TempLab: A Testbed Infrastructure to Study the Impact of Temperature on Wireless Sensor Networks *

C.A. Boano†, M.A. Zúñiga¶, J. Brown‡, U. Roedig‡, C. Keppitiyagama§, and K. Römer†

†Institute for Technical Informatics, TU Graz, Austria (E-mail: cboano@tugraz.at)
‡School of Computing and Communications, Lancaster University, United Kingdom
¶Embedded System Group, TU Delft, The Netherlands
§SICS Swedish ICT, Kista, Sweden

ABSTRACT

Temperature has a strong impact on the operations of all electrical and electronic components. In wireless sensor nodes, temperature variations can lead to loss of synchronization, degradation of the link quality, or early battery depletion, and can therefore affect key network metrics such as throughput, delay, and lifetime. Considering that most outdoor deployments are exposed to strong temperature variations across time and space, a deep understanding of how temperature affects network protocols is fundamental to comprehend flaws in their design and to improve their performance. Existing testbed infrastructures, however, do not allow to systematically study the impact of temperature on wireless sensor networks.

In this paper we present TempLab, an extension for wireless sensor network testbeds that allows to control the on-board temperature of sensor nodes and to study the effects of temperature variations on the network performance in a precise and repeatable fashion. TempLab can accurately reproduce traces recorded in outdoor environments with fine granularity, while minimizing the hardware costs and configuration overhead. We use TempLab to analyse the detrimental effects of temperature variations (i) on processing performance, (ii) on a tree routing protocol, and (iii) on CSMA-based MAC protocols, deriving insights that would have not been revealed using existing testbed installations.

1. INTRODUCTION

Research and industrial deployments have shown that the operations of wireless sensor networks are largely affected by the on-board temperature of sensor nodes. Temperature variations may significantly affect, among others, clock drift [1], battery capacity and discharge [2], as well as the quality of wireless links [3].

Depending on the packaging and deployment location, the electronics of wireless sensor nodes may experience a substantial temperature variation. Sunshine may easily heat a packaged sensor node up to 70 degrees Celsius – especially if the packaging absorbs infra-red (IR) radiation [4], and long term outdoor deployments have shown that the on-board temperature can vary by as much as 35°C in one hour and 56°C over a day [5]. This variation is sufficient to cause a frequency offset of more than 100 ppm on the crystal oscillator frequency [6], which can affect the rendezvous process of synchronous duty-cycled MAC protocols. Such temperature change is also enough to reduce the received signal strength between two sensor nodes by more than 6 dB [5], which can change the packet reception rate (PRR) of the link from 100% to 0%. Hence, a deep analysis of how temperature affects the operation of sensor networks is necessary to inform the design of robust and dependable applications.

Analytical models or simulation tools that can accurately predict the impact of temperature are hard to obtain due to the complexity of the involved physical processes. Similarly, setting up a pilot deployment of a sensor network to evaluate the impact of temperature can be very complex and time-consuming. On the one hand, meteorological conditions cannot be controlled, making it impossible to ensure repeatability across several experiments. On the other hand, temperature profiles that can be tested are highly specific to the deployment location and to the time of the year in which the experiment is carried out (i.e., to get a flavour of seasonal temperature variations, the pilot deployment should last several months). What is needed to overcome these limitations is an experimental facility that allows researchers and system designers to mimic the temperature variations normally found in outdoor deployments in a fast and simple way.

Traditional testbed facilities used to evaluate protocols and applications under realistic conditions in a cost effective manner such as MoteLab [7], TWIST [8], Kansei [9], and NetEye [10], do not allow the evaluation of temperature effects. To date, a low-cost flexible testbed infrastructure that allows the repeatable generation of predefined temperature patterns across a sensor network still does not exist. Industry makes heavy use of temperature chambers during device verification processes (e.g., to calibrate sensors and transceivers [11]), but such solutions are not suitable due to their high cost and because they target individual components and not a network of nodes, which is necessary to disclose limitations at the communication level. We aim to close this gap and design tools to make a testbed capable of

*This is the authors version of the work, and it is not meant for redistribution. The definitive version was published in: Proceedings of the 13th International Conference on Information Processing in Sensor Networks (IPSN). Berlin, Germany. 978-1-XXXX-XXXX/XX.

reproducing real-world temperature profiles.

Augmenting a testbed with the ability to reproduce temperature profiles is not a trivial task. Firstly, we need to recreate in a faithful manner the temperature variations that each node would experience in a real-world deployment over time. Secondly, these temperature profiles must be applied in such a way that no other property of the setup besides temperature is altered. Thirdly, the temperature profiles reproduced in the testbed need to be repeatable in order to allow a systematic quantification of the impact of temperature, and should ideally emulate daily or seasonal changes within a few hours, allowing fast prototyping and experimentation. All these goals should be met while minimizing costs and efforts.

In this paper we present TempLab, an extension for sensornet testbeds that allows the on-board temperature of sensor nodes to be varied in a fine-grained and repeatable fashion. The contributions of this paper are the following:

- **TempLab infrastructure.** We describe testbed components, methods for implementing different temperature profiles, and evaluate TempLab to show that it can accurately reproduce temperature dynamics found in outdoor environments with fine granularity.
- **Use cases.** We use TempLab to examine and quantify the effects of temperature variations on several applications and protocols, showing that they can drastically change the topology of a network and lead to network partitions, reduce significantly the performance of MAC protocols, as well as increase the processing delay in the network. These findings represent challenges to the research community and may open up a new research area of "temperature-awareness".

This paper proceeds as follows. The next section motivates the need for a testbed solution to evaluate the impact of temperature on wireless sensor networks. Sect. 3 describes the requirements of such a testbed infrastructure. We describe the design and implementation of TempLab in Sect. 4, and investigate its performance in Sect. 5, showing that temperature dynamics found in typical deployments can be accurately reproduced. Thereafter, in Sect. 6, we use TempLab to analyse the detrimental effects of temperature variations on sensornet applications and protocols. After describing related work in Sect. 7, we conclude our paper in Sect. 8.

2. TEMPERATURE MATTERS

Temperature affects the operations of the most basic elements in *all* electric and electronic circuits: from resistors and capacitors to clocks and transistors. Due to this impact, assessing the effect of temperature on *individual* devices is usual practice in industry, and most electronic devices are given an operational range. Temperature also matters at the *network* level, but the effect of temperature on *inter-device* operation is far less understood.

A few studies have started to evaluate the effect of temperature on network operations. Bannister et al. [3] showed

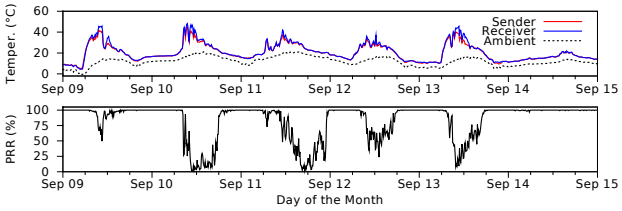


Figure 1: Temperature has a strong impact on link quality in outdoor deployments. Even the normal temperature fluctuations during a day can render a good link useless [12].

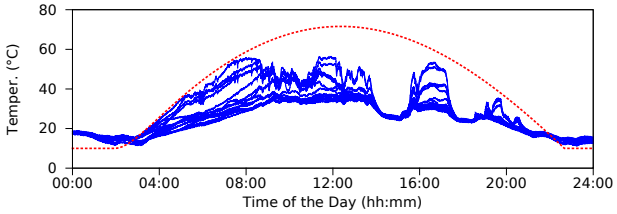


Figure 2: Temperature profiles over the course of a day of 16 nodes deployed in an outdoor setting (blue curves), and maximum temperature profile obtained with the model presented in Sect. 4.2.3 (red dashed curve).

that temperature has a significant impact on link quality, and Boano et al. [5] validated these claims with a more systematic study. The most powerful case highlighting the importance of temperature at the network level is probably given by Wennerström et al. [12], who report insights from a long-term study showcasing the impact of meteorological conditions on the quality of 802.15.4 links. Fig. 1, based on traces recorded during Wennerström's outdoor deployment, shows the on-board temperature of a transmitter and receiver pair, and the packet reception rate of their link: even the normal temperature fluctuations occurring during a day can transform a perfect link (100% PRR) into an almost useless one.

