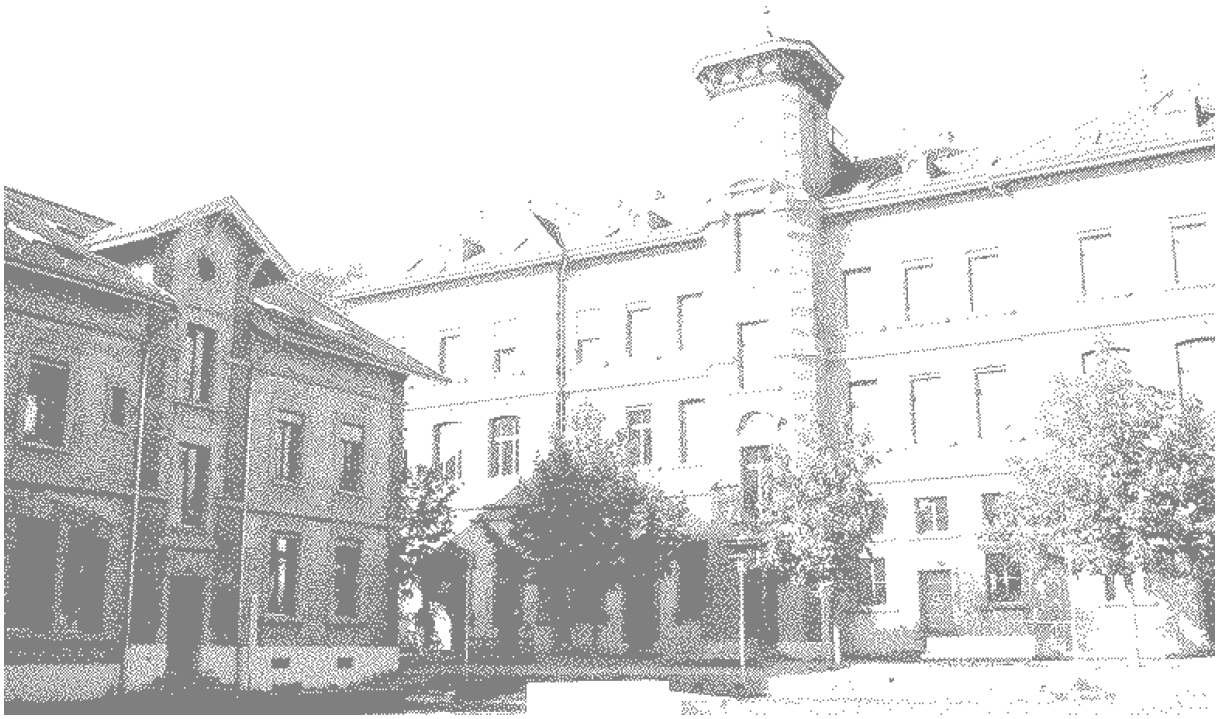
UNIVERSITÄT
BERN

# Data filtering and aggregation in the Location Based Analyser testbed

**I.F.R. Noppen, D.C. Dimitrova, T. Braun**

Technischer Bericht IAM-11-004 vom 1. Februar 2012

Institut für Informatik und angewandte Mathematik, www.iam.unibe.ch



# Data filtering and aggregation in the Location Based Analyser testbed

**I.F.R. Noppen, D.C. Dimitrova,T. Braun**

Technischer Bericht IAM-11-004 vom 1. Februar 2012

**CR Categories and Subject Descriptors:**
C.2.3 [Computer-Communication Networks]: Network Operations; C.2.4 [Computer-Communication Networks]: Distributed Systems

**General Terms:**
Algorithms, Measurement, Performance

**Additional Key Words:**
sensor networks

Institut für Informatik und angewandte Mathematik, Universität Bern

# Abstract

The main challenge in wireless networks is to optimally use the confined radio resource to support data transfer. This holds for large-scale deployments as well as for small-scale test-environments such as test-beds. We investigate two approaches to reduce the radio traffic in a test-bed, namely, filtering of unnecessary data and aggregation of redundant data. Both strategies exploit the fact that, depending on the tested application's objective, not all data collected by the test-bed may be of interest. The proposed design solutions indicate that traffic reduction as high as 97% can be achieved in the target test-bed.

# Contents

# 1 Introduction

Wireless sensor networks provide excellent means for monitoring and data gathering in a large range of application areas. One such application is the use of radio-enabled sensor nodes for (indoor) positioning in which the sensor nodes collect signal measurements of user devices using radio transmissions, e.g., Bluetooth. For outdoor localisation of devices, global positioning system (GPS) is widely accepted as the de facto standard. For indoor localisation, however, various technologies and localisation techniques have been proposed by the research community. Among the most used radio standards are the IEEE 802.11 (with commercial name WiFi) and Bluetooth standards. Processing of the collected measurements can derive the location coordinates of the transmitting device. Potential use cases of a positioning application include, but are not limited to, analysis of visitor behaviour in shopping malls, tailored discount dissemination in attraction parks and evaluating staff efficiency in hospitals.

Inspired by the many use case opportunities, the Location Based Analyser (LBA) project addresses the indoor localisation challenge by leveraging radio frequency (RF) based technologies, namely WiFi and Bluetooth, see [1]. More specifically, we use multiple sensor nodes at known positions to collect measurements on the received signal strength indicator (RSSI) from personal devices on the premises. The collected measurements are periodically sent to a central database server where they are sorted per observed device and processed to determine the current position of each device. There are various techniques to map RSSI to distance, the most often cited being (multi)lateration and fingerprinting

As part of the development process we set up a test-bed for the purpose of testing the implementation and performance of the developed localisation system. Typically, test-beds are designed at a smaller scale than the final, deployed system but even then challenges related to the access to and congestion of the radio medium arise. In order to ensure undisrupted operation and system scalability one needs to take care when managing the radio resources. A technically advanced approach towards resource management is the use of contention-based mechanisms for medium access. Another, more intuitive, approach is classifying the wireless traffic and identifying what data is pertinent for the needs of the application. In this approach strategies such as filtering of unnecessary data, aggregation of redundant data and data compression can be helpful. In particular, data aggregation in WSNs has been largely studied [2, 3, 4] and evaluated in WSN testbeds [5, 6, 7] for the purposes of reducing traffic volume and

energy consumption.
This report describes the LBA test-bed and how we adopt filtering and aggregation to minimise wireless traffic in the test-bed. Contrary to other studies, e.g., [3], which address hierarchical aggregation in the network, we are only interested in a local (on a single node) aggregation. We consider the requirements of a localisation system and the limited resources of sensor nodes. Typically, sensor nodes do not have the same processing and energy resources as normal desktop- or server machines. Storage capacity is also often limited, posing the need implementation decisions to be taken carefully.

