ROBOTS IN ASSISTED
LIVING ENVIRONMENTS

UNOBTRUSIVE, EFFICIENT, RELIABLE AND MODULAR
SOLUTIONS FOR INDEPENDENT AGEING

Research Innovation Action

Project Number: 643892 Start Date of Project: 01/04/2015 Duration: 36 months

DELIVERABLE 5.7

Large-scale and privacy-preserving data fusion and interpretation II

Dissemination Level	**Public**
Due Date of Deliverable	Project Month 30, September 2017
Actual Submission Date	15 May 2018
Work Package	WP5, Overall architecture of the RADIO ecosystem of services for the medical care institutions and informal care-givers
Task	T5.3, RADIO overall data management process T5.4, Development of large-scale data aggregation and interpretation methods
Lead Beneficiary	NCSR-D
Type	R
Status	Submitted
Version	Final

Project funded by the European Unions Horizon 2020 Research and Innovation Actions

Abstract

This report describes the related data management processes concerning both technical related controls to protect against respective privacy/security issues and attacks as well as data management procedures aiming to defend the RADIO platform against soft issues such as information misuse and unauthorized access.

History

Version	Date	Reason	Revised by
01	13 Jun 2016	Document structure, Introduction (Sect. 1), and a summary of D5.6 (Sect. 2.1).	NCSR-D
02	12 Jul 2017	Updates to the RASSP Protocol (Sect. 2.3) and implementation (Sect. 2.2)	NCSR-D
03	16 Jan 2018	Implementation of the non-aggregative data access and visualization using InfluxDB and Grafana (Section 3).	NCSR-D
04	4 Apr 2018	Pre-final version given for review	NCSR-D
05	5 Apr 2018	Internal review	S&C
06	11 May 2018	Addressing review comments (mostly Sect. 2) and new efficiency evaluation results (Sect. 2.4).	NCSR-D
Fin	15 May 2018	Final document preparation and submission.	NCSR-D

Executive Symmary

RADIO Home deployments interact and exchange data beyond the boundaries of the local network. Actually, it is envisaged that RADIO Home deployments will seamlessly integrate in the RADIO ecosystem as nodes and collectively provide data mining capabilities to medical research institutions. The data exchange and processing between the entities of the RADIO ecosystem should employ techniques and methods for preserving the privacy of the data and protecting the data for misuse and unauthorized access.

This report follows up on D5.6 which presented (a) the RASSP protocol for supporting a privacy preserving data mining system; and (b) the technical controls and the procedures for protecting from authorized access. In the current report, we present lessons learned from a large-scale deployment and testing of the RASSP protocol; and (b) present an implementation of the technical controls for the InfluxDB/Grafana solution for storing and visualizing timeseries.

Abbreviations and Acronyms

AAL	Ambient Assisted Living
ADL	Activities of Daily Living
RASSP	RADIO Secure Summation Protocol
REST API	Representational state transfer applications programming interface
IoT	Internet of Things
IPSec	Internet Protocol Security is a protocol suite for secure IP
VPN	Virtual Private Network
RPC	Remote Procedure Call

CONTENTS

List of Figures

1 INTRODUCTION

1.1 Purpose and Scope

The purpose of this deliverable is to provide the methods and techniques for the management of the data exchanged between the RADIO ecosystem entities in a privacy-preserving and secure way.

This includes the RASSP protocol for scalable, privacy-preserving access to aggregations (Section 2) and the access controls for accessing individual end-users' data by competent parties (Section 3).

1.2 Approach

Task 5.3 tackles all aspects of coordination and communication system emphasizing on security and privacy issues. During its first phase (reported in D5.6, Section 5), this task focused on data management processes concerning both technical related controls to protect against respective privacy/security issues and attacks as well as data management procedures aiming to defend the RADIO platform against soft security issues such as information misuse, unauthorized access, accidental error etc. During the (currently reported) second phase, this task applied the technical aspects of these controls to implement a system for accessing individual end-users' data by competent parties—in our case the medical personnel responsible for each individual end-user (Section 3). This work also includes a demonstration of these access controls for the InfluxDB database management system and the Grafana data visualization platform.

Task 5.4 develops and prototypes the methods needed by medical care institutions in order to aggregate and interpret the detailed ADL and mood recognition results into trends and averages at the right level of abstraction for inspection by medical personnel. During its first phase (reported in D5.6, Sections 2, 3, and 4) this task developed the RASSP protocol for scalable, privacy-preserving access to aggregations. During the (currently reported) second phase, this task investigated possible exploits due to corner cases in node availability and adapted the protocol accordingly.

