Tales from the C130 Horror Room

A Wireless Sensor Network Story in a Data Center

Ramona Marfievici
Nimbus Centre, CIT, Ireland
ramona.marfievici@cit.ie

Pablo Corbalán
University of Trento, Italy
p.corbalanpelegrin@unitn.it

David Rojas
Nimbus Centre, CIT, Ireland
f.rojas-calvente@mycit.ie

Alan McGibney
Nimbus Centre, CIT, Ireland
alan.mcgibney@cit.ie

Susan Rea
Nimbus Centre, CIT, Ireland
susan.rea@cit.ie

Dirk Pesch
Nimbus Centre, CIT, Ireland
dirk.pesch@cit.ie

ABSTRACT

An important aspect of the management and control of modern data centers is cooling and energy optimization. Airflow and temperature measurements are key components for modeling and predicting environmental changes and cooling demands. For this, a wireless sensor network (WSN) can facilitate the sensor deployment and data collection in a changing environment. However, the challenging characteristics of these scenarios, e.g., temperature fluctuations, noise, and large amounts of metal surfaces and wiring, make it difficult to predict network behavior and therefore network planning and deployment. In this paper we report a 17-month long deployment of 30 wireless sensor nodes in a small data center room, where temperature, humidity and airflow were collected, along with *RSSI*, *LQI*, and battery voltage. After an initial unreliable period, a connectivity assessment performed on the network revealed a high noise floor in some of the nodes, which together with a default low CCA threshold triggered no packet transmissions, yielding a low *PDR* for those nodes. Increasing the CCA setting and relocating the sink allowed the network to achieve a reliability of 99.2% for the last eight months of the deployment, therefore complying with the project requirements. This highlights the necessity of using proper tools and dependable protocols, and defining design methodologies for managing and deploying WSNs in real-world environments.

CCS CONCEPTS

• **Computer systems organization → Sensor networks**; • **Networks →** *Network experimentation*; *Network performance analysis*;

KEYWORDS

Wireless Sensor Networks, Low-power Wireless Communications

ACM Reference format:
Ramona Marfievici, Pablo Corbalán, David Rojas, Alan McGibney, Susan Rea, and Dirk Pesch. 2017. Tales from the C130 Horror Room. In *Proceedings of FAILSAFE'17, Delft, Netherlands, November 5, 2017,* 8 pages.
https://doi.org/10.1145/3143337.3143343

1 INTRODUCTION

Wireless sensor networks (WSNs) are nowadays seen as a key flexible infrastructure able to monitor the environment in which they are immersed for application domains that span engineering, scientific, medical and other disciplines. Domain experts have enormous expectations from this technology as an enabler of previously impossible scenarios as well as credible replacement for established solutions. Examples of real-world successful WSN deployments for environmental [2, 3, 18] and animal habitat monitoring [9, 16], or in residential [10] and clinical [13] environments, exist in the literature. However, these experiences demonstrated how difficult it is to run and manage a WSN deployment in the real-world, and that taking out from the laboratories, solutions that have been extensively tested in simulators and controlled testbeds, and immersing them in real-world environments, brings a great deal of complication. This is partly due to the fact that the behavior of the communication channel is affected by the characteristics of the environment (e.g., noise, temperature, humidity, presence of vegetation) in which the nodes are embedded. As a consequence, the behavior of the links, protocols and applications is affected, especially their reliability and energy efficiency. The absence of quantitative evidence about the target application environment, is further limiting the understanding of the behavior of the low-power wireless links and the development and tuning of the systems and the protocols to be well-suited to the specific environment. Moreover, WSN developers are left in the dark without guidelines to drive their deployment. Thus, WSN design and deployment is based on lessons learned from previous deployment experience, from reported experiences in the literature, or from experiments run in simulators and testbeds that cannot work out the many aspects of real-world scenarios.

This paper reports on the entire life of a 30-node WSN deployment for airflow monitoring in a university data center (a tough RF environment due to the high metal contents of the servers, racks, cables and railing), run for a total of 17 months, from October 23, 2014 to March 23, 2016. Fig. 1 provides a concrete idea of: *i*) the extent to which the reliability of the network was affected by the limited understanding of the impact of the target environment on the communication links together with a series of unfortunate events, i.e., battery depletion and gateway outage, during the first eight LOSSY months of the deployment, from October 2014 to June 2015, and *ii*) the RELIABLE eight months period, from July 2015 to March 2016, in which all the nodes were reporting nearly continuously, pushing network's reliability to 99.2%. In between these two chapters in the life of the deployment, LOSSY and RELIABLE, there is a short period

---

FAILSAFE'17, November 5, 2017, Delft, Netherlands

ACM ISBN 978-1-4503-5482-0/17/11…$15.00
https://doi.org/10.1145/3143337.3143343

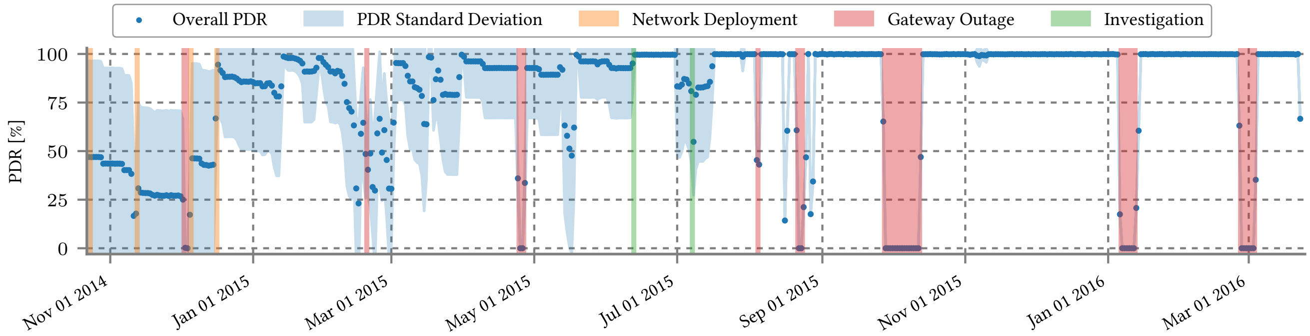


Figure 1: Overall PDR of the network per day, computed considering all 29 nodes, during the complete life of the deployment.