# 2 Localisation WSN test-bed

We designed an indoor localisation system that relies on sensor nodes at known positions, which collect signal measurements from personal devices. A central server processes the collected measurements to derive the location of the devices. We are interested in signals from personal devices and in signals from the sensor nodes. The latter are used to monitor the quality of the radio channel and to improve the performance of the system.
In order to test and evaluate the system we built inside a single room a test-bed which reflects the system design. The test-bed contains 16 sensor nodes, which form a 4x4 grid at 0.5 meter below the room ceiling, see Figure 1. The sensor nodes scan continually for WiFi and Bluetooth signals and record the RSSI levels. Periodically, each sensor node sends its measurements to a gateway node, which collects all measurements and forwards them to the database server. At the server the measurements are stored and analytically processed. The RSSI is directly related to received signal strength and therefore believed to allow the retrieval of distance. In the rest of the paper the term sensor node and sensor are used interchangeably.

## 2.1 Test-bed challenges

One the one hand, the test-bed faces challenges related to the efficient use of the radio channel. In our proposed localisation system the amount of wireless traffic depends on the number of deployed sensor nodes, the frequency at which measurements are reported, and on the number of detected or tracked devices. Additionally, there is interference traffic from

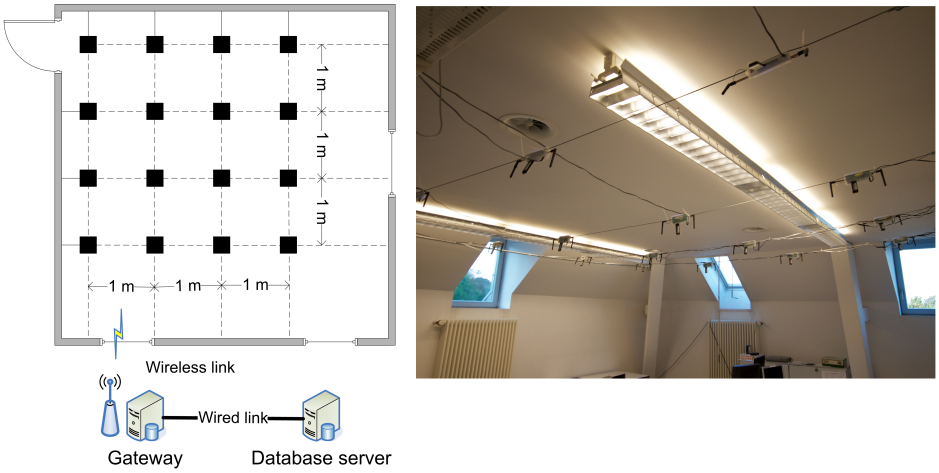



Figure 1: LBA test-bed architecture: multiple sensor nodes, connected over WiFi to a gateway. Measurements are stored in a database server.

other devices that may use the same wireless medium. Since the test-bed is located in the Computer Science building of the University of Bern, there are other experimental wireless networks and wireless access infrastructures that use the same radio channels. We consider traffic from such networks and WiFi access points, e.g., beacons, as non-informative since measurements on it contribute neither to the localisation of devices nor to the radio channel monitoring. Therefore, these are unnecessary measurements and while we cannot always avoid their collection we can prevent their transmission to the central server. To this end we apply filtering.

Another factor that directly affects the busyness of the radio channel is the proportion of collected measurements per device. For each device several measurements may be collected by one sensor while a single measurement would be sufficient for localisation. Clearly, reporting all measurements would be redundant and we would like to minimise the radio channel utilisation. This can be achieved by decreasing the amount of data to send. Therefore, we explore the use of data aggregation.

On the other hand, we should also take into account the limited resources of the sensor nodes. Sensor nodes are typically not as powerful as normal desktop- or server machines, leaving a specific set of constraints for the sensor software to operate within. Typically, the available memory in such nodes is limited, forcing us to rethink what information is important to store in-memory and how we can keep memory usage as low as possible. For example, using the file-system as a temporary storage for often used data structures is not appropriate because a file-system normally resides on a

relatively low speed flash memory (flash memory is limited in the number of write sequences).

## 2.2 Test-bed implementation

As sensor nodes we used Gumstix Overo Fire devices [8] with integrated on-board Bluetooth and WiFi interfaces, which support the scanning of the medium. Moreover, we attached an additional WiFi card, which is used for communication between a sensor and the gateway.
On the sensor nodes, we use a light Linux kernel and several lightweight packages to keep the sensor end of the system as lightweight as possible. This sensor software is built with the Administration and Deployment of Adhoc Mesh (ADAM) framework [9, 10] which includes the custom packages needed for running the Bluetooth and WiFi scanners. WiFi measurements are collected by capturing packets with *libpcap*, an application programming interface for capturing network traffic, on the wireless interface and by hopping through the WiFi channels with an interval of one second. Bluetooth measurements are recorded by using output from the *bluez* Linux Bluetooth library. In both cases, when the measurement buffer is full or a predetermined period has ended, the measurements are flushed to the gateway. This gateway also periodically flushes the measurements to a database server for the purpose of storage and analytical processing.
The gateway and the database server both run on a regular Linux distribution on a standard desktop machine. The gateway can also reside on a sensor node, as long as there is a wired connection available for the communication to the database server. At the database server, we use a regular MySQL database to store all the measurements. The gateway and the database server communicate through a SOAP service.

# 3 Filtering

In this section we discuss filtering as a way to deal with unnecessary measurements. Filtering is a method that can successfully omit the collection and transmission of unnecessary measurements. In order to select a filtering solution that best fits the objectives of our test-bed we need to answer three *design questions*: what to filter, where to filter and how to filter.

## What to filter

We need a filtering solution that can identify unnecessary measurements and only allow the transmission of measurements on user devices (used for localisation) and reference sensor nodes (used for channel estimation). As discussed earlier, we consider as unnecessary the measurements on signals from experimental wireless networks and WiFi infrastructure, e.g., access points (APs). We refer to the former group as 'always-on devices' and to the latter - as 'fixed infrastructure'. Each group requires different filtering strategies as it is explained later.

## Where to filter

There are three places in our system where we can employ filtering: the sensors, the gateway and the database server. Filtering at the sensors has a direct impact on buffer occupation and on wireless traffic. At the same time, sensors generally have limited resources and it is challenging to decide on what to blacklist, for which we need large sets of measurements. The only benefit of filtering at the gateway is that there is less data transfer between the gateway and the database server. However, since they are usually connected through wired connections, the bandwidth here is not a problem. Moreover, the gateway does not store any data and is thus also not appropriate for the decision process. On the database server, we have both the capacity and the measurements at our disposal to support a decision making for filtering. Hence, it is a more appropriate for the blacklist decision process.

## How to filter

Filtering can be based on black- or whitelisting of certain MAC addresses. When a certain MAC address is blacklisted, all measurements of that MAC

address are discarded. When a MAC address is whitelisted, all measurements related to it are collected. Whether black- or whitelisting is the best strategy for a project, depends entirely on the setup and scope of the project. When we have a controlled testbed for instance and we only want to see measurements from a limited set of MAC addresses, whitelisting of these addresses is the obvious choice. In that case all other measurements would be discarded and our set of measurements would only contain data we are interested in. In real life situations and larger experiments, however, whitelisting is not feasible because we do not know the MAC addresses beforehand and the task of collecting this data would be cumbersome.
Another classification criteria is how the decision what to filter is taken. If we collect the MAC addresses and enter them manually into the filtering system we use *static filtering*. Static filtering is time consuming, requires effort and does not scale well. A better alternative is *dynamic filtering*, which introduces certain intelligence in the system. Such system integrates decision making processes to analyse incoming measurements and decides what MAC addresses to filter out.

