

# WSAN4CIP

## Deliverable D4 . 2

### Specification and testing of elementary dependable services

Editor:	Claude Castelluccia
Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	31 August 2010
Actual delivery date:	31 August 2010
Suggested readers:	Research community
Version:	1.0
Total number of pages:	22
Keywords:	elementary dependable services, clock skew, fingerprinting, covert channels, security.

---

#### Abstract

This deliverable describes services that are based on the clock skew to identify and detect anomalies in wireless sensor networks.

The clock skew, an inherent property of clock crystals of physical devices, is defined as the rate of deviation of a device clock from the true time. The frequency of a device's clock actually depends on its environment, such as the temperature, humidity, vibration, electromagnetic interference, as well as the type of crystal. The main contributions of our work are two-fold. First, we experimentally validate that MICAz and TelosB sensor motes have different and unique clock skews. Furthermore, the clock skew of a node can easily be monitored, even via a multi-hop Wireless Sensor Network (WSN). We argue that this feature can be used for identification of the nodes, detection of Wormhole and Sybil attacks. Second, we show that the clock skew of a sensor node varies with the temperature. We explain how this property can be used to detect malicious and mal-functioning nodes and to geo-localize them.

Using 7 MICAz and 2 TelosB motes we performed extensive experiments to validate our approaches.

Note that this deliverable describes only the elementary security service, namely *Skew-based service*, that was implemented and provided as input to WP5. As described in D4.1, several other elementary security services have been designed within task 4.1 of the WSAN4CIP project. Since they will not be implemented they are not reported in this deliverable (but are partially described in D4.1).

---

---

**Disclaimer**

---

This document contains material, which is the copyright of certain WSAN4CIP consortium parties, and may not be reproduced or copied without permission.

*In case of Public (PU):*

All WSAN4CIP consortium parties have agreed to full publication of this document.

**Impressum**

Wireless Sensor Networks for Protection of Critical Infrastructures

WSAN4CIP

WP4 “Service Protection”

Specification and testing of elementary dependable services

[Editor: Name, company] Claude Castelluccia, INRIA

**Copyright notice**

©2009 Participants in project WSAN4CIP

## Executive summary

This deliverable contains the specification and the analysis of modules for elementary modules for dependable services. This deliverable will be input to Deliverable D4.3 as well as work package WP5.

Note that this deliverable describes only the elementary security service, namely *Skew-based service*, that was implemented and provided as input to WP5. As described in D4.1, several other elementary security services have been designed within task 4.1 of the WSAN4CIP project. Since they will not be implemented they are not reported in this deliverable (but are partially described in D4.1).

This deliverable describes services that are based on the clock skew to identify and detect anomalies in wireless sensor networks.

The clock skew, an inherent property of clock crystals of physical devices, is defined as the rate of deviation of a device clock from the true time. The frequency of a device's clock actually depends on its environment, such as the temperature, humidity, vibration, electromagnetic interference, as well as the type of crystal. The main contributions of our work are two-fold. First, we experimentally validate that MICAz and TelosB sensor motes have different and unique clock skews. Furthermore, the clock skew of a node can easily be monitored, even via a multi-hop Wireless Sensor Network (WSN). We argue that this feature can be used for identification of the nodes, detection of Wormhole and Sybil attacks. Second, we show that the clock skew of a sensor node varies with the temperature. We explain how this property can be used to detect malicious and mal-functioning nodes and to geo-localize them.

Using 7 MICAz and 2 TelosB motes we performed extensive experiments to validate our approaches.

**List of authors**

<b>Company</b>	<b>Author</b>
INRIA	Claude Castelluccia
INRIA	Daniele Perito

## Contents

<b>Executive summary</b>	<b>3</b>
<b>List of authors</b>	<b>4</b>
<b>1 Introduction</b>	<b>8</b>
<b>2 Clocks On Wireless Sensor Motes</b>	<b>8</b>
<b>3 Clock Skew and Temperature</b>	<b>9</b>
3.1 Definitions . . . . .	9
3.2 Estimation of Clock Skew . . . . .	9
<b>4 Background and Related Work</b>	<b>10</b>
<b>5 Experimental Setup</b>	<b>11</b>
5.1 Single-hop WSN Setup . . . . .	11
5.2 Multiple-hop WSN Setup . . . . .	11
5.3 Programming the motes with nesC TinyOS . . . . .	12
5.4 Setup for Clock Skew Measurement in Constant Temperature . . . . .	12
5.5 Setup for Clock Skew Measurement with Varying Temperature . . . . .	12
<b>6 Experiments and Results</b>	<b>13</b>
6.1 Estimation of Clock skew at constant Temperatures . . . . .	14
6.2 Analysing the Effect of Temperature on Clock Skews . . . . .	16
<b>7 Potential Applications</b>	<b>18</b>
<b>8 Conclusion and Future Work</b>	<b>19</b>
<b>9 Summary of Deliverable</b>	<b>20</b>

## List of Figures

1	Timing diagram showing equal delays (i.e., $d_1 = d_i$ ) . . . . .	10
2	Single-hop WSN Setup with 7 MICAz motes. Node 1 is the sink node connected with computer through MIB510 serial interface. Nodes 2-7 are acting as end-nodes and sending packets to sink over the wireless channel. . . . .	11
3	Multiple-hop WSN Setup with 7 MICAz (nodes 1-7) and 2 TelosB motes (nodes 8-9). Node 1 is the sink node. Nodes 2 and 3 are forwarding nodes on WSN and remaining nodes are end-nodes. . . . .	12
4	Clock skew estimation on room temperature in single-hop WSN using Microsecond level TimeStamps. Colored points indicate offset values of corresponding nodes and the black straight lines above the offset points indicate the estimated clock skews for the corresponding nodes . . . . .	14
5	Clock skew estimation on room temperature in single-hop WSN using 32kHz level TimeStamps	14
6	Clock skew of MICAz Motes in Multi-hop WSN using 32kHz level TimeStamps . . . . .	15
7	Estimated clock skew of MICAz and TelosB Motes in multiple-hop WSN using 32kHz level TimeStamps. Nodes 4-7 are MICAz motes and nodes 8-9 are TelosB motes. . . . .	15
8	Estimated clock skew of 4 MICAz motes in multiple-hop WSN while varying the temperature (from 4-60 degree centigrade). Colored points show the offset values of corresponding motes and black straight lines above the offset values show the estimated clock skews . . . . .	16
9	Estimation of Constant Clock Skew from the offset values. Green dots indicate offset values and Upper black straight line shows the estimated clock skew . . . . .	17
10	Denoising of Variable Skew using a total of 8 windows and 3 passes . . . . .	17
11	Estimated Drift (pink dashed line) and Measured Temperature (red line) . . . . .	18
12	Estimated Drift (pink dashed line) of a MICAz mote (upper graph) matches with measured temperature (red line) and Drift of a TelosB (lower graph) mote matches with temperature for higher than room temperature but for lower temperature it is overcompensated. . . . .	18

## List of Tables

1	Estimated Clock Skew using 32kHz and Microsecond level timestamps for single-hop WSN using MICAz motes as of setup Figure 2 . . . . .	15
2	Estimated Clock Skews (using 32kHz level timestamps) of multiple hops MICAz and TelosB motes as of setup Figure 3 . . . . .	16

## 1 Introduction

Clock skew has been used for remote fingerprinting of hosts on computer networks [10]. Variation of clock skew, i.e. clock drift, with respect to temperature has been used for revealing hidden services on Tor network [17]. However, this technique was never used in the context of wireless sensor nodes yet.

