European
Commission

JRC TECHNICAL REPORTS

Sensor network field trials

Validation of signal percolation through multiple hops. Studies of physical latency on small-scale networks.

E. Gutiérrez, G. Renaldi

2016

EUR 28267 EN

Joint
Research
Centre

Contact information
Name: Eugenio Gutiérrez
Address: TP 480, Via E.Fermi, 2749, I-21027 Ispra (VA), ITALY
E-mail: eugenio.gutierrez@jrc.ec.europa.eu
Tel.: +39 0332 78.5711

JRC Science Hub
https://ec.europa.eu/jrc

JRC104388

EUR 28267 EN

PDF ISBN 978-92-79-64109-1 ISSN 1831-9424 doi:10.2788/610616

Luxembourg: Publications Office of the European Union, 2016

How to cite: E. Gutiérrez, G. Renaldi; Sensor network field trials: Validation of signal percolation through multiple hops. Studies of physical latency on small-scale networks; EUR 28267 EN; doi:10.2788/610616

Deliverable Administration & Summary		WP 852: CSNWI		CONTAINSTECH	
No & name	D7 2016 **Sensor network field trials** Validation of signal percolation through multiple hops. Studies of physical latency on small-scale networks.				
Status	FINAL	Due	15/12/16	Date	30/11/16
Author(s)	E. Gutiérrez, G. Renaldi				

This page is purposely left blank.

Contents

List of Figures

SUMMARY

In this report we present some preliminary results on the percolation of signals through a multi-hop wireless network. Such networks are candidates for low cost, low maintenance, systems for the deployment of energy-autonomous sensor nodes used for environmental and structural diagnostics of constructions and components of the built environment. We conduct field-trial tests with a view to study the latency and success rate of the transmission of datagrams using low power, short range, transceiver stations capable of linking a remote source to a central base station. In particular, we report on how the network performance is affected by the duration of the effective duty cycle and topology and compare these to simple probabilistic percolation models of the network.

BACKGROUND

The deployment of wireless sensor networks presents an opportunity to monitor the performance of natural and anthropogenic systems in a wide range of field environments that hitherto had not been possible due to the high cost of electronic devices (1) (2) (3) (4). But whereas the cost of information technology components has been coming down—Moore's law-like—over the decades, battery life and the ever-increasing costs associated with maintenance schedules, place severe constraints on the scale-ability of deployment for swarms consisting of hundreds or thousands of sensor transceiver nodes.

We can distinguish two main classes of wirelessly connected objects on the basis of their energy autonomy.

On the one hand those objects permanently plugged to an energy source such as the electricity grid, an automobile or equivalent. For these it is fair to say that the design of the energy consumption protocols, other than when they affect directly their physical performance, or contravene certain regulations, such as for example the power or bandwidth of the radio transceiver, is a not a primary issue.

A second class of devices concerns those that possess a limited energy resource provided by a battery or energy cell of some kind. Here too it is safe to say that whereas the energy performance protocols have improved considerably over the years (especially on improving battery life and processing power per mAh capacity), the manufacturers of the billions of 'smart' phones rely on billions of unpaid, but highly motivated, users to keep the system going. Without that essential recharge-ability link in the chain our, so-called, smart world and all the industries that depend on it would either soon collapse or undergo a systemic change.

Our study concerns how to create a network of wirelessly interconnected devices that do not rely on an unlimited energy source or supply of willing end-users to make it work. Such a network would have to be very parsimonious with its energy resources, and under such circumstances it becomes necessary to tabulate the rate of energy consumption for each type of process (sensing, computation, transceiving) (5) (6) (7). This in turn obliges the node designer to consider what the purpose of the unit is and what the functionality criteria are. For example, what type of information is the network supposed to gather, and at what rate? Is it cheaper to sample and send much raw data, or to process much and transmit less? In all cases all considerations are underpinned by the fact that whereas computational processing power has increased exponentially over the past decades, the power needed to transmit radio signals with omnidirectional antennas —such as those used in mobile devices—grows approximately with the square of the distance (directional antennas can, in the appropriate circumstances improve on this).

Some rough but usefully demonstrative figures: Let's assume that our typical 4 G smart-phone is equipped with a 2500mAh battery over 3.6 V; a perfect battery of these characteristics would store just under 32kJ of energy. Supposing a typical 4G phone requires of the order of 30dBm (about 1Watt) to function in the cell network; then talking for one hour would consume 10% of the battery (these are just ball-park figures and do not include the background computation, ADC/DAC conversion and other 'smart' features running in the background). Considering that this would probably require of the order of a five-minute plug-in to the mains adapter, it is clear how dependent such networks are reliant on their end-users.

As a counter-example consider the wireless nodes used in this study, which use transmission powers of a 0dBm (1mWatt). This system is purposely left low on power in order to save on battery life and allow a low capacity energy-harvesting device to recharge the system during its duty cycle, the drawback is that if a signal is to be sent to some far-off location then some other means of transmitting the signal must be found.

The development of appropriate ICT systems for sensor networks has developed considerably over the past two decades or so and is still the subject of much research (8) (9) (10) (7) (11) (12) (13). Whereas much of the development has been concentrated on the technical systems themselves, and whereas such developments requires in-depth knowledge in the electronics and information technology components , there is still a case for a non-specialist analysis of the performance of such systems to be made by comparing real-life field performance to simple mathematical models. We use the term simple mathematical models, however this does not mean that we may expect simple behaviour.

In this, our first, study we shall concentrate on how reducing the duration of the live duty cycle affects latency, and failure rate. We will also keep an eye open to the issue of topological connectivity and make a nod to the implications of Bayesian inference on the successful percolation paths. Ultimately, we wish to collect data to both calibrate numerical simulations and devise generic mathematical (probabilistic) models for system performance.

ARCHITECTURE AND FEATURES OF FIELD NETWORK

The system architecture is shown in Figure 1 and the duty cycle structure is shown in Figure 2. The network system WiSP-node presented here was developed in-house at the JRC (14) for the purpose of calibrating ballistic propagation in multi-hop networks. Allied to this is a numerical simulation program (*see* (15) for more details on *Dylink* programme), also developed at the JRC, with which to simulate networks of generic size and dimension with capability to simulate not just the connectivity but also the network functionalities such as battery drain and degraded communication channels due to interference and collision.

