HealthCom2011 - IEEE International Conference on e-Health Networking, Applications and Services, Columbia, MO, USA

vNurse: Using virtualisation on mobile phones for remote health monitoring

Devan Rehunathan, Saleem Bhatti
School of Computer Science
University of St Andrews, UK
dr,saleem@cs.st-andrews.ac.uk

Ommena Chandran
Croydon Health Services
National Health Service, UK
ommena.chandran@nhs.net

Pan Hui
Deutsche Telekom Laboratories
Berlin, Germany
ben@net.t-labs.tu-berlin.de

***Abstract*—We present vNurse, a system based on a smartphone platform that permits comprehensive, secure and modular patient remote monitoring outside a clinical environment, e.g. in the home. Using both virtualisation of the phone OS and virtual mobile networks of sensors with full Internet Protocol (IP) connectivity, we enable real-time remote sensor readings of patient Wireless Body Area Networks (WBANs) to be stored, processed and forwarded securely to healthcare practitioners based at clinical sites, while patients are remote or mobile.**

I. Introduction

vNurse builds upon and extends technologies that are inexpensive and readily available to allow for better patient monitoring. Our proof-of-concept system leverages three technologies, (i) virtualisation; (ii) network mobility; and (iii) wireless sensor networking. We first utilise existing wireless sensors to collect relevant patient sensor readings, such as temperature, heart-rate and environmental readings such as GPS location. These readings are streamed 'live' from the patient to the healthcare practitioner, or stored on a smartphone device (which acts as both a data collector and mobile router) for later upload or retrieval on request. The smartphone connections can be maintained even while the patient is mobile (within the home or travelling outside), permitting 24-hour monitoring. This functionality is achieved by aggregating the WBAN sensors as a single *mobile network*, utilising mobility protocols such as Mobile IP[1] and NEMO[2] to achieve this.

A. Motivation

Healthcare practitioners have a number of standard steps to progress the process of *diagnosis* of medical problems[3], which includes the following forms of measurement and monitoring:

- Conducting tests and measurements on patients: it is often standard practice to gather such information as heart-rate, temperature and blood-pressure (amongst other things).
- Admitting patients for observation: in some cases, the diagnosis may require the careful and continuous monitoring of a patient, for example, periodic sampling of measurements as above.

[1]http://ipv6.com/articles/mobile/Mobile-IPv6.htm
[2]http://datatracker.ietf.org/wg/nemo/charter/
[3]Based on the process in the UK: similar processes exist elsewhere.

Additionally, ongoing patient care, especially for longer term (e.g. chronic) problems may also require further (prolonged) use of both steps. Overall, this may be expensive in human, medical and financial resources, as well as being inconvenient and stressful for the patient.

The need for accurate patient information gathering, retrieval and dissemination is immediately evident. In cases of remote patient monitoring, technical and possibly social barriers make this even more challenging, even though such information can sometimes be time and/or life critical [3].

Important application areas include those listed below. In each case there is improvement in the process of information gathering, reduction of costs, and the potential to improve the quality of healthcare:

- *Home monitoring of the elderly.* As the population of the world ages, remote monitoring (e.g. at home) will reduce the burden on clinical resources.
- *Monitoring of ill children.* Managing illnesses in children (especially chronic illnesses) through detailed monitoring instead of relying solely on possibly inaccurate reports of symptoms from the young patients.
- *Rural healthcare.* In many parts of the world, healthcare centres may be rare or be mobile centres, however mobile phone coverage is increasingly available.

II. Proposed Architecture

A. Overview

Our architecture (See Figure 1) has been designed to solve the problem of remote monitoring in eHealth scenarios. This system comprises of four main components (i) WBAN, (ii) virtualisation, (iii) network mobility and (iv) egress connectivity.