In this work, we performed several experiments and measured the clock skew of several wireless sensor nodes. We found out that different motes have different and unique clock skews. These clock skews are different enough to be used to fingerprint sensors.

Clock skew based fingerprinting has some advantages over MAC address based fingerprinting of wireless sensor motes. For example, MAC address can be easily spoofed [23] while the clock skew of a node is very difficult to alter.

We also show that the Clock skew of a node varies with temperature [13]<sup>1</sup>. The variation of the clock skew (a.k.a. clock drift) increases when the temperature rises and decreases when temperature lowers. Clock drift is therefore a linear function of the temperature, and can be used to estimate the temperature of a node. This has two possible applications. First, the clock skew can be used to evaluate the temperature of the node's environment, and possibly geo-localize it. Second, it can be used, in WSNs that are deployed to measure temperatures (to detect fire for example), to verify that the temperature reported by a node is consistent with the clock skews. This could be useful to detect lying or malfunctioning nodes.

**Our Contributions.** We verified that clock skew based fingerprinting of sensor nodes can be successfully performed on WSN. Clock skew based mote identification can be utilized for detecting Wormhole and Sybil attacks. We also show that clock skew *variations*, i.e. clock drifts, can be used to detect malicious or malfunctioning mote. Using 7 MICAz and 2 TelosB motes, we created several single-hop and multiple-hop WSNs and validated our techniques.

**Organization.** The rest of the report is organized as follows. In section 2 we discuss about the implementation of clocks on wireless sensor motes. In Section 3 we details how the time skew can be computed. Section 4 reviews the related work. In Section 5, we describe our experimental setup. Section 6 presents the results. In section 7, we discuss the possible applications of our results. Finally, Section 8 concludes this report.

## 2 Clocks On Wireless Sensor Motos

Clocks of available wireless sensors are either implemented in hardware or in software. A hardware clock consists of an oscillator that ticks at a nominal frequency and a counter that records the number of ticks. Microcontroller oscillators are of four different types- Crystal, Ceramic Resonator, RC(Resistor, Capacitor) and Silicon Oscillators. Among these oscillators, crystal oscillators and ceramic resonators are the most accurate and the cheapest. However, they have the disadvantage of being sensitive to environmental conditions as electromagnetic interference (EMI), vibration, temperature, humidity etc.

All of the commercially available wireless sensor motes are based on low power microcontrollers: MICA and IRIS<sup>2</sup> family motes are based on Harvard architecture Atmel ATmega128 AVR microcontroller, Telos family motes are based on Texas Instruments MSP430 microcontroller and Intel Imote2<sup>3</sup> family motes are based on Intel PXA271 microcontrollers<sup>4</sup>. From the datasheets of all the above mote families and their microcontrollers, it has been found that all of the microprocessors recommended external crystal oscillators for their timer circuit and all the motes use external crystal oscillators in their timer circuit.

The actual clock frequency of a mote depends on the environmental conditions (as electromagnetic interference-EMI, vibration, temperature, humidity etc.) and properties of the clock crystals.

Software-based clocks are implemented in the operating system installed on the motes. Almost all of the commercially available motes support open source TinyOS<sup>5</sup> operating system. TinyOS Extension Proposal

<sup>1</sup>However, this variation does not impact fingerprinting.

<sup>2</sup>IRIS Datasheet. Available at [http://www.xbow.com/Products/Product\\_pdf\\_files/Wireless\\_pdf/IRIS\\_Datasheet.pdf](http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/IRIS_Datasheet.pdf)

<sup>3</sup>Imote2 Datasheet. Available at [http://www.xbow.com/Products/Product\\_pdf\\_files/Wireless\\_pdf/Imote2\\_Datasheet.pdf](http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2_Datasheet.pdf)

<sup>4</sup>Crossbow Technology Inc. Wireless Sensor Networks. [http://www.xbow.com/Products/Wireless\\_Sensor\\_Networks.htm](http://www.xbow.com/Products/Wireless_Sensor_Networks.htm)

<sup>5</sup>TinyOS: An open-source OS for the networked sensor regime. <http://www.tinyos.net/>

(TEP) 102 [21] includes an implementation of different clocks and timer counter components of different ranges and frequencies.

A software clock is essentially provided by multiplying the hardware counter by a factor that depends on the desired clock frequency. For speed and performance reasons, this factor is often equal to some power of 2, and therefore introduces some errors in the software clock. For example, the clock crystal of a TelosB mote has a frequency of 32768 Hz. To obtain a millisecond level clock, the frequency is divided by 32(= $2^5$ ); thus on a TelosB mote one second corresponds actually to 1024 milliseconds (i.e. an error of 24 milliseconds is introduced every second). Similarly, a MICAz mote have a crystal frequency of 28800Hz; and thus one seconds corresponds actually to 900 milliseconds (i.e. an error of 100 milliseconds is introduced every second ).

### 3 Clock Skew and Temperature

#### 3.1 Definitions

A clock is a piecewise continuous function that is twice differentiable. If  $C(t)$  is the time reported by a clock at time  $t$ , there exists  $C'(t) \equiv dC(t)/dt$  and  $C''(t) \equiv d^2C(t)/dt^2$  for  $t \geq 0$ .

We used the following nomenclature from [16] to describe the clock characteristics. Let  $C_1$  and  $C_2$  be the two clocks:

- **offset:** the difference between the time reported by two clocks. If time at two clocks  $C_1$  and  $C_2$  are  $C_1(t)$  and  $C_2(t)$  respectively, the offset of the clock  $C_1$  relative to clock  $C_2$  at time  $t \geq 0$  is  $C_1(t) - C_2(t)$ .
- **frequency:** the rate at which the clock progresses. The frequency at time  $t$  of  $C_1$  is  $C_1'(t)$ .
- **clock ratio ( $\alpha$ ):** the frequency ratio between two clocks. The ratio of  $C_1$  relative to  $C_2$  at time  $t$  is  $\alpha = C_1'(t)/C_2'(t)$ .
- **skew ( $\delta$ ):** the difference in the frequencies of two clocks. The skew of  $C_1$  relative to  $C_2$  at time  $t$  is  $\delta = C_1'(t) - C_2'(t) = \alpha \times C_2'(t) - C_2'(t) = (\alpha - 1) \times C_2'(t)$ .
- **drift:** the drift of clock  $C_1$  relative to the clock  $C_2$  at time  $t$  is  $C_1''(t) - C_2''(t)$ .

