Fafoutis, X., Elsts, A., Oikonomou, G., & Piechocki, R. (2018). SPHERE deployment manager: A tool for deploying IoT sensor networks at large scale. In *Ad-hoc, Mobile, and Wireless Networks: 17th International Conference on Ad Hoc Networks and Wireless, ADHOC-NOW 2018, Saint-Malo, France, September 5-7, 2018. Proceedings* (pp. 307-318). (Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 11104 LNCS). Springer, Cham. https://doi.org/10.1007/978-3-030-00247-3_27

SPHERE Deployment Manager: A Tool for Deploying IoT Sensor Networks at Large Scale

Xenofon Fafoutis[1,2,⋆], Atis Elsts[1], George Oikonomou[1] and Robert Piechocki[1]

[1] Department of Electrical and Electronic Engineering,
University of Bristol, Woodland Road, BS8 1UB, UK
{xenofon.fafoutis,atis.elsts,g.oikonomou,r.j.piechocki}@bristol.ac.uk

[2] DTU Compute, Technical University of Denmark,
Richard Petersens Plads, 2800 Kongens Lyngby, Denmark
xefa@dtu.dk

Abstract. Internet of Things (IoT) technology has the potential to revolutionise several domains of everyday life, including the healthcare sector. In order to reach its full potential, IoT technology needs to be evaluated in the real world, beyond controlled environments, such as laboratories and test-beds. SPHERE is an experimental sensing platform for healthcare in a residential environment. Unlike other similar smart home health systems, SPHERE is deployed in a large number of properties of volunteers. Based on our experiences and lessons learned from SPHERE's large-scale deployments, this paper focuses on the challenge of effectively managing the sensor installation overhead, aiming at supporting our deployment technicians with achieving a satisfactory deployment throughput. In this context, this paper presents the SPHERE Deployment Manager: an open-source tool that facilitates the deployment of bespoke IoT networks by technicians that are not experts in IoT technology. We believe that the SPHERE Deployment Manager is a tool that can accelerate future IoT research deployments of similar nature and scale.

Keywords: Deployment Tools, Sensor Deployments, Internet of Things

1 Introduction

In the era of the Internet of Things (IoT), networked embedded sensing devices are going to be the foundational building block of several critical infrastructures of high societal and economic impact [1]. In this context, there is a growing interest in the academic community in evaluating IoT enabling technologies in the real world, rather than in controlled environments, such as laboratories and test-beds. At an infrastructure level, there is evidence that the environment plays a critical role in the performance of an IoT sensor network [21]. It is, therefore, vital to test these technologies in real environments in order to validate

---

[⋆] Work done prior to joining DTU Compute.

their effectiveness and robustness in a variety of contexts. At a data level, the robustness of machine learning models heavily depends on the quality of the training data. Indeed, models that are trained on input data that are based on small groups of participants, performing scripted activities, under ideal data collection conditions, tend to perform very well [10, 20]. Yet, such models would be challenged in the wild, where the input data are imperfect (*e.g.* missing data, noise, etc. [14, 19]) and derive from large groups of people performing free-living activities [3]. At an application level, several research hypotheses require input from dozens of individuals to be tested. This is particularly relevant for the health care domain. Indeed, there is increasing evidence that behaviour plays a critical role in the development of chronic health conditions, such as depression [18] and dementia [22]. In this context, large-scale IoT deployments have the potential to provide healthcare professionals with an unbiased and quantitative mechanism to assess the long-term behaviour of their patients.

SPHERE (a Sensing Platform for HEalthcare in a Residential Environment) is a multipurpose, multi-modal platform of non-medical home sensors that aims to provide data that would allow researchers to learn the behavioural patterns of the residents, and, enable them to conduct data-rich clinical studies [25]. Different to many other smart home health systems evaluated in controlled environments [13, 15], the SPHERE platform is intended to be deployed in a large number of properties in the Bristol area for a period of up to 12 months (see SPHERE's 100 homes study[3]). At the time of writing, the SPHERE platform has been successfully deployed in the houses of 45 volunteers, and, indeed, the current rate of deployments is only constrained by the rate of recruiting. The purpose of the SPHERE deployments is three-fold. Firstly, a large number of IoT deployments in the wild, would allows us to understand and overcome the weakness and limitations of state-of-the-art IoT enabling technologies: for example, scheduling [6] and interference avoidance [4] in IEEE 802.15.4-2015 TSCH (time slotted, channel hopping) networks. Secondly, data collected from large-scale deployments in the wild, would enable the development of robust machine learning models that are able to operate effectively in real world situations and under missing data [14]. Thirdly, the produced dataset will be shared with medical professionals, enabling clinical research studies.