1.3 Relation to other Work Packages and Deliverables

This deliverable complements *D5.6 Large-scale and privacy-preserving data fusion and interpretation I* and provides to *D5.9 Integrated RADIO prototype* the prototypes of the data management and sharing techniques. The architectural considerations stemming from this deliverable are recorded in *D5.3 Architecture of the RADIO Ecosystem III*.

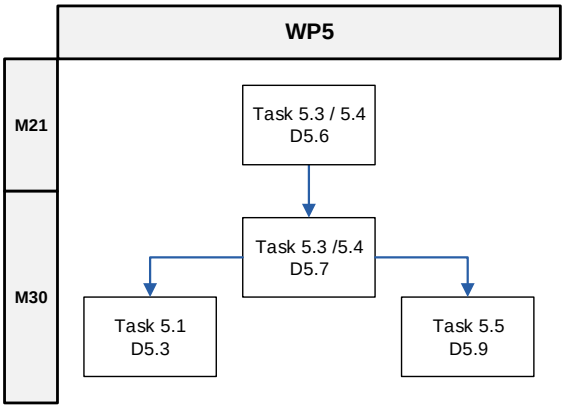

Figure 1: Dependencies between this deliverable and other deliverables.

2 THE RADIO DATA MINING SYSTEM

2.1 RASSP Overview

The insights gained by the large-scale analysis of health-related data can have an enormous impact in public health and medical research, but access to such personal and sensitive data poses serious privacy implications for the data provider and a heavy data security and administrative burden on the data consumer. Cryptography and distributed computation can provide methods for computing aggregates and statistics without revealing the specific data values involved in the computation, offering stronger guarantee of privacy than anonymization.

RASSP is such a distributed computation protocol. RASSP is defined as a stack of three layers: the *Medical Researcher's interface* provides bindings from the user's programming language to the back-end system (to provide a familiar environment to medical researchers, R is used in RADIO); the R computations are transformed appropriately in order to be passed to the next layer, which is the *Compilation Layer*. At that stage, the high-level parameters and commands of the statistical method are transformed into low-level instruction for accessing the private databases of the agents. An instruction represents an aggregation over a selection of data. Currently, the aggregation operation is summation. However, the aggregations that are both feasible by the system and safe for preserving privacy depend on the secure protocol used. These instructions will be eventually evaluated by the lowest layer of the architecture, the *Privacy Protocol Layer*. Figure 2 depicts the system architecture and the information exchanged between the layers.

2.1.1 The Compilation Layer

This layer is responsible for the communication between the two other layers. Specifically, it translates the arguments of the *secure statistic* to a suitable format, thus it defines the appropriate data that are going to be used for the statistic computation. Moreover, it converts the simple statistic equations to a set of summations; a compatible format to achieve the secure summation protocol. Therefore, a set of instructions is composed where each instruction represents a summation equation of the statistic with the appropriate parameters set for its computation. During the execution, the compilation layer gives to the privacy protocol layer a single instruction at a time and it receives its result. After the execution of the whole set, it computes the statistic and the analysis parameters. The statistic result is sent back to the Medical Researcher's interface layer.

2.1.2 The Aggregation Protocol

This layer executes the privacy protocol between the AAL agents. To deal with the concurrent computation of each instruction, we model our agents as actors. Each actor makes the appropriate computations with respect to the given instruction and its private data. These computations can easily be done since every AAL agent controls its corresponding health records. After the computation of the value, which represents the initial secret, the privacy protocol is executed. The protocol may involve all the actors to work collaboratively in order to compute the aggregation of their secrets without revealing the actual secrets to each other or the agent requesting the aggregation. The aggregated result is collected a designated actor. The selection of such actor is irrelevant and can be done randomly.

