication may employ protocols conforming to the Representational State Transfer (REST) architecture constraints [28], as the Constrained Application Protocol (CoAP) or HTTP, which does not require low RTT values to work. The use of RESTful protocols, particularly HTTP, also allows easy interoperability with external information systems as ThingSpeak [29] through HTTP methods GET, POST, PUT and DELETE.

V. CONCLUSIONS

This paper has presented a DS network able to distribute, collect information and control devices in real time. They are being deployed in public screens traveling in buses, and employed as relays for connecting IoT devices. The connection problem of these devices in mobility has been addressed.

After analyzing the causes of OpenSSH port-forwarding limitations it is concluded that bidirectional tunneling scheme based on port-forwarding lacks dynamic incoming buffers, generating undesirable effects when sharing SSH session with intensive throughput applications. However, the solution of this problem is feasible without modifying OpenSSH core implementation, by using different SSH sessions for each flow and applying Diffserv.

Results from tests performed conclude that this scheme works in environments with packet loss and mobility without degrading latency. As future work, we want to test that all security features provided by open SSH as a process-to-process protocol have barely noticeable performance degradation at application level. Although this approach cannot be deployed in a widespread sense, this communication scheme has immediate application on many existing devices on the market with enough hardware features, allowing fast development of new IoT devices and services.

REFERENCES

[1] B. Edson, "Creating the Internet of Your Things," Microsoft Corporation, 2014.

[2] CISCO, "The Internet of Things Reference Model," 15 November 2014. [Online]. Available: http://cdn.iotwf.com/resources/71/IoT_ Reference _Model _White _Paper_June_4_2014.pdf.

[3] C. Visual Networking Index, The Zettabyte Era: Trends and Analysis, San Jose, California, 2014.

[4] D. Evans, "The Internet of Things. How the Next Evolution of the Internet is Changing Everything," Cisco Internet Business Solutions Group (IBSG), 2011.

[5] V. Turner and J. F. Gantz, "The Digital Universe of Opportunities," in Rich Data and the Increasing Value of the Internet of Things, Framingham, MA, 2014.

[6] International Telecommunication Union, "ITU Internet Reports 2005: The Internet of Things," December 2006. [Online]. Available: http://www.itu.int/osg/spu/publications/internetofthings/.

[7] A. J. Jara, L. Ladid and A. Skarmeta, "The Internet of Everything through IPv6: An Analysis of Challenges, Solutions and Opportunities," Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, vol. 4, nº 3, pp. 97-118, September 2013.

[8] R. Want, B. N. Schilit, "Interactive Digital Signage," Computer, vol.45, no. 5, pp. 21-24, May 2012.

[9] D. Franck and A. Martin, Digital signage: the right information in all the right places, ITU-T Technology Watch Report, 2011.

[10] D. B. Kotak and W. A. Gruver, Distributed Intelligent RFID Systems, San Antonio, Texas: IEEE International Conference on Systems, Man, and Cybernetics, 2009.

[11] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, Fog Computing and Its Role in the Internet of Things, San Jose, California: Cisco Systems Inc., 2012.

[12] J. Jaehoon, P. Jungsoo and K. Hyoungjun, "Dynamic Tunnel Management Protocol," IEEE Xplore, vol. 7, pp. 4754 - 4757, 2004.

[13] W. Xiaoming, "A framework of enhanced local mobility routing," IEEE Xplore, vol. 3, pp. 2030 - 2034, 2003.

[14] E. A. Cooper, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, 2008.
[15] Heinanen and et al, "IETF Network Working Group, RFC2597: Assured Forwarding PHB Group," June 1999. [Online]. Available: http://tools.ietf.org/html/rfc2597.
[16] B. Davie and et al, "IETF Network Working Group, RFC3246: An Expedited Forwarding PHB," March 2002. [Online]. Available: http://tools.ietf.org/html/rfc3246.
[17] Wi-Fi Alliance, "Wi-Fi CERTIFIED™ for WMM™ - Support for Multimedia Applications with Quality of Service in Wi-Fi® Networks," 2 September 2004. [Online]. Available: http://www.wi-fi.org/wi-fi-in-your-life.
[18] Graf, Thomas, et al, "Simple, classless Queueing Disciplines," [Online]. Available: http://lartc.org/howto/lartc.qdisc.classless.html.
[19] A. Dayan and T. Hartley , "AirPi," 4 april 2013. [Online]. Available: https://airpies.wordpress.com/.
[20] Raspberry Pi org, "What is a Raspberry PI," Cambridge, UK, 2014.
[21] T. Ylonen and C. Lonvick, "The Secure Shell (SSH) Connection Protocol," 2006 [Online]. Available: https://tools.ietf.org/html/rfc4254.
[22] OpenSSH 6.9 Source Code (channels.h), 2015.
[23] Rapier, M. Stevens y B. Bennet, "High Performance SSH/SCP -HPN-SSH," Pittsburgh Supercomputing Center, 2012. [Online]. Available: http://www.psc.edu/index.php/hpn-ssh.
[24] M. Gates and et al, "Iperf," [Online]. Available: https://iperf.fr/. [Accessed 21 noviembre 2014].
[25] Linux Foundation, "NetEm: Network Emulation," 19 November 2009. [Online]. Available: http://www.linuxfoundation.org/collaborate/workgroups/networking/netem.
[26] J.A. Esquerra-Soto, J.A. Pérez-Díaz, I. Amezcua-Valdovinos and C.F. García-Hernández "Performance Analysis of 3G+ Cellular Technologies with Mobile Clients," Instituto Tecnológico de Monterrey, 2012.
[27] S. Mascolo and L. De Cicco, "TCP Congestion Control over HSDPA: an Experimental Evaluation," arXiv, December 2012.
[28] A. Aijaz and A. Hamid Aghvami, F, "Cognitive Machine-to-Machine Communications," IEEE Internet of Things Journal, vol. 2, nº 2, April 2015.
[29] ThingSpeak Community "ThingHTTP", [Online]. Available: http://community.thingspeak.com/documentation/apps/thinghttp/.