The basic idea of this type of network is quite simple: A source node broadcasts a datagram containing some pertinent information (an alarm or a status etc.), if the datagram is registered by some neighbouring node, it rebroadcasts and thus forwards it through the network. These are referred to as flooding protocols; they try to permeate the network without any topology or location-based forwarding strategy and the signal may or may not reach the sink (destination). Clearly there are cleverer ways of implementing a forwarding protocol, which we will stand to examine in subsequent studies, however, our simple protocol is easier to express mathematically and simulate numerically.

SYSTEM ARCHITECTURE

A convenient deployment of *WiSP* nodes has been employed to carry out the tests described in this document. The nodes distribution covers an area around and inside of building 48 at the JRC Ispra site. The testing setup involves special devices and software applications besides the *WiSP* nodes. Such devices are represented in Figure 1 as green striped rectangular boxes. There now follows a brief description of these devices.

The *Signal Generator* device transmits packets at specific time intervals. The outgoing packets payload is an alarm code followed by two parameters that represent the time when the packet was been transmitted. The *Signal Generator* is composed by a radio transmitter (operating on the same frequency as the *WiSP* nodes), a GPS receiver and a microcontroller. The firmware running on the microcontroller is designed to acquire the GPS signals, to synchronize its time base with the GPS time pulse and to transmit packets punctually.

On the opposite side of the deployment of the WiSP nodes, there is a Receiver device. Its scope is to acquire the packets coming out at the end of the propagation process through the established network. Immediately following the acquisition of a packet, a set of operations is carried out by the microcontroller to generate an accurate timestamp to be associate to the incoming datagram. At this point, a formatted string comprising the original packet and other data, is sent to the application server where an ad hoc designed *Sniffer* node receives the string via a USB connection.

Some parametric functionalities of the WiSP nodes are remotely configurable using commands that, themselves, can be propagated through the nodes. In order to do so, another dedicated node has been designed to inject such commands into the network; we refer to this as the *WiSP TRX*, and is connected to the application server as well as the *Receiver*. For the test campaign described below, the *WiSP TRX* has been used to set or to check the nodes' *Level of Activity*. This device is managed by the *WiSP Node Manager* software application.

The *Spy Node* and the *Control Network TRX*, as a whole, constitute a structure that gives another point of view from which the packets propagation through the *WiSP* network can be monitored. In particular, the Spy Node is within range of the Signal Generator transmissions. The device is composed by a receiver operating within the same frequency band as the *WiSP* nodes and a transceiver operating within the SRD 868 MHz frequency band. Both radio devices are managed by a microcontroller that basically receives packets from the *WiSP* network, processes them and retransmits to the Control Network TRX. Once a packet reaches the *Control Network TRX*, it is forwarded to the *Communication Logger* software that is running on the *Application Server*.

As mentioned above, three software applications run at the same time to manage the tests. More specifically, the *Sniffer* compiles report files composed of detailed and time-reliable records for every packet that reached the *Receiver*, i.e. the packet target destination. In addition to this, the *Communication Logger* application compiles report files composed of traces left by packets propagating through a significant subset of *WiSP* nodes (specially the *Signal Generator*). Both these applications write their reports on a shared repository where team members can analyse the test results.

As is shown in Figure 2, said devices surround a set of *WiSP* nodes, which are immersed in a noisy environment. This is a typical real-life scenario. For example, some nodes are fully operating and repeating packets, whereas others are either inactive, not retransmitting due to external interference or in planned off-duty because they have already retransmitted (this is purposely designed in order to avoid infinite retransmission of already-received datagrams)
The testing system architecture is designed to be expandable and currently available for further tests and differing environmental scenarios.

Duty cycle implementation

The level of activity of a *WiSP* node, namely the duty cycle, represents the ratio between the period in which the node is active and the reference period. In order to have nodes sufficiently reactive, a period of 5 seconds has been fixed as reference period. Statistically, the length of this period is irrelevant if the number of packets sent is big enough.

Figure 2 represents a simplified flowchart of the duty cycle implementation. After a number of initialization operations, a random number that can range from 1 to 100 is extracted. At this point, the microcontroller goes into an endless loop. At the same time, a counter is updated by an interrupt service routine whose execution is triggered by the reception of an interrupt generated on a precise time base. The counter increment is strictly sequenced to the reference period.

The duty cycle mechanism is as follows: If the previously extracted number is equal or lower than the fixed activity level, it is established that the node is active, otherwise the node is inactive. If the counter is greater than the reference period then a new random number has to be extracted, and the counter has to be reset. At this point, if the node has been set as inactive, the radio modem is switched off, otherwise it is switched on. In the former case, the execution restarts from the beginning of the loop (in the latter case the radio modem buffer is checked). If the radio modem buffer does not contain data, the execution restarts from the beginning, otherwise a buffer content analysis starts.

