Context-aware IPv6 Flow Mobility for Multi-Sensor based Mobile Patient Monitoring and Tele-consultation

Norbert Varga\textsuperscript{a}, László Bokor\textsuperscript{a,}\textsuperscript{*}, András Takács\textsuperscript{b}

\textsuperscript{a}Budapest University of Technology and Economics, Department of Networked Systems and Services (HIT), Multimedia Networks and Services Laboratory (MediaNets), Magyar Tudósok krt. 2, H-1117, Budapest Hungary

\textsuperscript{b}Hungarian Academy of Sciences, Computer and Automation Research Institute, Kende u. 13-17, H-1111, Budapest, Hungary

Abstract

Recently, the rapid growth of multi-access mobile devices interconnected with different wearable biosignal sensors via Body Area Network (BAN) play an increasingly important role in healthcare systems and provide significant solutions for scenarios of home healthcare, real-time remote/mobile patient monitoring and ubiquitous tele-consultation. The proliferation of different radio access technologies make possible mobile healthcare (mHealth) services to actively benefit from the advances of heterogeneous and overlapping wireless networks. The varying characteristics of mHealth applications (particularly different real-time monitoring solutions) in means of the required network resources and Quality of Services parameters invoke elaboration of effective flow-based mobility management algorithms. Aiming to provide a transparent and efficient mobility infrastructure for the mHealth field we designed and developed a context-aware IPv6 flow mobility management scheme for Android focusing on mobile patient monitoring and tele-consultation use-cases in a multi-sensor environment. The proposed solution incorporates a cross-layer optimization platform and relies on MIP6D-NG, which is a client-based, multi-access, flow-aware Mobile IPv6 implementation. We have implemented a real-life multi-sensor testbed architecture where the proposed system with its adaptive flow handover scheme was evaluated.

© 2014 The Authors. Published by Elsevier B.V.

1. Introduction

Recently, the rapid growth of multi-access mobile devices interconnected with different wearable vital sensors via Body Area Network (BAN, also referred to as a Body Sensor Network or BSN) play an increasingly important role in healthcare and provide significant solutions for home healthcare, remote patient monitoring and real-time tele-consultation [1]. The spreading of heterogeneous and overlapping wireless access technologies make possible to mobile healthcare applications to use the available network resources efficiently and call into being the ubiquitous Internet connection for these applications. These facts motivated us to design and implement a context-aware IPv6 flow mobility management scheme for multi-sensor based mobile patient monitoring and tele-consultation. We designed and implemented a cross-layer optimization platform and a flow-aware, client-based mobility management mechanism for Android-based systems interconnected with different medical body sensors and Smartphone CCD video cameras. However this paper refers to our previous work [2], currently we focus on context-aware flow mobility mechanisms for multi-sensor mHealth services and the proposed decision engine together with the evaluation and measurement scenarios have been put into the context of ubiquitous tele-consultation applications with real-time patient monitoring support. We have also performed several enhancements and modifications in the core devices of the evaluation testbed. The latter fact also required the porting of the proposed architecture to newer Android and kernel version.

The rest of paper is organized as follows. In Section 2 we present the related work on the existing solutions for context-aware mobile patient monitoring and tele-consultation. Section 3 introduces our cross-layer optimized mobile patient monitoring and tele-consultant framework for Android devices in details. Also the overall client-based, flow mobility management scheme is depicted. Section 4 introduces our testbed setup, measurement scenarios and performance evaluation results. In section 5 we conclude the paper and describe our future work.
2. Background and Related Work

Nowadays, mobile health has been receiving more and more attention [3] thanks to the spectacularly increasing number of high-performance, multi-access mobile devices and wireless access technologies. Mobile health is a sub-segment of electronic health (eHealth), which aims to use mobile devices and communication technologies to enhance traditional medical services. The mHealth paradigm includes real-time patient monitoring, diagnostic and treatment support, disease and epidemic outbreak tracking, fitness and training applications, tele-consultation and awareness services as well. In this paper we focus on multi-sensor based patient monitoring and ubiquitous tele-consultation.

Mobile patient monitoring is a special type of mHealth services, which is used to measure, collect and transfer biosignal data provided by wearable sensors from the patients to the hospital or other healthcare center [4]. Authors of [5] categorize the remote patient monitoring systems, try to identify the most important mHealth stakeholders and their requirements for a generic system architecture. In [6] comprehensive report is presented about the evolution of mobile healthcare. The summary includes extensive desk research and 20 in-depth interviews from healthcare providers and payers, technology and telecommunications companies and industry organizations.