However, besides these initial studies, temperature has not received (at the network level) the same level of attention that it received at the device level, but it definitely should. Temperature introduces a *dynamic heterogeneity* across the network: two nodes with the same parameters, but with different on-board temperatures, will perform differently. It is important to analyse this temperature-based heterogeneity, because even nodes that are physically close can have vastly different temperature profiles.

In Wennerström's deployment [12], indeed, all the nodes are within each-other's transmission range, and experience highly different temperatures. Fig. 2 depicts the on-board temperature of sixteen of these nodes over the course of a summer day [12], and Fig. 3 depicts the temperature density function for two of them. One node is much "hotter" than the other, and this hot node will have a shorter transmission coverage [3], [5], a larger clock drift [1], whereas the lifetime of the cold node will be much shorter [2].

How do all these temperature effects, and others that are yet uncovered, affect the operation of network protocols?

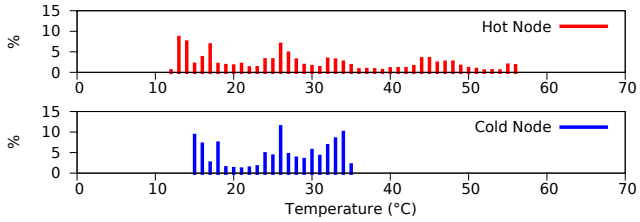


Figure 3: The temperature profile of nodes can be highly different even if nodes are in proximity of each other. This difference can affect the overall performance of the network.

To evaluate these effects, we need to provide the sensor network community with a simple, yet accurate, low-cost testbed infrastructure enabling the study of the effects of temperature variations on the network performance in a precise and repeatable fashion.

3. REQUIREMENTS

Such a testbed solution should essentially have the ability to control the on-board temperature of wireless sensor nodes. However, in order to accurately reproduce the temperature dynamics that can be found in typical deployments, it is not simply enough to choose off-the-shelf heating and cooling elements and connect them to the testbed. The choice of the hardware, as well as the design of the infrastructure should meet a number of requirements that we describe below.

Large temperature range. Ideally, the testbed would be able to reproduce temperature patterns covering the complete operating range of sensor nodes. For example, in the case of the off-the-shelf TelosB platform, this would imply to heat sensor nodes up to 85°C, but also to cool them down to -45°C. While it is perhaps not necessary to cool all the way down to -45°C, it is important to reach temperatures below 0°C to reproduce the conditions that can be found in a real deployment during the coldest times of the year.

Fine-grained temperature control. As shown in Fig. 2, the temperature of a node deployed outdoors can continuously vary depending on the presence of sunshine and obstacles (e.g., clouds or buildings). These effects cause *continuum* gradients of temperature, i.e., the jumps of temperature are not sudden and discrete, but smooth. Since our goal is to recreate temperature traces in the most faithful manner, the testbed infrastructure should be able to precisely tune the on-board temperature of a sensor node with a high resolution.

Fast temperature variations. In a real deployment, temperature can change quickly due to meteorological effects such as wind, rain, and snow, as well as due to the presence of clouds or sunshine. In the deployment shown in Fig. 2, for example, a node that receives the first sun-rays at the beginning of the day increases its temperature as much as 1.98°C/minute. An important requirement for the infrastructure that we want to build is hence the ability of reproducing these variations as fast at they occur in the real-world. This requirement has a strong effect on how accurately temperature dynamics can be reproduced.

Time scaling. It is often desirable to compress the time scale of an experiment to save evaluation time (as long as such time compression does not depend on the rate of the temperature change, but only on the absolute temperature values). One may want to time-lapse the recreation of real-world traces and playback, for instance, in a few hours the profile of a full day. This poses stronger requirements on the ability of the testbed to quickly heat up and cool down nodes.

Per-node temperature control. As observed in Fig. 2, the profile of each node can be highly different. Hence, placing all the nodes into a single chamber would not be realistic because all nodes would follow the same temperature profile. Temperature must be controlled individually on each node.

Unaltered system behaviour. The extension of the existing infrastructure should ideally not alter the behaviour of the system in any way, as this may lead to unwanted (and unexpected) system failures. For example, the use of metal casings should be restrained, as RF propagation should be minimally affected. Similarly, the use of I/O ports of a sensor node to control heating or cooling devices has to be avoided if this would affect the operations of the system.

Scalability. Although it may not be necessary to augment all nodes of an existing infrastructure with temperature control, it should be ideally possible to extend an entire testbed. Commonly used testbeds such as MoteLab [7], TWIST [8], and NetEye [10] have typically up to 200 nodes, and our testbed solution should be able to scale at these levels.

Low cost. All the above requirements have to be satisfied while minimizing the cost of the solution, in order to make it applicable on a large-scale.

4. TEMPLAB: ARCHITECTURE AND IMPLEMENTATION

In this section, we present the general architecture of TempLab, our low-cost extension of testbed facilities capable of reproducing real-world temperature profiles with fine granularity, and describe the hardware and software components that we use in our implementation.

4.1 Architecture

In order to study the effects that temperature variations have on the operation of wireless sensor networks and their protocols, the infrastructure needs to be able to reproduce specific temperature profiles on several nodes. This requires (i) temperature profiles to be reproduced, (ii) actuators to control the on-board temperature of each sensor node, and (iii) a controller that uses the actuators to instantiate the desired profiles.

4.1.1 Temperature Profiles

In order to support a wide range of experimentation techniques, TempLab can generate temperature profiles using three different approaches.

Firstly, one can re-play temperature traces collected in-situ at a given deployment site, such as those in Fig. 2. Such *trace-based* temperature profile instantiation can accurately reflect the temperature variations over time with fine granularity if long-term measurements from one or more nodes are available. However, traces are not always at one's disposal, or they may be incomplete: trace-based profiles can be used only if one or more sensor nodes deployed previously actually collected temperature data with a frequency sufficiently high to capture the dynamics of temperature changes.

A second possibility is, therefore, to use a *model-based* temperature profile to have an estimation about the temperature dynamics at a certain location without the need of traces collected in-situ. A model-based approach uses models to estimate the temperature profile of objects using basic environmental information such as the maximum solar radiation and the minimum temperature during a day (that is readily available from satellites and meteorological stations). We derive such a model in Sect. 4.2.3.

A third possibility is to use TempLab to vary the temperature of sensor nodes using specific *test patterns*. For example, a user may not be interested in recreating a specific profile and needs instead only to verify whether a high temperature variation has an impact on the operation of a given protocol. In this case, TempLab can be fed with on-off patterns (e.g., a series of cold and warm periods) or jig-saw patterns that vary temperature with a specified frequency, allowing a quick debugging of protocols' behaviour.

4.1.2 Actuators

To heat-up and cool-down the on-board temperature of sensor nodes, one or more actuators are required for each node. Actuation can be applied *out-of-band* or *in-band*. Out-of-band means that the sensor node is not involved in the control of its temperature, i.e., additional processing hardware is needed. In-band methods, instead, make use of the sensor node to vary its on-board temperature, e.g., by using its I/O pins to control heating or cooling devices. Although in-band methods have the advantage of avoiding extra-hardware (and reduce testbed costs), they may alter the system behaviour and violate the corresponding requirement.