## 3.1 Filtering solution

Taking into account the requirements of the current experimental test-bed we chose a dynamic blacklisting strategy with static elements and static whitelisting support.

### 3.1.1 Static blacklisting

Static blacklisting refers to the filters that are directly installed at the sensor to filter out measurements of signals from the fixed infrastructure (APs). An AP contributes significantly to the wireless traffic because (i) it typically sends a beacon message every 100ms and (ii) it serves multiple clients in parallel.
Filtering of the fixed infrastructure is quite easily done at the sensor nodes. Since they already use *libpcap* to capture packets at the WiFi interface, we only need to create an additional rule to discard all packets with the type 'Beacon' or with distribution system (DS) flags 10 or 11. The DS flag in a WiFi packet [11], shown in Figure 2, indicates the traffic pattern. Flags 10 and 11 indicate traffic from an AP to a mobile device or another AP respectively. Flags 00 and 01 indicate traffic from a device to another station or an AP respectively. Hence, by filtering the flags 10 and 11, we

can directly disable measuring most of the fixed infrastructure traffic. Static blacklisting is implemented using the existing *libpcap* functionality.

| 2 | 2 | 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| Protocol version | Type | Sub-type | To DS | From DS | More flag | Retry | Power mngt | More data | WEP | Order |

Figure 2: Format of the MAC header of a WiFi packet frame.

### 3.1.2 Dynamic blacklisting

Static blacklisting on top of *libpcap* is not feasible for the identification of always-on devices that behave as any other device but are active continually or for long periods of time. Instead we use a dynamic blacklisting technique that combines a decision making process, which periodically generates blacklists, and a dissemination process, which distributes the blacklists to the sensor nodes.

### 3.1.3 Decision making

The decision making process is situated on the central server and is responsible for the generation of the blacklists - one per each sensor node. The process relies on one commonality between all always-on devices, namely, they are generally connected 24/7. Therefore, if we analyse the collected measurements over a long period we should be able to identify always-present MAC addresses that correspond to always-on devices. Besides the previously mentioned experimental networks (in our specific case), examples of always-on devices are devices forming an ad-hoc network for a long period or large displays with a thin client fixed to them for advertisement purposes (e.g., in a mall).
Formally a device in our test-bed can be identified by its MAC address and *activity level*, i.e., the percentage of time in which measurements of its MAC address were received. The activity level is calculated over a specific *evaluation period*, which is the timespan over which the list of blacklisted MAC addresses is generated. For example, if a device was active for two hours within an evaluation period of eight hours it has an activity level of 25%. If we define an activity level threshold and a device's activity level is above it we can deduce that this is an always-on device. The choice of the threshold is very important and related to the duration of the evaluation period. For instance, it is fair to say that a threshold of 80 or 90% should allow the identification of always-on devices.

An easy way to calculate the activity level would be to count the number of unique timestamps when a certain MAC address was registered and divide this by the total number of seconds in the evaluation period. However, since our WiFi scanner hops channels every second, this will not have the desired effect. Furthermore, it is possible that a 'fixed device' is only connected to the network and not actively transmitting data, in which case only few measurements will be collected of the device. To correct for this, we divide the evaluation period into *activity periods* of the same length. Per MAC address, we check within each activity period whether there is at least one unique timestamp of this address. If this is the case we mark the period with true, otherwise we mark it with false. If we now count the number of 'true' activity periods and divide that by the total number of activity periods in the evaluation period (equation 1) we will get the percentage of time that this MAC address has been active. We can derive the number of activity periods by dividing the evaluation period by the activity period, both measured in seconds.
A simple comparison of the activity level threshold with the activity levels of all MAC addresses detected within the evaluation period will give us the MAC addresses to include in the blacklist.

$$ActivityLevel = \frac{count(ActivityPeriod\_True)}{EvaluationPeriod/ActivityPeriod} \quad (1)$$

### 3.1.4 Dissemination

The dissemination of blacklists is pull-based. The procedure is shown in Figure 3. The sensors request the blacklists from the central database server via the gateway node. The server can answer, also via the gateway, either with a new blacklist, when available, or with an empty message, when the sensor polled too early and no update is available yet. Note that the new blacklist from the server can contain no MAC addresses when there are none to filter out. If a sensor should blacklist certain MAC addresses it filters out their measurements but keeps statistics on each address. Periodically this information is sent back to the server, where it is used to re-evaluate whether the MAC address should stay blacklisted. Without these statistics the decision support process will lead to a repetitive adding and removing of MAC addresses to the blacklist.
The dissemination procedure is implemented by extending the test-bed functionality and introducing three new message types, namely, blacklist request, blacklist update and blacklist aggregate messages, see Figure 3. Either on start-up or after a timer expires the sensor nodes request

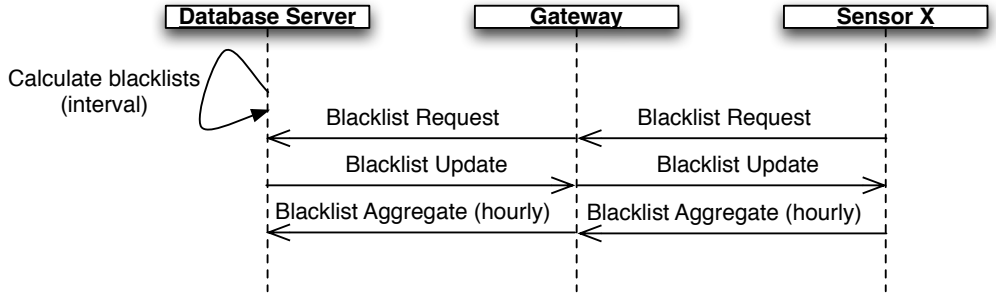
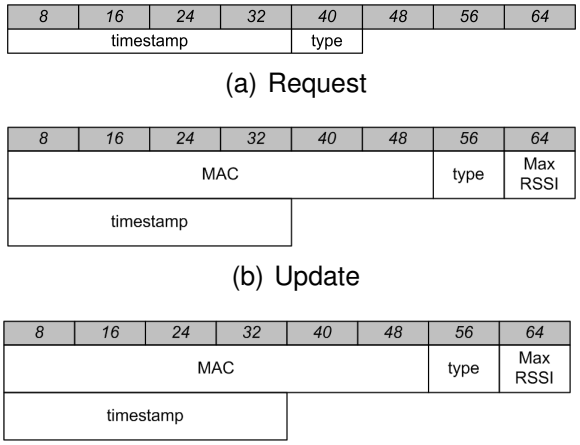



Figure 3: Message exchange between sensors, gateway and database server regarding the blacklist




(a) Request

(b) Update

(c) Aggregate

Figure 4: Message types for filtering with blacklists.

a blacklist through the gateway using a *blacklist request* message, see Figure 4(a). The message contains a *timestamp* of the current blacklist at the sensor and the *type* of blacklist the node is requesting (Bluetooth or WiFi).