There are several challenges with IoT deployments of that scale. From software development to remote monitoring, the challenges of deploying IoT sensor networks in outdoor and indoor environments have attracted the interest of the research community over the last decades [2, 12, 17]. Leveraging experience and insight gain from the SPHERE deployments, our previous works discuss challenges, experiences and lessons learned from making SPHERE's bespoke IoT sensing platforms [7], as well as developing IoT networking software for them [5]. Extending our previous work, the focus of this paper is on effectively managing the overhead of preparing IoT sensors for installation in a large-scale context.

Indeed, managing the overhead of preparation and installation is crucial for maintaining a satisfactory deployment throughput. In SPHERE, the preparation

---

[3] http://irc-sphere.ac.uk/100-homes-study

and installation of the IoT sensing platforms is conducted by a small number of deployment technicians that are not experts in the deployed IoT technology. To accelerate the deployment process and facilitate the job of the deployment technicians, we have designed and developed the SPHERE Deployment Manager. The SPHERE Deployment Manager is a tool that supports the deployment technicians with preparing bespoke IoT networks, tailored to the characteristics of each of the participating houses. The contributions of this paper can be summarised as follows. Firstly, we present requirements and challenges regarding the preparation of IoT sensors for deployment from the SPHERE perspective. We believe that IoT deployments of similar nature share the same challenges to a great extent. Secondly, we design, develop and present the SPHERE Deployment Manager: a supporting tool that facilitates the deployment of bespoke IoT networks by technicians that are not IoT experts. Lastly, with this paper, the SPHERE Deployment Manager is released as an open source project[4].

The remainder of the paper is structured as follows. Section 2 summarises the IoT embedded devices and networks of the SPHERE system, including details on security, configurability, and deployment requirements. Section 3 presents the SPHERE Deployment Manager. Lastly, Section 4 concludes the paper, providing deployment statistics.

2 The SPHERE System

In a nutshell, the SPHERE System has three distinct sensing modalities, namely environmental sensors, video sensors, and wearable sensors [25]. The focus of this paper is on the embedded IoT sensing devices of SPHERE, namely the environmental and wearable sensors. For further details on the video sub-system of SPHERE we refer the reader to [11]. The data generated from these sensing modalities is collected in a central server within each deployed house, named SPHERE Home Gateway, and stored in an encrypted solid state drive. Whilst sensor data are saved locally, monitoring data is transmitted over the cellular network to the University of Bristol for remote monitoring.

Each deployed house is uniquely identified in an anonymous manner with a 4-digit identifier, namely the House ID (`HID`).