Therefore, we design TempLab following an *out-of-band* approach based on infra-red heating lamps and cooling enclosures that allow to vary the on-board temperature of wireless sensor nodes in the range [-5, +80] °C. TempLab can have two types of nodes with different capabilities as shown in Fig. 4: *LO* and *PE* nodes. *LO nodes*, which stands for lamps-only nodes, are heating-only devices that have the capability of warming the sensor nodes between room temperature and their maximum operating range. They are based on IR heating lamps and they do not have any capability to cool-down the nodes below room temperature. *PE nodes*, which stands for Peltier enclosure nodes, are hard temperature-isolating Polystyrene enclosures with an embedded IR heating lamp and an air-to-air Peltier module to heat-up and cool-down the inner temperature of the casing. To control the intensity of the IR lamps and the operations of the Peltier module, we borrow existing home automation solutions and use *wireless dimmers* to vary the intensity of the lamps and *on-off wireless switches* to control the Peltier modules embedded in the enclosure. To make sure that the temperature control system does not interfere with the existing testbed communication, we select home automation solutions working on a ISM frequency band that is different from the one used by the sensor nodes.

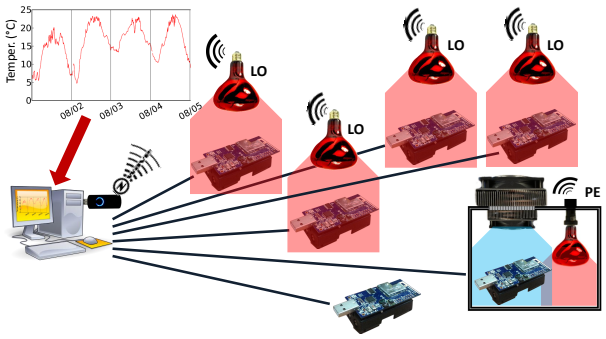


Figure 4: Sketch of TempLab's architecture.

This approach can easily scale to large testbeds as PE and LO nodes only need to be plugged into wall power and require no further cabling. Furthermore, home automation solutions such as Z-Wave allow to connect up to 256 wireless dimmers in a multi-hop fashion, and can can in principle scale to large buildings with many devices. If a very large number of nodes need to be supported, it is possible to partition the control network and use several controllers.

4.1.3 Controller

To instantiate a temperature profile and control heat lamps and Peltier modules, TempLab uses different controllers running on a centralized testbed gateway computer.

Open-loop controller. The simplest one is an open-loop controller that varies the intensity of the light bulbs in LO and PE nodes according to a pre-computed calibration function[1]. This is possible if the impact of each dimming level on the on-board temperature of a node is known based on a previous calibration. In this case, the open-loop controller can instantiate a given profile without further processing. The key advantage of this approach is hence that no sensors are needed to measure the actual temperature of the motes during the experiment. For an accurate replay of temperature

[1]For PE nodes, one can vary the intensity of the heat lamps while the Peltier module is constantly active. As we show in Sect. 5, the IR lamp can change the temperature much quicker than the Peltier module, and a constantly active Peltier module does not slow down the heating from the IR lamp significantly.

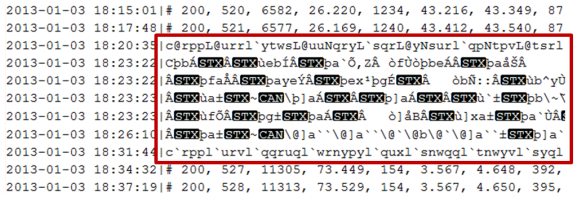


Figure 5: Temporarily unreadable USB serial output in the presence of sudden thermal variations.

dynamics, however, the surrounding environment as found during calibration would need to remain constant, as the controller would not account for external factors influencing temperature such as open windows or sun shining in the room hosting the testbed.

Closed-loop controller. To precisely regenerate trace- or model-based temperature profiles, TempLab uses a closed-loop proportional-integral (PI) controller that tries to minimize the difference between the desired temperature profile and the on-board temperature of the sensor node of interest. The controller should hence receive a periodic feedback with frequency F_U about the on-board temperature of the sensor node in order to minimize the error with respect to the desired temperature profile. The reading of the on-board node temperatures can be carried out either *out-of-band* through the use of an external device or *in-band* using the sensor node itself to measure the temperature and forward it to the controller. As most off-the-shelf wireless sensor nodes carry on-board a temperature sensor, it is very tempting to use an in-band approach to provide up-to-date temperature measurements without adding extra-costs. However, it has to be ensured that system behaviour is not altered. TempLab uses an in-band approach using the USB back-channel to periodically convey temperature readings to the controller. This task is carried out using a low-priority routine executing only when the processor is idle.

During our experiments, we have observed that common USB serial connections used in testbeds for data logging and node programming may be unable to cope with very fast temperature fluctuations, as they result in de-synchronization of the USB sender and receiver. In the presence of such variations, the USB serial port looses synchronization with the mote and the characters forwarded to the USB back-channel become temporarily unreadable, as shown in Fig. 5. Since standard nodes do not handle this issue autonomously, TempLab either re-initializes the USB port or piggybacks the temperature readings onto regular data packets. In this way, other nodes that do not suffer from this issue can report the temperature to the controller over the USB back-channel.

4.2 Implementation

We now describe the hardware and software components that we used to extend our local university testbed based on Maxfor MTM-CM5000MSP nodes (TelosB replicas).

4.2.1 Hardware

In our implementation, we use Philips E27 infra-red 100W light bulbs that can be remotely dimmed using the Z-Wave wireless home automation standard. The latter operates on the 868 MHz ISM band, and hence does not interfere with the communications between the wireless sensor nodes (that use the 2.4 GHz ISM band)[2]. To vary the intensity of the light bulbs, we used Vesternet EVR_AD1422 Z-Wave Everspring wireless dimmers, which provide 100 dimming levels.

LO nodes are only controllable using dimmers. PE nodes have the capability of going below room temperature thanks to enclosures made of hard Polystyrene foam embedding, in addition to the IR heating bulb, an ATA-050-24 Peltier air-to-air assembly module by Custom Thermoelectric. The latter allows on-board temperatures of -5°C when operated at room temperature, and can be controlled through Vesternet EVR_AN1572 Z-Wave Everspring on-off wireless switches. The Polystyrene hard foam isolating box has a minimal impact to the radio propagation of sensor nodes and supports temperatures up to +85°C. The overall hardware cost is €65 for each LO node, and €293 for a PE node.

4.2.2 Software

Actuators. We control the Z-Wave network with a C++ program that uses the Open Z-Wave stack to vary the intensity of dimmers and duty cycle the Peltier modules. Commands to the control network are sent through the Aeon Labs Series 2 USB Controller deployed within the testbed facility.

Controller. Each node runs Contiki, and contains a low-priority process that periodically measures temperature using the on-board SHT11 sensor, and communicates the readings over the USB back-channel. This can be also easily implemented in TinyOS or other operating systems, since it needs only basic building blocks such as reading and outputting temperature. To select the sampling frequency F_U, i.e., how often should the controller receive feedback about the on-board node temperature, we use the fastest temperature variation observed in the outdoor deployment shown in Fig. 2, and compare it to the accuracy of the on-board temperature sensors. In our case, the nodes carry SHT11 sensors that have an accuracy of 0.4°C. According to the profiles shown in Fig. 2, such a variation can be reached within 12 seconds.

The PI controller is implemented as a standalone multi-threaded C++ application executing on the testbed gateway that receives as input a file with two columns: the first one contains the time of the day, the second one describes the on-board temperature that the node should have at that time. The controller is agnostic to the type of trace (whether derived empirically or from a model): as long as the file adheres to the two column format, it will (try to) recreate such tem-

[2]We have also implemented a TempLab version that uses the LightwaveRF standard operating on the 433 MHz ISM band, in case the sensor nodes in the testbed operate on the 868 MHz ISM band. In the rest of the paper we refer to the Z-Wave implementation.