in which during two INVESTIGATION campaigns performed using TRIDENT, a tool for in-field connectivity assessment for WSNs, the impact of the target environment on the low-power wireless links was assessed. Although the packet delivery rate (*PDR*) of the links is the most directly informative indicator of the quality of the link, this time it was not useful as during the INVESTIGATION all links were perfect ($PDR = 100\%$). On a closer inspection of TRIDENT traces, the noise floor measurements highlighted a set of nodes always reporting values of noise floor higher than −90 dBm. This value was in turn the clear channel assessment (CCA) threshold used by the application to decide on clear channel for sending packets. Adapting the CCA threshold used by the application to be well-suited to the target environment and relocating the sink to create more line-of-sight links, transformed the deployment into a successful one, complying with the domain experts' requirements.

This experience highlights the need to assess the characteristics of the links in the target environment, as this supports the WSN deployment and informs the selection of protocol/application parameters to ensure they are well-suited to the environment. Moreover, dependable protocols that can mitigate the impact of the environment (i.e., temperature, noise) are needed to increase the reliability and efficiency of WSNs in harsh environments.

The remainder of the paper is organized as follows. Section 2 introduces the motivating application and its main requirements, while Section 3 presents in detail the deployment. The tales of the deployment unfold in Section 4. We end by discussing the necessity of a methodology for designing and deploying WSNs in Section 5, followed by brief concluding remarks in Section 6.

2 MOTIVATING APPLICATION

The motivation for the WSN deployment presented here stems from the Globally Optimized Energy Efficient Data Centers (GENiC) project [1] involving computer scientists, control system engineers and building engineers, with the aim of developing a management and control system for data center wide optimization of energy consumption by integrating monitoring and control of the IT workload, data center cooling and energy. The control system integrates a thermal management component, responsible for monitoring the thermal environment and cooling system in the data center, predicting temperature profiles and cooling demand, and optimally coordinating and actuating the cooling system. One of the contributions to the project revolves around the most important element of the data center thermal management, the airflow management.

Given that the most crucial objective of data center operations is system uptime, the control strategy to maintain the required environmental parameters (i.e., airflow, temperature, humidity) is vital. For this, chilled air supplied by computer room air conditioning (CRAC) units is provided in the air-cooled small data center via a raised floor plenum through grille vents placed directly in front of the racks. The air is heated as it passes through the IT equipment, then the hot air exhausted from the air outlets at the rear of the racks intermixes with ambient air, eventually circulating back to the CRAC units through the room. The airflow loop in the environment is illustrated in Fig. 2. Temperature of the cooling air actually available for IT equipment depends on the airflow dynamics between the perforated tiles and the equipment inlet. Equipment will draw air as needed and, if sufficient cooling air is unavailable, warm exhaust air will be recirculated over the racks or around the row ends. It is therefore essential that perforated tiles located near the equipment provide sufficient air cooling.

Traditionally, data centers are managed based on accrued experience or best practices, which often lead to an overly conservative thermal management approach, at the cost of wasted cooling resources. Reducing energy consumption and carbon footprint of data centers, on the other hand, requires a fundamental principles based approach. Nowadays, data center engineers supplement prior experience with conceptual understanding of thermodynamics, computational modeling, and data acquisition and processing. In this context, computational fluid dynamics (CFD) have been established as an important tool that enables engineers to examine the airflow and its momentum in data centers. CFD models are key for several, intertwined goals: supporting the analysis and optimization of the cooling performance, by identifying places where the cold air is undersupplied or mixed with the reverse flow, bypass or recirculation; informing the planned modifications in a data center, to ensure optimal cooling configurations and performance and investigate potential failure modes in the data center; informing the disaster recovery planning. For this, quantitative evidence on air flow characteristics in the data center needs to be collected and correlated to the characteristics of the environment. This led to a WSN experimental setup whose deployment was informed by the domain experts in the team, who were interested in monitoring air velocity at key locations in the data center.

Next, we outline the key requirements for the deployment and the application running atop established in collaboration with the domain experts:

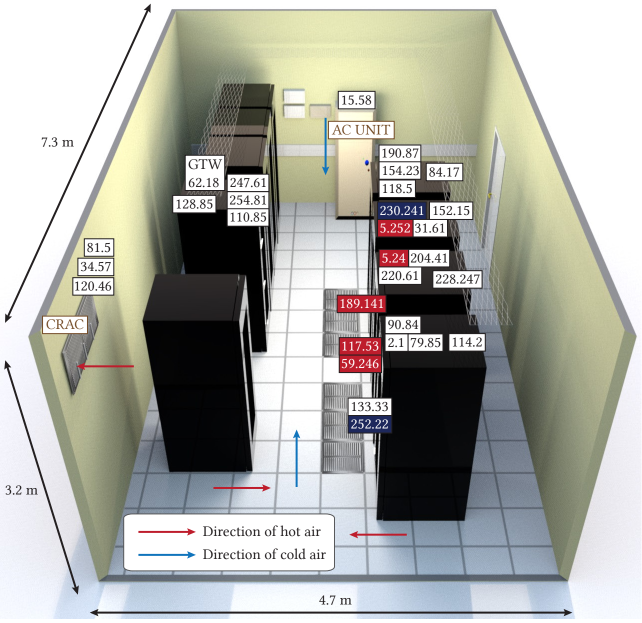


Figure 2: TelosB sensor nodes and gateway distribution in the room. Problematic nodes marked with red and blue.