We are proposing to use the smartphone as a data collector, aggregating and partially processing (e.g. [1]), data gathered from the sensors, and as a mobile router for the sensors. It will host a wireless network via its WiFi interface to provide network connectivity to local wireless sensors (and of course, it can use other technology such as bluetooth). It will also provide uplink capability via the existing 3G interface. As patients roam, we leverage the prevalence of 3G coverage to provide network connectivity at all times. To accomplish this, the smartphone will also be acting as a NEMO mobile router,

in this way, existing TCP/IP connections will be handed over as the smartphone moves from cell to cell.

The smartphone's Home Network will be the server to which the healthcare practitioner wishes the data to be sent back. A Home Agent is a server at the Home Network which runs the NEMO protocol that has a public IP address that allows the smartphone to connect to. It serves as a central repository and secure distribution point for the patients readings to other interested parties such as the patients relatives or friends who have been authorised.

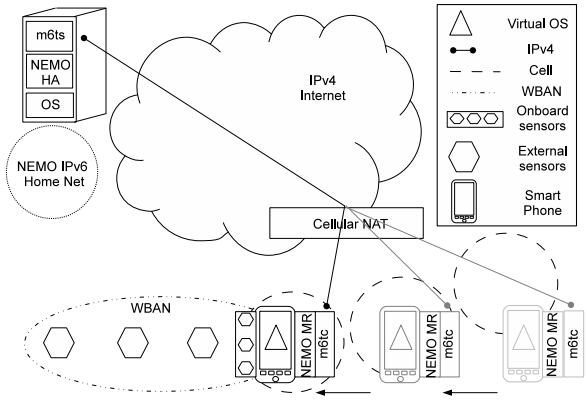

Fig. 1. A broad overview of our proposed architecture. An Android smartphone is configured as a NEMO mobile router to provide network connectivity to its local wireless sensors using a custom virtual file system on the phone. The m6t protocol is used to tunnel the IPv6 NEMO packets through the IPv4 3G connection on the phone to the NEMO Home Agent provided by the healthcare provider. As the WBAN roams from cell to cell, the sensor information can still be accessed and existing connections remain persistent.

B. WBAN Component

Mobile WBANs have a unique set of requirements that should be considered to ensure their successful deployment. Requirements such as bio-sensor design and sensor system design [2] are important for our medical scenario. For bio-sensor design, our scenario needs sterile sensors as the intended target could be someone who is ill or someone who is recovering from illness. Similarly, the sensors should also be as comfortable as possible as they are likely to be attached to the subject for a considerable period of time, both day and night. It would also be beneficial if the sensors were discrete and/or possibly visually unobtrusive, for patient comfort. This is to reduce to any negative psychological sentiment that might result from the sensors to the patient.

For the sensor system, the sensors should be made from off-the-shelf components to keep costs low. We believe that an open market and standard architecture will encourage early adoption by hospitals. This concept fits well into existing platforms proposed such as the Continua Alliance[4] and IHE[5]. It will also mean that practitioners and patients will be able to select the appropriate sensors to incorporate into the mobile WBAN. An open market would encourage a large variety of sensors to be available. Such a market does not exist today, and we take the position that showing the feasibility of a low-cost, easy-to-use mobile network platform enabled using a consumer device such as a mobile phone would encourage such a market [6].

In this architecture, the patient may wear any number of sensors required. These external sensors along with those on the smartphone constitute our WBAN. As the subject moves location, the readings from the sensors are recorded, possibly encoded (e.g. compressed and/or encrypted) and forwarded to the healthcare practitioner. Including additional environmental sensors, such as those that measure room temperature, allows for some possible correlation between how a patient feels with relation to different surroundings in a quantitative way.

Figure 2 outlines the basic setup of our mobile WBAN platform. The large box labelled *Host OS* is an HTC G2 Touch smartphone running Android 2.2 (with a custom 2.6.29 kernel). We assume that the patient carries a smartphone at all times. We chose the Android platform because it has a growing market share and is arguably the most open mobile platform available at the time of writing. The box labelled *Guest OS* refers to a mounted image of a Debian file system that we built and placed on the SDcard of the phone.