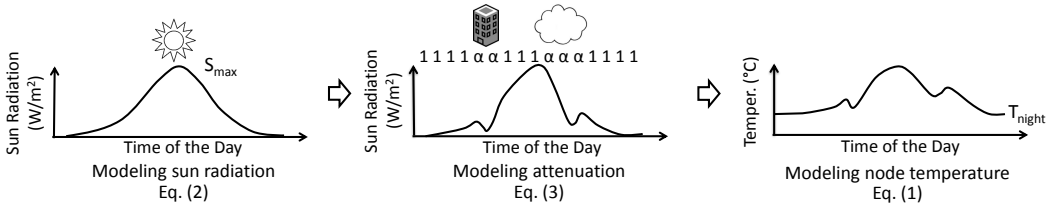


Figure 6: Model-based temperature profile generation.

peratures based on this information and the feedback signals from the motes. In case the user chooses to time-lapse the experiment, the controller skips rows accordingly, e.g., for a 2x speed, the controller skips every other line. We found experimentally that an optimal configuration of the controller is P=2 and I=0.01 to achieve fast and self-stabilizing control.

The controller allows users to manually assign the available traces to the temperature-controlled nodes in the network. If a non-implementable mapping is created, e.g., when mapping a trace containing negative temperatures to a LO node, the controller will signal an error.

4.2.3 Deriving Model-based Temperature Profiles

Using thermodynamic equations we now derive a temperature model suitable to create temperature profiles for nodes. We focus on outdoor deployments where IR radiation from the sun and air temperature are the most significant factors.

Basic thermodynamic equations. In essence, objects heat up by absorbing solar radiation and cool down by constantly releasing energy to their surrounding. The balance between these processes determines the object temperature.

Energy absorption and dissipation. An object that is exposed to the sun, absorbs energy according to: $E_{in} = S\alpha A\Delta t$, where S is the solar radiation, α is the attenuation of the solar radiation, A is the exposed area of the object and Δt is the amount of time exposed to the solar radiation. On the other hand, objects release energy according to: $E_{out} = sT^4A\Delta t$, where s is the Boltzmann constant and T is the temperature of the object in Kelvin.

Energy balance. Considering the energy absorption and energy dissipation of an object, its change of temperature ΔT is determined by the heat energy equation: $H = C_pm\Delta T = E_{in} - E_{out}$, where C_p is the specific heat of the object and m its mass. The temperature of an object cannot be less than air temperature at any given time t (T_t^{air}). Hence, at time $t + \Delta t$, the object temperature is given by:

$$T_{t+\Delta t} = \max\{\ T_t + \frac{(S_t\alpha_t - sT^4)}{C_pm}A\Delta t,\ \ T_t^{air}\ \} \quad (1)$$

Considering a standard mote with parameters m = 50 grams, C_p = 0.5 $\frac{J}{gC}$, A = 20 cm^2; the model only requires the sun radiation S_t, the air temperature T_t^{air} and the attenuation α_t ($0 \leq \alpha_t \leq 1$).

Modelling sun radiation and cloud obstruction. In the absence of any obstructions, the sun radiation throughout the day can be modelled by a gaussian-like shape [13]:

$$\begin{aligned} S_t &= \frac{S_{max}}{\max\{\mathcal{N}(0,\sigma)\}}\frac{1}{\sqrt{2\pi}\sigma}\exp^{-(t-\delta)^2/2\sigma^2} \\ &= S_{max}\exp^{-(t-\delta)^2/2\sigma^2}, 0 \leq t \leq 2\delta \end{aligned} \quad (2)$$

where S_{max} is the maximum sun radiation during the day, and $t = 0$ and $t = 2\delta$ represent the 00 hrs and the 24 hrs. The number of hours with sun light (length of day) can be fine-tuned with σ and δ. To further simplify Eq. 1, instead of considering the air temperature throughout the day (T_t^{air}), we use only the minimum temperature in the day (night temperature T_{min}). Hence, at this point, the only information that we need to model the *clear sky* temperature of a node is the maximum radiation and minimum *air* temperature.

Few locations, however, receive constant sun radiation throughout the day. In most scenarios, clouds block the sun radiation and cause sudden variations of temperature. The length of clouds and the length of the clear sky between clouds are known to have exponential distributions $\lambda\exp^{-\lambda x}, x \geq 0$, with $\frac{1}{\lambda}$ representing the average cloud (or inter-cloud) length [14]. Denoting $\overrightarrow{\alpha}$ as an *attenuation vector* where all elements are α and its length is given by the exponentially random length of a cloud. And denoting $\overrightarrow{1}$ as a *clear-sky vector* where all elements are 1 (i.e. $\alpha = 1$) and its length is equal to the random length of an inter-cloud period; the variable α_t in Eq. 1 is the t^{th} element of the vector:

$$\overrightarrow{v} = \{\overrightarrow{\alpha_1}, \overrightarrow{1_1}, \ldots, \overrightarrow{\alpha_i}, \overrightarrow{1_i}, \ldots\}. \quad (3)$$

At each t in Eq. 1, the t^{th} element is used to capture the amount of sun radiation attenuated during the respective period Δt. The shade of events that are specific to the scenario of interest (trees, buildings, etc), can be included in $\overrightarrow{v}$ by inserting attenuation elements (α) in the vector.

The model can be easily coded using any programming or scripting language. In TempLab, we use Matlab, making sure that the output of the model adheres to the requirements of the PI controller. To compute a temperature value at time t, Eq. 1 is evaluated for the respective value of Δt. Fig. 6 captures the steps followed by the model, and the outcome is a curve similar to the red one shown in Fig. 2, or one with random fluctuations due to shades.

The model allows the user to test a wide range of scenarios. The user can test the worst-case temperature with clear skies, generate shades of any length at any time (to test temperature gradients), and generate random instances for each node by varying the model parameters.

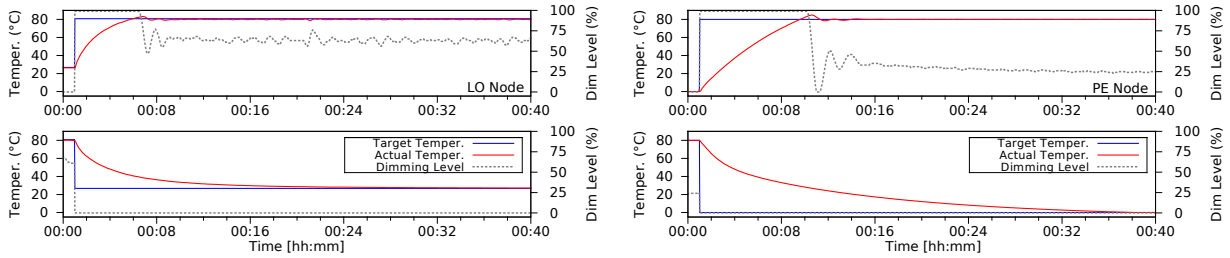


Figure 7: Limits in the speed of heating and cooling for LO and PE nodes.

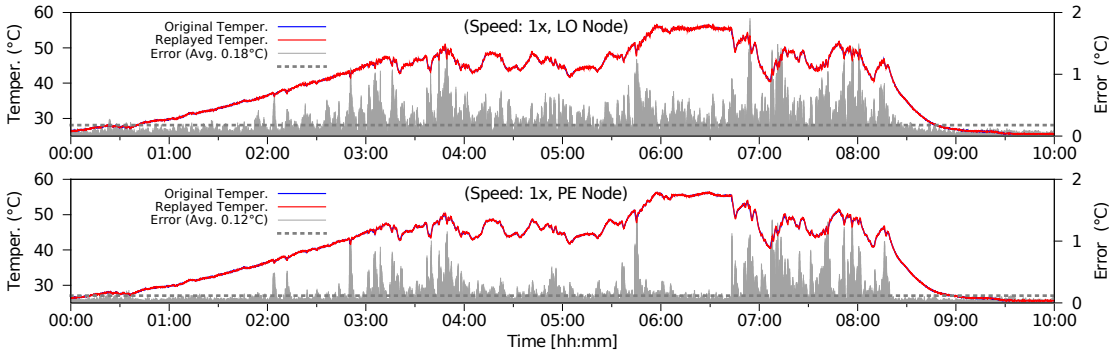


Figure 8: Accuracy of LO and PE nodes in replaying a real-world trace captured during summer.