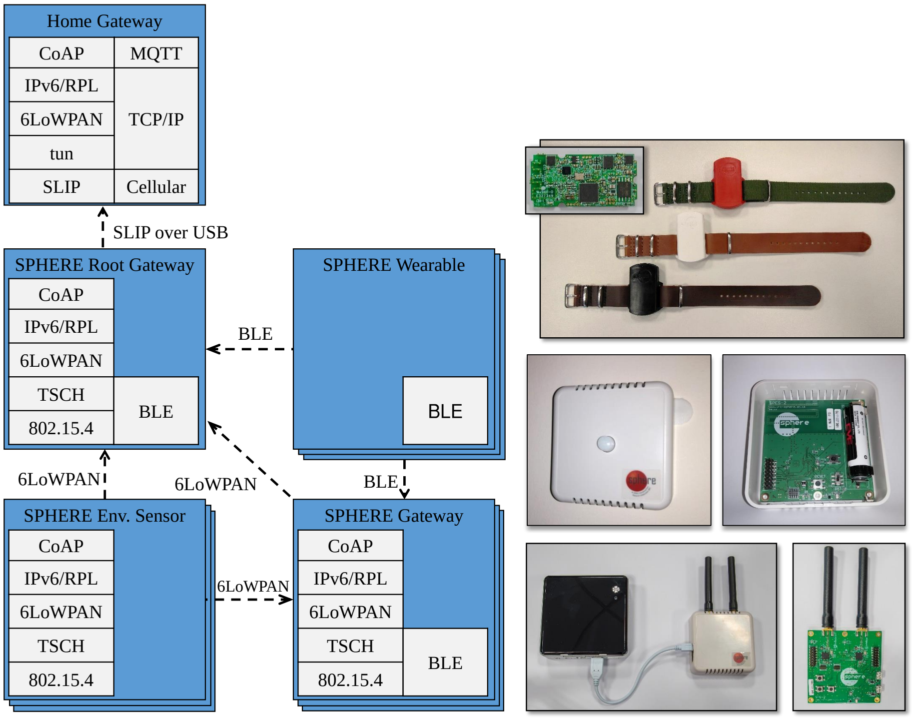
2.1 IoT Embedded Devices and Networks

The IoT sub-system of SPHERE is composed of three types of embedded devices, namely the SPHERE Environmental Sensors (SPES-2), the SPHERE Wearable Sensors (SPW-2), and the SPHERE Gateways (SPG-2), as shown in Fig. 1b.

The SPHERE Environmental Sensors are battery-powered devices and they are responsible for collecting environmental data, such as temperature, humidity, light levels, and mobility levels, at room-level granularity [8]. One of Environmental Sensors also hosts a water flow sensor [24]. The SPHERE Wearable

---

[4] http://www.github.com/irc-sphere/sphere-deployment-manager



(a) SPHERE IoT Architecture [5]

(b) SPHERE Platforms [7]

Fig. 1: (a) Data generated by the IoT sensing platforms of SPHERE are collected over a backbone 6LoWPAN mesh network. (b) The bespoke IoT sensing platforms of SPHERE: SPHERE Wearable Sensor, SPW-2 (top); SPHERE Environmental Sensor, SPES-2 (middle); SPHERE Gateway, SPG-2 (bottom).

Sensors are wrist-worn acceleration-based activity sensors [9]. One wearable sensor is provided to each resident of the house. Lastly, the SPHERE Gateways are mains-powered dual-radio nodes that form a backbone low power network that is responsible to collect the sensor data and forward them to the SPHERE Home Gateway for long-term storage.

In particular, the SPHERE Gateways form a 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) over IEEE 802.15.4 TSCH mesh network in each home [5], as shown in Fig. 1a. One of gateways is connected to the SPHERE Home Gateway, playing the role of the root gateway of 6LoWPAN. All the remaining gateways act as data forwarders. The SPHERE Environmental Sensors are connected directly to the 6LoWPAN network as leaf nodes. The SPHERE Wearable Sensors communicate the acceleration data by broadcasting Bluetooth Low Energy (BLE) non-connectable undirected advertisement packets periodically. The BLE advertisements are received by the secondary radio of the SPHERE Gateways, which operates in BLE mode. Acting as gateways

0	1	2	3	4	5	6	7
`xx`	`xx`	`xx`	`xx`	`xx`	`PID`	`NID`	`DID`

0	1	2	3	4	5
`xx`	`xx`	`xx`	`PID`	`NID`	`DID`

`PID: Project ID`; `NID: Network ID`; `DID: Device ID`

Fig. 2: Custom MAC Addresses: IEEE 802.15.4 (top) and BLE (bottom).

from the BLE to the 6LoWPAN network, the SPHERE Gateways forward the wearable sensor data to the SPHERE Home Gateway. Lastly, the system is leveraging the broadcasting nature of BLE advertisements to collect multiple copies of the wearable data packets. The signal strength of the advertisement packets, as received by multiple gateways, is leveraged for room-level indoor localisation.