Ubiquitous tele-consultation aims to create an environment for preliminary consultation between patients and nurses/doctors associated with possibility of sending medical data of patients, such as ECG and other medical examinations results, CT scans, MRI images and ultrasound using heterogeneous wireless networks [7]. The benefits of tele-consultation services reduce overhead costs of consultation, while also make appointments quicker and make doctors to be able to conveniently access patients who may be at far distance from the hospital.

The efficiency of any mHealth service can be increased by using the network resources of all the available wireless networks in a heterogeneous multi-access environment. Benefits of such systems can only be exploited if mobility between the different radio access networks is efficiently handled. Efficient mobility management can be achieved by assigning application flows to the appropriate interfaces using intelligent decisions and adaptivity based on the available network resources. Vertical handover and network flow mobility algorithms are the basics of an optimal and cross-layer driven mobility management method for future heterogeneous mobile networks. In this paper we focus on vertical handover solutions which incorporate with flow-mobility management. Most of the papers in the subject discuss the definition and management of different flows in protocol level [8].

In our architecture the MIPD6-NG [9] solves all the protocol level questions by relying Mobile IPv6 extensions, namely Flow Binding [10] and MCoA [11], thus we do not detail in this paper. In [12], Haw et al. examine a multi-criteria VHO decision mechanism to manage the network flows efficiently. In their flow mobility scenario two flows were defined (FTP and VoIP), however the flow mobility was managed by the operator side and in the context of the mostly theoretical content centric networking (CCN). Contrarily we designed and implemented an IPv6 client-based mobility management which provides more freedom to the end users and relies on the practical IPv6 networking schemes. The first publicly available Flow Bindings implementation was designed for Linux distributions by the authors of [13], however their implementation supported only NEMO environments, regular mobile nodes were not able to register or update network flows. Francois Hoguet et al. [14] showed a Linux based flow mobility environment and the possibilities of porting it to Android Smartphones. They used the UMIP’s MIPv6 implementation and a proprietary flow binding solution. Ricardo Silva et al. [15] examine the mobility management on Android systems. They created a custom Android ROM to use the 3G and Wi-Fi interfaces simultaneously. IEEE 802.21 Media Independent Handover framework [16] is applied to support IPv6 based mobility. From this article also the flow mobility and the flow based decision mechanism are missing compared to our architecture.

Several researches dealt with remote patient monitoring solutions based on modern Smartphone devices and wireless networking architectures. K. Wac et al. [17] proposed a tele-monitoring and tele-treatment system, which allows the incorporation of different medical sensors via wireless connection, and live transmission of the measured vital signs to healthcare providers. Also authors of [18] designed and developed two mobile software platform. The first one collects the patient’s ECG and blood pressure data and sends the measured value to the mobile phone of patient where the first level analysis is performed and to a central server as well. The second platform allows to view patient’s health reports on the mobile phones form a central database. In this project a HTC Nexus One was chosen for development. Dusit Niyato et al. [19] propose a patient monitoring service architecture using heterogeneous wireless access and remote monitoring devices attached to the patient. In [20] a model system for remote patient monitoring using Java-enabled mobile phones is presented. They developed an information service, which contains EEG data samples and the doctors can browse these information using their mobile phones.

Despite the active involvement of the research community in the development of advanced mobile patient monitoring and tele-consultation schemes, there is no available work in the literature, which would consider context-aware flow-level mobility management for optimization of such mHealth applications.

3.1. Multi-sensor BAN

In our framework the patient monitoring and tele-consultation strongly relies on the collection of biomedical data from two wearable sensors, namely a Samsung Gear Fit Smartwatch and a Zephyr HxM heart rate monitor, and in addition transfers high-resolution video stream using the Smartphone’s internal CCD video camera. In the following sub-sections we introduce the aforementioned sensors in details.

3.1.1. Samsung Gear Fit

The Samsung Gear Fit is a hybrid of a fitness tracker and a Smartwatch†. It has the ability to monitor and log the patient’s heart rate and daily exercise activities, like running, walking or cycling, and also it is able to receive notifications from its host Smartphone. Samsung provides a Mobile SDK‡, which allows developing applications for wearable devices such as the Gear Fit.