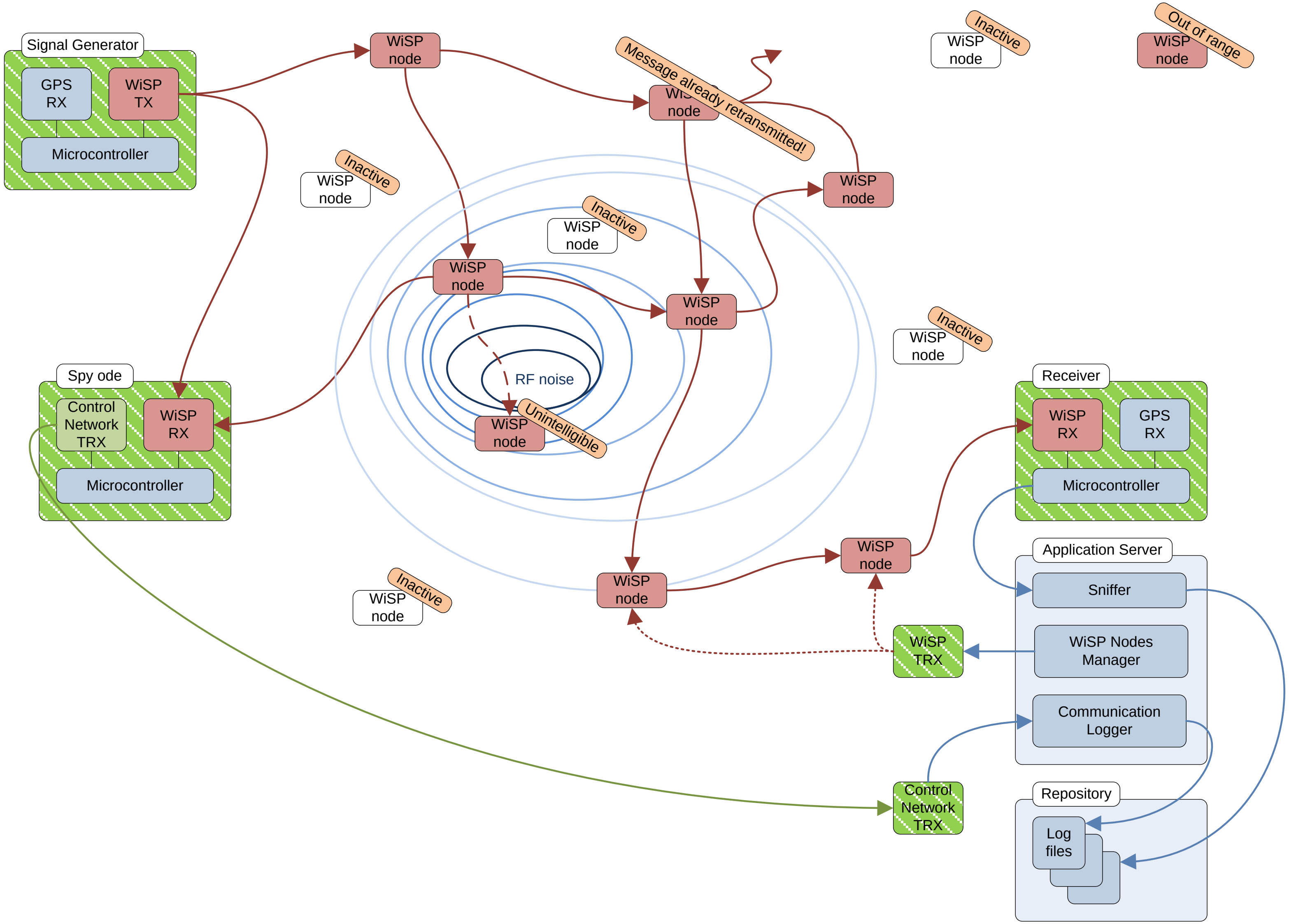

Figure 1 Outline of network architecture and connectivity functionality: Set up for the test campaign.

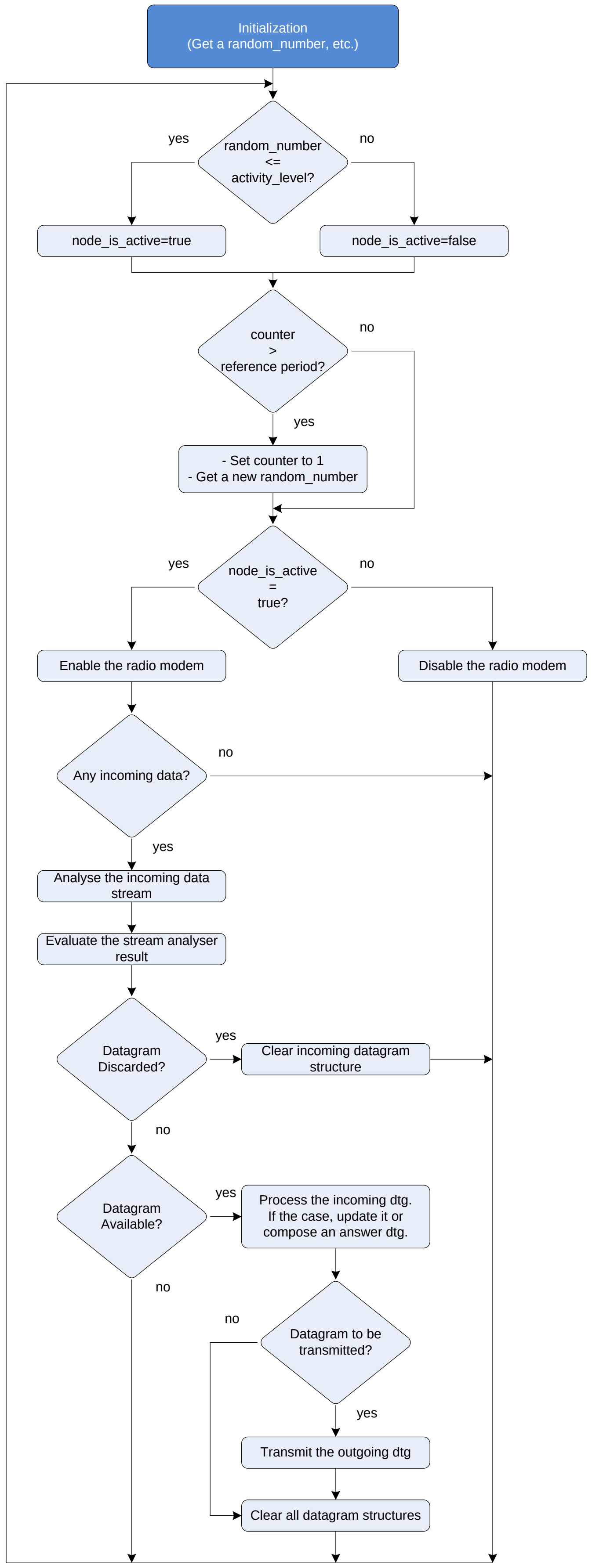

Figure 2 Simplified flowchart of the duty cycle implementation.

Basic Combinatorics of Percolation Paths

In our field example we wish to transmit a signal (in this case a short datagram) from a source node *A* to some *distant* base station *Z*. By *distant* we mean that the range of the transceiver of *A* is not sufficient to reach *Z* without resorting to an intermediate node(s) b,d,e…. Thus the problem of connectivity is posed in a completely arbitrary metric range.