The clock skew is the difference between the clock frequencies of two clocks. When two clocks run at different frequencies, the same physical time duration measured by the two clocks will be different. When a delay measurement involves two clocks, the synchronization between those clocks has a tremendous impact on the accuracy of the measurement. For example, let us consider the case of measuring a packet delay between two nodes of WSN. The sending node inserts a timestamp to a packet, and the receiving node appends its receiving timestamp upon its reception. If the clocks of the two nodes are perfectly synchronized, the difference between the two timestamps is the end-to-end delay that the packet experienced in the WSN. If the clocks on the two nodes have a non-zero offset, but no skew, the difference between the two timestamps will be the sum of the end-to-end delay and the offset. In a one-way delay measurement, the offset cannot be distinguished from the measurement. If the clocks have a non-zero skew, not only is the end-to-end delay measurement off by an amount equal to the offset, but it also gradually increases or decreases over time depending on whether the sender clock runs slower or faster than the receiver's clock.

#### 3.2 Estimation of Clock Skew

Let us assume that measurer obtained a trace of  $N$  packets having the sending timestamps of an end-node and receiving timestamps of the sink node. Let us assume (as Figure 1) that:

- Sender(end-node) clock  $C_s$  runs at frequency  $H z_s$ .
- Receiver (sink node) clock  $C_r$  runs at frequency  $H z_r$ .

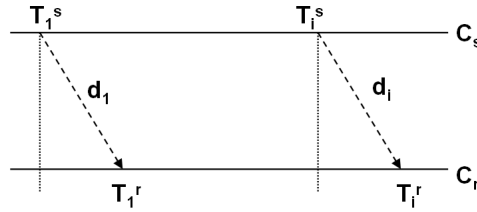


Figure 1: Timing diagram showing equal delays (i.e.,  $d_1 = d_i$ )

- Sending and receiving timestamps of 1st packet are  $T_1^s$  and  $T_1^r$ .
- Sending and receiving timestamps of  $i$ -th packet are  $T_i^s$  and  $T_i^r$ .
- Propagation delays of 1st and  $i$ -th packets  $d_1$  and  $d_i$  are equal (i.e.,  $d_1 = d_i$ ).
- Elapsed sending time of  $i$ -th packet measured by  $C_s$  is  $t_i^s = (T_i^s - T_1^s)/Hz_s$ .
- Elapsed receiving time of  $i$ -th packet measured by  $C_r$  is  $t_i^r = (T_i^r - T_1^r)/Hz_r$ .
- Offset of  $i$ -th packet  $\tilde{o}_i = t_i^s - t_i^r$ . Now, we have  $N$  pairs of  $(t_i^r, \tilde{o}_i)$  for  $i \in \{1, \dots, N\}$ .

As in [10], we estimate the skew, by drawing a line that goes above all offset measurements and minimizes the sum of errors, distances between the line and the offset points. This line is obtained by applying the linear programming algorithm described in [16]. This algorithm finds an estimate of the upper bounds of the linear offset components-  $\hat{o}(t) = \hat{\delta} \times t + \gamma$  that minimize the following expression:

$$\frac{1}{N} \sum_{i=1}^N (\hat{o}(t_i^r) - \tilde{o}_i)$$

where  $\hat{\delta}$  is the estimate of the skew.

## 4 Background and Related Work

There exists a significant amount of prior work on clock skew-based host identification in computer networks. However, no prior work exists in the context of WSNs.

It has long been known that seemingly identical computers have different clock skews and clock skew varies with temperature. The NTP [15] protocol specifies a method to reduce clock skews by performing periodic synchronizations with time servers.

[19] initiated a line of research geared toward eliminating clock skew from network measurements. The linear programming algorithm for clock skew measurement we use is actually based on a descendent of the Paxson paper[16]. The linear time algorithm of the above linear programming approach is solved by the linear time linear programming algorithms of [2] and [14].

[10] used timing information from remote computers to fingerprint their physical identity. By examining timestamps from different machines they estimated the clock skews of the machines, showed that different machines have different clock skews and it might be a viable means of identifying physical machines based on clock skew.

[17] first showed clock skew varies with variation of temperature on network hosts. He used this property to identify hidden services in Tor network. He showed that three collaborating servers providing a hidden service on Tor network can be easily identified by receiving timestamps from the candidate servers. As three servers in Tor networks sharing the same service will have similar load, and therefore experience similar temperature changes, they will all experience similar changes in their clock skew as well. Therefore three servers serving the same hidden service in Tor network can be identified. Their anonymity can then be compromised.

[24] devised an improved clock skew measurement technique based on synchronized sampling of clocks. He shows that synchronized sampling of clocks reduces quantization errors and makes the estimated clock skew more accurate. Synchronized sampling also helps measuring the clock skew for low resolution timestamps (e.g., HTTP servers having Hz level clocks thus timestamps in second level resolution).

[9, 1] explored the feasibility and reliability of using clock skew to fingerprint wireless local area network Access Points (AP) and detect fake APs. However, their methods are based on single-hop settings between the terminals and APs.

## 5 Experimental Setup

We performed two different experimental setups to validate our two techniques.

In both setups, we performed our experiments with single and multiple hops wireless sensor networks using two different types of commercially available sensor motes: MICAz and TelosB motes. MICAz is based on Atmel AVR Atmega128 8-bit microcontroller and TelosB is based on Texas Instruments MSP430 16-bit microcontroller. MICAz has a 51-pin expansion slot to connect with computers through a MIB<sup>6</sup> serial interface and TelosB has an USB interface to connect directly with a computer. Both motes run on the open source TinyOS operating system and operate over the 2.4 GHz IEEE 802.15.4 protocol<sup>7</sup>, compliant to Direct Sequence Spread Spectrum (DSSS) radio. We used a total of 9 motes (7 MICAz motes and 2 TelosB motes).

The sink node was connected through a MIB510 serial interface with the RS232 serial port of a desktop computer running on open source OS Ubuntu. TinyOS 2.1.0 package was installed on that machine. A serial packet sniffer TinyOS 2.1.0 *MsgReader* was used to capture transmitted packets from the sink node over the serial interface.

### 5.1 Single-hop WSN Setup

In the single-hop WSN setup, one sink node was connected with the desktop computer and the remaining nodes were configured as end-nodes. The end-nodes only transmitted packets containing a timestamp to the sink node periodically. Upon reception of these packets, the sink node appended its receiving timestamp and transmitted them instantaneously to the desktop computer over the serial interface. Figure 2 shows our setup for the single-hop wireless sensor network.