The server answers with a *blacklist update* message, see Figure 4(b), which contains the *timestamp* of the list in the update, or the timestamp of the blacklist request if no newer update is available. The *checkback-time* field suggests how many seconds will pass between the timestamp and the time a new blacklist will be available. The *list size* field tells us how many MAC addresses the complete blacklist contains. The *update* flag indicates if a new list is sent (flag 1) or if the sensor polled and there is no

update (flag 0). When there are no MAC addresses to blacklist the flag is 1 (true) but the list size is zero and the MAC addresses field is empty. The length of the MAC addresses field for a non-empty list depends on the list size field.
Upon receiving the blacklist update message the sensor replaces the old blacklist with the new one and resets the timer according to the checkback-time field. For each blacklisted MAC address the sensor collects statistics and reports them hourly back to the server in a *blacklist aggregate* messages. The blacklist aggregates packet, shown in Figure 4(c), contains one or more structures depending on the number of blacklisted MAC addresses. Each structure contains the MAC address of the blacklisted device along with the first-seen and last-seen timestamp, the number of measurements between the two timestamps and the average RSSI. The count is used to make a decision whether a MAC address has to remain blacklisted.

### 3.1.5 Static whitelisting

Static whitelisting is used to ensure the collection of measurements on the anchor sensor nodes. As explained, these measurements are used for channel evaluation in the test-bed. For that purpose the MAC addresses of all sensor nodes are identified and a specific whitelist for each sensor is kept at the central server. The server is responsible to check that a MAC address from the whitelist does not get blacklisted.

## 3.2 Experimental analysis

In this section we present results on the data reduction that filtering can bring but first we discuss some parametrisation issues.

### 3.2.1 Parametrisation

Integrating the proposed filtering solution requires setting up some parameters such as the blacklist evaluation period and the activity period. For our purposes we selected an evaluation period of 24 hours, which aligns easily with human activity. Choosing a good value for an activity period is more challenging. In order to analyse this, we set up a test, where we included a fixed WiFi device (laptop) in idle mode into the test-bed. The device was only connected to a wireless network with no data traffic exchange. We let the sensor nodes collect measurements over 65 hours and

calculated the activity level of the idle and the most frequently seen device at each of the sensor nodes for an activity period of 60 and 300 seconds. Corresponding box plots over all sensors are given in Figures 5.

A successful deployment should be able to filter out the idle device's MAC address as well as other high activity MAC addresses (most seen device). As we can see in Figure 5, an activity period of 60 seconds will not lead to a successful identification of the idle device as 'fixed' since its activity level reaches only about 42% on average. When we change the activity period to 300 seconds the activity period of the idle devices rise up to 90% and it can be easily identified for blacklisting. The reason for the above behaviour is the idle status of the device in which case it communicates to the networks once every few minutes. Note that the most seen device is less vulnerable to short activity periods and easily reaches 80-90% of activity because it is actively transmitting.

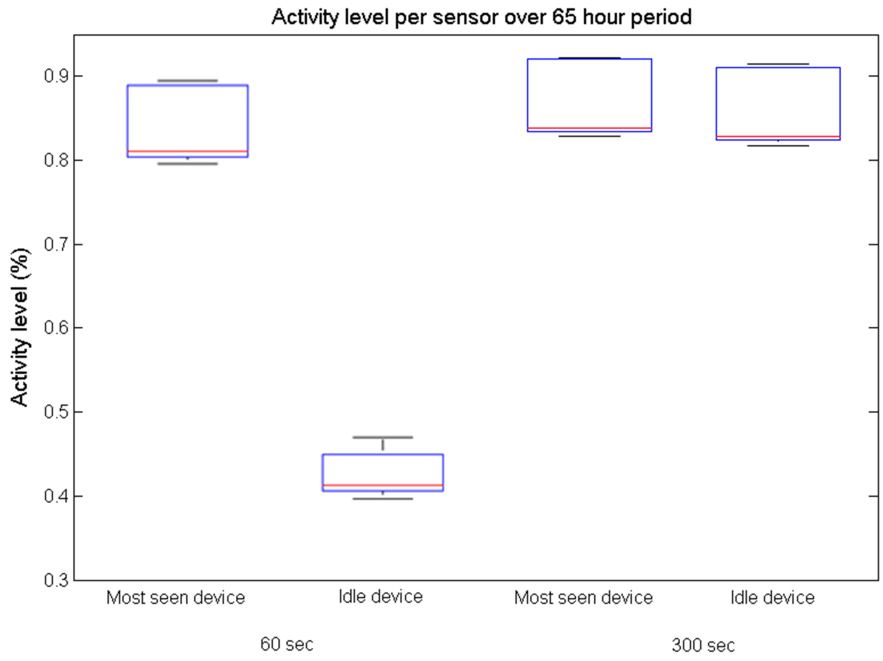



Figure 5: WiFi: Activity level over all sensors for the most active MAC address and an idling device when the activity period is 60 seconds

### 3.2.2 Traffic reduction

To quantify the gains in terms of reduced number of measurements we conducted the following experiment. First, the test-bed ran for full 24 hours, after which both Bluetooth and WiFi blacklists were generated for each node. Then, in a second 24 hours run no filtering was directly applied but the generated blacklists were used to calculate, for the same data set, what would the measurement reduction be. This provides us a common base for comparison since we are using the same data set. In the filtering decision the parameters are: $activity\ level > 0.8$, $activity\ period = 300$, $evaluation\ period = 86400$ seconds (24 hour).

Table 1 provides detailed statistics on the measurement reduction per sensor node. The reduction is the percentage of measurement numbers that can be saved if using filtering. Interestingly, the size of the generated blacklist is rather small although the test-bed location would suggest much larger wireless activity. We explain that with the fact that the experiments were conducted on a weekend when there are significantly less people, and hence always-on devices, in the building.

In terms of reduced measurement values the results show that the effect of filtering is significant. The reduction includes whitelisting of the MAC addresses of the other sensor nodes for reasons discussed earlier. For deployments where whitelisting is not needed the gains in reduction would be even bigger. This trend is better visible in Figure 6 for WiFi - the mean measurement reduction per sensor without whitelisting is about 93%, more than 10 percentage points higher than the mean reduction with whitelisting.

For Bluetooth we registered even higher measurement reduction with 99.76% on average. We explain that with the smaller (six times) proportion of Bluetooth devices in our test-bed environment compared to the number of WiFi devices. As result one Bluetooth MAC address contributes more to the total number of measurements. Note that whitelisting is not included because there are no addresses to be whitelisted for Bluetooth (we do not use Bluetooth signals in channel characterisation).