In our example, in order to connect A to Z we have five intermittent nodes at our disposal. We can think of each path as a word, of sorts, where the letters are the nodes. We are interested in testing the network in order to find the most common routes and perform statistical analyses on both the percolation latency and their success rate.

We first need to perform some simple combinatorics analysis of the possible paths; this will provide the basis for the probabilistic analysis before comparing that to the real-life statistical performance. What are the probable paths then?

At this stage it is important to say that, in its current deployment, it is perhaps physically possible for the signal to propagate from A to Z using only one of the intermediate nodes. Moreover, once a node has been selected as an element of the path route, it cannot be re-used again later on in the path. However, that same node may serve as a fork in the path for other possible paths leading to Z. So, in general we have five nodes that can be combined, without duplication, but can be selected in combinations using from one up to five nodes. The time for each path will depend on both the path length and the traffic associated with the protocol.

For example the following paths are valid permutations: {AbZ}, or {AbcdefZ}, or {AfZ}, or {AdeZ}, or {AedZ}, noting from the last two examples that the **order is important**. Moreover, it is also possible for the signal to arrive at destination through more than one path, i.e. with more than one possible permutaion; for example: {AbZ} and {AbdZ} and {Acef} are possible, but {AdZ} and {AbdZ} are not probable for a concurrent datagram because node d cannot be reselected in {AbdZ} having already, presumably, appeared first in {AdZ}.

In the field trials presented we deployed a total of two fixed start and end nodes, and five intermediate nodes. Given that the fixed nodes are not exchangeable, the permutations can only be generated from the five intermediate nodes combining selections of up to five nodes at one time of the form:

$$(1) \quad P_{n,r} = \frac{n!}{(n-r)!}$$

We have that for r=[1:5] and n=5, the total number of permutations is

$$(2) \quad P_{total} = \sum_{r=1:5} P_{5,r}$$

$$(3) \quad P_{total} = 120_{(5,5)} + 120_{(5,4)} + 60_{(5,3)} + 20_{(5,2)} + 5_{(5,1)} = 325$$

This is the number of combinatorically possible paths. In principle, if no other factors influence the path choice, each path has a 1/325 chance of occurring.

The actual paths recorded from our field trials will reflect the impact that physical location, and effective range have on the likelihood of generating a successful datagram transfer.

As we shall show later, although all such paths are theoretically feasible (all we need do is to physically deploy the nodes in such a way to make the connectivity possible), the transmission protocol and the physical location of the devices generate a form of Bayesian probability, which is implicit in the transceiver protocol, such that the likelihood of some paths is much smaller than that given by Eq (3).

THE PROBLEM WITH DATA

The term Big Data, some say, is often overused. Well, the following is an example of how, if we do not consider the implications of the combinatorics of path sequences, the task of allocating storage space to cater for all possible percolation paths quickly builds up exponentially. In our field example of 5 nodes we have shown above that it is possible to generate 325 path combinations, but were we to use 10 nodes the figure rises to just under ten million paths, 26 nodes (letters) would generate about 1e27 paths (words), and with fifty nodes we arrive at 8.2674×10^{64} path combinations. Just as a rough estimate, the number of stars in the Universe is of the order of 1e24 (source ESA[1])

---

1 http://www.esa.int/Our_Activities/Space_Science/Herschel/How_many_stars_are_there_in_the_Universe

Why is this important? The reason lies primarily with our wish to both quantify and catalogue the paths generated by our procedure. It is not enough to say that we have so many paths of length *n*, but also to uncover their structure and evolution. In order to do this in a very abstract and general way we make recourse to matrix algebra, for this reason the dimensionality of our path numbering and ordering becomes a key issue.

This may be useful to identify the role of certain nodes in the chain both in a topological and geographic sense. This information may, in turn, assist us in understanding the overall performance of the network. If, on this basis, other than the combinations presented earlier, we make no restrictions on either the length or node combination of possible paths, so we must be able to catalogue both the number and arrival times of a possibly very large range of combinations.

Coming back to the words analogy, think of this as a literary text—a novel, for example—where each word (path) is made up of a concatenation of letters from our alphabet (the list of node numbers). We can construct a universal 'dictionary' made up of all possible combinations of words from *n* letters.

Suppose we start reading a text and we want to assign to each word a number for the purpose of cataloguing and filling our transmission report; we have two options. We can enumerate the words and assign a number in order of their appearance in the text; for example, our novel might contain only ten words. That's a pretty reasonable number; the only problem is that if we wanted to compare two texts the numbering of their respective word ordering—or words for that matter—may not match. For example, the word '*the*' might be *Word 1* in one text but *Word 3* in another. We need something more generic.

Another option is to look up a word's number in the universal dictionary: that which contains all the possible words conjugated with our limited alphabet. This might sound reasonable in a dictionary of natural human language as there might be of the order of a few hundred thousand words. But as we have shown above, creating a universal path dictionary of just 10 nodes already generates nearly a million paths.

If we wish to keep the flexibility of tracking the evolution of possible path names over field trials involving even only tens of nodes, then we require a flexible data manipulation format.

DATA COLLATION METHODS

An example of the catalogue of datagram percolations of a field trail in our experiment is presented in Table 1.

It consists of four major headings, in turn divided into pertinent subheadings. In the *timestamp* section we have date and time up to millisecond accuracy. This is followed by a *datagram* identification group giving the id number and source and end nodes (fixed). The next section gives the payload.

The last section concerns the path of the successfully-received datagrams, which, as we explained earlier, can be up to five columns long. As can be seen in the extract, path lengths can range over the full five-column set, and more than one path can be performed for each datagram (indeed it is not unusual to see up to three). A path may be repeated but only for a new datagram, noting that paths are read from right to left.

Our network design was conceived to cater for up to one-hundred intermediate, ad-hoc, nodes bookended by the source (102) and sink (101) nodes; currently we have about twenty intermediate nodes available, but for this study we have used nodes 3,4,5,6 and 9.

Whereas datagrams are sent on multiple minute intervals of the clock (either one or five), the duration of the percolation is given Table 1, columns 6-7, expressed in seconds and milliseconds; for example datagram with id=6 received two percolations: respectively (102->3->6->101) and (102->3->6->9->101), datagram id=7 received three percolations from paths (102->5->3->6->101), (102->5->3->6->4->101) and (102->5->3->6->9->101).