5. EVALUATION

In this section, we carry out an experimental evaluation of the capabilities of our TempLab implementation. First, we investigate the performance of TempLab in terms of implementable temperature profile dynamics and highlight the limitations on how fast nodes can be heated or cooled. Thereafter, we show that temperature dynamics found in typical deployments can be accurately reproduced despite the low-cost infrastructure, even when compressing the time scale of an experiment to save evaluation time.

5.1 Heating and Cooling Limits

To verify how fast LO and PE nodes can be heated and cooled, we carry out an experiment in which we let the closed-loop PI controller heat the nodes to 80°C. The initial temperature is room temperature for LO nodes and 0°C for PE nodes, respectively. After reaching a stable temperature, the controller cools the nodes down to their original value.

LO nodes. Fig. 7 (left) shows that LO nodes can heat from room temperature (26°C) to 80°C in less than 5 minutes, with an average heating slope of 11.3°C/minute. As LO nodes do not have cooling capabilities, their cooling is rather slow: they need only 7 minutes to decrease from 80°C to 35°C, but they require the same time to decrease from 35°C to 30°C, and 20 more minutes to get back to 26°C.

PE nodes. Fig. 7 (right) shows that PE nodes can heat from 0°C to 80°C in less than 9 minutes, with an average heating slope of 9.3°C/minute. PE nodes are obviously much more efficient in cooling than LO nodes: they need only 6 minutes to decrease from 80°C to 35°C, and 10 minutes to decrease to ambient temperature (26°C). Overall, they can vary the temperature from 80°C to 0°C in less than 35 minutes.

5.2 Regeneration of Traces

We now evaluate TempLab's ability of reproducing a given temperature profile. We compute the accuracy of TempLab by computing how close the instantiated temperature profile P_I follows the given profile to be reproduced P_G. The overall accuracy Q_n of the reproduced temperature profile at node n can be expressed as:

$$Q_n = \frac{1}{T}\int_0^T |\mathrm{P_I(t) - P(t)}|\,\mathrm{d}t \qquad (4)$$

where T is the duration of the experiment. Besides the requirement to follow a temperature profile over time, it is also important to ensure that the rate of temperature changes is reflected accurately. At no point in time the instantiated temperature curve at a node n should deviate too much from the given temperature profile. The maximum deviation q_n can be expressed as:

$$q_n = \max_t |P_I(t) - P(t)| \qquad (5)$$

The smaller the value of Q_n, the better the instantiation of the temperature profile, whereas the smaller q_n, the better the dynamics of the temperature change are reflected.

We take as a reference for our evaluation two temperature traces collected in an outdoor deployment in Sweden [12]: one taken during summer (August), and a "colder" one taken in the end of October, when temperature approaches 0°C.

Summer trace. Fig. 8 shows that both LO and PE nodes can instantiate the desired temperature profile on the sensor

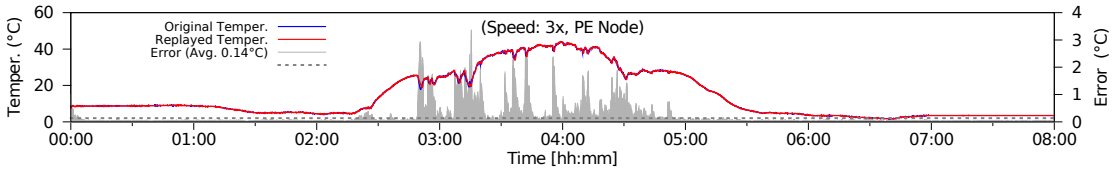


Figure 9: Accuracy of PE nodes in replaying a real-world trace captured during winter.

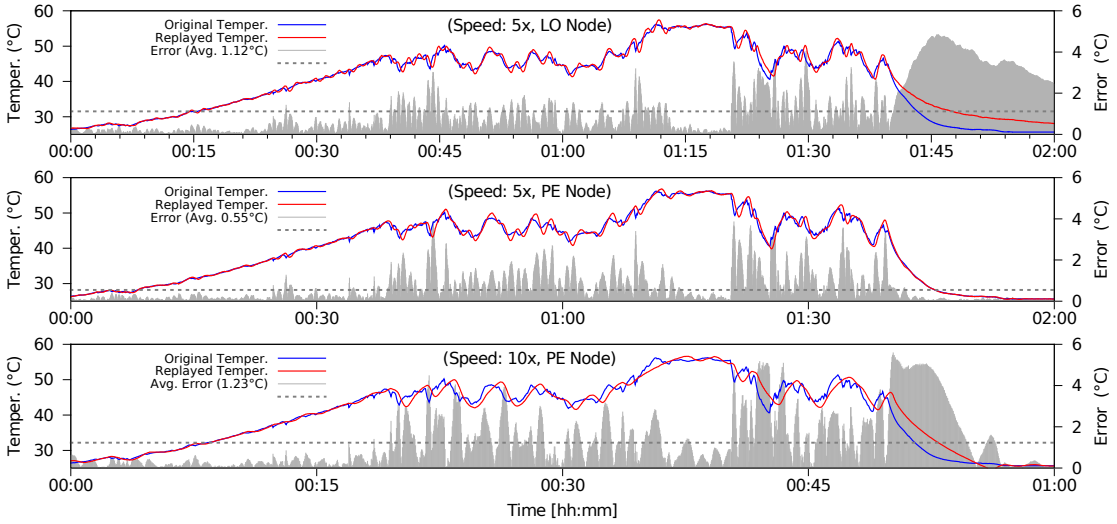


Figure 10: Accuracy of LO and PE nodes when compressing the time-scale of the experiment.

nodes with very high accuracy. The average error Q_n equals 0.18°C and 0.12°C, whereas q_n is 1.90°C and 1.43°C for LO and PE nodes, respectively. This is a remarkable accuracy, and shows that despite the use of low-cost components (LO nodes), TempLab can still reproduce with high accuracy real-world temperature profiles above room temperature.

Winter trace. During winter time, the sun can quickly heat up the temperature in the package hosting the sensor nodes. We replay a trace captured during October 2012 [12], in which the on-board temperature of a node has a significant variation from 45°C during daytime to 0°C in the evening, and see how accurately PE nodes can instantiate this temperature profile on sensor nodes. Fig. 9 shows the results: the average error Q_n equals 0.14°C, whereas $q_n = 3.36$°C.

Accuracy of time-lapsed traces. The accuracy of the replay shown in Fig. 9 is even more remarkable if we consider that we have compressed the original 24-hour trace into 8 hours playback time, i.e., we used a compression factor of 3. We now show the accuracy of LO and PE nodes in the regeneration of traces in which the time has been compressed even further. Fig. 10 shows the results: when instantiating the same trace used in Fig. 8, LO nodes show evident limits due to the lack of cooling capabilities. Compared to the error of 0.18°C when regenerating at normal speed, the average error Q_n raises to 1.12°C when the time is compressed by a factor of 5, whereas Q_n is 0.52°C and 1.90°C when replaying a trace compressed with factor 3 and 10, respectively.

PE nodes, instead, can replay a trace 5 times faster than the original speed with $Q_n = 0.55$°C (the error is halved compared to the LO nodes) and $q_n = 3.84$°C. When compressing time by a factor of 10, however, we can start to observe that the Peltier modules reach their limit, and cannot properly cool down in only 4 minutes what in reality takes 45 minutes. Nevertheless, Q_n is only 1.23°C, and q_n = 5.57°C.

6. TEMPLAB IN ACTION

In this section we present a series of experiments carried out using TempLab. We demonstrate that temperature has a significant impact on processing and protocol performance, and show that TempLab is an ideal tool to investigate these effects. Our aim is not to give a complete solution to the issues that we reveal, but rather to highlight to the community several research challenges that require attention. We believe that TempLab can play a significant role in this emerging research area.