2.1.3 Example

We will use a simple example to better demonstrate the proposed system. Suppose that a medical researcher needs to run a t-test to assess whether the means of two groups are statistically different from each other, that is to compute t in Eq. 1:

$$t = \frac{\bar{X} - \bar{Y}}{\sqrt{|X|^{-1}\,\sigma_X^2 + |Y|^{-1}\,\sigma_Y^2}} \tag{1}$$

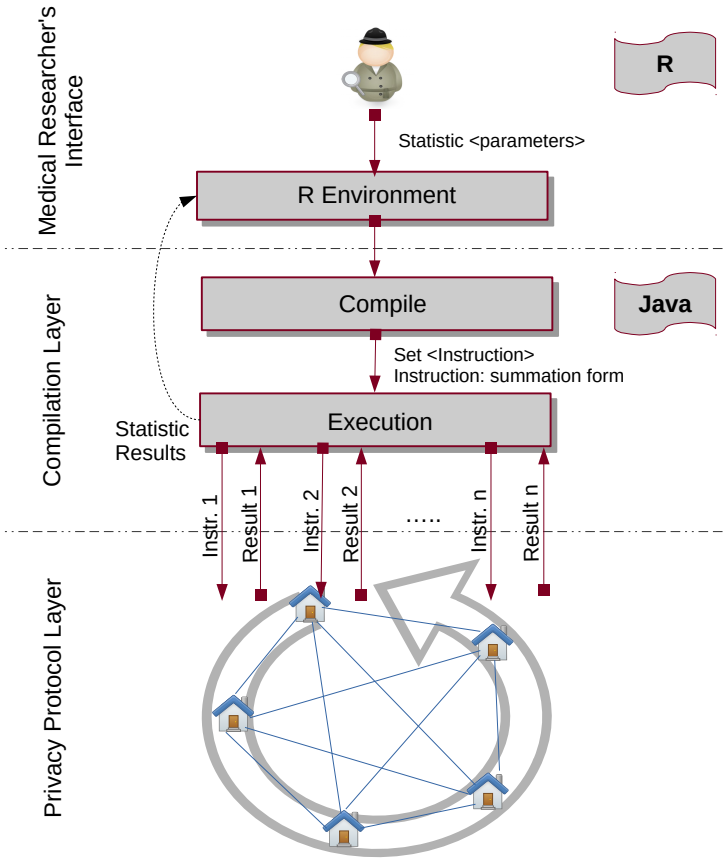

Figure 2: The architecture of the RADIO data mining system

where X and Y are the datapoints of the two groups, $\bar{X}$ and $\bar{Y}$ are their means, $|X|$ and $|Y|$ are their cardinalities, and σ_X^2 and σ_Y^2 their variances.

Assume, for instance, that a researcher wants to test the effect of medicine M_1 (Group 1) and medicine M_2 (Group 2) on blood pressure, with the further restriction that participants in Group 2 should be above 65 years old. A workflow using the R language would be:

- Select from a database the instances that match Group 1 criteria and store them in variable X
- Select from a database the instances that match Group 2 criteria and store them in variable Y
- Decide on the conditions of the T-test, such as the confidence level and alternation, and store them in variable C
- Pass X, Y, C as arguments to an implementation of t-test

Our architecture allows this workflow to remain essentially unaffected, except for the contents of X and Y. Instead of holding actual data arrays these now contain a representation of the Group 1 and Group 2 criteria, so that the selection can be executed in distributed manner. Using this representation, a privacy-aware implementation of t-test can produce the exact same result as the conventional implementation, except without ever accessing any individual data.

This representation declares a list of dependent variables and a list of eligibility criteria of the sample groups, as a set of $(\text{variable}, \text{operator}, \text{value})$ tuples. In our example, we assign to X and Y the criteria that we would have used to assign to them a value array if we had full access to the data:

- $X = [(\texttt{"medicine"}, =, \texttt{"}\mathrm{M}_1\texttt{"})]$

Function	Definition
$\mathrm{add}(C)$	$\sum_i s_i(C)$, where $s_i(C)$ is the secret value of the i-th AAL agent if condition C is satisfied, 0 otherwise
$\mathrm{add}^2(C, k)$	$\sum_i (s_i(C) + k)^2$, where k is a constant and $s_i(C)$ is same as above
$\mathrm{cnt}(C)$	$\sum_i c_i(C)$, where $c_i(C)$ is 1 if the i-th AAL agent satisfies condition C, 0 otherwise.

Table 1: Characteristic instructions provided by the RASSP Protocol.