(R1) *Monitoring key points.* Assessing air velocity at tile, rack inlet, rack outlet and through the CRAC/DX coil is essential for building the CFD models. Therefore, air velocity sensors must be installed as follows: one in each perforated floor tile, three at the CRAC/DX coil, three per rack inlet, at the bottom, middle and top, i.e, 0.35 m, 1.0 m and 1.65 m, at least for every second rack in each row, and at least one sensor per rack outlet for every second rack in each row in the data center.

(R2) *Periodic acquisition, accuracy and reliability.* The time resolution of air velocity measurements is 5 minutes and required accuracy and reliability are 5% and 95%, respectively.

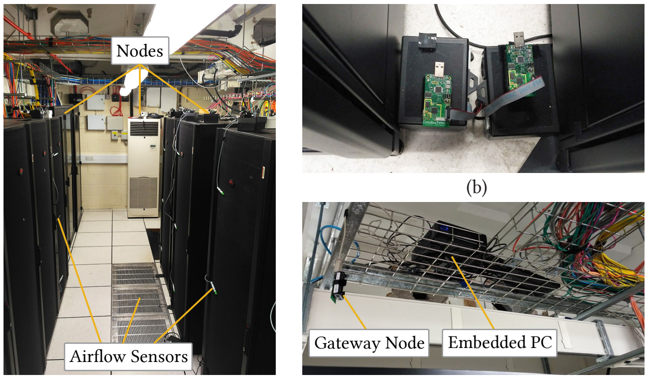
3 DEPLOYMENT SCENARIO

We present the selected location and describe the WSN deployment with the hardware/software components and their functionality.

Location. The deployment area is a typical medium size university data center room that hosts large communications equipment for the campus, as well as the main e-mail and DNS server. The layout of the windowless room, with a floor area of 34 m^2 (7.3 m x 4.7 m), is depicted in Fig. 2. There are 8 server racks in the room, arranged in two rows, forming one cold aisle in the center of the room and two hot aisles between the rear of the racks and the walls. Cooling is provided by a CRAC unit from the adjacent room via underfloor vents through four perforated floor tiles in front of the racks of the cold aisle. A backup air conditioning (AC) unit, i.e., a Mitsubishi PSA-RP140GA split system, is placed in the sever room itself. There are two electrical panels in the room: one fed from the UPS supply via the distribution board in a different room and one fed directly from the main supply. The power is fed to the cabinets from four bus bars, which run above the two rows of cabinets, as shown in Fig. 3(a). Each cabinet has a UPS supply and a main supply.

Hardware and software. The deployment consists of 30 TelosB nodes, including a sink node, containing a TI CC2420 IEEE 802.15.4 standard-compliant radio and an SHT11 temperature and relative humidity sensor chip. To measure the air velocity, each node incorporates a compact airflow probe sensor, either the EE575-V2B1 or the newer EE671-V2XDKD, which extends the operating temperature/humidity conditions and the air velocity range, and slightly decreases the current consumption. The integration of these sensors enables flexible positioning and reduces wiring, resulting in an ideal solution to avoid disturbing data center maintenance. To interconnect the airflow sensors with the nodes, a custom PCB was designed, allowing nodes to switch on/off the sensors to minimize power consumption, and adapting the airflow voltage readings to the ADC voltage reference of the TelosB nodes. Moreover, due to the voltage requirements of the airflow sensor, an external pack of 12 AA batteries was added to supply enough power to the sensor and the mentioned PCB, while the TelosB was powered by the standard 2AA battery pack. Fig. 4 shows our complete hardware platform integrating a TelosB node and an airflow sensor with the external battery pack.

The TelosB nodes run a custom Contiki application that measures and reports airflow, temperature, humidity and battery voltage of both the node itself and the airflow battery pack. The data rate of the application is configurable and set to 5 minutes by default as per requirement (R2). Due to the room size, multihop communication was discarded, and nodes simply form a one-hop network,



(a) (b) (c)

Figure 3: TelosB node and airflow sensor placement.

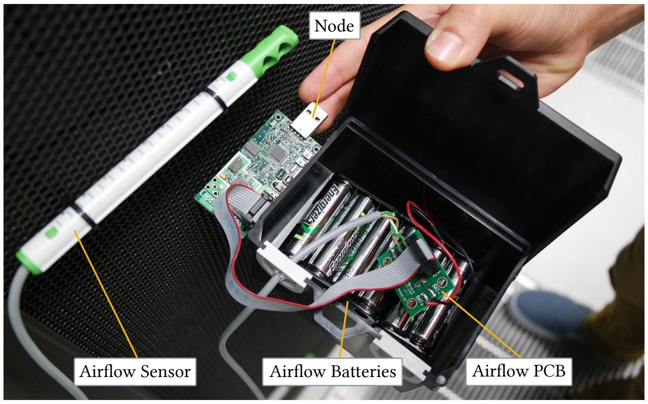


Figure 4: The hardware platform integrates a TelosB with an airflow sensor. The black box contains a battery pack and a custom conditioning PCB for the airflow sensor.