A custom MAC (Medium Access Control) addressing scheme is adopted for the embedded devices of SPHERE. As illustrated in Fig. 2, the three least significant bytes of the BLE and IEEE 802.15.4 MAC addresses are modified as follows. The third-to-last byte is the Project ID (`PID`), allowing the same scheme to be used in future deployments; SPHERE uses `PID = 0x00`. The second-to-last byte is the Network ID (`NID`). The `NID` uniquely identifies each deployed house, *i.e.* the House ID (`HID`) is uniquely mapped to a `NID`. The least significant byte is the Device ID (`DID`). The `DID` uniquely identifies each embedded device, and it can also be used to identify the type of the device. In particular, the IEEE 802.15.4 radio of the SPG-2 is assigned a `DID` in `[0x01-0x3F]`; the BLE radio of the SPG-2 is assigned a `DID` in `[0x40-0x7F]`; the SPES-2 is assigned a `DID` in `[0x80-0xBF]`; and the SPW-2 is assigned a `DID` in `[0xC0-0xFF]`. The `DID 0x01` is reserved for the root SPG-2 gateway.

The purpose of the custom addressing scheme is multifold. Firstly, MAC filtering is implemented to prevent external devices from joining the IEEE 802.15.4 TSCH network. This is particularly relevant in cases of overlapping TSCH deployments. Secondly, the custom MAC addresses are used to form the TSCH schedule and ensure that battery-powered devices operate as leaf nodes, *i.e.* do not forward traffic [6]. Thirdly, the custom MAC addresses are used as unique identifiers in the database. This ensures a logical coherence in the sensor data. For example, if a wearable sensor breaks and gets replaced, the same MAC address is used by the replacement and the sensor data from the two devices can be directly linked to each other. It is noted that, globally unique MAC addresses are considered personally identifiable information [16], whilst the custom MAC addresses of SPHERE are equivalent to pseudo-anonymous identifiers.

2.2 Security Requirements

Before deploying the technology in the houses of volunteers, SPHERE went through a process of ethics review and approval. One of the ethical commitments of SPHERE is that all sensor data that are transmitted over the air will be encrypted using state-of-the art encryption.

All the embedded IoT devices of SPHERE use the CC2650 system-on-chip, which incorporates hardware-accelerated AES (Advanced Encryption Standard) encryption. We implement this security commitment using hop-by-hop symmetric encryption, using the hardware-accelerated AES-128 module of CC2650 and one hard-coded encryption key per house. We consider that one encryption key per deployed house provides a sufficient level of security for the following reasons. Firstly, sensor data from the embedded IoT devices are transmitted using short-range wireless technology, and thus, can only be eavesdropped locally. Note that the location and identity of the participants are treated as highly confidential and kept secret even from the SPHERE researchers themselves. Therefore, the threat is limited to the direct neighbourhood of each deployment. Secondly, contrary to using a SPHERE-wide encryption key, a unique key per deployed house secures the confidentiality of the sensor data of participants from each other, should their identity is leaked via their social circles.

An interesting peculiarity of the SPHERE IoT network (see Section 2.1) is that the BLE network is unidirectional: the SPHERE Wearable Sensors communicate their data using non-connectable undirected BLE advertisements. The advantages of this approach are primarily energy-efficiency and efficient mobility support. However, as a result, the SPHERE Wearable Sensors are unable to receive any information. Hence, the encryption key must be installed to the device together with the firmware.

2.3 Configurability Requirements

Every house is different. The SPHERE houses differ in their building era (ranging from Victorian to contemporary), number of rooms (ranging from two-bedroom apartments to detached houses with more than ten rooms), number of floors (from one to four), materials used (different types of locally-sourced sandstone and limestone, bricks, or wood) and in other parameters. The number of SPHERE participants range from one to five per house. Some of SPHERE's houses are close to the centre of the city and receive interfering WiFi signals from a large number of nearby access points; others are in rural areas beyond the boundaries of Bristol.

This diversity makes it challenging to create one-fits-all software solution. In an embedded networking stack, there is a large number of configurable MAC-layer parameters that affect the performance of the system, and that ideally would be tuned for each house separately [23]. However, the large number of deployments makes such fine-tuning impractical in SPHERE. The solution used in SPHERE instead is to (1) enable runtime adaptations (for example, the SPHERE TSCH networking stack automatically selects which wireless channels to use in order to minimise interference from external sources [4]) and (2) to pre-configure the number of sensor devices of each type to be deployed in a SPHERE house, and use that as a constant MAC-layer parameter. Knowing

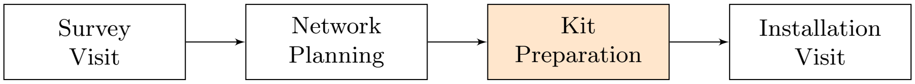


Fig. 3: Overview of the deployment process. The SPHERE Deployment Manager supports the highlighted *Kit Preparation* stage.

this number of devices of each type allows to statically reserve slots in the TSCH schedule in a fair manner and with no run-time overhead [6].