- Y = [("medicine", =, "M_2"), ("age", >, "65")]

The compilation layer converts the t-test implementation into a set of instructions. Recall that each instruction is an aggregation over the private data of each agent, under the given selection restrictions. Table 1 defines the instructions needed to implement the t-test (Eq. 1), which is then implemented using the following pseudo-code:

1. X = [("medicine", =, "M_1")];
   X is a representation of the secret values of all AAL agents where medicine M_1 is used.
2. Y = [("medicine", =, "M_1"), ("age", >, "65")];
   Y is a representation of the secret values of all AAL agents where medicine M_2 is used and age is above 65.
3. $N_1 = \mathrm{add}(X); N_2 = \mathrm{add}(Y)$;
   N_1 is the sum of the secret values X and N_2 is the sum of the secret values Y.
4. $C_1 = \mathrm{cnt}(X); C_2 = \mathrm{cnt}(Y)$;
   C_1 is the number of AAL agents with non-zero values in X and C_2 is the number of AAL agents with non-zero values in Y.
5. $\bar{X} = N_1/C_1; \bar{Y} = N_2/C_2$;
   This uses the values above to calculate means.
6. $\sigma_X^2 = \mathrm{add}^2(X, -\bar{X}); \sigma_Y^2 = \mathrm{add}^2(Y, -\bar{Y})$;
   This uses the values above to calculate variances.
7. $T = \left(\bar{X} - \bar{Y}\right) / \mathrm{sqroot}\left(\sigma_X^2/C_1 + \sigma_Y^2/C_2\right)$;

Each instruction is executed with the use of the secure summation protocol, obtaining the aggregate values specified in the instruction without obtaining the values themselves. From the perspective of the R interface user, the t-test functions operate as if they had been passed the actual value matrices as parameters.

2.2 Implementation

The system has been implemented as a fully distributed system in Scala and Java. The system has been based upon the actor model using the library Akka[1].

The project's source code is organized in three modules, each one implementing one of the layers in the architecture above:

- `proto` implements the *aggregation protocol*

[1] https://akka.io/

- `stats` is the implementation of statistical analysis primitives over an aggregation protocol, and implements the *compilation layer*
- `RStats` implements the R interface for the medical researcher over the compilation layer.

To execute the aforementioned example using our implementation, the medical researcher executes the following code in the R interface:

```
# Describe the two groups in GroupStat structures:
group1 <- GroupStat(list(c("med","=","A")))
group2 <- GroupStat(list(c("med","=","B"), c("age",">","65")))
# Set dependent variables and groups in a Parameters structure:
p <- Parameters(list("bloodPr"), list(group1, group2))
# Execute the normal t-test using the Parameters structure p:
ttest(p, varEq=TRUE)
```

What is important to note in the example is that our implementation of the `ttest()` function presents an interface identical to the standard R implementation of the t-test. The underlying difference is that the `Parameters` structure does not point to actual data matrices but to instances of our `GroupStat` structure, which hold the information needed by the compilation layer in order to distribute the computation to the participating nodes.

2.3 Robustness

A notable issue that arises when implementing the theoretical model of the aggregated protocol is the robustness of the system, that is, how it handles corner cases and the potential exploits that arise from them. More specifically, an information leak can happen by isolating a RADIO Home as the only secret holder in an aggregation where everybody else colludes.

This places the RADIO data mining in the following position when one or more of the RADIO Home nodes are off-line:

- If RADIO Homes accept to execute the protocol with the remaining nodes, the protocol becomes more vulnerable because it becomes increasingly more likely that a RADIO Home might be singled out as the only non-colluding contributor to an aggregation.
- If RADIO Homes refuse to execute the protocol when even one RADIO Home is unreachable, (a) they need to be able to verify that all RADIO Homes in the network are legitimate; and (b) the system becomes nearly unusable system as it practically never respond for a non-trivial number of participants.

In order to create a usable yet secure system we have created the following organization of the RADIO ecosystem peers:

- Each peer is randomly assigned to a group of peers. The assignment, is not permanent but configurable and subject to change in order to avoid malicious activities. The number and the size of the groups are also configurable and depend on the total number of the peers in the RADIO ecosystem.
- Every group executes the protocol independently if and only if every peer in the group is reachable. In the unfortunate event that some of the peers are not reachable at the beginning of the computation then the computation is canceled and the result of the specific group is not considered. The same sequence of events happens when some peers appear unreachable during the protocol procedure.
- The intermediate results of the groups are gathered and aggregated to produce the final result.