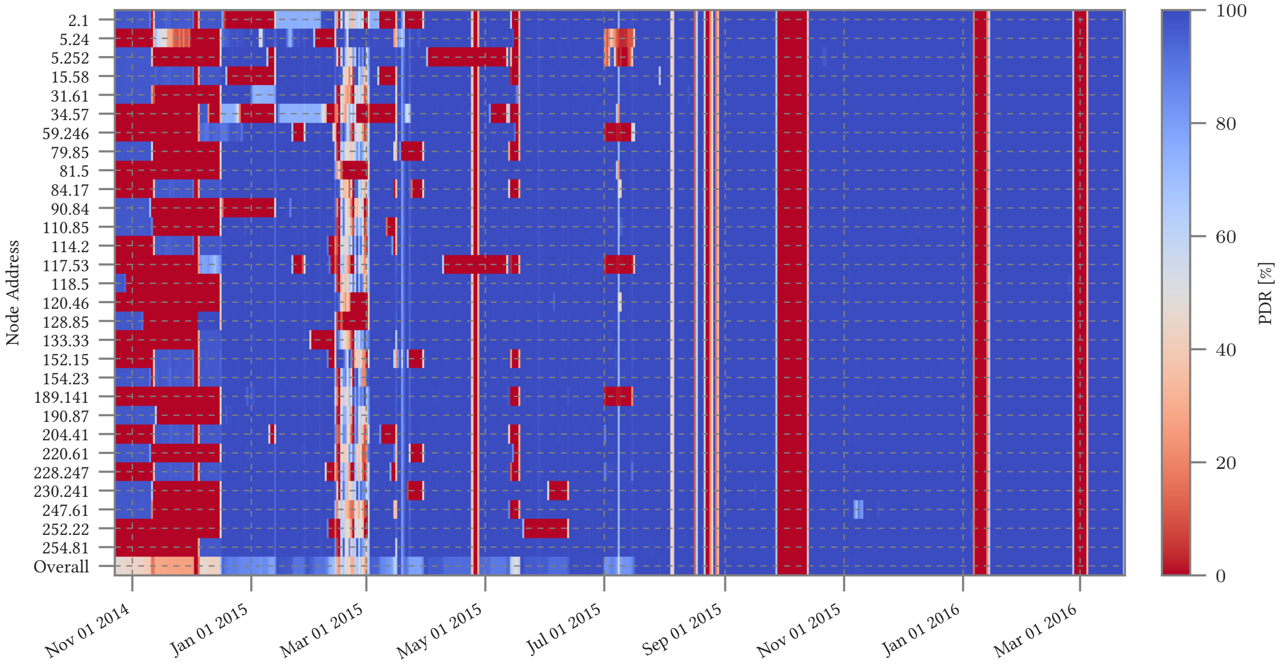


Figure 5: Overall PDR per node per day during the complete life of the deployment.

using the reliable unicast (runicast) primitive of Contiki's Rime stack [8] to report the measured information to a sink node. To reduce power consumption, the nodes only switch on the radio to transmit the packet every reporting period and retransmit it up to four times when necessary. The sink keeps the radio always on, as it is USB-powered. To avoid collisions, nodes employ a CSMA MAC layer and also randomize the exact transmission time within the last four seconds of the reporting period. Furthermore, nodes transmit at the maximum power (0 dBm) and use channel 26 to avoid cross-technology interference. Upon receiving a packet, the sink measures the received signal strength indicator (*RSSI*) and link quality indicator (*LQI*) for offline link analysis, and sends this information in the packet via serial interface to a gateway (i.e., an embedded Linux machine). The gateway parses the received packets and encodes their information in a JSON message published via RabbitMQ to a cloud-based integration platform. This platform includes a dynamic data distribution service, which forwards the monitored data to a storage service with a PostgreSQL database for recording historical sensor values, and also exposes the live data to other components. In addition, the gateway also runs a web-based dashboard tool that exposes the latest information received from every node, allowing network maintainers to detect potential issues during and post deployment.

Airflow sensor and TelosB node placement. To fulfill requirement (R1), the WSN employs fixed airflow sensors deployed in the data center at the positions indicated by the domain experts. The size of the air velocity sensors allowed to be fixed in the predetermined positions using cable ties, at rack inlet and outlet, below the tiles in front of the racks and at the CRAC unit, as shown in Fig. 3(a). However, because of the packaging design and batteries, the TelosB nodes were placed either on top of the racks or in between them (Fig. 3(b)). The exception being the nodes connected to the CRAC unit and the sink, which were placed on top of the metal bars due to constraints induced by the airflow and gateway cables' length respectively.

4 TALES OF THE DEPLOYMENT

To put things into perspective, this section presents the entire life of the WSN deployment and its nodes, as depicted in Fig. 5, run from October 23, 2014 to March 23, 2016, for a total of 17 months, split in three main periods: *i*) Lossy from the inception of the deployment to June 12, 2015; *ii*) Investigation from June 12 to July 16, 2015, and *iii*) Reliable from July 17, 2015 to the end of March 2016. The story of the deployment will unfold in a chronological order and the events are presented as they took place from the perspective of the team investigating the deployment and turning it from a *lossy* into a highly *reliable* one. Using the information gathered by the system we show all the failure modes of the deployment during Lossy. The Investigation shows the approach taken in investigating the deployment using Trident, the decisions informed by the assessment w.r.t. the deployment and the application and the changes done. Then, we show that during the last eight months of Reliable operation, the overall loss rate always remained below 0.08%, which is striking if compared to the average yield of the deployment, 76.89%, before the Investigation.

4.1 Lossy

This section provides a series of important events in the life of the deployment, from its inception on October 23, 2014 to our first Investigation on June 10, 2015. Events are presented as they occurred to the best of our knowledge, from discussions with project members and analysis of the data. The network was installed in four phases from October 23 to December 16, 2014, annotated on Fig. 1 as network deployment. These coincided with the purchasing of the

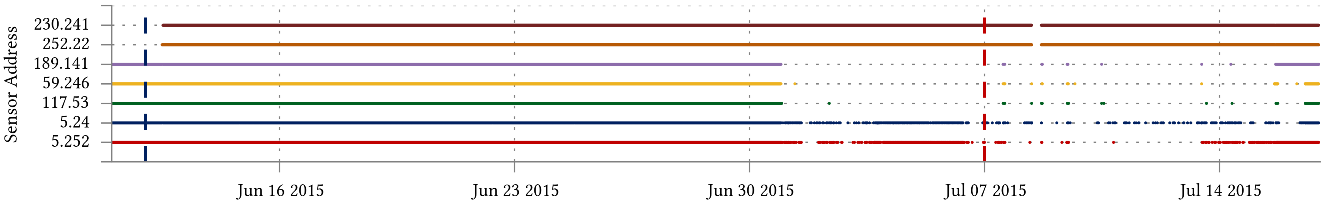


Figure 6: Packets received from problematic nodes during the period with problems reported.

old and new generation of the airflow sensors, and firmware updates towards the final version of the application presented in Section 3. Clearly, the first deployment months represent a trial-and-error period characterized by: *i)* changes in software configuration, i.e., testing different MAC/RDC and Rime primitives for communication, *ii)* airflow sensor testing, and *iii)* rechargeable batteries draining. All these had an effect of "polarizing" the network into dead ($PDR = 0\%$) and close to perfect ($95\% \leq PDR < 100\%$) links, as it can be seen in Fig. 5, and induced overall *PDR* variations, illustrated in Fig. 1. Note that the overall *PDR* is computed considering always all 29 nodes, as this represents the reliability defined in the requirements.