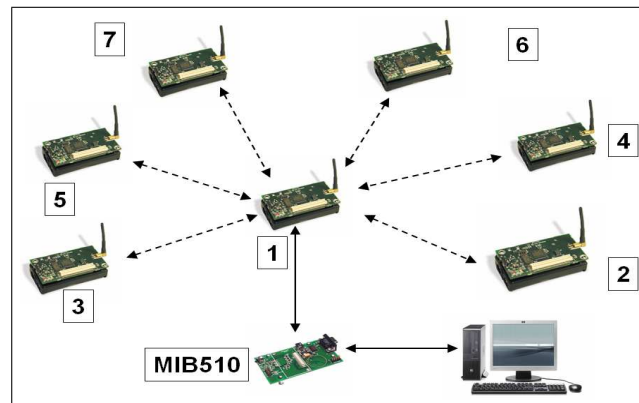


Figure 2: Single-hop WSN Setup with 7 MICAz motes. Node 1 is the sink node connected with computer through MIB510 serial interface. Nodes 2-7 are acting as end-nodes and sending packets to sink over the wireless channel.

### 5.2 Multiple-hop WSN Setup

In the multiple-hop WSN setup, there were 3 types of nodes - the sink, the forwarder and the end-nodes. The end-nodes only transmitted packets with timestamps periodically. The forwarding node captured them and forwarded the packets to the sink or to another forwarder towards the sink node over the wireless channel without modifying the content of the packets. The sink node captured all the packets transmitted to it, appended its timestamps, and forwarded them to the desktop computer over the serial MIB510 interface. Timestamps were only sent from the end-nodes to the sink. The sink didn't send any timestamps to end-nodes. The sink-to-end-nodes paths were only used for sending requests from the measuring machine to start and stop the transmission of packets.

<sup>6</sup>MPR-MIB Sensorboards Users Manual. Revision A. PN:7430-0021-08. [http://www.xbow.com/Support/Support\\_pdf\\_files/MPR-MIB\\_Series\\_Users\\_Manual.pdf](http://www.xbow.com/Support/Support_pdf_files/MPR-MIB_Series_Users_Manual.pdf), June 2007.

<sup>7</sup>IEEE Wireless medium access control and physical layer specifications for low-rate wireless personal area networks. IEEE Standard, 802.15.4-2003.

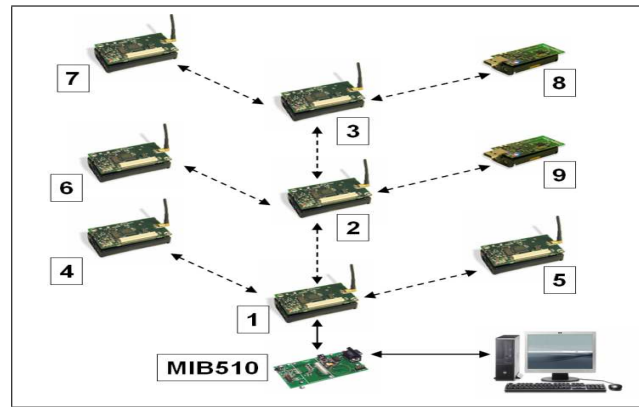


Figure 3: Multiple-hop WSN Setup with 7 MICAz (nodes 1-7) and 2 TelosB motes (nodes 8-9). Node 1 is the sink node. Nodes 2 and 3 are forwarding nodes on WSN and remaining nodes are end-nodes.

Figure 3 depicts our setup for the multiple-hop WSN with a fish-bone topology. Nodes with id 1-7 are 7 MICAz motes and nodes with id 8-9 are 2 TelosB motes. Node id 1 is the sink node, attached with computer through MIB510 Serial interface. Nodes 2 and 3 are the forwarding nodes and nodes 4-9 are six end-nodes. So, we have 4 MICAz (nodes 4-7) and 2 TelosB (nodes 8-9) motes as end-nodes. Multiple hops of sensor network were implemented in an office building placing different-hop motes in different rooms of the building. Forwarders were placed in the corridor of the building.

### 5.3 Programming the motes with nesC TinyOS

The transmitter, forwarder and sink node functionalities were implemented in TinyOS [11] using nesC [3] language. Programs for the different functionalities were installed on the corresponding motes. TinyOS packets do not have any built-in timestamps. We implemented a tinyOS packet timestamping feature on applications to insert emission and reception timestamps in packets. Along with timestamps, sending node id, sending packet sequence number, current temperature etc. were sent by the end-nodes. The sink node appended its node id, receiving timestamp, packet count number, temperature and forwarded the packets to the desktop computer over the serial interface.

The current implementation of TinyOS 2.1.0 has both 32 kHz and Micro level *CounterToLocalTime* components- (*Counter32khz32C* and *CounterMicro32C*) for MICAz motes; but only 32kHz *CounterToLocalTime* component- *Counter32khz32C* for TelosB motes. Default size of all the timer counters is 32-bit.

In the single-hop WSN setup (with only MICAz motes), we sent both the Micro and 32kHz level timestamps from end-nodes to sink node. We sent timestamps of two precisions (i.e., Micro and 32kHz frequencies) to test whether clock skew varies much with the precision of the timer counters. As a sink node, we used a MICAz mote in both single-hop and multiple-hop setups.

### 5.4 Setup for Clock Skew Measurement in Constant Temperature

In this setup, the temperature of the sensor mote environment was kept constant. Packet transmission and capturing were done at normal room temperature. Both single- and multiple-hop WSN configurations were used to capture data for fingerprinting of wireless sensor motes. Micro and 32kHz level timestamps were used in the single-hop WSN (using only MICAz) and only 32kHz level timestamps were used in the multiple-hop (using both MICAz and TelosB) WSN. Both types of timestamps were used to test the effect of precision (frequency) of timestamps on skew calculation; both types of motes were used to test the skew calculation on sensor network with heterogeneous motes.

### 5.5 Setup for Clock Skew Measurement with Varying Temperature

For testing the effects of temperature on mote clock skews, the temperatures of end-nodes were varied. First, the motes were set to the room temperature for a certain period of time to get a constant skew. Afterward, the temperature of each mote was changed by heating up and cooling down the mote environment. We used

an air heater system to heat up and a cooler to cool down the motes. The maximum temperature was about 55-60 degree celsius and the minimum about 4-5 degree celsius. The highest temperature of the experiment is approximately near to the ever reported highest temperature (136 degree Fahrenheit i.e., about 58 degree centigrade) on the Earth<sup>8</sup>.

Both MICAz and TelosB motes were used to run this experiment so that the effect of the temperature on both of them can be compared. Both single-hop and multiple-hop WSNs were used in these tests. Same thermal conditions were applied to all the nodes of the network i.e., all the motes were heated up or cooled down simultaneously.