It is apparent that the above procedure will lead to a greater rate of responses. Moreover, this hierarchical organization of the system can work satisfactorily under the assumption that the size of the group is

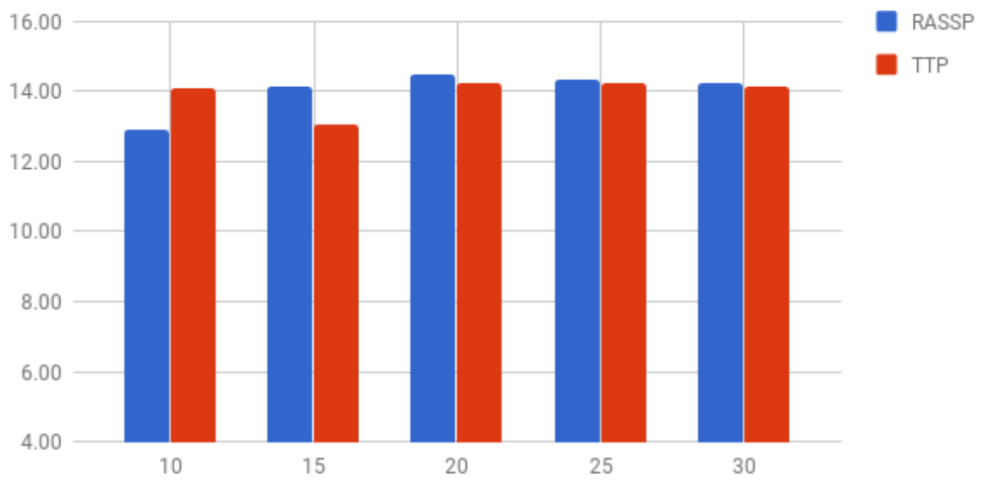

Figure 3: Efficiency of the RASSP protocol: execution time (sec) of RASSP and trusted third party, as a function of the number of nodes. An identical computation is performed in both setups. The times shown given are the average of ten runs.

Number of nodes	10	15	20	25	30
Successful runs	8/10	8/10	8/10	10/10	10/10

Table 2: Number of successful runs (out of ten) as a function of the number of nodes.

sufficiently large that the intermediate results do not reveal any particular individual data points. The benefit is that since the members of groups that contain at least one unreachable peer refuse to answer, they are safe from an attack that tries to isolate them as the last responding member who is not a colluding adversary.

2.4 Efficiency

The RASSP protocol imposes a higher volume of network traffic and a higher number of computations to compute the same result, by comparison to less secure protocol such as *trusted third party (TTP)* methods where a trusted intermediary aggregates private values into shareable statistics. On the other hand, RASSP is more distributed than centralized trust protocols so that more nodes are also involved in this larger communication and computational load. This offers *scalability*, as the network nodes undertake more of the processing than in TTP protocols.

In a RASSP network with N nodes, each node needs to transmit $N-1$ secret shares to the rest of the network and receive $N-1$ secret shares from the rest of the network, and then send one message with the summation of the shares it has received. Using a trusted third party (TTP) protocol, by contrast, requires only one message to be send by each node to the TTP. This implies that the expected behaviour is that RASSP execution time is linear to the size of the network and the TTP protocol is constant. In both protocols, we also expect a constant-time (with respect to the number of nodes) overhead that is caused by the translation of the R-level language constructs to the underlying protocol and the communication between the R runtime and the underlying Java runtime.

In order to empirically test the theoretical expectations above, we deployed RASSP over virtual machines at GRNET, the Greek academic network and cloud infrastructure. All machines have public IPs and all communication goes through the normal internet routing, to minimize the effect to our measurements of the fact that all nodes are hosted at the same data centre.

In the first experiment, we compared RASSP and TTP execution times as a function of the number of nodes. As be seen in Figure 3, the effect of the peer-to-peer communication imposed by the RASSP