October 23–November 11, 2014. The first 15 nodes, including the sink, were deployed on October 23 running a Contiki application that exploited a different Rime primitive than the final one and reported only the sensor data (i.e., temperature, humidity, and airflow) and the external battery pack voltage. The network then ran with most nodes achieving $PDR \geq 95\%$, as shown in Fig. 5.

November 12–December 15, 2014. The second installation took place on November 12, adding six new nodes to the deployment. However, on this date, most of the initially deployed nodes were taken back to the lab for further testing, due to a soldering problem that appeared in the air flow sensors during the initial deployment, leaving only three nodes (2.1, 15.58, and 154.23) operative since the beginning. As a result, many nodes (e.g., 31.61) stopped reporting data (Fig. 5), decreasing the overall *PDR* (Fig. 1). Further, from December 2 to 4, there was a gateway outage that was solved on December 5, when five new nodes were added to the deployment and another fixed node was redeployed.

December 16, 2014–February, 2015. On December 16, the removed nodes were redeployed and four new nodes (81.5, 120.46, 189.141, 252.22) were installed in the CRAC unit and grille vents, completing the 30-node deployment. Moreover, there was a firmware update using already the Rime runicast primitive. A few days later, however, three nodes, 2.1, 15.58, and 90.84, stopped reporting. The first two nodes, which were running since the beginning, depleted their battery, while the latter died because of unknown reasons. This triggered the replacement of some batteries on January 12.

February, 2015–June, 2015. This was a hectic period for the deployment, characterized by dramatic *PDR* variations and packet losses. During this period, there were two gateway outage events on February and May. Also, the application was enhanced: to sample *RSSI* and *LQI* at the sink upon packet reception, and, because of the many problems triggered by battery depletion, to report the battery level of the nodes inside the same packet used for sensor data and external battery pack voltage. The latter proved to be beneficial, as a battery replacement strategy was developed and an improvement in the overall *PDR* could be observed. However, the network was still far from the required reliability.

4.2 INVESTIGATION

The first contacts between the domain experts and us, the investigation team, took place on June 10, 2015, when the former reported about a decrease in reliability of the network below 90% and consecutive failures of a couple of nodes from the deployment, as can be seen in Fig. 6. At the same time, conversations with a new colleague, part of the investigation team, revealed that an assessment of the characteristics of the low-power wireless links in the target environment, using TRIDENT, might help. The assessment not only informs about the areas of the deployment with connectivity problems but also helps to understand the impact of the environment on the channel behavior and as a consequence on the performance and reliability of the network. A part of the project members thought—as probably some of the readers—that this is not going to help much as problematic nodes were already identified using the web-based dashboard and key metrics like *PDR*, *RSSI* and *LQI* were available. However, what we distilled from the analysis of the traces collected with TRIDENT went beyond expectations. The main findings reported next were gathered during tests of *short* (e.g., few minutes) and *long* (e.g., few hours) duration, run between June 12 and July 16, 2015. Since the location of the deployment was the university data center we had limited access for performing the tests.

Tool support. TRIDENT is a tool to support the in-field assessment of connectivity. It automatically produces the code to be installed on the TelosB nodes, based on the experiment configurations input by the user. Each node can be configured to behave as sender, listener or both. Nodes are time-synchronized and links are probed by having senders transmit messages in round-robin to avoid collision, and listeners record packet reception. For each packet, the sender logs the ambient noise floor before transmission, and the receiver logs *RSSI*, *LQI* and *RSSI* noise floor. Environmental parameters (e.g., temperature and humidity) from on-board sensors can also be acquired. TRIDENT allows splitting a test into a set of rounds, each characterized by a set of parameters—time interval between two consecutive transmissions, transmission power, radio channel and number of messages per sender—configured during the experiment design step. The results of the connectivity tests are stored in the external flash memory of the node. All the interactions with the nodes are done over-the-air [11].

Test setup and execution. All 30 nodes were configured both as listeners and senders in TRIDENT. *Short* tests consisted of a 20-minute round, during which each node sent 40 packets, at a rate of 1 packet/s. We kept the interval of 1 s between the transmissions to avoid possible clock drifts that can cause collisions among senders given that no MAC protocol was used. We used channel 26 and transmission power 0 dBm, same as the application running on the deployment. For each link, the traces collected consist of raw

Table 1: Noise floor and RSSI of problematic nodes.

Node address	Short		Long	
	Noise [dBm]	RSSI [dBm]	Noise [dBm]	RSSI [dBm]
189.141	-84	-51	-84	-59
117.53	-85	-57	-85	-57
5.252	-87	-48	-85	-59
59.246	-84	-43	-81	-52
5.24	-88	-48	-84	-49
220.61	-88	-48	-89	-48
133.33	-88	-45	-89	-46
228.247	-90	-45	-90	-47
152.15	-90	-48	-91	-48

packets along with per-round and overall statistics. *Long* tests were 12 consecutive rounds with the same characteristics as *short*. Although the experimental setting allowed us to probe 870 links, next we report only the characteristics of the 29 links that correspond to the actual links used by the application, having as senders each node in the deployment and a unique receiver, the sink.