Technically, we achieve this per-house configurability though pre-building SPHERE firmware images for each possible type of configuration. This means that a large number of new firmware images are produced after each software update; it would be impossible to test all these images manually. Therefore we integrated a fully automated regression testing step in the build process. We use the Cooja network simulator for this testing step. As an example, the testing process verifies that for each configuration, the simulated network achieves 100 % packet delivery rate in several different multihop network topologies.

In most cases, the deployment technicians do not need to explicitly configure MAC-layer parameters, since they are implied by the numbers of devices.

2.4 Deployment Process and Requirements

The deployment process, illustrated in Fig. 3, is summarised as follows. Upon successful recruitment, the SPHERE Deployment Officer arranges a Survey Visit, and assigns a randomly generated `HID` to the new participating house, as well as a letter (A, B, $\ldots$) to each resident of the house. During the survey visit, a SPHERE Deployment Technician creates a house plan, identifies a tentative location for the SPHERE Home Gateway, and identifies all available power plugs for the SPHERE Gateways, amongst other tasks that are out of the scope of this paper. The Network Planning stage follows next. In this stage, the technician plans the deployment, identifying the number of required embedded IoT devices and their location. The Kit Preparation stage follows. In this stage, the SPHERE Deployment Technician uploads the firmware on the embedded devices, installs the encryption key, and sets up the respective custom MAC address, as specified in Section 2.1. The IoT devices are then ready for the Installation Visit. During the installation of the devices, the SPHERE Deployment Technician marks the exact location of the deployed sensors. Lastly, the deployment technician gives to each participant their designated wearable sensor.

To facilitate their installation and future maintenance, all deployed embedded IoT devices must be easy to track by the deployment technicians in a quick and efficient manner. We address this requirement with QR (Quick Response) code labels that contain a unique identifier for each device, *i.e.* the custom MAC address for the Wearable Sensors, and the custom IPv6 address of Environmental Sensors and Gateways. Hence, by scanning the QR code label of an IoT device, the technicians can easily inspect relevant information about the device such its `DID` and `NID`. Moreover, they can directly query the contents of the QR code in the database, should they need to look deeper into the monitoring data and sensor data generated by the device. The labels for the wearable sensors also contain the letter identifier (A, B, $\ldots$), assisting the participants with identifying their personal wearable sensor.

It is important to highlight that SPHERE is deployed in the private houses of participants, who are interrupting their everyday life to volunteer to a scientific experiment. It is therefore very important to keep both the duration and the number of the visits to the absolutely minimum. In addition, the time of the technicians is a very limited resource, and the efficient usage of their time is vital for sustaining a good deployment throughput.

3 The SPHERE Deployment Manager

The SPHERE Deployment Manager is a tool that primarily aims to assist the deployment technicians with *Kit Preparation* stage of the deployment, as described in Section 2.4. In addition, the SPHERE Deployment Manager is also used as a quick reference to the active `HIDs` and to the number of IoT devices deployed to each property. An illustration of the folder structure of the SPHERE Deployment Manager is provided in Fig. 4a.

In its current implementation, the SPHERE Deployment Manager supports up to 256 deployments. Indeed, during its installation, it generates a key file that contains 256 randomly generated 128-bit encryption keys. The key file itself is encrypted and password-protected.

The SPHERE Deployment Manager has three types of users, namely the *Researchers*, the *Deployment Officers*, and the *Deployment Technicians*.

3.1 Firmware, Table of Deployments, and Configuration Files

The researchers are responsible for creating and releasing new firmware versions. A new firmware version is released by adding a new directory under the `firmware` directory, titled after the release version number, as shown in Fig. 4a. The directory contains the raw images of each type of deployed device, including a different image for each static TSCH schedule (see Section 2.3). The version number is incremented for each new version, following a `major.minor` pattern – at the time of writing, the current deployed version is `elmer.4`. Old versions are not removed, allowing the deployment technicians to deploy a previous version if faults are identified in the newly released version of the firmware.