The SDK consists of 16 independent packages, including the Remote Sensor package. Using this package we are able to collect remote sensor data, such as user activity information, pedometer data and wearing state from the Gear Fit device, which is connected to a host Smartphone (in our case to a Samsung Note 3) via Bluetooth. Unfortunately, the SDK is currently in beta release state, thus it is not allowed to collect heart rate information with Remote Sensor package. In our framework Gear Fit provides the user activity status (running, walking or cycling) and wearing state information for the host Smartphone. Due to the Android permissions, the Gear Fit Manager application and the Remote Sensor Service are required on the host. The package contains five classes and interfaces, which allow to initialize our special sensor application (see later), get the list of available sensors and register/deregister events of the wearable device. This Remote Sensor package is supported on Android 4.3 or above.

3.1.2. Zephyr HxM

Zephyr HxM§ is a heart rate monitor and fitness accessory for Smartphones. Uses Bluetooth to transmit heart rate, speed, distance and RR interval (R wave to R wave interval, which is the inverse parameter of heart rate) towards Android devices. In our framework this device is applied as a continuous heart rate monitor, and – similarly to the data coming from the Gear Fit – the system forwards the measured values to the hospital or medical centre.

When using this sensor we have to pair the device with the host by relying on the Android SDK’s BluetoothDevice class. Also Zephyr provides an SDK**, which allows us to easily collect and process body vital data on the host Smartphone. The SDK provides the ZephyrProtocol and HRSpeedDistPacketInfo classes; using these we are able to communicate and receive the necessary information (heart rate data and timestamp) with the Zephyr HxM sensor node.

3.1.3. Camera video streaming

To provide high-quality video stream to the hospital/specialized medical center we apply the internal CCD camera of the Smartphone. The Libstreaming†† external android library is used to stream the camera and microphone data of the Android device with RTP protocol over UDP. The library supports H.264, H.263, AAC and AMR encoders (we have chosen H.264). On the client side (i.e., in the hospital) a Wowza Media Server ‡‡ plays the streamed camera video originated from the Smartphone. Also an HTTP server (Apache 2) was deployed at the hospital side to manage the playout of the stream. The library allows the phone to play the role of an RTSP server. In that case the smartphone is waiting for a RTSP client to request a stream. It is possible to use Libstreaming without RTSP, thus the session using only SDP over arbitrary transport protocol. Libstreaming requires Android 4.1 or above.

3.2. Network and Application Context Discovery

In the Java layer of our highly customized Android system we devised and implemented a modular Android application comprising three main parts. The so called Network and Application Context Discovery module is designed to measure the

---

† Samsung Gear Fit: http://www.samsung.com/global/microsite/gear/gearfit_features.html
‡ Samsung Mobile SDK: http://developer.samsung.com/develop
** Zephyr SDK: https://www.box.net/shared/c169gssedk2rugu441f
†† Android Libstreaming Library: https://github.com/fyhertz/libstreaming
different parameters from multiple layers of the available networks (e.g., ESSID, signal strength, delay and packet loss) and discover the required network resources and QoS parameters of the running applications. The packet loss and delay are calculated by pingm6, which is an IPv6-based test tool of MIP6D-NG, the signal strength (RSSI) is provided by Android’s WiFiManager API and TelephonyManager API. Using these modules we are able to discover any kind of access networks supported by the Smartphone (IEEE 802.11 a/b/g/n, 3G, 4G/LTE). We have created a pre-defined flow profile in the application layer, which contains the preferred interface and required QoS parameters of the different type of data flows. In particular, the CCD camera stream requires high bandwidth in this profile set, so Wi-Fi is the preferred interface, while for transmission of measured medical data bandwidth of 3G is enough, however this flow doesn’t not tolerate the packet loss and the also frequently handover is not suggested.

3.3. Flow Mobility Management

The architecture of our highly customized Android-based system is relies on MIP6D-NG, which realizes a flow-aware, cross-layer optimization ready, client-based IPv6 mobility management mechanism. To use MIP6D-NG two important usage requirements are required. First of all MIP6D-NG requires special kernel source, which contains Mobil IPv6 modules and MIP6D-NG patches. The other one is the simultaneous usage of network interfaces. For multi-access communications, the MN needs the ability to communicate via 3G and Wi-Fi at the same time with IPv6 support. Even the newest versions of Android OS (Android 4.4 at the time the preparation of this article) do not fulfill aforementioned requirements, thus we had to modify the source code of Android OS and kernel as well. Further details on our cross-layer optimized flow mobility solution can be found in Section 3.4 and in our previous work [2].