For collecting the temperature values from the motes, we used the motes' thermometers. MICAz motes do not have built-in on board temperature sensor but TelsoB motes do. For collecting temperature values from MICAz motes we attached a MTS300<sup>9</sup> sensorboard. Four MTS300 sensorboards were attached with four end-nodes MICAz motes (motes 4-7 of figure 3) using their on board 51-pin expansion connectors. A MTS300 sensorboard has a light sensor, a temperature sensor, a microphone and a buzzer on it. The application, that runned on MICAz motes, implemented the functionality of reading temperatures from the MTS300 sensor. The temperature sensor on the MTS300 board is a thermistor (i.e., changes resistance with change of temperature). Thermistor readings (extracted from the received packets) are converted to celsius values in the desktop computer using the following formula:

$$\frac{1}{T(K)} = a + b \times \ln(R_{thr}) + c \times [\ln(R_{thr})]^3$$

and  $T(C) = T(K) - 273.15$

where:

$$R_{thr} = \frac{R1 \times (ADC\_FS - ADC)}{ADC}$$

$$a = 0.00130705, b = 0.000214381$$

$$c = 0.000000093, R1 = 10k\Omega$$

$$ADC\_FS = 1023$$

$T(K)$  = Value of temperature in Kelvin

$T(C)$  = Value of temperature in Celsius

$ADC$  = output value from Mote's ADC thermistor measurement.

The two TelosB motes have built-in on-board temperature, humidity and light sensors. The temperature sensor on TelosB is Sensirion SHT11<sup>10</sup> which is also a thermistor. In the application installed on the two TelosB motes, the thermistor value is read and inserted into each transmitted packet. The thermistor values (extracted from received packets) are converted to celsius temperatures by the measuring desktop computer using the following formula:<sup>11</sup> and also in TinyOS 2.1.0 *SensirionSht11C* component documentation.

$$T(C) = -38.4 + 0.0098 \times data$$

where:

$T(C)$  = Value of temperature in Celsius

$data$  = output value from Mote's ADC thermistor measurement.

## 6 Experiments and Results

The experiments were conducted with two main goals. The first goal was to find out whether the motes can be uniquely identified from their clock skew. The second goal was to study the impact of temperature on the clock

<sup>8</sup>NCDC: Global Measured Extremes of Temperature and Precipitation. Available at <http://www.ncdc.noaa.gov/oa/climate/globalextremes.html##hightemp>

<sup>9</sup>MTS-MDA Sensor Board Users Manual. Revision A, June 2007. PN: 7430-0020-05. [http://www.xbow.com/Support/Support\\_pdf\\_files/MTS-MDA\\_Series\\_Users\\_Manual.pdf](http://www.xbow.com/Support/Support_pdf_files/MTS-MDA_Series_Users_Manual.pdf)

<sup>10</sup>Datasheet of Sensirion SHT11: Humidity and Temperature Sensor on Telosb mote. Available at [http://www.sensirion.com/en/pdf/product\\_information/Datasheet-humidity-sensor-SHT1x.pdf](http://www.sensirion.com/en/pdf/product_information/Datasheet-humidity-sensor-SHT1x.pdf)

<sup>11</sup>Specification of Sensirion SHT11: Humidity and Temperature Sensor on Telosb mote. Available at <http://www.parallax.com/dl/docs/prod/acc/SensirionDocs.pdf>

skew.

### 6.1 Estimation of Clock skew at constant Temperatures

In the single-hop WSN setup, six MICAz motes were used as end-nodes and one MICAz mote was used as a sink node. End-nodes sent packets to the sink node every 4096 milliseconds. Both Micro and 32kHz timestamps were sent from the end-nodes in the single-hop WSN setup but only 32kHz timestamps in the multiple-hop setup. The test was run for about 10K seconds. Offsets were then obtained from the sniffed packets. Linear time linear programming based skew estimation algorithm was used to calculate the skew as explained in Section 3.2.

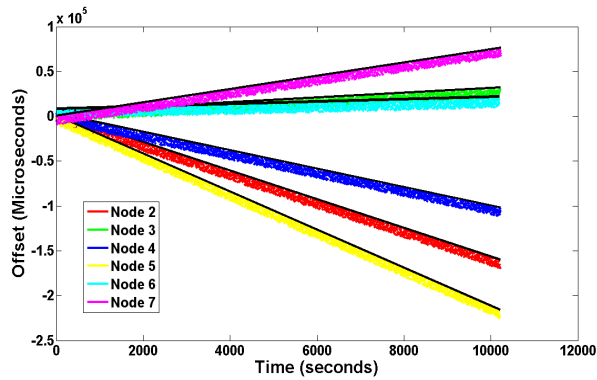


Figure 4: Clock skew estimation on room temperature in single-hop WSN using Microsecond level TimeStamps. Colored points indicate offset values of corresponding nodes and the black straight lines above the offset points indicate the estimated clock skews for the corresponding nodes

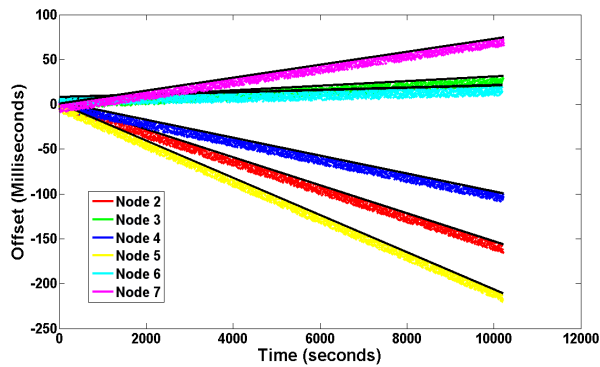


Figure 5: Clock skew estimation on room temperature in single-hop WSN using 32kHz level TimeStamps

We performed the skew estimation with both Microsecond and 32kHz level timestamps. 32kHz timestamps were later converted to millisecond level timestamps by dividing by 32. Figure 4 and 5 display the estimation of the clock skew of six MICAz motes with respect to a sink MICAz mote. Figure 4 uses microsecond level timestamp and figure 5 uses millisecond level timestamps. Table 1 shows the calculated skews for the end-nodes using 32kHz and microsecond level timestamps. The results show that each node has a different and unique clock skew. Therefore, the clock skew seems to be a good measure to fingerprint motes.

Similar experiments were performed with multi-hop WSNs, configured as shown on figure 3. Several packets were sent from end-nodes 4-9 to the desktop computer through the sink node for about 7000 seconds. The clock skews were then computed as in the single-hop setup using linear time linear programming approach.

The clock skews of the four MICAz motes are plotted in figure 6. The results show that nearby nodes (few hops away from sink node) have smaller delay offset variations than nodes further away from the sink node. Indeed, nodes 4 and 5, which are only one hop away from the sink, (red and green delay offsets) have smaller delay offsets than nodes 6 (two-hop away) and 7 (three-hop away).