6.1 Testing Processing Performance

Many sensornet applications require a significant amount of on-node processing, so that data is filtered, analysed, or aggregated before being delivered over the network. Heavy processing is often also required for compression, i.e., to reduce the volume of data that has to be transmitted. The processing time required to compress, filter, or analyse data is very significant, as it defines the achievable sampling rate and determines if deadlines can be fulfilled. In [15], the ex-

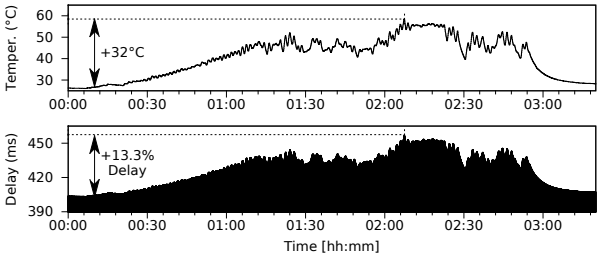


Figure 11: Inter-arrival times follow temperature variations.

ecution of an object detection algorithm requires 240 ms for an image size of 128x128 pixel on an ATmega128 running at 7.3728 MHz. In structural health monitoring application such as [16], accelerometer samples are compressed before transmission, which requires 17.32 ms on a platform employing an MSP430 running at 4 MHz. In medical applications such as ECG monitoring, depending on the algorithm used, the compression of two seconds of ECG data (512 samples) can require up to 580ms [17]. Similarly, applications that require encryption algorithms also require significant processing: software based encryption and authentication of a packet with 56 byte payload requires 17.3 ms on a TelosB platform employing an MSP430 running at 4 MHz [18].

We will show that temperature can have a significant impact on the processing capabilities of a node, and that these execution times may significantly vary when the processing node is deployed outdoors. It is therefore vital to ensure that an application satisfies requirements regarding real-time deadlines and jitter despite the temperature variations found in the target deployment site.

Evaluation using TempLab. We use TempLab to mimic the operations of the class of applications previously discussed. We develop a Contiki application in which data is processed using a fixed-effort of 1 million processor cycles and the result is transmitted to a sink node. To this end, we use Maxfor MTM-CM5000MSP nodes emplying *nullrdc*, a simple MAC protocol without duty-cycling. This makes sure that we avoid protocol-specific effects.

First, we run the application in a testbed without any temperature variation, i.e., we leave the nodes at room temperature. The inter-arrival time of messages at the sink is, on average, 404.35 ms with a tiny variation of 0.88 ms.

Next we assume that the application will be used outdoors, and we expose the processing node to temperature variations. In particular, we use the same summer trace profile used in Sect. 5.2 to mimic a temperature profile to which a node would be exposed during summer. Fig. 11 shows the obtained inter-arrival times when the processing node is cycled through a time-lapsed version of the 10-hour trace. At the (lowest) temperature of 26°C the inter-arrival time is 402.9 ms, whilst at the highest temperature of 58°C an inter-arrival time of 456.5 ms is observed. This represents a change of 13.3% for an increase of 32°C, hence the variation in temperature introduced a significant change.

Closer investigation reveals that processing on nodes requires significantly more time on hot nodes than on cold ones. We use TempLab to test a simple application toggling a GPIO pin after a fixed amount of processor cycles, and recording the time required to complete this amount of work. The anomalous behaviour is indeed caused by the temperature-dependent drift of the processor clock: when temperature is increased from 21 to 54°C, we observe that the processor speed drops by roughly 13%.

Although this outcome is not really surprising, it would not have been possible to verify that the application performance would be largely affected in the expected target area (and assess by how much) when using a standard testbed without temperature control. Using TempLab, the analysis of sensornet performance under varying temperature becomes very simple and helps to identify crucial performance aspects. Although in this paper we do not discuss a solution, TempLab can also be used also to find a solution to the problem and to evaluate its effectiveness, e.g., a periodic recalibration of the processor clock with the temperature stable external crystal.

6.2 Testing Protocol Performance

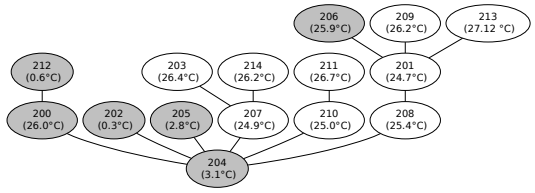
In this section, we use TempLab to highlight the strong impact of temperature on wireless communication, routing topologies, and MAC protocols. Especially relevant when analysing protocol performance is TempLab's ability to generate specific test patterns, as well as the possibility of heating individual nodes (e.g., transmitters-only or receivers-only), which is fundamental to systematically study the impact of temperature on different protocol components.

6.2.1 Impact of Temperature on Routing Protocols

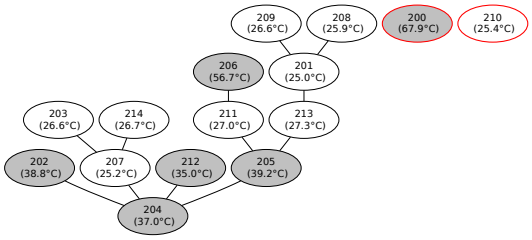
Earlier work has shown that temperature affects the efficiency of low-power wireless radios and hence the quality of links [3]. However, an experimental evaluation of how temperature variations affect network protocols is, to date, still missing. We now use TempLab to show how temperature fluctuations can affect the behaviour of the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [19].

We program fifteen Maxfor MTM-CM5000MSP nodes in our local testbed with a basic Contiki application that uses ContikiRPL [20]. Each node sends a message to the root node (id 204) every minute and logs the transmitted and received packets, as well as the on-board temperature and the Expected Transmission Count (ETX) of the active links. We use TempLab to evaluate the impact that daily fluctuations of temperature can have on the RPL topology with a test pattern that gradually increases the temperature of the designated root node and of one third of the other testbed nodes.

Fig. 12(a) shows a snapshot of the RPL topology at the beginning of the experiment, when nodes are kept at low temperature: all nodes are connected to the sink within a maximum of three hops. Fig. 12(b) illustrates a snapshot of the RPL topology after temperature has increased: temperature-



(a) Topology before heating (at time 00:10)



(b) Topology after heating (at time 01:00)

Figure 12: An increase in temperature can lead to drastic changes in the RPL topology, including a network partition and an increase in network diameter.

controlled nodes are shaded in gray. *The increase in temperature led to drastic changes in the topology of the network, including a network partition and an increase in network diameter*. Nodes 200 and 210 had a direct link to the root node when temperature was low (Fig. 12(a)), but these links are isolated from the network once temperature has increased.

As we have highlighted in Sect. 4.2.3, the presence of direct sunshine on nodes or clouds can quickly vary the on-board temperature of sensor nodes. ContikiRPL attempts to construct a tree by minimizing the ETX sum along the paths to the root. However, ETX changes abruptly with fast temperature changes, especially when packets are exchanged sporadically. In our experiments, we can indeed observe a sudden increase of ETX in links 200 $\rightarrow$ 204 and 210 $\rightarrow$ 204 (Fig. 13), which will lead to a sudden network partition.

These results emphasize the need for techniques that infer the information about the on-board temperature of sensor nodes to the routing layer, so that the most stable tree can be computed before drastic temperature changes [21], for example using the first-order SNR model by Boano et al. [5]. Because of the stochastic nature of the topology formation on RPL (it depends on the *trickle timers* used on nodes to announce DAG information object messages), it is very important to test protocols against several temperature profiles, and TempLab can be a very handy tool to conveniently control and repeat temperature patterns.

