SEMComm: Sharing Electronic Medical Records using Device to Device Communication

Tomasz Kalbarczyk and Christine Julien
The University of Texas at Austin
{tkalbar, cjulien}@utexas.edu

ABSTRACT

This demonstration showcases SEMComm, an Android application that allows an individual patient's personal device (e.g., smartphone) to collect health data from nearby medical IoT devices and to share pieces of medical records with the devices of nearby medical personnel (e.g., doctors and nurses) using direct device-to-device (D2D) links. SEMComm uses XD, a middleware that enables device discovery, context sharing, and data transmission using heterogeneous D2D communication technologies. Current approaches for sharing electronic medical records use onerous HIPAA-compliant cloud-based solutions that are costly for hospitals and require patients to release sensitive medical records to an external server. SEMComm allows patients to maintain fine-grained control over who has access to their electronic medical data, while simultaneously allowing the patient's record to collect data from multiple medical devices all without the need for an external network or cloud storage. Our demonstration shows how XD enables SEMComm to collect data from a blood pressure cuff and a heart-rate monitor and then to share medical data with neighboring devices using a mixed set of D2D communication links.

1. SEMCOMM

As IoT devices have grown in popularity, one sector that has seen considerable activity is the medical field. Many devices can monitor patient attributes, such as heart rate, blood pressure, and glucose level; applications to collect data from patients exist for most of these devices. However, existing approachs upload data to a cloud server for all access and analysis. Accessing electronic medical records (EMRs) is typically done through HIPAA-compliant cloud solutions, which require patients to be comfortable with storing medical information electronically in a remote location. The recent growth of device-to-device (D2D) communication presents an alternative for sharing EMRs: patients can directly own and store their own EMR data and exercise fine-grained control over who has access to the data. Recent studies show that patients have a desire for granular privacy control over what medical information is accessible and by whom [2]. D2D communication such as BLE and Wifi-Direct have ubiquitous support across modern mobile devices. These technologies make it possible for a patient's personal devices (e.g., smartphones) to collect medical data directly from IoT devices, for example in a clinic, and to share EMRs (including data collected from fitness applications) directly with nearby medical personnel. Our SEMComm Android application implements such a strategy that collects, stores, and shares medical data, all using resources local to the patient's device or connected directly to that device via D2D links. SEMComm showcases the XD middleware [4], which abstracts heterogeneous available D2D communication technologies into streams of *context* and *data*, hiding the details of D2D communication from the application. Using XD also enables SEMComm to rely on other forms of context, including, for example, using location to restrict transmission of medical data to within an exam room.

Related Work. Many cloud services, such as CureMD and AthenaHealth, manage electronic medical records. Most of these suffer from concerns of cost, privacy, data security, and data interoperability. Recent work aims to solve these issues through more suitable encryption techniques [6] and detailed reference models to consolidate varied EMRs [1]. However, all of these approaches use the cloud to share EMRs unlike SEMComm, which uses D2D links.

Some work has facilitated communication with medical devices. InHealth [3] focuses on communication between advanced monitoring devices in hospitals, but does not touch on medical IoT devices. Beetle [5] improves communication between mobile phones and BLE peripheral devices (including most IoT medical devices on the market) by virtualizing the hardware layer and providing a generic API for interacting with the devices. While Beetle is useful to abstract communication channels, it does not solve the issue of sharing the EMRs themselves. SEMComm provides both.

The Application Scenario. To concretely demonstrate the problem SEMComm solves, we describe it in the context of a particular patient who has been monitoring his heart rate over several months using a commodity BLE heart rate monitor. Upon advice from his doctor, he also takes daily blood pressure readings. He was initially using a wireless blood pressure cuff connected to his smartphone through his home's WiFi network; he recently replaced this cuff with a more modern one that uses BLE to directly send measurements to his phone. SEMComm, relying on XD, supports this daily data collection. Further, because SEMComm fo-

Middleware Posters and Demos '16 December 12-16 2016, Trento, Italy

ACM ISBN 978-1-4503-4666-5/16/12.
DOI: http://dx.doi.org/10.1145/3007592.3007609

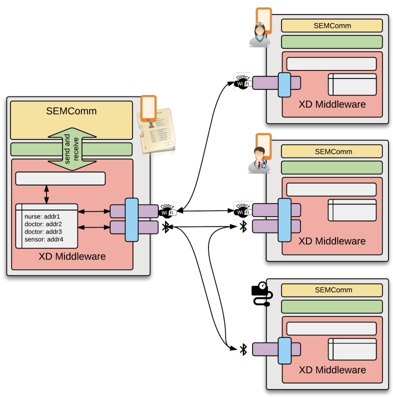

Figure 1: SEMComm in context of D2D Network

cuses on application-level context and data streams, the change from a WiFi-based cuff to a BLE one is completely transparent to the application. This data is obviously private from the patient's perspective and should be available only to himself and potentially to medical providers, assuming it is relevant to some treatment.