Mote	Skew (T32kHz)	Skew (TMicro)
Node 2	-16.7375 ppm	-16.7359 ppm
Node 3	3.8091 ppm	3.8054 ppm
Node 4	-10.6432 ppm	-10.6398 ppm
Node 5	-21.1137 ppm	-21.1148 ppm
Node 6	3.0436 ppm	3.0517 ppm
Node 7	7.8206 ppm	7.8194 ppm

Table 1: Estimated Clock Skew using 32kHz and Microsecond level timestamps for single-hop WSN using MICAz motes as of setup Figure 2

However, the results show that each node has still a unique and different clock skew. This result is very interesting, since it demonstrates that clock skew-based fingerprinting is also feasible in multiple-hop wireless sensor networks.

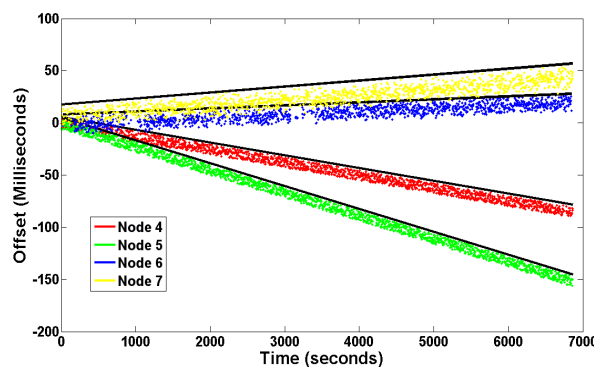


Figure 6: Clock skew of MICAz Motes in Multi-hop WSN using 32kHz level TimeStamps

In order to confirm our results, similar experiments were performed with different types of sensors (TelosB and MICAz). Note, however, TelosB and MICAz motes have different measurements of the same physical time. As explained in Section 2, on a TelosB mote, 1 second corresponds actually to 1024 milliseconds, whereas on a MICAz node 1 second corresponds to 900 milliseconds. As a result TelosB nodes have larger skews than MICAz sink nodes. To correct this effect, TelosB skews were scaled down by a factor of 900/1024. Figure 7 and Table 2 display the clock skews of 4 MICAz motes (nodes 4-7) and two TelosB motes (nodes 8-9), measured on the network configuration presented on Figure 3.

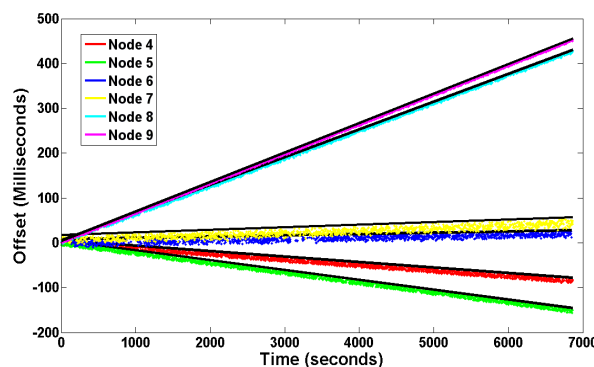


Figure 7: Estimated clock skew of MICAz and TelosB Motes in multiple-hop WSN using 32kHz level TimeStamps. Nodes 4-7 are MICAz motes and nodes 8-9 are TelosB motes.

These results clearly show that different motes (similar or different types) have different clock skews. Furthermore, clock skews can be measured over single-hop or multiple-hop networks with micro, 32kHz or

Mote	Clock Skew (T32kHz)
Node 4 (MICAz)	-11.3785 ppm
Node 5 (MICAz)	-26.1601 ppm
Node 6 (MICAz)	7.6290 ppm
Node 7 (MICAz)	11.9188 ppm
Node 8 (TelosB)	126.7999 ppm
Node 9 (TelosB)	118.9366 ppm

Table 2: Estimated Clock Skews (using 32kHz level timestamps) of multiple hops MICAz and TelosB motes as of setup Figure 3

millisecond level timestamps. This makes clock-skew fingerprinting a very reliable and robust identification scheme.

## 6.2 Analysing the Effect of Temperature on Clock Skews

This section analyses the effect of the temperature of the clock skew of a node. We performed some tests using the multiple-hop WSN scenario of Figure 3 by varying the temperatures of the end-nodes. In all these experiments, the temperature of the intermediate nodes were kept constant. We only varied the temperatures of the end-nodes by heating up or cooling down their surrounding environment with a heater or a cooler. We collected traces of about ten thousand seconds. Figure 8 displays the estimation of clock skew in the multiple-hop scenario while varying the temperature from 4 to 60 degree celsius. These results show that the clock skew of a node fluctuates when its temperature varies, but still exhibits a constant component. In other words, temperatures add some small noise to the skew without really modifying its value; Temperature variations do not affect our clock skew based fingerprinting. This is an important first result.

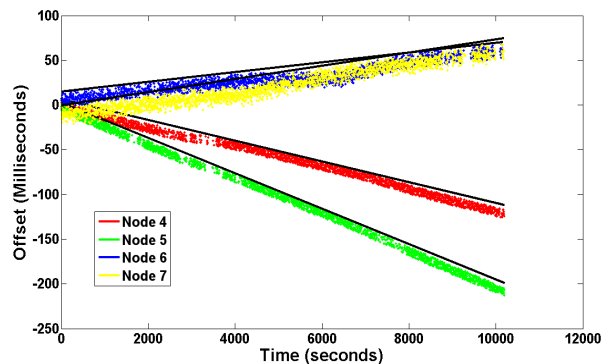


Figure 8: Estimated clock skew of 4 MICAz motes in multiple-hop WSN while varying the temperature (from 4-60 degree centigrade). Colored points show the offset values of corresponding motes and black straight lines above the offset values show the estimated clock skews

The other goal of this section was to find out whether the temperature had any effect on the clock that could be measured and predicted. Since the temperature does not affect the skew, we decided to study the clock drift variation according to the temperature.

We then computed the node clock drift as follows:

### - Step 1: Estimation of the Clock Skew Variable component

The first step consists in removing the constant component of the clock skew of a node to retrieve its variable part, i.e. the fluctuation added by the temperature variation. The constant component was computed, as explained previously, by exchanging several timestamps and computing the clock skew using a linear programming algorithm [16].

Figure 9 shows the clock skew, the estimated constant clock skew (upper black straight line) component of node 7. It also shows the variation of temperatures during our experiment (red curve).

The resultant variable skew component obtained by subtraction the constant component (black line) from the measured skew is plotted in figure 10 (green dots) along with the corresponding temperature (red curve).