When a datagram is lost it is implicitly recorded; for example, the datagram column jumps from id=7 to id=9, i.e. id=8 was broadcast from the source node but none of the intermediate nodes picked it up, and hence was not transmitted along any path. This last item brings up another factor we are interested in monitoring: the success rate or percolation.

Table 1 Datagram details from a field-trial over three days using five intermediate nodes (labelled Nos. 3 4 5 6 9).

Timestamp at the arrival of the datagram to the Sniffer							Datagram identification			payload			List of repeaters (reverse order)				
year	month	day	hour	minute	second	millisecs	id	Source node	destn. node	code	param. 1	param. 2	last	…	…	…	…
…																	
16	10	14	15	10	4	982	6	102	101	7	15	10	6	3			
16	10	14	15	10	10	28	6	102	101	7	15	10	9	6	3		
16	10	14	15	15	8	857	7	102	101	7	15	15	6	3	5		
16	10	14	15	15	9	390	7	102	101	7	15	15	4	6	3	5	
16	10	14	15	15	10	997	7	102	101	7	15	15	9	6	3	5	
16	10	14	15	25	7	112	9	102	101	7	15	25	6	3			
16	10	14	15	25	8	944	9	102	101	7	15	25	4	6	3		
16	10	14	15	25	11	706	9	102	101	7	15	25	9	6	3		
16	10	14	15	30	3	113	10	102	101	7	15	30	6	3			
16	10	14	15	30	5	696	10	102	101	7	15	30	4	6	3		
16	10	14	15	30	6	754	10	102	101	7	15	30	9	6	3		
16	10	14	15	35	7	618	11	102	101	7	15	35	6	3	5		
16	10	14	15	35	8	0	11	102	101	7	15	35	4	6	3	5	
…																	
…																	
16	10	17	1	20	8	695	704	102	101	7	1	20	9	6	3		

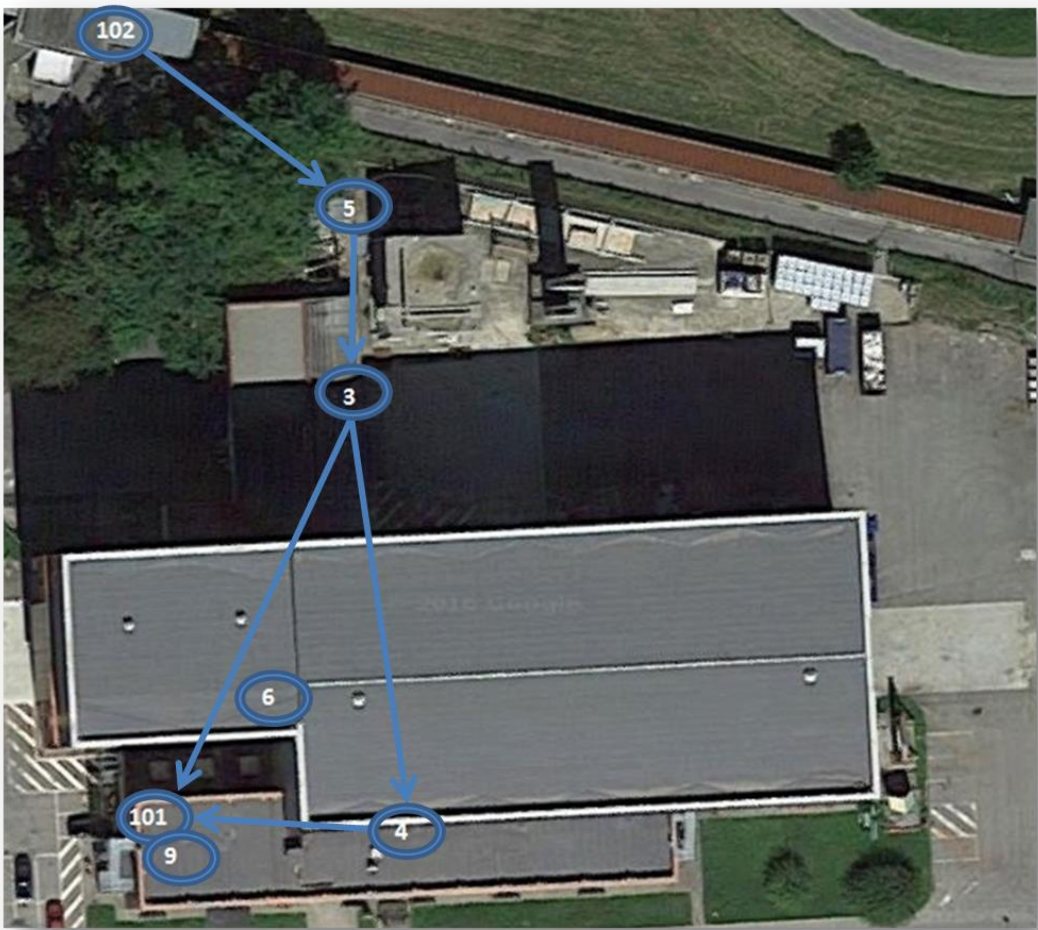

Figure 3 General view of location of nodes in the vicinity of the ELSA laboratory and path examples.

Source and sink nodes 102 (top left) and 101 (bottom left) are inside buildings shown. Intermediate nodes 5, 3, 6, 4 and 9 in order of proximity to source node 102. Nodes 3 and 5 are nearly free to air; node 6 is inside the main hall of the ELSA laboratory, and nodes 4 and 9 inside the annex office building on the same level as node 101. The percolations shown correspond to paths (102->5->3->101) and (102->5->3->4->101).

Photogrammetry based on Google Earth imagery

RESULTS

PATH CARDINALITY SETS AND SUCCESS RATE

In Table 2 we present the failure rates for the various activity levels; for each we have examined three distinct sets. The number of datagrams is arbitrarily different from set to set, in some cases lasting as much as three days. Datagrams were sent at minute and five minute intervals, but no significant difference was seen as a result. For each set we computed the cardinality set of sent datagrams, i.e. the number of unique datagrams broadcast from the source node (102). From these we are able to compute both the number lost and the number of repeats, that is, those arriving through more than one path from source to sink.