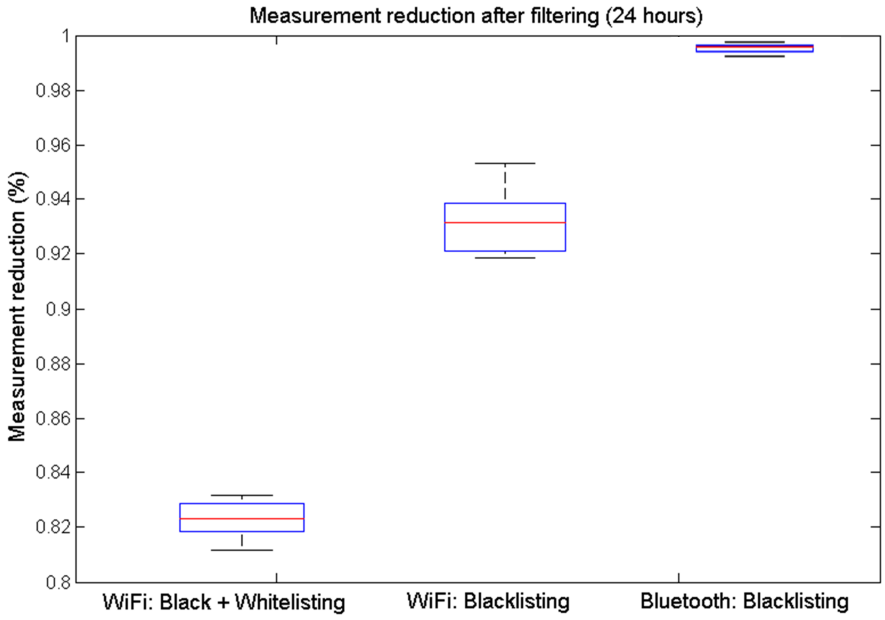



Figure 6: Measurement reduction comparison over all sensors after filtering

| Sensor | Total measurements | Reduced measurements | Reduction percentage | Blacklisted MACs |
|---|---|---|---|---|
| 1 | 408.453 | 74.569 | 81.7% | 26 |
| 2 | 458.547 | 83.204 | 81.9% | 26 |
| 3 | 425.139 | 76.017 | 82.1% | 26 |
| 4 | 416.985 | 72.839 | 82.5% | 25 |
| 5 | 432.813 | 72.880 | 83.2% | 27 |
| 6 | 404.987 | 71.233 | 82.4% | 26 |
| 7 | 418.635 | 74.164 | 82.3% | 25 |
| 8 | 412.666 | 76.420 | 81.5% | 25 |
| 9 | 441.695 | 83.182 | 81.2% | 26 |
| 10 | 390.836 | 70.235 | 82% | 26 |
| 11 | 394.006 | 71.564 | 81.8% | 26 |
| 12 | 441.728 | 78.060 | 82.3% | 25 |
| 13 | 427.409 | 72.928 | 82.9% | 26 |
| 14 | 364.902 | 62.817 | 82.8% | 26 |
| 15 | 268.988 | 63.144 | 82.9% | 25 |
| 16 | 414.818 | 63.914 | 84.6% | 26 |

Table 1: Overview of WiFimeasurements in the second 24 hour period in the experiment.

# 4 Aggregation

Aggregation of data (measurements) is another strategy that can improve the utilisation of the limited radio resource and decrease the chances of collision. Generally speaking aggregation is a technique to decrease the amount of measurements sent over the wireless channel while retaining the measurements credibility. In sensor networks aggregation has been proposed to decrease energy consumption [4, 12] or network congestion [2]. We are interested in using aggregation to decrease network traffic and improve scalability since the current deployment of electrically powered sensor nodes does not face energy consumption challenges.

## 4.1 Aggregation mechanism

Several approaches towards data aggregation are possible. One strategy is to let the sensors report only changes in measured values. If the value which we monitor would remain stable for prolonged times, we could simply send one measurement each time the value changes. Considering that RSSI values are anything but stable and dependent on external influences like noise, shadowing and reflection, this method is not a suitable candidate for aggregating our measurements.
Another aggregation method is to send a single measurement per timespan where the timespan duration largely depends on the type of application. For example, for monitoring of ambient temperature one measurement per hour may be sufficient while for target tracking timespan in the order of few seconds is more appropriate. We have chosen for aggregation over a timespan. The choice of timespan duration is investigated in Section4.2.
Next, we need to decide which value to report. In the case of RSSI we expect that the maximum value would be best since it is the least affected by propagation conditions and mostly related to distance. Other possible choices would be the last measured RSSI, the average RSSI or another statistic registered over the aggregation timespan, e.g., standart deviation.
To enable the chosen aggregation strategy in the test-bed two buffers are set at the sensor nodes - one that collects all raw measurements and another that keeps the reported aggregate values. When the first buffer is full, or at the end of a reporting period, the measurements are processed and a single measurement per MAC address, containing the maximum RSSI, is written into the second buffer. The packet format of an aggregate message is shown in Figure 7. It contains a number of measurement records where

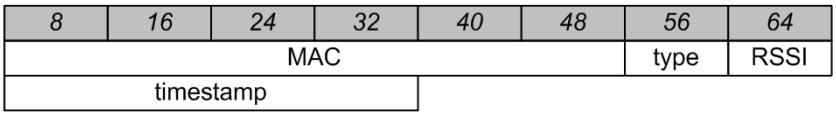
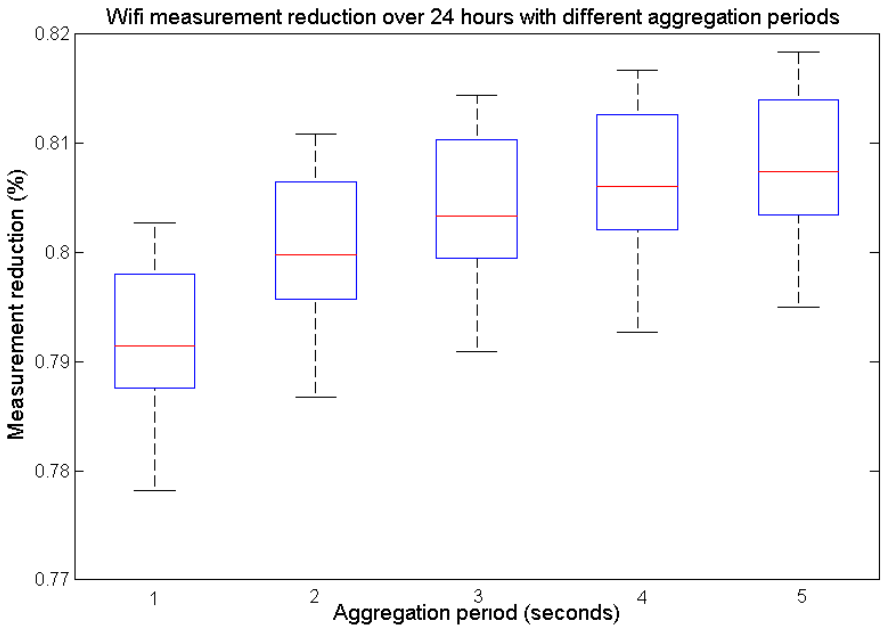



Figure 7: Packet format of an aggregate measurement.