First Investigation. The first *short* test was run on June 12, 2015, from 12:00 to 13:00. Special attention was given to nodes 252.22 and 230.241, marked with dark blue on the deployment Fig. 2, and to the assessment of the quality of their links to the sink as they were reported dead since the beginning of June, as shown in Fig. 6.

A quick look at the connectivity map built from the available collected traces from *short*, clearly indicated that all the 29 links of the network were perfect (*PDR* = 100%). Moreover, the *RSSI*/*LQI* metrics for links 252.22 → *sink* and 230.241 → *sink* were −56 dBm/106 and −44 dBm/107 respectively, indicating strong links. The noise floor reported by both nodes when acting as senders was −93 dBm. Nevertheless, we re-positioned the two nodes closer to the edge of the rack and facing the sink node and re-assessed connectivity with a quick test to make sure nodes were connected to the sink with highly reliable links. Then, all nodes were reprogrammed with the application and started delivering the packets, the network achieving 99.96% reliability, as depicted in Fig. 1, after the first green bar marking our first Investigation.

Back in the lab, we further dissected the 20-minute connectivity trace. In an attempt to characterize the target environment we looked at the noise floor. All but the group of nodes reported in Table 1 were exposed to low noise floor levels, below −90 dBm. Later discussions with the domain experts revealed that several nodes from Table 1 were among the ones failing several times. However, none of the nodes was on the list we received before the first Investigation. So far, so good.

Second Investigation. On July 7, 2015, the domain expert reported intermittent failures of nodes 5.252 and 5.24 and complete failure of nodes 117.53, 59.246 and 189.141, all five nodes marked with red on Fig. 2 and illustrated in Fig. 6 as not receiving packets starting July 1, 2015. Coincidence or not, these nodes match the nodes reporting the highest values of noise floor during the *short*. We did not rush out to run another *short* test but decided to run a *longer* one, equivalent of 12 consecutive rounds with the same characteristics as *short*, to account for the variations in time induced by the environment on the collected metrics. The *long* test was run on July 8, 2015, when we had access to the data center for six hours, from 10:00 to 16:00. While the nodes were running the *long* test, we investigated what the network reported from June 30 to July 7, 2015. As looking at the web-based dashboard allows one to grasp quickly which nodes are not reporting, but does not yield insights on what happened to the node before failing, we analyzed the packets sent by the application from nodes 5.252, 5.24, 117.53, 59.246 and 189.141. We provide this view through the lens of *PDR* and *RSSI* and the on-board temperature reported by the nodes in Fig. 7. This shows clearly a significant increase of the temperature triggered on June 30, 2015 at 12:00 sharp. The highest variation occurred over 10 hours, increasing the temperature by as much as 22°C. A zoom into that day, in Fig. 8, shows further that when the temperature in the environment is constant, until 12:00, link 59.246 → *sink* reports a constant *RSSI* of −39 dBm, being the strongest, as node 59.246 was in line-of-sight w.r.t. the sink. Link 5.252 → *sink* and 5.24 → *sink* report lower *RSSI* values, i.e., −43 dBm, −44 dBm and −45 dBm, and jump between these values. Link 117.53 → *sink* and 189.141 → *sink* follow with even lower *RSSI* values reported but higher jumps. These observations can be explained by looking at the node placement in the environment: nodes 5.252 and 5.24 had the shortest links to the sink, while node 117.53 is in the worst position w.r.t. the sink. Moreover, all these links exhibit multi-path effects, nodes being close to the ceiling or positioned in between the metal racks. After 12:00, each substantial increase in the temperature results in a decrease in the *RSSI*. On close inspection, 59.246 → *sink*, the strongest line-of-sight link, exhibited discrete steps of 1 dB in the relationship between *RSSI* and temperature. On the other hand, temperature increase induces more dramatic variations in the *RSSI* of the weaker links, up to 5 dB. For CC2420-based platforms, it has already been demonstrated [6, 11, 15, 17] that these variations can change a good link into a bad one. As the application running on the nodes was not reporting the noise floor, we do not have a measure of the Signal to Noise Ratio (SNR), therefore we can only conjecture that the links disappearing is an effect of temperature and induced high *RSSI* variations. When the heat wave ceased, only two links recovered, corresponding to nodes 5.252 and 5.24, that are on top of the racks and closer to the sink, i.e., thus forming shorter links. Similar behavior was observed during the next heat wave of July 4, 2015. When we looked at the overall *PDR* per node per day during that period, Fig. 5, two other nodes seemed to be failing, i.e., nodes 34.57 and 81.5 connected to the airflow sensors monitoring the CRAC unit area. Based on the discussions with the domain expert, this turned out to be caused by the nodes being taken to the lab for several hours, for soldering, and not by a node/link failure.

Once the *long* test ended and traces were downloaded, we started our analysis looking at the *PDR* computed for the 29 links. Across all 12 rounds, the *PDR* was stable at 100% for all the links. Trident reporting no packet losses was not matching the behavior of the links before Investigation as presented by the domain expert, and still did not provide us any clue. Therefore, we decided to focus on the noise floor recorded by the nodes and the *RSSI* upon their packet reception at the sink. And this was a wise decision since we observed differences in the noise floor across nodes during *short*. Values reported for the *long* test in Table 1 evidences coherent results with the ones from *short* during the first Investigation,