6.2.2 *Impact of Temperature on MAC Protocols*

Temperature variations can also drastically affect the performance of medium access control protocols. It is not difficult to envision that protocols relying on tight time synchronization, such as the ones based on time-division multiple

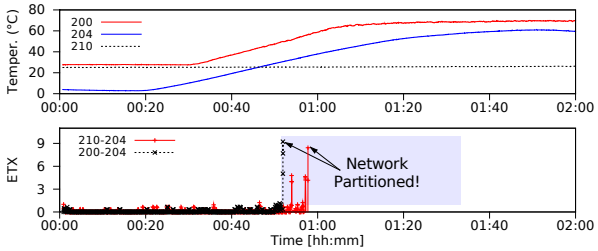


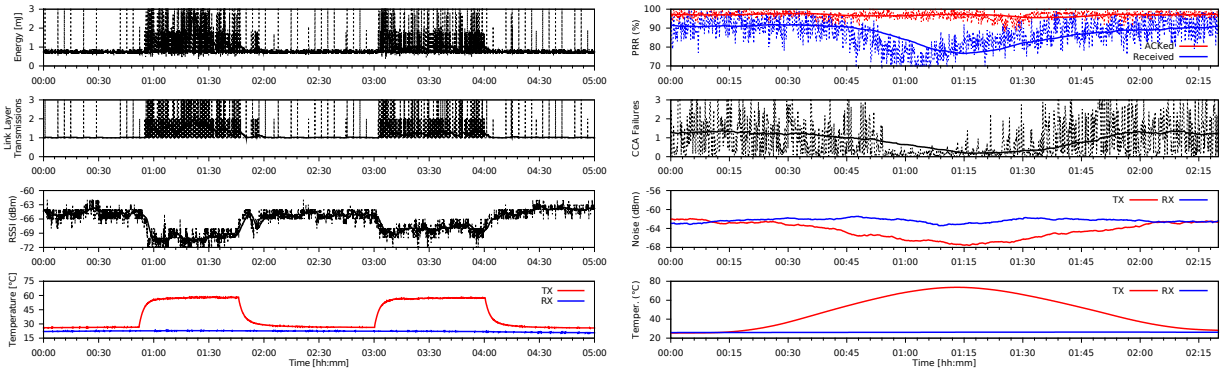
Figure 13: Sudden rise of ETX while temperature increases.

access (TDMA) schemes [22] [23], can be vulnerable to sudden temperature variations across the network due to clock drifts and slowdown of micro-controllers. In the context of TDMA protocols, TempLab can be used to experimentally find the optimal value for critical parameters such as slot size, guard time, and re-synchronization frequency, so that protocols can operate reliably despite challenging temperature variations that can occur at the final deployment site.

Less obvious is the fact that also the performance of carrier-sense multiple access (CSMA) protocols degrades because of temperature variations. In this section, we use TempLab to provide experimental evidence that *CSMA protocols may reduce their efficiency when operating at high temperatures*. We carry out experiments consisting of several transmitter-receiver pairs of Maxfor MTM-CM5000MSP nodes running a basic Contiki application, in which the transmitter node periodically sends packets to its intended receiver and collects statistics such as the overall energy expenditure at the link-layer and the noise in the radio channel. We compute the noise as the maximum of 20 consecutive cc2420_rssi() readings after the transmission of a packet. Receivers acknowledge the message reception and measure the RSSI of received packets as well as the noise in the channel after packet reception. We select ContikiMAC, Contiki's default MAC protocol, and use TempLab with a test pattern that progressively heats the transmitter while keeping the receiver at constant temperature.

Fig. 14(a) (top) shows that the overall energy spent at the link layer to successfully transmit a packet increases at high temperatures, as a result of an increased amount of link-layer transmissions, a behaviour that was found in several (but not all) transmitter-receiver pairs. The signal strength was sufficiently high for all links, therefore the impact that we observe is not connected to the decrease in signal strength at high temperatures observed in [3], [5]. The only difference among different pairs of nodes was the radio channel used for communication: each transmitter-receiver pair was assigned a different (orthogonal) channel. As the experiment was carried out in an indoor office testbed with several Wi-Fi access points, we can connect the increase of link-layer transmissions to the presence of interference in specific channels. However, we only notice an impact at high temperatures.

Further investigation led us to the identification of the problem: *the increase in link-layer transmissions was caused*



(a) Impact on MAC efficiency

(b) Impact on CCA efficiency

Figure 14: Impact of temperature on CSMA-based MAC protocols: the energy expenditure increases as well as the amount of link-layer transmissions at high temperatures, due to a reduced efficiency of CCA leading to a higher packet loss rate.

by a reduced efficiency of the clear channel assessment (CCA) operation at high temperatures. Fig. 14(b) shows that the strength of the measured noise at the transmitter decreases when temperature increases (whereas it remains constant at the receiver). As highlighted in [5], the radio's received power decreases at high temperatures, and so does the measured signal strength. This implies that a source of noise in the environment will be perceived as "weaker" by a heated node, i.e., the transmitter erroneously measures a weaker noise in the environment as a result of the increased temperature. CCA algorithms are typically based on a fixed threshold T_{CCA} below which the channel is considered clear (e.g., in the CC2420 radio, T_{CCA} is set by default to -77 dBm). At high temperatures, the strength of the measured noise decreases, and there are hence higher chances that it falls below T_{CCA}, leading to a "clear channel" and a consequent packet transmission. If this happens, there is a likelihood that the transmitted packets are going to be destroyed or corrupted by interference, and our experiments confirm this observation. We run the same experiment using Contiki's nullrdc (to avoid protocol-specific behaviours) and observe that the amount of CCA failures decreases at high temperatures, leading to a substantial loss of packets. Fig. 14(b) shows that up to 25% of the packets were lost as a result of wrong clear channel assessments.

While the loss rate would be even higher in interfered scenarios, it is important to highlight that *protocols that adapt CCA thresholds [24], [25], would be even more vulnerable to this issue*, as they would lower T_{CCA} when temperature increases as a result of the radio's decreased received power.

7. RELATED WORK

Traditionally, the wireless sensor networks research community relies on testbed facilities to evaluate and tune newly developed methods, protocols, and applications under realistic conditions in a cost-effective way. A large number of publicly available testbeds has been developed in the last decade, where registered users can typically upload the specifications of an experiment and collect traces directly via a web interface. Examples are MoteLab [7], Kansei [9], Indriya [26], TWIST [8], and NetEye [10].

The capabilities of testbeds have constantly evolved in the last years. Focus has been on reducing their management effort [27], allocating testbed resources to users that need them the most [28], accurately analysing the power consumption [29], improving data presentation and analysis [30], as well as on confederating multiple testbeds [31].

As the accuracy of a testbed experiment largely depends on how accurately environmental effects can be reproduced, recent efforts have looked at extending existing infrastructures with the emulation of environmental effects such as radio interference and mobility of nodes [32], [33], [34]. For example, in JamLab, Boano et al. [32] have added the ability to reproduce realistic interference patterns within a testbed without the need to add additional hardware equipment. In ViMobiO [33], Puccinelli and Giordano implemented a virtual mobility overlay to reproduce movement patterns of nodes during experimental evaluation.

One crucial environmental property, however, has not received significant attention in the community even though it can dramatically affect the performance of wireless sensor networks: temperature. A few works have reported the degradation of packet loss rate [35], signal strength [4], and link quality [12] as a consequence of an increase in ambient temperature, based on observations in real-world deployments or outdoor testbed facilities. Outdoor testbed facilities, however, do not allow to systematically analyse the impact of temperature [36], [37]. First, meteorological conditions cannot be controlled, making it impossible to ensure repeatability across several experiments. Second, the temperature profiles that can be tested are highly specific to the deployment location and to the time of the year in which the experiment is carried out.

Bannister et al. [3] have attempted to quantify the loss in

received signal strength between a pair of nodes using a temperature chamber, but did not have the possibility to carry out experiments on a larger scale. Experimenting inside thermal chambers is indeed extremely costly and targets only individual components and not a network of nodes with different individual temperatures (which is necessary to disclose limitations at the network level).

TempLab aims to solve the outlined shortcomings and provides the research community with a testbed capable of reproducing real-world temperature profiles.