The deployment officer is responsible for maintaining a table of deployments that is named `sphere-network-id.csv`, as shown in Fig. 4a. This table is composed of four columns, namely the `NID`, the `HID`, and two binary fields that indicate whether the corresponding `HID` is allocated to a particular deployment, and whether the deployment is active. The table has 256 rows that correspond to the `NIDs: [0-255]`. Each row implicitly corresponds to an encryption key in the key file. As highlighted in Section 2.4, following the recruitment of a new participant, the deployment officer randomly generates a unique `HID` for the new

```
sphere-deployment-manager
├── firmware
│   └── <system-version>
│       └── <raw-images>.hex
├── config
│   └── <HID>.cfg
├── out
│   └── <HID>
│       ├── img
│       │   └── <device>.hex
│       └── labels.pdf
├── sphere-network-id.csv
├── install.py
└── run.py
```

(a) Directory Structure

```
[DEFAULT]
house_id = 0000
version = elmer.4
tsch = 4_and_1_shared_schedule
total_gateways = 5
total_environmental = 8
total_wearables = 4
```

(b) Configuration File Example (`0000.cfg`)

Fig. 4: (a) Illustration of the SPHERE Deployment Manager's folder structure. (b) Example of a network configuration file for deployment `HID = 0000`.

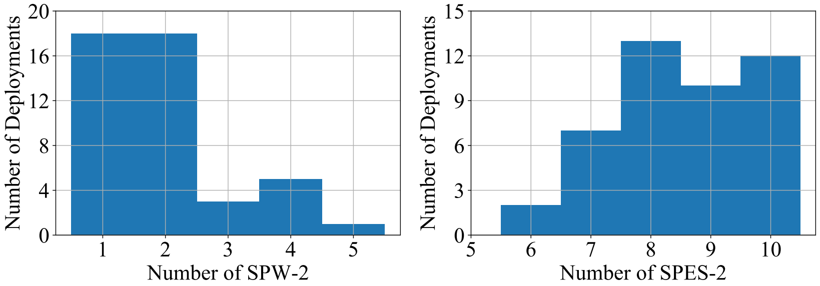
deployment and updates the table of deployments. In particular, the `HID` is assigned to an available `NID` and it is marked as allocated. Further to that, the deployment officer keeps the table up to date with updates in the status of the deployments. For instance, when a participant withdraws from the project, the `NID` is marked as inactive. This ensures that the same `NID`, and thus the same encryption key, will not be used in more than one deployment.

After the *Network Planning* stage of the deployment process (Fig. 3), the deployment technicians are responsible for preparing a configuration file for the deployment. This configuration file records the number of SPHERE Gateways, SPHERE Environmental Sensors, and SPHERE Wearable Sensors to be deployed. In addition, the file specifies the version of the firmware and the TSCH schedule to be used. An example configuration file for deployment `HID = 0000` is shown in Fig. 4b.

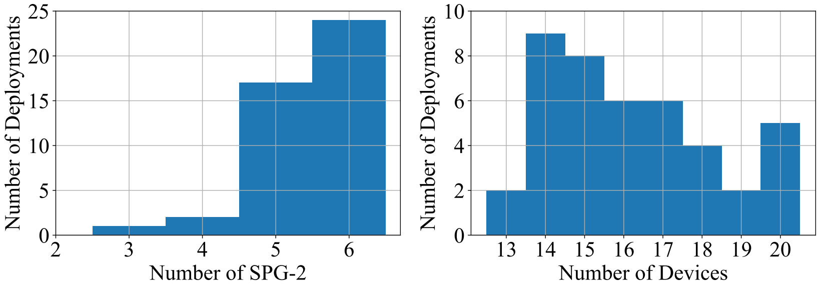
3.2 Operation and Output

During the *Kit Preparation* stage of the deployment process (see Section 2.4) the deployment technicians execute the script `run.py` of the SPHERE Deployment Manager, providing the `HID` as input argument. The technicians also provide the password required to access the encryption key file.