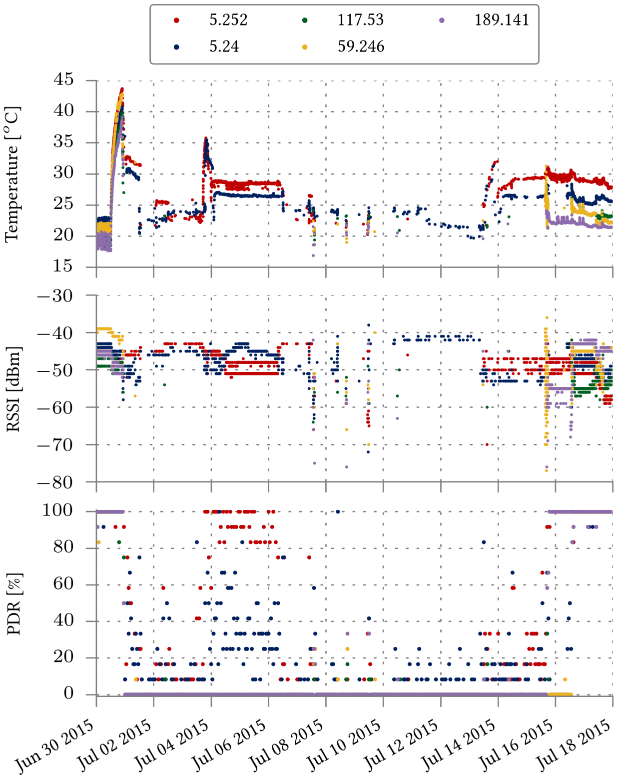


Figure 7: Temperature, RSSI, and PDR of the failing nodes during the second INVESTIGATION.

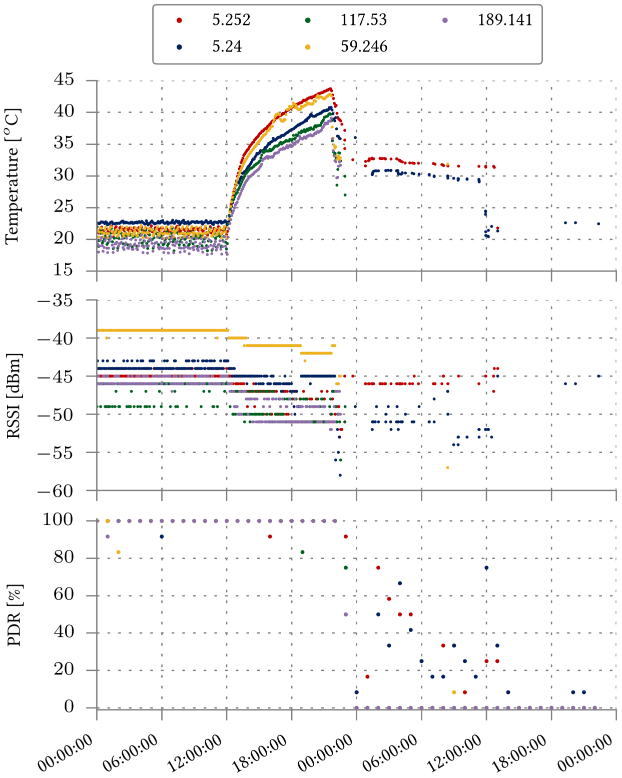


Figure 8: Temperature, RSSI, and PDR of the failing nodes during June 30 and July 1, 2015.

and confirms the higher noise experienced by these nodes compared with the rest of the nodes from the deployment. Moreover, when we looked at all collected traces: *i*) in contrast with the rest of the nodes, these nodes SNR is the lowest, indicating weaker links; *ii*) the sink reported a noise floor value of -82 dBm.

Eureka! Hours later and discussions with the programmer of the airflow application, we solved the puzzle and the clue was the CCA based on the sampled noise floor value before transmission and a programmable threshold. On one hand, all our TRIDENT experiments were run with the CCA check disabled, which translated in nodes sending their packets without checking the energy value in the channel and comparing the measured signal strength with a given CCA threshold. On the other hand, the application was relying on the default Contiki and Cooja (i.e., the emulator where the application was tested before the real-world deployment) CCA threshold of −90 dBm. This threshold is computed as the sum of *RSSI_OFFSET* and *CC2420_CONF_CCA_THRESH*, each having a default value of −45 dBm. This translates in nodes sending their packets *if* the sampled *RSSI* value before transmission is lower than −90 dBm. Looking at TRIDENT collected traces it was clear that some of the links, corresponding to sender nodes from Table 1, were not only weak but most of the times, when running the airflow application, the nodes were not even sending their packets because of the −90 dBm CCA threshold.

The data center was already a tough RF environment due to the high metal contents of the servers, racks, cables and railings. While certain nodes worked well in this target environment, sometimes a set of nodes did not connect at all to the sink which, after INVESTIGATION, we ascribed to the CCA threshold used by the application. The highest noise floor value reported by TRIDENT was −81 dBm. Because we wanted to account for possible variations below this value, variations that might not have been recorded due to the short duration of the tests, we decided to change the *CC2420_CONF_CCA_THRESH* that was used by the application from −45 dBm to −30 dBm, therefore pushing the CCA threshold to −75 dBm. Moreover, we decided to change the position of the sink, which was placed on top of a metal bar. We hanged it, as shown in Fig. 3(c). The new placement of the sink created more line-of-sight links in the deployment. On July 16, 2015, we returned to the deployment site and reprogrammed each node with the updated version of the application.

4.3 RELIABLE

Starting July 16, 2015 till March 23, 2016, that marked the end of the life of the deployment, all 29 nodes were reporting nearly continuously, the overall *PDR* of the network being 99.20%. The intermittent failures, marked in both Fig. 1 and Fig. 5, were caused by: *i*) power outages at the university, causing the gateway laptop to fail. During those times no data was logged by the gateway although the WSN nodes by themselves were operational, since all the nodes would report when the gateway restarted; *ii*) times when the deployment was stopped to replace the batteries.

5 DISCUSSION

Looking back at the WSN deployment from the university data center, we argue that the lack of evidence about the target environment and about how the low-power wireless communication is affected by its characteristics left the WSN developers in the dark, without specific guidelines to drive the deployment and tune the application, i.e., CCA threshold. Moreover, testing the application in a simulation environment like Cooja, which is still missing *realistic* models able to reproduce the behavior of network links under different temperature conditions, further limited the design of the WSN. Unfortunately, the lack of experience with real-world deployments and not devoting attention to the impact of the target environment on the reliability of the network did not help either.