Examining the data, it can be seen that the overall failure rate increases with reducing activity rate; with the nominal failure rate at 100% duty cycle being of the order of 2-3% whereas at 70% activity only 20 % of datagrams arrive at the sink. We also note that the range of failure rates per set are consistent with the general trend, but significant variations within a set cohorts may arise : e.g. at 95 and 85 two sets present failure rates of 33% ; such behaviour is expected of real finite-size realizations.

Also of interest is that the overall mean repeat rate of unique datagrams reduces from nearly 3 to 2 from 100 to 70 % activity levels respectively; the full range being 2.97 repeats for set 1 at 100% and 1.75 for set 2 at 70% respectively.

A reduction in the success rate is attendant to the duration of the duty cycle. From the data it would seem that whereas a concomitant reduction in the number of repeats is also present, we have yet to establish the statistical distribution of this trend. Clearly, less than one repeat is not possible, whereas it is certainly possible for a datagram to percolate to the sink through more than three paths; we presume that with reducing activity level the percolation options, and hence repeats, will dwindle. However, as we will discuss below, it would appear that the average path length is not dramatically affected by the activity level.

Even for a rather small network the path complexity produces some rather unexpected results. In Figure 4 we show the round-up for the third set at 95% activity level. Given the actual connectivity levels shown, it can be seen that starting from node 102 (for the sake of visual simplicity mark nodes 101 and 102 as nodes 1 and 2 respectively) we can create two 6-hop node paths; but then, searching 5-hop and 5-hop paths gets more difficult. One way to explore this complexity is by adopting the matrix storage system described earlier; we use this method in Figure 5 to show the 3-4-5 and 6-hop adjacency matrices for the complete concatenated data set.

Table 2 Datagram arrival multiplicity and failure rates.

Percentage activity level	Datagrams					Failure rate % of total datagrams	
	Total received per set	Repeat arrivals	Mean repeat arrivals	Lost	Total sent	Per Set	Mean
100	1892	2.97	2.88	11	647	1.7	3
	431	2.87		7	157	4.4	
	3757	2.8		50	1391	3.6	
95	1741	2.85	2.27	88	699	12.6	19
	1028	2.04		192	696	27.6	
	563	1.92		61	355	17.2	
90	1160	2.54	2.66	228	685	33.3	36
	943	2.68		199	551	36.1	
	573	2.77		125	332	37.7	
85	992	2.55	2.14	299	688	43.5	40
	319	1.64		157	351	44.7	
	422	2.22		94	284	33.1	
80	455	2.41	2.37	207	396	52.3	55
	571	2.41		330	567	58.2	
	570	2.30		309	557	55.5	
70	1596	2.18	2.00	2568	3299	77.8	81
	346	1.75		799	996	80.2	
	262	2.07		707	833	84.9	

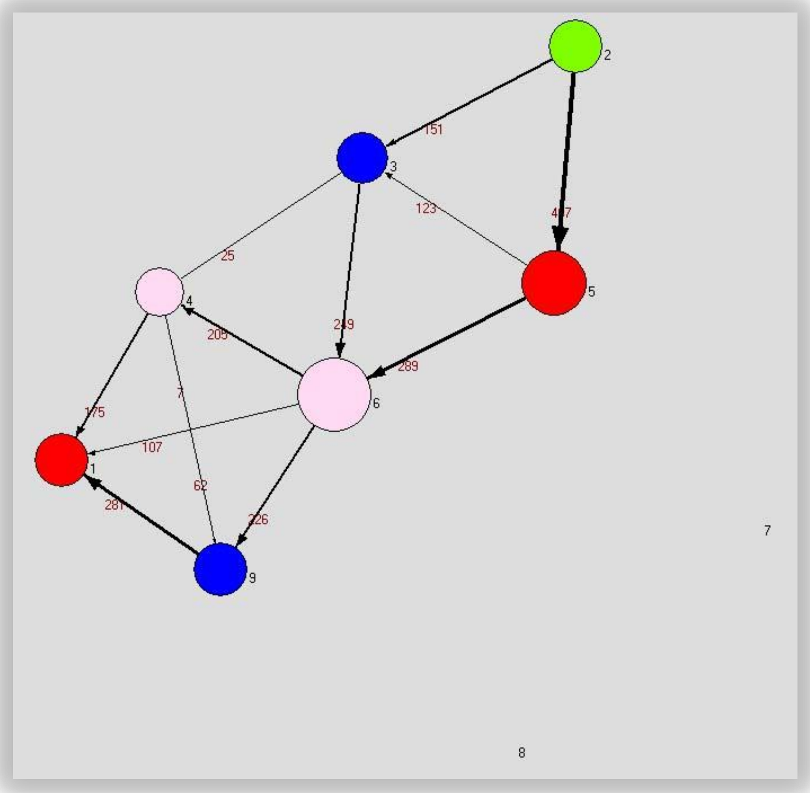

Figure 4 Example of accrued messages for network corresponding to third set of 95% (see Table 2 and Table 3) Node 2 and 1 are starting and end nodes respectively, nodes 7 and 8 were not used and hence are shown as isolates (<u>in the figure we refer to nodes 101 and 102 as nodes 1 and 2 respectively</u>).

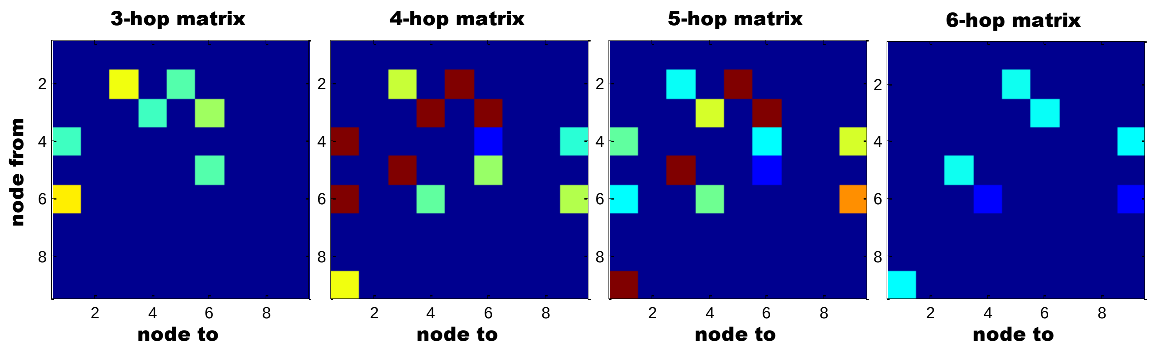

Figure 5 Adjacency matrices of the concatenated percolation series.