The boxes labelled *Network* refer to the network connectivity of the Debian and Android entities. We use this connectivity to pass data between the two entities. The oval labelled *ASE Server* refers to a python daemon that we have written to access local sensor readings through the Android API as well as act as a sink for external sensors wirelessly connected to the phone. The oval labelled *Controller* is an another python daemon that is running within the Debian file system. Its purpose is to push the reading across the network to the (remote) monitoring machine. Some data processing may occur here such as pre-analysis or data compression, plus encapsulation for security, e.g. through use of standard libraries such as SSL. Real time readings are also made available to the patient here.

C. Virtualisation

We used a virtual Debian file system to encapsulate the information processing, information forwarding and mobility software. By encapsulating relevant information processes and resources into a virtual file system, we made the solution portable from device to device. It will be possible to build an entirely custom virtual file system and populate it with the required programs for processing, monitoring and forwarding. These file systems can also be easily shared and distributed to patients who require monitoring. These patients only have to load the virtual file system and run it, they do not have to perform any configuration beforehand. We encapsulate the required functionality into a Debian image for the following reasons; (i) we have access to greater Linux functionality that is not available within the Android hosts OS; (ii) we have a natural way to port and customise our application directly without affecting the host operating system; (iii) by

[4]http://www.continuaalliance.org/index.html
[5]http://www.ihe.net/

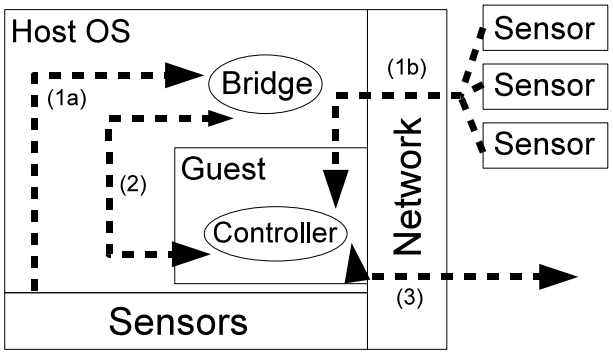

Fig. 2. This figure shows the flow of information in our mobile WBAN platform. Line (1a) represents the readings from the local sensors on the phone that is accessible via ASE Python API sensor facade. Line (1b) represents the readings from external sensors attached to the NEMO MR network. Line (2) represents the raw sensor readings being pulled into the *Controller* application running within the Guest OS. Line (3) represents the stream information that is then sent across the network to the healthcare personnel monitoring the patient. This stream also allows for management information to be passed to the controller.

manipulating the permissions of the file image, we can control the access rights to the information stored within the image and thereby have some level of security control for sensitive information. Similarly it should be possible to configure the rights of the file system to protect sensitive information on the host OS such as the messages and contacts.

The use of the VM image would also allow the monitoring capability to be placed easily onto the patient's personal mobile device (assuming he had a suitable device), for further convenience, if the patient was not willing to carry an additional device. The VM image sandboxing would protect the monitoring functionality, as well as prevent the potentially complex configuration of the monitoring system from interfering with the normal operation of the device if the application was installed on the host OS directly.

D. Mobile Networking

Network Mobility (NEMO) [6] requires a custom kernel that has specific options (such as IPv6) enabled. In order to get NEMO working on Android, we had to build a custom Android kernel and flash it onto the phone. Fortunately, the 2.6.29 kernel source [7] and other required binaries are freely available. We note that this set-up sometimes breaks the existing WiFi module. The workaround is to recompile the module and link it to the new kernel source. This module can then be manually installed onto the phone. For the NEMO userland, because we are installing the userland within the Debian VM file system, and we are not choosing to link the Android file system to it, there is an additional step of installing the new kernel modules into the Debian VM image. Only with these in place can the NEMO userland be successfully built and installed. Another issue we faced was with the *tun* module. This module is required for IP tunnelling. In a previous kernel configuration, where the *tun* module was built-in, we discovered that Android places it at */dev/tun* as opposed to the usual */dev/net/tun*. As a workaround, we built *tun* as a module and installed it directly onto the Debian image, and used *mknod* to create the appropriate directories.