The main purpose of the script is to prepare the firmware image for each IoT device to be deployed. To that end, a special 32-byte block is reserved in the flash memory of the CC2650 System-on-Chip. A explicit instruction in the linker script ensures that this memory block would be free in the raw images (see Section 3.1). The SPHERE Deployment Manager execution script then operates as follows. First, it matches the provided `HID` to the `NID` using the table of



(a) Deployed Wearable Sensors (b) Deployed Environmental Sensors



(c) Deployed Gateways (d) Total Deployed Devices

Fig. 5: Deployment histograms extracted from the configuration files. In total, there are 706 deployed devices in 45 deployments: 88 deployed wearable sensors, 376 deployed environmental sensors, and 242 deployed gateways (April 12, 2018).

deployments and extracts the corresponding encryption key from the key file. Then, for each device to be deployed, it generates the custom MAC address (see Section 2.1) and writes the address and the encryption key on the reserved flash memory block, effectively generating a bespoke image for each IoT device.

The SPHERE Deployment Manager execution script also generates labels for each device to be deployed. The label contains a QR code of the MAC or IPv6 address of the device, as well as the `DID` for quick access. The label of each wearable sensor also includes the respective letter identifier (see Section 2.4). The script then generates a LaTeX source file that places all the labels in order, and compiles a label document (`labels.pdf`), ready to be printed on a 5 × 13 label paper. The outputs of the script are saved in `out`, as shown in Fig. 4a.

The deployment technicians prepare the IoT devices for installation by programming them with their corresponding generated image, mounting them inside their enclosures and sticking the corresponding label on the enclosure.

4 Deployment Statistics and Concluding Remarks

This paper presents SPHERE Deployment Manager, a tool for supporting large-scale deployments of IoT sensing platforms. Indeed, since December 2016, the SPHERE Deployment Manager is used to support the deployment of IoT sensing technology in the houses of 45 volunteers in Bristol, UK. For further information, Fig. 5 plots deployment statistics, extracted from the configuration files of the deployed properties, and serves as evidence that the SPHERE Deployment Manager can effectively support large-scale real-world IoT deployments. It is the authors belief that the SPHERE Deployment Manager has the potential to accelerate future research deployments of similar nature and scale.

Acknowledgement

This work was performed under the SPHERE (a Sensor Platform for HEalthcare in a Residential Environment) IRC funded by the UK Engineering and Physical Sciences Research Council (EPSRC), Grant EP/K031910/1.

The authors would also like to thank the SPHERE Deployment Technicians, Paddy Uglow and Russell Knights, for their valuable feedback during the development of the SPHERE Deployment Manager.

References

1. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of things: A survey on enabling technologies, protocols, and applications. IEEE Communications Surveys Tutorials 17(4), 2347–2376 (2015)
2. Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M.: The hitchhiker's guide to successful wireless sensor network deployments. In: Proc. 6th ACM Conf. Embedded Network Sensor Systems. pp. 43–56 (2008)
3. Diethe, T., Flach, P.: Smart-homes for ehealth: Uncertainty management and calibration. In: NIPS Workshop on Machine Learning in Healthcare (12 2015)
4. Elsts, A., Fafoutis, X., Piechocki, R., Craddock, I.: Adaptive channel selection in IEEE 802.15.4 TSCH networks. In: 2017 Global Internet of Things Summit (GIoTS). pp. 1–6 (2017)
5. Elsts, A., Oikonomou, G., Fafoutis, X., Piechocki, R.: Internet of Things for smart homes: Lessons learned from the SPHERE case study. In: Global Internet of Things Summit (GIoTS). pp. 1–6 (2017)
6. Elsts, A., Fafoutis, X., Pope, J., Oikonomou, G., Piechocki, R., Craddock, I.: Scheduling High-Rate Unpredictable Traffic in IEEE 802.15. 4 TSCH Networks. In: Proc. Int. Conf. Distributed Computing in Sensor Systems (DCOSS). pp. 3–10. IEEE (2017)
7. Fafoutis, X., Elsts, A., Piechocki, R., Craddock, I.: Experiences and lessons learned from making iot sensing platforms for large-scale deployments. IEEE Access 6, 3140–3148 (2018)
8. Fafoutis, X., Elsts, A., Vafeas, A., Oikonomou, G., Piechocki, R.: Demo: SPES-2 – A Sensing Platform for Maintenance-Free Residential Monitoring. In: Proc. Int. Conf. Embedded Wireless Systems and Networks (EWSN). pp. 240–241 (2017)