The matrices show the weighted asymmetric nature of the cumulative percolation flows. Dark blue means there is no connection between nodes for the given hop length. Light blue, cyan green, yellow and red are highest. Red shifts indicate a higher weighting and hence a more pronounced use of the connected nodes. The 4-node matrix has the highest density of reds, indicative of the observation that 4-hop nodes are the most common, followed by the 5-hop and 3-hop. The rarest combinations are the 6-hop paths which appear as light blue and cyan. Nodes 7 and 8 are inactive and hence are always blue. We also note that node 2 is always a 'node from' (rows), whereas node 1 is always a 'to node' (columns). All other nodes can be either case reflecting that they both receive and re-broadcast information through the network. (<u>for the sake of visual simplicity we refer to nodes 101 and 102 as nodes 1 and 2 respectively</u>).

LATENCY AND PATH LENGTH

Here we refer to latency as the time it takes for a datagram to percolate from source to sink. The mean latency is about 8 seconds with a standard deviation of the order of 3 secs. It would seem that latency is not affected by the activity cycle. Nevertheless, after a drop in latency after 100%, for the 80 and 70% activity levels, the path lengths appear to be consistently longer and the concomitant latency lasts a little bit longer too. Perhaps for more complex systems, capable of offering many alternative routes, this may become more noticeable. This result could be consistent with the conjecture presented in (16) that the elimination of shorter paths induces the execution, wherever possible, of longer paths for shorter duty cycles. However, whereas this effect might be more obvious in larger complex sets where permutations for long walks are available, in our example, given the small size of the node set, this effect is not very pronounced.

Table 3 Path lengths and latency.

Percentage activity level	Statistical Path lengths		Latency (secs)	
	Set	average	X Set	Mean
100	4.12	4.2	7.98	8.3
	4.2		8.46	
	4.3		8.48	
95	4.12	4.03	7.99	7.9
	3.86		7.56	
	4.10		8.15	
90	4.05	4.05	7.83	8.1
	4.24		8.49	
	4.02		7.84	
85	3.99	4.05	7.75	8.1
	4.06		8.27	
	4.11		8.20	
80	4.03	4.15	8.08	8.3
	4.12		8.04	
	4.31		8.69	
70	4.27	4.37	8.44	8.5
	4.29		8.23	
	4.54		8.96	

PROBABILITY OF PERCOLATION SUCCESS

In the previous sections we discussed the combinatorics of the possible percolation paths. Here we turn our attention to the probability of successful percolations. We noted in the results section that the average path length for successful percolations was of the order of 4 steps. In part this is because the possibility of shorter paths is limited by the short range of the radio transmitters on the nodes. Also, the nodes are so disposed, and their number not so large, that most paths will travel along nodes 3 5 and 6. This means that many percolations consist of combinations starting from node 2 along nodes 3 and 5 and then either 6 or 4, before ending up in node 1. Taking a very simple chain-line system consisting of just five nodes (4 hops), the probability of unsuccessful percolation (failure) is given by

$$(4)\ P_f = 1 - p^n ,$$

where p is the percentage of the active duration cycle and n the number of hops. It can be seen from Table 4 that the data from the experiments matches quite well the failure rate for the 4-hop cycle which, in turn, is close to the average path length of the experimental cycles. In a previous study (16) we simulated the percolation of multi-hop communication networks assuming a transfer protocol based on that used in the field tests presented here. The study consisted of simple chain and regular grid topologies amenable also to mathematical analysis using simple probabilistic assumptions, from where we draw our conjecture above.

PATHS REALISED

In Table 5 we list the path cardinality and frequency for each set as well as the overall concatenated group spanning all the activity levels. The data serve as a cross-reference for the statistical analysis shown in Table 3. From Table 5 we can see that from the concatenated set, the total number of unique paths is 33, i.e., about 10 % of the total combinatorial possibilities (325). We also note that the highest variety of paths is not obtained for 100% activity but rather at the 95 and 90 percent activity levels—averaging 23 and 20 each. In all cases we are assuming that, even if the total number of *datagrams* of each set is not equal (see Table 2Error! Reference source not found.), we consider that the number is sufficient in all cases to conform to an acceptable statistical steady state.

For the 100% cases, although the mean path length is not substantially different from the others, there appears to be no single instance of a 6-hop path. In fact, 6-hops are very rear as they account for no more than a few percent of the total (at most 9.5 % for the 85 % activity level, which, coincidently has the lowest unique path count=9). What is consistent in all sets is the preponderance of 4-hop paths, which corroborates our earlier statistical findings in Table 3 . Based on these findings it is acceptable that the comparison between the 4-hop and field failure rates presented in Table 4 supports the assumption of parallel flows.

Table 4 Comparison of failure rates for theoretical and field trials

Duty cycle %	Failure rate %				
	Theoretical as a function of hops				Field
	3-hops	4-hops	5-hops	6-hops	
100	0	0	0	0	3
95	14	19	23	26	19
90	27	34	41	47	36
85	39	48	56	62	40
80	49	60	67	74	55
70	65	76	83	88	80

Table 5 Cardinality of unique paths observed from field trials.