E. 3G connectivity challenges

At the time of writing, there are some engineering issues with 3G connectivity that, whilst not invalidating the architectural approach we have followed, make practical aspects of the engineering challenging. Note that all of the issues listed below can be solved by the 3G provider adjusting the network configuration as required.

Commercial 3G networks often use Network Address Translation (NAT), so a server process (such as the Controller) on the VM image may not be reachable easily by connection requests ingress to the smartphone. For example, a technician wishing to perform some system maintenance on the VM image would typically initiate a connection request to the VM image for a management application.

Additionally, 3G providers may use transparent proxies, stateful firewalls, or simply block certain ports and protocols. This may perturb the operation of certain parts of the application, or at least require reconfiguration of the VM image or the host OS.

Most networks today run IPv4. We have created our proof-of-concept using IPv6 in order to use NEMO. In the short-term, various solutions for IPv6/IPv4 inter-operation exist, although they add to the engineering complexity, and create additional management and operational overhead. In the longer term, native IPv6 support will enable greater use of mechanisms such as NEMO. Also there is other work which suggests a more flexible approach to mobile networks which can be realised as a set of extensions to IPv6 [5] [4].

III. Evaluation

A. Power Consumption

To evaluate the feasibility of vNurse we have examined the power consumption of the smartphone in different configurations and scenarios during data collection. Because we do not want to bias our evaluation with the power effects of any network handovers, in the following tests, the smart phone is in a stationary position. Our main method of collecting battery information has been through the Python API on the Android Scripting Environment (ASE) [8]. This can be installed on Android platforms via the Android Market. Because we did not wish to run additional Python scripts outside of the Debian VM image, we exported the 'AP-PORT' values into the VM. This made it possible to access the Python Android API from within the shell of the Debian file system. Using this API we are able to get information about the battery levels in terms of an overall remaining percentage and also in terms of power. One issue using this API is that the LCD screen has to remain lit in order for the sensor readings to be collected. As a result

[6] http://umip.org/

[7] https://github.com/behnaam/HeRo-2.6.29-GoDmOdE

[8] http://code.google.com/p/android-scripting/

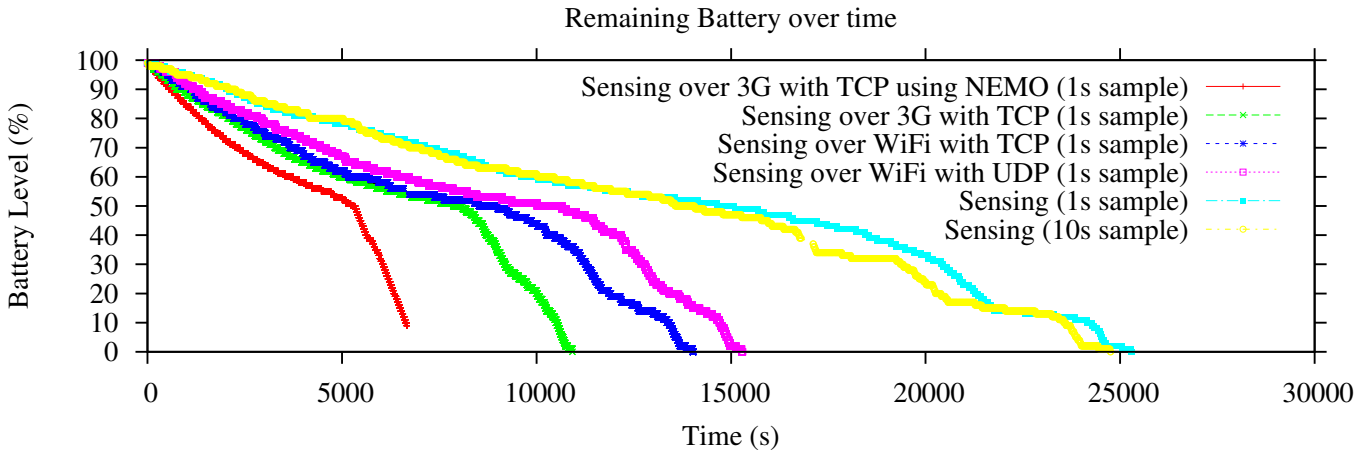

Fig. 3. This figure shows the impact on overall power consumption due to different features being turned on and also the impact of different protocols used to transmit the sensor information.

of this, we have forced the LCD to remain lit throughout all our experiments, (adversely) biassing power consumption figures. As a result, the figures should not be taken as absolute values but should be considered relatively.