Nevertheless, the findings reported by the INVESTIGATION team based on a few relatively short, 20 minutes and 4 hours, experimental campaigns, using TRIDENT, supported the transition of the deployment from LOSSY to RELIABLE by informing: of the high noise levels experienced, resulting in nodes' relocation; the selection of a CCA threshold that is well-suited to the target environment.

This deployment and the experiences revolving around it have reinforced that in the absence of a clear methodology the WSN design and deployment is still mostly an *art*, based on rules-of-thumb guidelines gleaned from experience, or lab-like testbeds. To support the principled design and deployment of WSNs, which constitutes the premise for WSNs to be a credible tool for domain experts, the WSN community needs to improve the understanding of how the environment affects the network stack and provide tools, models and protocols to address this impact. To this end, it is necessary to understand and characterize the behavior of the WSN in the target environment. The community already has *tools* (e.g., TRIDENT) for supporting the first step of the methodology, in-field collection of connectivity traces. This tool can support: the deployment of WSNs by helping determining a node placement enabling communication or to quickly evidence which nodes experience low *PDR* [11] or high noise values, as in our case; or support connectivity assessment for characterizing the target environment to inform the selection or design of the protocols [11]. *Testbed* infrastructures with realistic environmental effects are also available enabling the study of the impact of temperature (e.g., TempLab [6]) and interference (e.g., JamLab) on protocols.

Moreover, the last years witnessed the development of models that describe the influence of temperature on link quality [4, 15] or estimate radio signal attenuation in forests [7], along with models efficiently reproducing realistic network conditions for simulation of long-term behavior of protocols/applications by accounting for the influence of the environment on the network beforehand, e.g., [12, 15]. These models are key for reducing the gap between simulation and real-world performance of protocols and applications. Additionally, the design and implementation of environment-aware protocols to mitigate the impact of temperature [5] and the effects of radio interference [14] increases the dependability of WSNs deployed in harsh environments. Together these will significantly help by rendering the process of designing and deploying a WSN more repeatable and predictable.

6 CONCLUSIONS

This paper presented the life of a WSN deployment motivated by an application for airflow monitoring in a university data center. During the first nine months, a large number of unfortunate events, from battery depletion to temperature variations, and from node failures to gateway outages, severely affected the data yield of the network. Our most difficult challenge was to understand the impact of the environment on the network and application. This was achieved by means of a tool for connectivity assessment that, after two INVESTIGATION campaigns, showed a set of nodes was exposed to high noise floor values. Adapting the CCA threshold to the environmental conditions and relocating the sink led to an average network reliability of 99.20% for the last eight months of the deployment.

This experience emphasized that assessing the characteristics of the links in the environment where the WSN must be deployed is key for supporting the deployment and informing the selection of communication parameters that make the protocols and the application apt for the target environment. It also emphasized the necessity for a WSN design and deployment methodology and the need for dependable protocols.

ACKNOWLEDGMENTS

The authors would like to thank Alejandro Esquiva Rodríguez and Piotr Sobonski for their contributions to the GENiC project.

REFERENCES

[1] 2013. Globally Optimised Energy Efficient Data Centres (GENiC) project. (2013). http://projectgenic.eu/ Accessed: 2017-07-10.
[2] G. Barrenetxea et al. 2008. The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments. In *Proc. of SenSys*.
[3] Jan Beutel et al. 2011. X-SENSE: Sensing in Extreme Environments. In *Proc. of DATE*.
[4] Carlo Alberto Boano et al. 2013. Hot Packets: A Systematic Evaluation of the Effect of Temperature on Low Power Wireless Transceivers. In *Proc. of ExtremeCom*.
[5] Carlo Alberto Boano et al. 2014. Mitigating the Adverse Effects of Temperature on Low-Power Wireless Protocols. In *Proc. of MASSS*.
[6] Carlo Alberto Boano et al. 2014. TempLab: A Testbed Infrastructure to Study the Impact of Temperature on Wireless Sensor Networks. In *Proc. of IPSN*.
[7] Silvia Demetri et al. 2015. Estimating Low-Power Radio Signal Attenuation in Forests: A LiDAR-Based Approach. In *Proc. of DCOSS*.
[8] Adam Dunkels et al. 2007. An Adaptive Communication Architecture for Wireless Sensor Networks. In *Proc. of SenSys*.
[9] Vladimir Dyo et al. 2012. WILDSENSING: Design and Deployment of a Sustainable Sensor Network for Wildlife Monitoring. *ACM TOSN* 8, 4 (2012).
[10] Timothy W. Hnat et al. 2011. The Hitchhiker's Guide to Successful Residential Sensing Deployments. In *Proc. of SenSys*.
[11] Timofei Istomin et al. 2014. Trident: In-field Connectivity Assessment for Wireless Sensor Networks. In *Proc. of ExtremeCom*.
[12] Ankur Kamthe et al. 2013. Improving Wireless Link Simulation Using Multilevel Markov Models. *ACM TOSN* 10, 1 (2013).
[13] JeongGil Ko et al. 2010. Wireless Sensor Networks for Healthcare. *Proc. IEEE* 98, 11 (2010).
[14] Roman Lim et al. 2017. Competition: Robust Flooding Using Back-to-Back Synchronous Transmissions with Channel-Hopping. In *Proc. of EWSN*.
[15] Ramona Marfievici. 2015. *Measuring, Understanding and Estimating the Influence of the Environment on Low-Power Wireless Networks.* Ph.D. Dissertation. University of Trento, Italy.
[16] Gian Pietro Picco et al. 2015. Geo-referenced Proximity Detection of Wildlife with WildScope: Design and Characterization. In *Proc. of IPSN*.
[17] Kannan Srinivasan et al. 2010. An Empirical Study of Low-Power Wireless. *ACM TOSN* 6, 2 (2010).
[18] Geoffrey Werner-Allen et al. 2006. Deploying a Wireless Sensor Network on an Active Volcano. *IEEE Internet Computing* 10, 2 (2006).