Activity level %	Unique paths		Nr hops as fraction of total paths			
	Nr. Paths	Mean Paths	Nr. 3-hop	Nr. 4-hop	Nr. 5-hop	Nr 6-hop
100	12	11	0.144	0.587	0.268	0
	10		0.1	0.59	0.3	0
	12		0.03	0.63	0.33	0
95	21	23	0.21	0.46	0.32	0.006
	28		0.31	0.52	0.16	0.004
	19		0.14	0.63	0.22	0.009
90	21	20	0.22	0.52	0.25	0.01
	20		0.17	0.46	0.31	0.05
	18		0.24	0.51	0.23	0.02
85	18	14	0.25	0.53	0.20	0.016
	9		0.36	0.30	0.24	0.094
	16		0.25	0.44	0.24	0.06
80	13	16	0.23	0.52	0.22	0.02
	18		0.22	0.48	0.26	0.03
	17		0.08	0.54	0.35	0.02
70	20	17	0.1	0.55	0.32	0.026
	19		0.1	0.53	0.33	0.04
	13		0.008	0.49	0.46	0.05
Concatenated	33	N/A	0.15	0.54	0.29	0.015

CONCLUSIONS AND NEXT STEPS

We have conducted field studies of how the success rate and latency of the realization of possible communication-path combinations in short-range, multi-hop, wireless sensor networks are affected by the duration of the activity duty cycle.

In general terms, the success rate is consistent with the observed percolation topologies and the associated path lengths. Moreover, we have seen that even a simple probabilistic model is capable of predicting, to acceptable accuracy, the general percolation trend assuming parallel chain links from source to sink nodes. What needs to be studied in further field studies is to corroborate our previous numerical findings on percolation through larger complex grids, and the manner in which percolation is possible when multiple alternate paths are available.

Our results show that whereas it is possible to realize paths of up to six hops in length, the most probable and stable paths chosen are primarily 4-hop (over 50% overall). Another interesting feature is that whereas the success rate is clearly affected by the duty cycle length, the latency remains reasonably constant, of the order of 2 secs per hop, even for activity cycles as low as 70%.

Another important aspect concerns how environmental conditions such as the weather or the presence of electromagnetic fields (even not particularly strong ones) interfere with connectivity. In addition to the extended statistical analysis of the field studies presented herein, we shall be extending our mathematical analysis to include computer simulations using the JRC's own wireless sensor network simulation programme (15), to both calibrate our models and to simulate the deployment of such sensor fields over a wider area. Finally a key factor in the future development of such systems is, of course, their vulnerability to hacking (17).

ACKNOWLEDGEMENTS

Flavio Bono for the graphic interpretation of the percolation diagram shown on front cover.

BIBLIOGRAPHY

1. *Social network analysis for information flow in disconnected delay-tolerant MANET's.* **Daly, E.M., Haahr, M.,.** 5, May 2009, IEEE Trans. on Mobile Computing, Vol. 6, pp. 606-621.

2. *Environmental studies with the sensor web: Principles and practice.* **Delin, K.A. et al.** 2005, Sensors, Vol. 5, pp. 103-117.

3. *Five years of designing wireless sensor networks in the Doñana biological reserve (spain): An applications approach.* **Larios, . D., et al.** 2013, Sensors, Vol. 13, pp. 12044-12069.

4. *A survey of geosensor networks: Advances in dynamic environmental monitoring.* **Nittel, S.** 2009, Sensors, Vol. 9, pp. 5664-5678.

5. *Modelling and energy consumption evaluation of a stochastic wireless sensor network.* **Zhang, Y., Li, W.** 2012, EURASIP J. on Wireless Communications and Networking, Vol. 2012, p. 282.

6. *An estimation of sensor energy consumption.* **Halgamuge, M.N., Zukerman, M., Ramamohanarao, K.** 2009, Progress in Electromagnetic Research B, Vol. 12, pp. 259-295.

7. *Energy conservation in wireless sensor networks: A survey.* **Anastasi, G., Conti, M., Di Francesco, M., Passarellla, A.** 2009, Ad Hoc Networks, Vol. 7, pp. 537-568.

8. **Morais Cordeiro C., Agrawal D.P.** *Ad Hoc Sensor Networks: Theory and Applications.* Singapore : World Scientific, 2011.

9. *Wireless sensor networks: a survey.* **Akyildiz, I.F. Su, W., Sankarasubramaniam, Y., Cayirci, E.** 2002, Computer Networks, Vol. 38, pp. 392-422.

10. *A survey on routing protocols for wireless sensor networks.* **Akkaya, K., Younis, M.** 2003, Vol. 3, pp. 325-349.

11. *Efficient multi-path protocol for wireless sensor networks.* **Li, S., Neelisetti, R.K., Liu, C., Lim, A.** 1, 2010, Intl. J. of Wireless & Mobile Networks (IJWMN), Vol. 2, pp. 110-130.

12. *Efficient broadcasting in multi-hop wireless networks with a realistic physical layer.* **Wong, G.K.W., Liu, H., Chu, X., Leung, Y.W., Chun, X.** 2011, Ad Hoc Networks, Vol. 11, pp. 1305-1318.

13. *Strategies and techniques for node placement in wireless sensor networks: A survey.* **Younis, M., Akkaya, K.** 2007, Ad Hoc Networks, Vol. 6, pp. 621-655.

14. **Renaldi, G., Gutierrez, E.** *Wisp: s networkable autonmous sensor-equipped device to conduct experimental tests on sensor networks.* IPSC, Joint Research Centre. Ispra : European Commission, 2015. JRC99842.

15. **Renaldi, G.** *Dylink: a software to simulate container alarm detection triggering signal propagation through wireless networks in a container port scenario.* IPSC, Joint Research Centre. Ispra : European Commission, 2014.

16. **Strozzi, F., Renaldi, G., Gutiérrez, E.** *Network Segmentation and Spanning Sets, JRC 99540.* Joint Research Centre. Ispra : European Commission, 2015.

17. *Tampering woth Motes: Real-World Physical Attacks on Wireless Sensor Networks.* **Becher, A., Benenson, Z., Dornseif, M.** [ed.] J.A. Clark et al. Berlin Heidelberg : Springer-Verlag, 2006. pp. 104-118.

LB-NA-28267-EN-N

JRC Mission

As the science and knowledge service of the European Commission, the Joint Research Centre's mission is to support EU policies with independent evidence throughout the whole policy cycle.

EU Science Hub
ec.europa.eu/jrc

@EU_ScienceHub

EU Science Hub - Joint Research Centre

Joint Research Centre

EU Science Hub

Publications Office

doi:10.2788/610616

ISBN 978-92-79-64109-1