3.4. Overall Architecture

Fig. 3 presents our highly customized Android architecture for context-aware Mobile Patient Monitoring and Tele-consultation. The Smartphone must be able to run the MIP6D-NG daemon, therefore requires a special kernel environment. In order to achieve this, we modified the kernel part with the required extensions referred in Section 3. The porting of MIP6D-NG to Android systems was a non-trivial task, because it required libraries and header files that do not exist on Android OS or if exists, differ from their original GNU Linux implementations. To make up the missing requirements we created a cross-compiler toolchain which contains the ARM compatible versions of all the necessary components. We made the aforementioned compiler pack based on the NDK stand-alone toolchain and extended with our own libraries and header files. MIP6D-NG requires multi-access communications via two (3G and Wi-Fi) network interfaces (with IPv6 support) simultaneously. Despite the fact that recent Android devices usually possess multiple radio interfaces, the Android OS is currently pushing a solution which saves battery power so only one interface can be active at the same time. To change the default network interface management mechanism we modified the source code of Service module, namely the ConnectivityService class.

---

Figure 1 - Our highly customized Android architecture for context-aware Mobile Patient Monitoring and Tele-consultation

---

§§ Pingm6: http://www.mip6d-ng.net/documentation/pingm6/

Another issue to be solved was that the 3G interface doesn’t support native IPv6 on most Android devices. In order to solve this problem we configured an OpenVPN connection with a bridged interface on the Android Smartphone. In order to perform the required modifications inside the source code of the Android OS and the kernel, a build environment was created in which we were able to make a custom ROM image with our MIP6D-NG ready kernel source code and with our modified Android OS code. We used CyanogenMod Android sources and kernel tree distribution as a base code platform for our extensions. The result is a highly customized Android 4.4.2 and Kernel 3.4.89 with the appropriate patches and settings.

We flashed the aforementioned ROM to Samsung Note 3, thus our device is able to communicate via 3G and Wi-Fi simultaneously and run MIP6D-NG daemon. We designed and implemented a so called Mobile Patient Monitoring and Tele-consultation Application, which has four main parts. The Network and Application Context Discovery (NACD) module was already introduced in Section 3.2. The Wearable Sensor Data Aggregator collects data from Gear Fit and Zephyr sensors and sends the measured data towards a correspondent node. Gear Fit provides information about wearing state and user exercise activity, while Zephyr sends heart rate signal periodically. We defined two separated flows for these processes. Camera Sensor module streams the camera and microphone of Note 3 using Libstreaming library. A third flow belongs to this module, so in this scenario we have three separated flows. The fourth module of the application is the Handover Decision and Execution (HDE) module. HDE makes decision based on the information provided by NACD module. It is able to direct the Android OS to connect an available Wi-Fi network using WifiConfiguration and WifiManager APIs. The execution part communicates with the native MIP6D-NG, creates and sends flow register and flow update messages induced by the advanced decision algorithm. The first step of this decision algorithm of data flows to the default 3G interface using cross-layer communication between the application and the network layers and starts the phase of passive measurements of Wi-Fi networks. If there are no available networks, the algorithm holds the flows on the 3G interface and waits for the appearance of new Wi-Fi access points. Otherwise starts the cross-layer measurements, in which it measures the signal strength from link-layer, and packet loss provided by network layer. If the measured parameters are suitable for the QoS profile of the existing flows, the application sends flow update message, thus moves the flow from 3G to Wi-Fi interface. Cross-layer communication mechanism allows us to trigger and execute flow updates in a different layer of the stack which further increases the efficiency of our system. In each case when MIP6D-NG executes a flow registration or update, sends a Flow Binding Update (FBU) message to Home Agent, who sends back a Flow Binding Acknowledgement (FBA) message. More detailed description of our flow handover decision scheme can be found in [2].

4. Testbed and Measurements

4.1. Testbed Topology

Fig. 2 presents the overall testbed topology designed and implemented for evaluating our multi-sensor based mobile patient monitoring and tele-consultation system. The Home Agent is realized by a Dell Inspiron 7720 notebook running a MIP6D-NG daemon configured for Home Agent functionality.

A Samsung Galaxy Note 3 Smartphone plays the role of the MN. The core parameters of our Smartphone: 3G&LTE/Wi-Fi connectivity, FullHD display resolution, 3 GB RAM, Qualcomm Snapdragon 800 Quad-core 2.3 GHz processor, 13 MP primary and 2 MP secondary camera, BT4.0LE, GPS.
Medical data are provided by two different wearable devices: a Zephyr HxM and a Samsung Gear Fit Smartwatch. The Gear Fit communicates via Bluetooth 4.0 LE with host Android Smartphone. It has a 1.84” Curved sAMOLED touchscreen display and possesses different built-in sensors, such as accelerometer, gyroscope, pedometer, and heart rate monitor. Also Zephyr prefers Bluetooth communication, is able to measure heart rate, speed, and distance. Bluetooth low energy or Bluetooth LE (or BT Smart) is intended to provide reduced power consumption. BT LE is used in the healthcare, fitness, security, and home entertainment applications. It uses 2.4 GHz frequencies, and the maximum of data rate is 1 Mbit/s.