1) On the Effect of Network Protocol: Due to the NAT-ed nature of the 3G service provision on the phone, we are so far unable to run a UDP server/client successfully between the smartphone and the central server. To gauge if this would have an impact, we created two versions of our client/server sensors. The first uses UDP and the other uses TCP. Based on our results (see Figure 3, where we ran the exact same client/server programs, with the sole exception being the default network protocol), we show that a TCP connection consumes a larger amount of power over time compared to UDP.

2) On the Effect of Wireless Medium: We also wanted to gauge the power impact of running the sensor application on different physical connectivity. In our proposed architecture, we use 3G for the uplink interface and WiFi for the 'internal' interface to the mobile network. In the future, there might exist smartphones with two or more Wifi interfaces. If this were the case, it would be possible to multi-home between the 3G interface and one of the Wifi interfaces, the other WiFi interface could be reserved for the NEMO mobile network. In such an event, Figure 3 also shows the difference in power consumption between the 3G and WiFi mediums.

B. On the Effect of NEMO

Here we compare the power consumption of a smartphone running the sensor application over 3G and a smartphone running the sensor application while acting as a NEMO Mobile Router. This basically involves creating a wireless hotspot for wireless mobile sensors. In order to create this hotspot, we enabled the WiFi and 3G interfaces simultaneously and started the NEMO daemon. Figure 3 shows the additional power cost of running the NEMO Mobile router daemon as well as maintaining the wireless mobile hotspot.

IV. Conclusion

Utilisation of commercial off-the-shelf technology in WBANs in healthcare opens up the possibility of using inexpensive, unobtrusive and unsupervised monitoring for patients during their daily activities for prolonged periods of time. The benefits of such applications are numerous. Patients will now be able to have continuous monitoring in the comfort and privacy of their homes. Less medical manpower per patient will be required, which means that healthcare workers will be able to manage their time with greater efficiency, thus providing better care at a reduced cost. Constant remote monitoring would also permit faster response times in the event of a medical emergency, and the potential for speedier and/or more accurate diagnosis in some cases. Doctors will also have much more feedback to make more accurate diagnoses. In this paper we have shown the feasibility of our proposed system (vNurse), with the main challenges being power consumption and availability of inexpensive wireless medical sensors.

References

[1] I. Homed, A. Misra, M. Ebling, and W. Jerome. Harmoni: Context-aware filtering of sensor data for continuous remote health monitoring. In *PerCom 2008 – 6th Annual IEEE International Conference on*, pages 248 –251, Mar 2008.

[2] P. Kulkarni and Y. Öztürk. Requirements and design spaces of mobile medical care. *Mob. Comput. Commun. Rev.*, 11(3):12–30, 2007.

[3] J. S. Raj Gururajan, San Murugesan. Bringing mobile technologies in support of healthcare. Cutter IT Journal: The Journal of Information Technology Management, August 2005.

[4] D. Rehunathan, R. Atkinson, and S. Bhatti. Enabling mobile networks through secure naming. Proc. MILCOM2009 – 28th IEEE Military Communications Conference, Oct 2009.

[5] D. Rehunathan and S. Bhatti. A Comparative Assessment of Routing for Mobile Networks. Proc. WiMob2010 – 6th IEEE International Conference on Wireless and Mobile Computing, Oct 2010.

[6] D. Rehunathan and S. Bhatti. Application of Virtual Mobile Networking to Real-Time Patient Monitoring. Proc. ATNAC2010 – 2nd Australasian Telecomm. Networks and Applications Conf., Nov 2010.