9. Fafoutis, X., Vafeas, A., Janko, B., Sherratt, R.S., Pope, J., Elsts, A., Mellios, E., Hilton, G., Oikonomou, G., Piechocki, R., Craddock, I.: Designing Wearable Sensing Platforms for Healthcare in a Residential Environment. EAI Endorsed Trans. Pervasive Health and Technology 17(12) (Sept 2017)
10. Gao, L., Bourke, A.K., Nelson, J.: Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems. Medical Engineering and Physics 36(6), 779–785 (2014)
11. Hall, J., et al.: Designing a Video Monitoring System for AAL applications: the SPHERE Case Study. IET Conference Proceedings (2016)
12. Hnat, T.W., Srinivasan, V., Lu, J., Sookoor, T.I., Dawson, R., Stankovic, J., Whitehouse, K.: The hitchhiker's guide to successful residential sensing deployments. In: Proc. 9th ACM Conf. Embedded Networked Sensor Systems. pp. 232–245 (2011)
13. Islam, S.R., Kwak, D., Kabir, M.H., Hossain, M., Kwak, K.S.: The internet of things for health care: a comprehensive survey. IEEE Access 3, 678–708 (2015)
14. Karadogan, S., Marchegiani, L., Hansen, L., Larsen, J.: How efficient is estimation with missing data? In: Proc. 2011 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). pp. 2260–2263. IEEE (2011)
15. Ko, J., Lu, C., Srivastava, M.B., Stankovic, J.A., Terzis, A., Welsh, M.: Wireless sensor networks for healthcare. Proc. IEEE 98(11), 1947–1960 (2010)
16. Krishnamurthy, B., Wills, C.E.: On the leakage of personally identifiable information via online social networks. In: Proc. 2nd ACM Workshop on Online Social Networks. pp. 7–12. ACM, New York, NY, USA (2009)
17. Langendoen, K., Baggio, A., Visser, O.: Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In: Proc. 20th IEEE Int. Parallel Distributed Processing Symp. pp. 1–8 (2006)
18. Livingston, G., Blizard, B., Mann, A.: Does sleep disturbance predict depression in elderly people? A study in inner London. Br J Gen Pract 43(376), 445–448 (1993)
19. Marchegiani, L., Posner, I.: Leveraging the urban soundscape: Auditory perception for smart vehicles. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 6547–6554 (May 2017)
20. Maurer, U., Smailagic, A., Siewiorek, D., Deisher, M.: Activity recognition and monitoring using multiple sensors on different body positions. In: Int. Workshop on Wearable and Implantable Body Sensor Networks (BSN) (2006)
21. Mottola, L., Picco, G.P., Ceriotti, M., Gună, S., Murphy, A.L.: Not all wireless sensor networks are created equal: A comparative study on tunnels. ACM Trans. Sen. Netw. 7(2), 15:1–15:33 (Sep 2010)
22. Pfeffer, R., Kurosaki, T., Harrah Jr, C., Chance, J., Filos, S.: Measurement of functional activities in older adults in the community. J. Gerontology 37(3), 323–329 (1982)
23. Strübe, M., Lukas, F., Li, B., Kapitza, R.: DrySim: simulation-aided deployment-specific tailoring of mote-class WSN software. In: Proc. 17th ACM Int. Conf. Modeling, Analysis and Simulation of Wireless and Mobile Systems. pp. 3–11 (2014)
24. Vafeas, A., Elsts, A., Pope, J., Fafoutis, X., Oikonomou, G., Piechocki, R., Craddock, I.: Energy-Efficient, Noninvasive Water Flow Sensor. In: Proc. 4th IEEE Int. Conf. Smart Computing (SMARTCOMP) (2018)
25. Woznowski, P., Burrows, A., Diethe, T., et al.: SPHERE: A Sensor Platform for Healthcare in a Residential Environment. In: Designing, Developing, and Facilitating Smart Cities: Urban Design to IoT Solutions, pp. 315–333. Springer International Publishing (2017)