Upon arriving for a routine visit at his doctor's office, the nurse and doctor treating the patient need to view various elements of his records. The patient's policy for releasing information may require the doctor or nurse to be co-located (e.g., in the same exam room) with the patient to view his medical data. The nurse needs to take the patient's blood pressure using an in-clinic medical grade cuff, which, using XD, is automatically inserted into the patient's health data on his own device. SEMComm also allows the patient to specify which data is released to which personnel, depending on their roles and the context of the visit. The nurse may only need to see the in-clinic readings, while the doctor may need to also see the full set of measurements done at home. Since SEMComm ensures that the EMRs are only stored on the patient's device, the nurse will immediately be able to see in-clinic measurements via the D2D link with the patient's device, but the measurements done at home will only be transmitted to the doctor's device. Figure 1 provides a visualization of the scenario from an IoT / D2D perspective.

We assume the existence of a hospital web server that acts as a central authority for registering and authenticating doctor, nurse, and patient devices. However, none of the medical data is sent through the server. SEMComm also uses XD to perform device discovery among the devices belonging to the patient, doctor, nurse and hospital. XD performs D2D transmission between devices, including exchanging authentication information, requesting medical data, and transmitting EMRs.

The User Experience. SEMComm provides two user interfaces: one for the patient and one for medical personnel. The patient views a menu for collecting data from medical IoT devices and a menu to indicate what items in his EMR he wishes to share and with what types of users. To simplify the demo, the only types of users he shares data with are nurses and doctors. The patient can also see which nurses and doctors are nearby, allowing him to select specific nurse or doctor devices if he wishes to exercise finer granularity in sharing his data. Finally, the patient can also select whether he needs to be in the vicinity of an examination room, as indicated by a BLE location tag, in order for the data to be available to the specified users.

The medical personnel view lists all neighboring patient devices that have EMRs available. The care provider can select a device and request to view the EMRs that are viewable by that staff member. The staff device provides an authentication key for validation. Upon success, the patient device initiates transmission of relevant medical data over a D2D link. All device discovery and D2D transmission is handled by the XD middleware, including selecting what mode of communication to use; all of these details are hidden from the application.

Usage of XD Middleware. The user interfaces rely heavily on the XD middleware to facilitate device discovery and D2D communication. XD allows applications to periodically exchange lightweight context and to transmit heavyweight data on-demand. SEMComm uses both features. For the patient device to notify nearby medical staff devices that it has data to share, SEMComm includes an identifier that is periodically transmitted as lightweight context using XD. All requests and authentication by the staff device as well as EMR transmission by the patient device is done using the XD API for heavyweight or on-demand data transfer.

Hardware Support. SEMComm works on all Android devices that support BLE. For our prototype and for the demo, we adapted the Withings Wireless Blood Pressure Monitor and the Withings Pulse Ox Tracker to communicate using the Bluetooth GATT standard. In theory, any medical IoT device that implements one of these profiles should work seamlessly with SEMComm. In the future, we aim to extend SEMComm to work for other GATT-based specifications such as Health Thermometer, Weight Scale and Glucose.

Acknowledgments

This work was funded in part by the NSF, Grant #CNS-1218232 and by the DoD.

2. REFERENCES

[1] A. Bahga and V. K. Madisetti. A cloud-based approach for interoperable electronic health records (EHRs). *IEEE J. of Biomed. and Health Info.*, 17(5):894–906, 2013.

[2] K. Caine and R. Hanania. Patients want granular privacy control over health information in electronic medical records. *J. of the American Medical Info. Assoc.*, 20(1):7–15, 2013.

[3] M. Görges. InHealth: A rapid medical software development platform using "internet of things" (IoT) communication standards for medical device interoperability. In *Anesthesia and Analgesia*, volume 122, 2016.

[4] T. Kalbarczyk and C. Julien. XD (exchange-deliver): a middleware for developing device-to-device mobile applications. In *Proc. of Mobilesoft (Demos)*, 2016.

[5] A. A. Levy et al. Beetle: Flexible communication for bluetooth low energy. In *Proc. of MobiSys*, 2016.

[6] M. Li et al. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE Trans. on Parallel and Distributed Sys.*, 24(1):131–143, 2013.