In our testbed the core router is an ASUS WL500 with DD-WRTv24 OS (CrushedHat distribution). Two OpenVPN daemons are running on this router. On one hand an OpenVPN Server provides an appropriate IPv6 address for the 3G connection of Android Smartphone using RADVD. On the other hand an OpenVPN client operates as an IPv6 over IPv4 or IPv6 over IPv6 tunnel, interconnecting the testbed with our IPv6 domain, independently of the router’s actual IP access. It means that the overall architecture could be portable and in the worst case only recovers legacy IPv4 connection for the core router.

4.2. Measurement Scenarios and Results

In order to present the efficiency of our mHealth framework and to evaluate the proposed context-aware, flow mobility architecture, we designed and implemented different measurement scenarios. We have put our focus on the performance characteristics of the flow vertical handover process of the proposed scheme. In the first scenario we measured the quality of the transmitted video from the mobile node towards the correspondent node using MSU Video Quality Measurement Tool††. We sent a YUV video file with 4:2:0 sampling rate and with QCIF (176x144) resolution.

Both of the sender and receiver side were developed in Java, using Datagram Socket implementation. We added a unique sequence number to each transmitted package to guarantee the in-order delivery of video frames on the receiver side.

The received and transmitted videos are compared based on three different video metrics: APSNR, VQM and MSE. PSNR metric is used often in practice, called “peak to peak signal to noise ratio” To take simple average of all the per frame PSNR values, we can use APSNR [21].

VQM measures the perceptual effects of video impairment including blurring, global noise, block and color distortion, and combines them into a single metric [21]. MSE is the mean squared error in relation to the maximum possible value of the luminance [21]. If the frame is flawless the value of APSNR is 100.0, the VQM and the MSE are 0.0.

Fig. 3 shows the three calculated video metrics as function of the number of handovers. In this test case we transmitted an 11 Mb size video, which took approximately 90 s. The average of total handover process from the sending of flow update message in Java layer to the receiving of Binding Acknowledgement message was 2.2s, however the native execution of the handover was only 0.083s.

In the second measurement scenario we defined two different flows: a UDP flow for video transmission and another UDP flow for heart rate sending. Fig. 4 shows the heart rate (in beats per minute (BPM)) and the APSNR metric of the video in two different scenarios. The graph on the left depicts a scenario, when flows of the video stream and of the heart rate are assigned to the 3G interface, because 3G is the only available network in this case. The bandwidth of 3G is not enough for both of flows, thus the quality of video falling off and also the heart rate is downgrading, until our framework detects a new Wi-Fi AP, and moves the flow of the video stream form 3G to Wi-Fi. The bandwidth of Wi-Fi can be inconvenient for the simultaneous transmission of data flows (e.g., high resolution adaptive video stream and heart rate transmission with fast interval), thus the usage of all available accesses has been recognized as a key mechanism to provide the necessary QoS for mHealth services. The scenario, when the two flows are assigned to different interfaces (3G and Wi-Fi) is presented by the second graph.

5. Conclusions

In this paper we aimed to present a transparent and efficient mobility infrastructure for multi-sensor based mobile patient monitoring and tele-consultation. The proposed system incorporates a cross-layer optimization platform and relies on MIP6D-NG and provides prompt solution for mHealth field in heterogeneous wireless networks. We designed and implemented a mobile patient and tele-consultation application for Android devices. We confirmed the applicability and the performance of our architecture with the help of real-life measurements. We performed a video quality test using VQMT tool with different video metrics and a flow-mobility measurement with different flows for heart rate monitoring and video streaming. As a future work we plan to integrate our cross-layer optimized system with adaptive video streaming solutions like MPEG-DASH. We also plan to extend our client-centric solution with network-based mobility management approaches, and to have more attention on network-side cross-layer measurement inputs, e.g., from a Network Information Service (NIS) node.

Acknowledgements

The research leading to these results has received funding from the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n° 288502 (CONCERTO project). The authors are grateful to the many individuals whose work made this research possible.

References

Norbert Varga et al. / Procedia Computer Science 40 (2014) 222 – 229