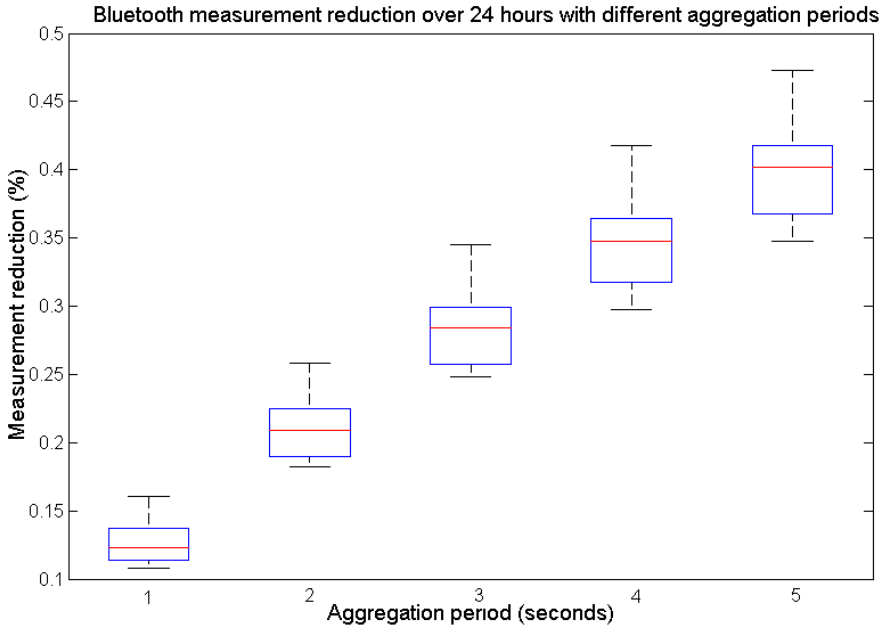
each measurement record contains the maximum RSSI, record-type (Wifi or Bluetooth), the timestamp of the measurement and the MAC address.
An alternative approach to decrease wireless traffic is data compression [13]. Instead of using the redundancy in measurements data compression gains from redundancy in the data itself by applying appropriate encoding. Although beneficial it also requires additional processing.

## 4.2 Experimental analysis

In order to analyse the measurement reduction we can get by applying aggregation, we used the experiment setup from the filtering experiment. We took measurements over the first 24 hours and calculated the measurement reduction if each sensor would apply aggregation. Since the current test-bed deals with tracking of moving objects we chose timespans of one to five seconds. The maximum RSSI value is reported since we believe it is least affected by propagation factors. The calculations were done for both WiFi and Bluetooth signals.
Figure 8(a) shows measurement reductions for all sensors over the 24 hour period for WiFi. We are able to reduce the number of measurements by more than 79% on average by aggregating over a one-second timespan. Increasing this timespan does only marginally improve the reduction to just shy of 81% (five-seconds timespan). The reason for this is two-fold: not all devices are broadcasting every second and in the WiFi scanning process channels are hopped every second . Because devices communicate with a network on one channel, we will not see their MAC addresses after this second again until we completed the cycle of channel-hopping.
For Bluetooth, the measurement reduction shows a different pattern. Figure 8(b) shows that the measurement reduction has an almost linear increase when we increase the timespan over which we aggregate the measurements. However, the number of measurements we collect for Bluetooth are far fewer than the measurements collected for WiFi. The 12% to 40% measurement will therefore have a less dramatic impact than is the case for WiFi. For Bluetooth we can choose an optimal value of the aggregation period depending on the application which suits our needs.




(a) WiFi




(b) Bluetooth

Figure 8: Measurement reduction over different timespans of aggregation.

# 5 Combined data reduction

While the individual measurement reduction of both filtering and aggregation shows great promise, it will be interesting to know if we can gain even more by applying both techniques in the same sensor network. To analyse this, we used the results of the estimated measurement reduction experiment for the filtering and calculated the total measurement reduction when we apply aggregation (with one-second timespan) on top of that.

Figure 9 shows the results of this analysis for WiFi measurements. For clarity we included the results for the filtering experiment as shown in Figure 6. As visible, we can achieve a higher measurement reduction when we apply blacklisting, whitelisting and aggregation than when just using black- and whitelisting. As expected, when disabling whitelisting the measurement reduction is even higher because more MAC addresses will be blacklisted. In our specific case, given we chose to apply a combination of black- and whitelisting, the combined reduction will be on average just shy of 94%.

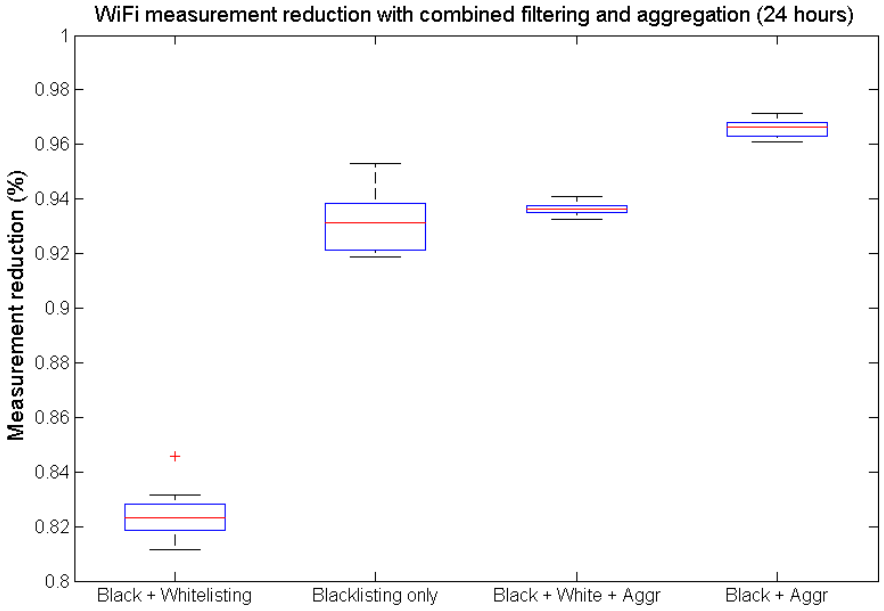



Figure 9: Combined Wifi measurement reduction over all sensors for combinations of blacklisting, whitelisting and aggregation

For Bluetooth, Figure 10 shows the results of combining filtering and aggregation. For reference, the graph for Bluetooth with filtering only is included. The graphs show that aggregation only slightly improve performance, the reasons being the few Bluetooth devices that the system detects and the efficient filtering of always-on devices that already brings measurement reduction of almost 99.8%. Although we are aware that the results are sensitive to the specific system deployment, we expect that aggregation will lead to smaller reduction in measurements for Bluetooth than for WiFi due to the lower number of Bluetooth devices. Note that there is no whitelisting for Bluetooth since in the current deployment it is not used for channel estimation.