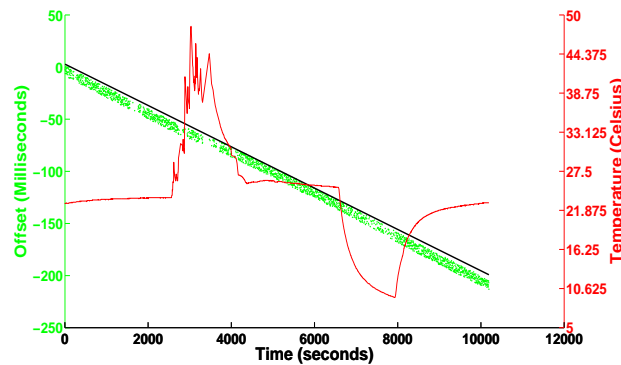


Figure 9: Estimation of Constant Clock Skew from the offset values. Green dots indicate offset values and Upper black straight line shows the estimated clock skew

The results shows that the variable component is quite noisy and needs to be denoised.

- **Step 2: Denoising the Variable Skew** We then denoised the variable skew component using a multiple-pass sliding window based linear time programming algorithm [16, 2, 14]. The final denoised line (upper blue line of Figure 10) is the average of the denoised lines for several passes of the algorithm.

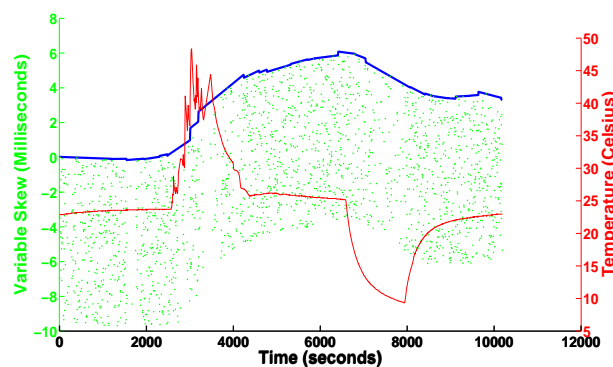


Figure 10: Denoising of Variable Skew using a total of 8 windows and 3 passes

- **Step 3: Clock Drift computation**

We then computed the clock drift of the node. The clock drift is calculated by taking the first derivative of the denoised variable skew curve (i.e. upper blue curve of Figure 10). The resulting drift is plotted on Figure 11 (dashed pink line) along with the corresponding temperature variation (red curve). The drift was multiplied by a factor in order to have similar ranges than the temperature. The obtained result is very interesting: the variation of the clock drift almost follows perfectly the variation of the temperature. This means that the temperature variation of a node can be obtained by evaluating the drift of its clock. This result can be useful in many applications.

The above steps 1-3 were performed for all end-nodes (i.e., nodes 4-9 of Figure 3). We found out that the number of hops between the end-node and the sink does not affect the measure of the clock drift.

We also discovered, as shown on Figure 12, that the variation of the clock drift according to the temperature was different according to whether the node was a TelosB or a MICAz mote. In fact, with MICAz nodes, the matching between the temperature and the clock drift variation was almost perfect. With TelosB nodes, the matching happened only when the temperature was positive, but the drift increased

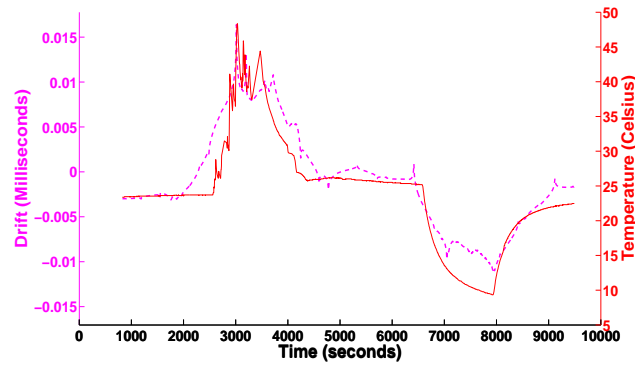


Figure 11: Estimated Drift (pink dashed line) and Measured Temperature (red line)

when the temperature was going below zero. We believe that this is the result of a compensation mechanism implemented on TelosB nodes, that automatically compensates the clock drift when the temperature drops below zero. For some reasons, this compensation is not performed when the temperature is larger than 0.

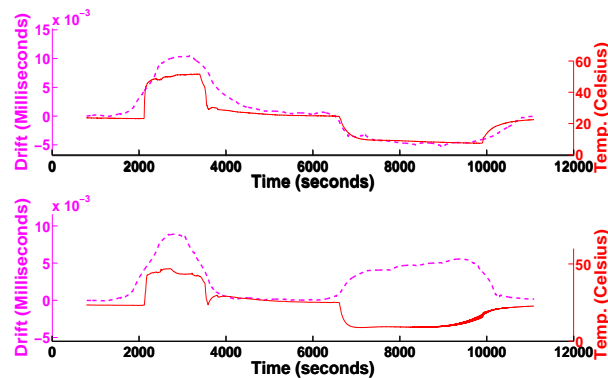


Figure 12: Estimated Drift (pink dashed line) of a MICAz mote (upper graph) matches with measured temperature (red line) and Drift of a TelosB (lower graph) mote matches with temperature for higher than room temperature but for lower temperature it is overcompensated.

## 7 Potential Applications

This section discusses several potential applications of our results.

- **Fingerprinting (Identification) of Wireless Sensor Motes** One of the results of our work is to demonstrate that different motes have different and unique clock skews (see Tables 1 and 2, and Figures 4 to 7). We also showed that clock skews can be measured over single-hop and multiple-hop WSNs with both milli and micro level timestamps. Furthermore clock skews are not affected by varying temperatures, are quite difficult to alter [4] and are quite stable over time.

We believe that all these properties makes the clock skew a good candidate to remotely identify nodes. By measuring the clock skew of a node, a verifier can identify it amongst a set of other nodes.

Clock skew-based identification could also potentiall be used to detect malicious nodes that try to impersonate other nodes (such as in wormholes or sybil attacks).

- **Malicious/Malfunctioning Node Identification:**

We showed that the clock drift varies according to the temperature (see Figure 11), and that the temperature of a node can potentially be computed from his clock drift.

Therefore, the clock drift can be potentially be used to compute the temperature of nodes that do not have onboard temperature sensors. Furthermore, it can be used, for nodes that have temperature sensors, to detect malfunctioning (or malicious) nodes that report, intentionally or not, bogus temperature values. A compromised node that reports fake temperatures in order to, for example, generate an alarm can then potentially be detected with this approach.

- **Geolocation:** [6] proposed ‘Sundial’- an on board opto-electric sensor based geolocation estimation for WSNs that uses day light variations. Temperature varies with the variation of day light. Since the temperature varies according to the day light, we believe that a similar geolocation estimation technique can be devised by replacing the day light measurements with estimated temperatures derived from the clock drift. The longitude of a place could be estimated by measuring daily peak temperatures. Latitude can be estimated by measuring the lengths of days over a reasonably long period of time.