8. SUMMARY AND OUTLOOK

The central tenet of our study is that the important role played by environmental temperature in the performance of sensor networks can (and must) be analysed in a systematic way. To achieve this goal, we have designed and implemented TempLab, an extension for wireless sensor network testbeds that allows to vary the on-board temperature of sensor motes and study the effects of temperature variations on the network performance in a precise and repeatable fashion. We have shown that TempLab can accurately reproduce traces recorded in outdoor environments with an average error of only 0.1°C, and demonstrated that temperature has a significant impact on processing and protocol performance. Hence, we believe that TempLab can play an important role in studying the effects of temperature variations on the performance of wireless sensor networks, as it can reveal system limitations that would not have been visible when experimenting with existing testbed installations.

9. REFERENCES

[1] T. Schmid. *Time in Wireless Embedded Systems*. PhD thesis, University of California, 2009.

[2] C. Park, K. Lahiri, and A. Raghunathan. Battery discharge characteristics of wireless sensor nodes: An experimental analysis. In *Proc. of the* 2^{nd} *SECON Conf.*, 2005.

[3] K. Bannister et al. Wireless sensor networking for hot applications: Effects of temperature on signal strength, data collection and localization. In *Proc. of the* 5^{th} *HotEmNets Worksh.*, 2008.

[4] C.A. Boano, J. Brown, N. Tsiftes, U. Roedig, and T. Voigt. The impact of temperature on outdoor industrial sensornet applications. *IEEE Trans. Ind. Informatics*, 6(3), 2010.

[5] C.A. Boano et al. Hot Packets: A systematic evaluation of the effect of temperature on low power wireless transceivers. In *Proc. of the* 5^{th} *ExtremeCom Conf.*, 2013.

[6] Hewlett Packard. Fundamentals of quartz oscillators. Application Note 200-2, 1997.

[7] G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: a wireless sensor network testbed. In *Proc. of the* 4^{th} *IPSN Conf.*, 2005.

[8] V. Handziski, A. Köpke, A. Willig, and A. Wolisz. TWIST: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *Proc. of the* 2^{nd} *RealMAN Worksh.*, 2006.

[9] E. Ertin, A. Arora, R. Ramnath, M. Sridharan, and V. Kulathumani. Kansei: A testbed for sensing at scale. In *Proc. of the* 5^{th} *IPSN Conf.*, 2006.

[10] X. Ju, H. Zhang, and D. Sakamuri. NetEye: A user-centered wireless sensor network testbed for high-fidelity, robust experimentation. *Int. J. Commun. Syst.*, 25(9), 2012.

[11] Abhishek Chattopadhyay. Basic RF testing of CCxxxx devices. Application Report SWRA370, 2011.

[12] H. Wennerström et al. A long-term study of correlations between meteorological conditions and 802.15.4 link performance. In *Proc. of the* 10^{th} *SECON Conf.*, 2013.

[13] F.O. Hocaoğlu, Ö.N. Gerek, and M. Kurban. Hourly solar radiation forecasting using optimal coefficient 2-D linear filters and feed-forward neural networks. *Solar Energy*, 82(8), 2008.

[14] D.E. Lane, K. Goris, and R.C.J. Somerville. Radiative transfer through broken clouds: Observations and model validation. *Journal of Climate*, 15(20), 2002.

[15] M. Rahimi, R. Baer, O.I. Iroezi, J.C. Garcia, J. Warrior, D. Estrin, and M. Srivastava. Cyclops: In situ image sensing and interpretation in wireless sensor networks. In *Proc. of the* 3^{rd} *SenSys Conf.*, 2005.

[16] M. Ceriotti et al. Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment. In *Proc. of the* 9^{th} *IPSN Conf.*, 2009.

[17] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst. Compressed sensing for real-time energy-efficient ECG compression on wireless body sensor nodes. *IEEE Trans Biomed Eng*, 58(9), 2011.

[18] I.E. Bagci, S. Raza, T. Chung, U. Roedig, and T. Voigt. Combined secure storage and communication for the internet of things. In *Proc. of the* 10^{th} *SECON Conf.*, 2013.

[19] T. Winter et al. RPL: IPv6 routing protocol for low-power and lossy networks. Technical report, IETF, 2012.

[20] N. Tsiftes, J. Eriksson, and A. Dunkels. Low-power wireless IPv6 routing with ContikiRPL. In *Proc. of the* 9^{th} *IPSN Conf.*, 2010.

[21] C. Keppitiyagama et al. Temperature hints for sensornet routing. In *Proc. of the* 11^{th} *SenSys Conf., poster session*, 2013.

[22] L.F.W. van Hoesel and P.J.M. Havinga. A lightweight medium access protocol (LMAC) for WSN. In *Proc. of the* 1^{st} *INSS Worksh.*, 2004.

[23] P. Suriyachai, J. Brown, and U. Roedig. Time-critical data delivery in wireless sensor networks. In *Proc. of the* 6^{th} *DCOSS Conf.*, 2010.

[24] M. Sha, G. Hackmann, and C. Lu. Energy-efficient low power listening for wireless sensor networks in noisy environments. In *Proc. of the* 12^{th} *IPSN Conf.*, 2013.

[25] W. Yuan, J.-P.M.G. Linnartz, and I.G.M.M. Niemegeers. Adaptive CCA for IEEE 802.15.4 wireless sensor networks to mitigate interference. In *Proc. of the IEEE WCNC Conf.*, 2010.

[26] M. Doddavenkatappa, M. Chan, and A.L. Ananda. Indriya: A low-cost, 3D wireless sensor network testbed. In *Proc. of the* 7^{th} *TridentCom Conf.*, 2011.

[27] R. Crepaldi, S. Friso, A. Harris, M. Mastrogiovanni, C. Petrioli, M. Rossi, A. Zanella, and M. Zorzi. The design, deployment, and analysis of SignetLab: A sensor network testbed and interactive management tool. In *Proc. of the* 3^{rd} *TridentCom Conf.*, 2007.

[28] B.N. Chun, P. Buonadonna, A. AuYoung, C. Ng, D.C. Parkes, J. Shneidman, A.C. Snoeren, and A. Vahdat. Mirage: A microeconomic resource allocation system for sensornet testbeds. In *Proc. of the* 2^{nd} *EmNetS Worksh.*, 2005.

[29] I. Haratcherev, G. Halkes, T. Parker, O. Visser, and K. Langendoen. PowerBench: A scalable testbed infrastructure for benchmarking power consumption. In *Proc. of the* 1^{st} *IWSNE Worksh.*, 2008.

[30] A.R. Dalton and J.O. Hallstrom. A file system abstraction and shell interface for a wireless sensor network testbed. In *Proc. of the* 3^{rd} *TridentCom Conf.*, 2007.

[31] I. Chatzigiannakis, S. Fischer, C. Koninis, G. Mylonas, and D. Pfisterer. WISEBED: An open large-scale wireless sensor network testbed. In *Proc. of the* 1^{st} *Sensappeal Conf.*, 2009.

[32] C.A. Boano, T. Voigt, C. Noda, K. Römer, and M.A. Zúñiga. JamLab: Augmenting sensornet testbeds with realistic and controlled interference generation. In *Proc. of the* 10^{th} *IPSN Conf.*, 2011.

[33] D. Puccinelli and S. Giordano. ViMobiO: Virtual mobility overlay for static sensor network testbeds. In *Proc. of the* 4^{th} *EXPonWireless Worksh.*, 2009.

[34] D. Johnson, T. Stack, R. Fish, D.M. Flickinger, L. Stoller, R. Ricci, and J. Lepreau. Mobile Emulab: A robotic wireless and sensor network testbed. In *Proc. of the* 25^{th} *INFOCOM Conf.*, 2006.

[35] J. Sun and R. Cardell-Oliver. An experimental evaluation of temporal characteristics of communication links in outdoor sensor networks. In *Proc. of the* 2^{nd} *RealWSN Worksh.*, 2006.

[36] P. Dutta et al. Trio: Enabling sustainable and scalable outdoor WSN deployments. In *Proc. of the* 5^{th} *IPSN Conf.*, 2006.

[37] R. Lim et al. FlockLab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In *Proc. of the* 12^{th} *IPSN Conf.*, 2013.