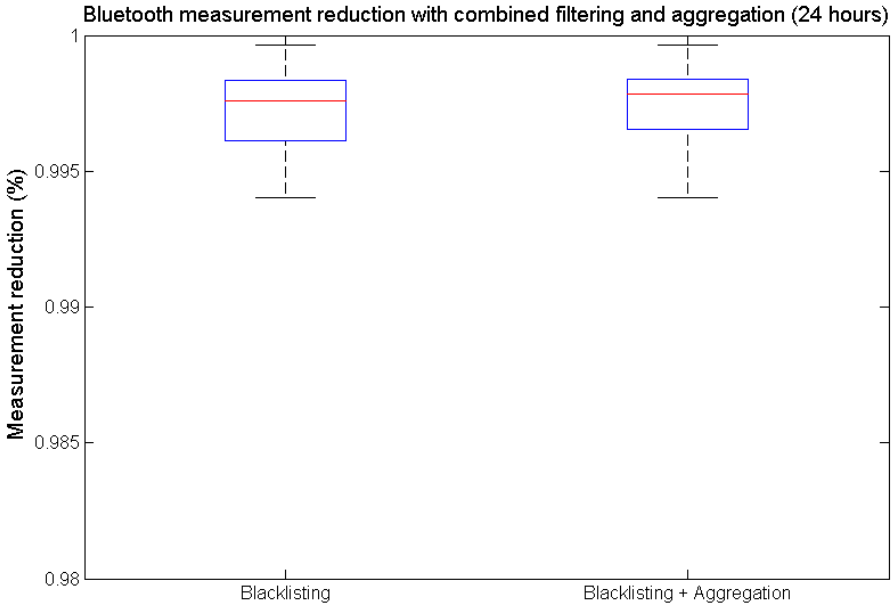



Figure 10: Combined Bluetooth measurement reduction over all sensors for combinations of blacklisting and aggregation

# 6 Conclusion

This work is dedicated to radio traffic challenges arising in wireless sensor test-beds. We showed how the traffic volume can be greatly reduced by leveraging the use of filtering and aggregation independently and combined. We achieved reductions of 80% on average with a peak above 95%, depending on used settings. Without this reduction in traffic, the testbed would not be able to scale well when extended to large testing sites due to the limited resources on the sensors and congestion of the radio medium. The reductions were achieved for our specific experimental test-bed consisting of 16 sensor nodes deployed indoors in a small-scale building. The test-bed purpose is to test an indoor positioning system based on WiFi and Bluetooth technologies. We quantified the reduction in measurements that filtering and aggregation can bring independently and in combination with each other.

The presented evaluation and results have relevance beyond the scope of radio-based test-beds. We are confident that filtering and aggregation strategies can also help other test-beds as well as real-world deployments to scale better and to make better use of the limited radio resources. We are aware that both mechanisms have a downside, e.g., wrongly identifying a device as always-on in filtering or loosing measurements details in aggregation, but we believe that a careful parametrisation can contain these effects. In addition, compression techniques could further bring the size of the transferred data down, which is worth investigating.

# References

[1] D. Dimitrova, U. Bürgi, G. Martin Dias, T. Braun, and T. Staub, "Inquiry-based bluetooth parameters for indoor localisation - an experimental study," pp. 13–21, 5th ERCIM workshop on e-Mobility, 2011.

[2] Z. Chen and K. Shin, "Opag: Opportunistic data aggregation in wireless sensor networks," in *Real-Time Systems Symposium, 2008*, pp. 345 –354, 2008.

[3] V. Kumar, J. McCarville-Schueths, and S. Madria, "A test-bed for secure hierarchical data aggregation in wireless sensor networks," in *Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on*, pp. 762 –764, nov. 2010.

[4] Z. Taghikhaki, N. Meratnia, and P. Havinga, "Energy-efficient trust-based aggregation in wireless sensor networks," in *IEEE INFOCOM: Workshops*, pp. 584 –589, 2011.

[5] "Honk kong university, internet and mobile computing laboratory test-bed." `http://www.comp.polyu.edu.hk/en/research/centres_labs/internet_and_mobile_computing_laboratory/index.phpm`. Accessed: 27/01/2012.

[6] "Greenorbs test-bed." `http://greenorbs.org`. Accessed: 27/01/2012.

[7] R. Murty, G. Mainland, I. Rose, A. Chowdhury, A. Gosain, J. Bers, and M. Welsh, "Citysense: An urban-scale wireless sensor network and testbed," in *Proc. of IEEE Technologies for Homeland Security*, pp. 583 –588, 2008.

[8] "Gumstix overo." `https://www.gumstix.com/store/product_info.php?products_id=227`. Accessed: 27/01/2012.

[9] D. Balsiger, "Administration and deployment of wireless mesh networks," Master's thesis, University of Bern, 2009.

[10] T. Staub, S. Morgenthaler, D. Balsiger, P. Goode, and T. Braun, "Adam: Administration and deployment of adhoc mesh networks," 3rd IEEE Workshop on Hot Topics in Mesh Networking (IEEE HotMESH 2011), 2011.

[11] "IEEE 802.11-2007, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications." http://standards.ieee.org/about/get/802/802.11.html, 2007.

[12] L. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proc. of Distributed Computing Systems Workshops*, pp. 575 – 578, 2002.

[13] K. Dolfus and T. Braun, "An evaluation of compression schemes for wireless networks," pp. 1–6, International Congress on Ultra Modern Telecommunications and Control Systems, 2010.

# A Blacklist decision support

When we first start the blacklist decision support process, it will calculate a blacklist for all sensors which were active in the past 24 hours and save the blacklists sperately for Bluetooth and Wifi. In this case sensor 98 was active and 11 MAC addresses are blackllisted for Wifi. Since the Bluetooth scanner was not running, zero MAC addresses are blacklisted for Bluetooth. Note that MAC addresses have been masked to comply with any privacy concerns.

```
Aggregate server is listening on port 23654
Calculating blacklists over interval 1323618300-1323704700
Connected to database lba
Adding 1 sensors to the blacklist-processing
Adding sensor 98 to the list of sensors
Calculating blacklist for sensor 98
Adding 0 MAC-addresses for sensor 98 to Bluetooth blacklist
Adding 11 MAC-addresses for sensor 98 to WIFI blacklist
MAC,activity_level,stddev
00:xx:xx:xx:CF:D4:,0.888889,10.6986
00:xx:xx:xx:3D:E4:,0.888889,9.70247
00:xx:xx:xx:B4:BF:,0.888889,23.9726
00:xx:xx:xx:3E:9B:,0.888889,8.08426
00:xx:xx:xx:CF:C8:,0.885417,12.742
00:xx:xx:xx:F2:E0:,0.875,9.3976
00:xx:xx:xx:CF:D6:,0.875,12.2034
00:xx:xx:xx:CF:C4:,0.861111,12.8039
CA:xx:xx:xx:CA:CA:,0.854167,13.274
00:xx:xx:xx:32:90:,0.756944,8.92944
00:xx:xx:xx:31:B0:,0.701389,11.003
Saving wifi list in location /opt/blacklist/98_wifi.bin
Saving bluetooth list in location /opt/blacklist/98_bt.bin
Going to sleep for 86230 seconds
```