## 8 Conclusion and Future Work

In this report, we explored the clock skews and drifts of different commercially available wireless sensor motes. We showed that different motes have different clock skews and that these skews are not affected by time and temperatures. Therefore, clock skew-based fingerprinting of wireless sensor motes is possible. This fingerprint technique can be used to identify nodes, but also to detect virtual motes on virtual honeynets, wormhole and sybil attacks. We also showed that the clock skew of a node varies with the variation of temperature. This property can be used to detect malfunctioning or malicious nodes.

The work performed in this task was mainly experimental. The experiments required significant resource and time. They show that clock skew-based identification is feasible. However, the design of an actual identification scheme still requires further work. Similarly, the design of a geolocalization scheme still requires some effort. We believe however that the achieved results are very interesting and valuable.

## 9 Summary of Deliverable

This deliverable contains the specification and the analysis of modules for elementary modules for dependable services. This deliverable will be input to Deliverable D4.3 as well as work package WP5.

Note that this deliverable describes only the elementary security service, namely *Skew-based service*, that was implemented and provided as input to WP5. As described in D4.1, several other elementary security services have been designed within task 4.1 of the WSAN4CIP project. Since they will not be implemented they are not reported in this deliverable (but are partially described in D4.1).

This deliverable describes services that are based on the clock skew to identify and detect anomalies in wireless sensor networks.

The clock skew, an inherent property of clock crystals of physical devices, is defined as the rate of deviation of a device clock from the true time. The frequency of a device's clock actually depends on its environment, such as the temperature, humidity, vibration, electromagnetic interference, as well as the type of crystal. The main contributions of our work are two-fold. First, we experimentally validate that MICAz and TelosB sensor motes have different and unique clock skews. Furthermore, the clock skew of a node can easily be monitored, even via a multi-hop Wireless Sensor Network (WSN). We argue that this feature can be used for identification of the nodes, detection of Wormhole and Sybil attacks. Second, we show that the clock skew of a sensor node varies with the temperature. We explain how this property can be used to detect malicious and mal-functioning nodes and to geo-localize them. Using 7 MICAz and 2 TelosB motes we performed extensive experiments to validate our approaches.

## References

- [1] Arackaparambil, C., Bratus, S., Shubina, A., and Kotz, D., 2010, On the Reliability of Wireless Fingerprinting using Clock Skews, *The ACM Conference on Wireless Network Security (WiSec)*
- [2] Dyer, M. E., 1983, Linear time algorithms for two- and three variable linear programs, *SIAM Journal on Computing*, 31–45.
- [3] Gay, D., Levis, P., von Behren, R., Welsh, M., Brewer, E., and Culler, D., June 2003, The nesC language: A holistic approach to networked embedded systems, *Proceedings of Programming Language Design and Implementation (PLDI 2003)*.
- [4] Gehani, A., and Chandra, S., 2006, PAST : Probabilistic Authentication of Sensor Timestamps, *Annual Computer Security Applications Conference (ACSAC)*.
- [5] Gerdes, R. M., Daniels, T. E., Mina, M., and Russel, S. F., 2006, Device identification via analog signal fingerprinting: A matched filter approach, *The 13th Annual Network and Distributed System Security Symposium (NDSS)*.
- [6] Gupchup, J., Musaloiu-E., R., Szalay, A., and Terzis, A., 2009, Sundial: Using Sunlight to Reconstruct Global Timestamps, *European Conference on Wireless Sensor Networks (EWSN)*.
- [7] Hall, J., Barbeau, M., and Kranakis, E., 2003, Detection of transient in radio frequency fingerprinting using signal phase, *Wireless and Optical Communications*.
- [8] Huang, D.-J., Teng, W.-C., Wang, C.-Y., Huang, H.-Y., and Hellerstein, J. M., 2008, Clock Skew Based Node Identification in Wireless Sensor Networks, *IEEE Globecom*.
- [9] Jana, S., and Kasera, S. K., 2008, On fast and accurate detection of unauthorized wireless access points using clock skews, *ACM international conference on Mobile computing and networking (MobiCom)*.
- [10] Kohno, T., Broido, A., and kc claffy, May 2005, Remote physical device fingerprinting, *Proceedings of IEEE Symposium on Security and Privacy*, 211–225.
- [11] Levis, P., June 2006, TinyOS Programming. Available at <http://csl.stanford.edu/~pal/pubs/tinyos-programming-1-0.pdf>
- [12] Maheshwari, R., Gao, J., and Das, S. R., 2007, Detecting Wormhole Attacks in Wireless Networks using Connectivity Information, *IEEE INFOCOM*.
- [13] Martinec, M. Temperature dependency of a quartz oscillator:. <http://www.ijs.si/time/#temp-dependency>.
- [14] Megiddo, N., November 1983, Linear-time algorithms for linear programming in  $r^3$  and related problems, *SIAM Journal on Computing*, 759–776.
- [15] Mills, D. RFC1305: Network Time Protocol (Version 3) Specification, Implementation:. RFC1305, 1992.
- [16] Moon, S. B., Skelly, P., and Towsley, D., Mar 1999, Estimation and removal of clock skew from network delay measurements, *Proceedings of the Conference on Computer Communications (IEEE Infocom)*.
- [17] Murdoch, S. J., October 2006, Hot or not: Revealing hidden services by their clock skew, *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS'06)*, 27–36.
- [18] Newsome, J., Shi, E., Song, D., and Perrig, A., 2004, The Sybil attack in sensor networks: analysis & defenses, *Information Processing in Sensor Networks (IPSN)*.
- [19] Paxson, V., June 1998, On calibrating measurements of packet transit times, *Proceedings of SIGMETRICS' 98*.

- [20] Rasmussen, K. B., and Capkun, S., September 2007, Implications of Radio Fingerprinting on the Security of Sensor Networks, *International Conference on Security and Privacy in Communication Networks (SecureComm)*.
- [21] Sharp, C., Turon, M., and Gay, D. TinyOS Extension Proposal(TEP) 102: Timers:. Available at [http://tinycvs.sourceforge.net/\\*checkout\\*/tinycvs/tinycvs-2.x/doc/html/tep102.html](http://tinycvs.sourceforge.net/*checkout*/tinycvs/tinycvs-2.x/doc/html/tep102.html).
- [22] Uddin, M. B., and Castelluccia, C., 2010, Toward Clock Skew Based Wireless Sensor Node Services, *International Workshop on Ubiquitous Body Sensor Networks (UBSN)*.
- [23] Wright, J. Detecting Wireless LAN MAC Address Spoofing:. Available at <http://www.net-security.org/dl/articles/wlan-mac-spoof.pdf>.
- [24] Zander, S., and Murdoch, S. J., 28 July-01 August 2008, An Improved Clock-skew Measurement Technique for Revealing Hidden Services, *17th USENIX Security Symposium*.

[end of document]