All running sensors will request a blacklist at some point in time. When there is no available blacklist, the decision process will suggest that the sensor checks back in an hour for a new one. If there is a calculated blacklist, the server will send the blacklist along with the timestamp of generation and the number of seconds (from the blacklist timestamp) in which a new blacklist is expected to be available.

```
Incoming connection from (130.92.66.188:57099)
Waiting for data
Got header (4), waiting for payload (5)
Got 5 of 5 bytes
Received blacklist request for node 14
Sending tstamp: 0, update: 0, list_size: 0, checkback: 3600
Connection closed
Incoming connection from (130.92.66.36:56085)
Waiting for data
Got header (4), waiting for payload (5)
Got 5 of 5 bytes
Received blacklist request for node 98
Sending tstamp: 1323704870, update: 1, list_size: 66, checkback: 86400
Connection closed
```

Regularly, sensors with a blacklist will send blacklist-aggregate messages to the decision support process for all the blacklisted MAC addresses. Typically this happens each hour, but it can be configured differently if need be.

```
Incoming connection from (130.92.66.36:56410)
Waiting for data
Got header (4), waiting for payload (297)
Got 297 of 297 bytes
Received 11 blacklist aggregates
Connection closed
```

After the blacklist evaluation period (in this case 24 hours), a new blacklist is calculated. For node 98, we are interested in being able to filter out sensor 10 and 14, with MAC addresses ending in AA:85 and 02:8E. As is visible below, the decision process correctly identified these MAC addresses and added them to the blacklist.

```
Calculating blacklists over interval 1323704700-1323791100
Adding 4 sensors to the blacklist-processing
Adding sensor 13 to the list of sensors
Adding sensor 10 to the list of sensors
Adding sensor 14 to the list of sensors
Adding sensor 98 to the list of sensors
Calculating blacklist for sensor 13
Adding 0 MAC-addresses for sensor 13 to Bluetooth blacklist
```

```
Adding 0 MAC-addresses for sensor 13 to WIFI blacklist
Saving wifi list in location /opt/blacklist/13_wifi.bin
Saving bluetooth list in location /opt/blacklist/13_bt.bin
Calculating blacklist for sensor 10
Adding 0 MAC-addresses for sensor 10 to Bluetooth blacklist
Adding 42 MAC-addresses for sensor 10 to WIFI blacklist
MAC,activity_level,stddev
** list cut out **
Saving wifi list in location /opt/blacklist/10_wifi.bin
Saving bluetooth list in location /opt/blacklist/10_bt.bin
Calculating blacklist for sensor 14
Adding 0 MAC-addresses for sensor 14 to Bluetooth blacklist
Adding 42 MAC-addresses for sensor 14 to WIFI blacklist
MAC,activity_level,stddev
** list cut out **
Saving wifi list in location /opt/blacklist/14_wifi.bin
Saving bluetooth list in location /opt/blacklist/14_bt.bin
Calculating blacklist for sensor 98
Adding 0 MAC-addresses for sensor 98 to Bluetooth blacklist
Adding 4 MAC-addresses for sensor 98 to WIFI blacklist
MAC,activity_level,stddev
00:xx:xx:xx:00:00:,0.979167,10.0706
00:xx:xx:xx:35:AF:,0.947917,12.1544
00:xx:xx:xx:AA:85:,0.871528,19.5804
00:xx:xx:xx:02:8E:,0.854167,19.0385
Saving wifi list in location /opt/blacklist/98_wifi.bin
Saving bluetooth list in location /opt/blacklist/98_bt.bin
Going to sleep for 86388 seconds
```

# B Estimated reduction of filtering

Among the categories of measurements we want to filter out, are fixed infrastructure and always-on devices. The characteristic these two categories have in common is the high number of measurements. The always-on devices because they are at least always connected to the network and sometimes transmit data; the fixed infrastructure because an AP usually serves multiple clients causing a lot of measurements.
To get a feeling for what we can expect in the reduction of measurements through the means of filtering, we estimated the reduction in measurements for a 24 hour period. We analysed data from a one-day experiment where sensors were collecting measurements of both WiFiand Bluetooth signals. Our approach is as follows: we identified the ten MAC addresses per sensor which generated the most measurements for the 24-hour period. To get the expected reduction in measurement volume, we calculate the fraction of measurements generated by these ten MAC addresses versus the total number of measurements. Figure 11 shows the expected measurement reduction per sensor.
As it is clearly visible in Figure 11, filtering in our sensor testbed has high potential. With a mean measurement reduction of roughly 73.25% per sensor we can be fairly certain that the impact of filtering on congestion and scalability will be positive. Although the way we choose the MAC addresses is not an informed decision process, we find our estimation realistic since the MACs with most measurements are most probably MAC addresses of access points, which are of little interest to us.

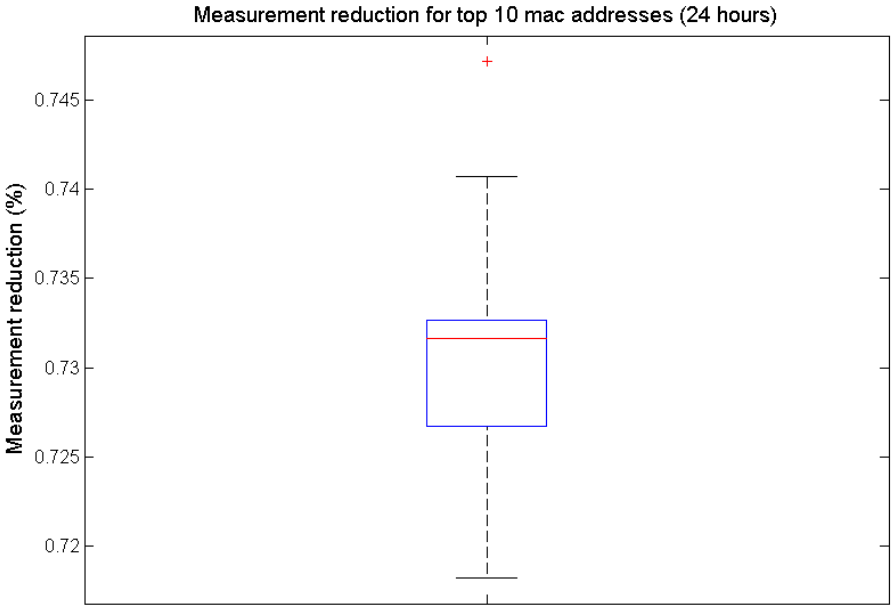



Figure 11: Boxplot over all sensors of measurement reduction in a 24 hour period when the 10 MAC addresses with the most measurements (per sensor) would be filtered out.