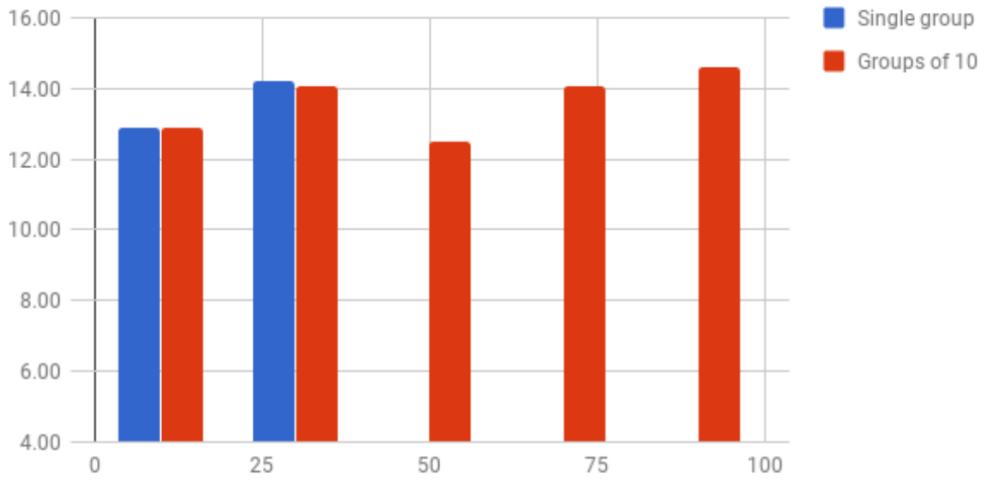

Figure 4: Efficiency of the grouped RASSP protocol: execution time (sec) of placing all RASSP nodes in a single group and in groups of 10 nodes, as function of the total number of nodes in all groups. An identical computation is performed in both setups. The times shown given are the average of ten runs.

protocol is minimal. The measurements appear to be constant as a function of the number of nodes, inferring that constant overheads dominate the node communication time.

Furthermore, groups of as many as 30 nodes are stable and almost always provide an answer, with only 4/50 runs aborted (Table 2.3) due to delays and, in general, timing that caused some of the nodes to activate the protection against the possible exploit presented in Section 2.3 above. This is even less of an issue in the grouped RASSP protocol, where *some* answer is practically always returned, although it might not involve all of the groups. Finally, the extra computation layer imposed by the grouped RASSP protocol has a negligible impact to the overall runtime (Figure 4).

3 INDIVIDUAL DATA ACCESS METHODS

The formal caregivers of the RADIO ecosystem must have access to the recorded data of their assigned subjects for health monitoring reasons. In contrast to the health researchers, the formal caregivers are interested in a more detailed and focused report about the activities of their assigned subjects.

In this chapter we describe the system and access methods developed in the RADIO ecosystem that a formal caregiver can use to access this kind of information. We give a brief description of the available data and visualizations and we provide a discussion about the authentication and authorization mechanisms

3.1 Database

The events that occur in each RADIO Home are recorded in a separate database, so that the access rights of each RADIO Home (and, by extension, the data of each specific subject) are managed by independent access controls. At the same time, RADIO provides a unified front-end to the formal caregiver. This organizational decision reflects the privacy requirements reported in Deliverable 5.3 and enables the authentication and authorization mechanisms discussed later in Section 3.3.

The nature of the recorded data is measurements of the same phenomenon (here, activity of daily living) over time. This makes a time-series database management systems the predominant choice for using in the RADIO ecosystem. For the RADIO demonstrator we used InfluxDB[2], an open source time-series database, as the backend of the RADIO ecosystem data management.

In InfluxDB, the database is a collection of time series (i.e., tables). A time series can be used to record events (i.e., rows). It is mandatory for a time series to contain a timestamp field that correspond to the time of the recorded event. The row is made by field keys and field values. The field keys are usually strings and are used to categorize the events while field values are usually numerical and correspond to measurable amounts (for example, temperature, distance, wind speed) recorded during the occurance of the event.

The schema of each database consists of multiple time series that contain the following information:

Time of the occured event.

Participant an alphanumeric identifier used as a *field key* to identify the subject of the event. This is mainly used for visualization and reporting reasons. Since each database corresponds to a single SubjectID, the field is expected to be constant throughout the time-series. However, the existence of this field helps during the reporting of multiple subjects.

Measurement is a *field value* that contain the measured quantity of the event. The measurement type and value range depends on the type of the event.

Currently, there are several events recognized and recorded by the Main Controller of each RADIO Home. If a new event type needs to be recorded, the database schema can be easily extended accordingly. The recorded events are:

- Chair transfer. The measurement for this event is the elapsed time (in seconds) needed to perform the action.
- Bed transfer. The measurement for this event is the elapsed time (in seconds) needed to perform the action.
- Four-meters walk. The measurement for this event is the elapsed time (in seconds) needed to perform the action.

[2]https://github.com/influxdata/influxdb

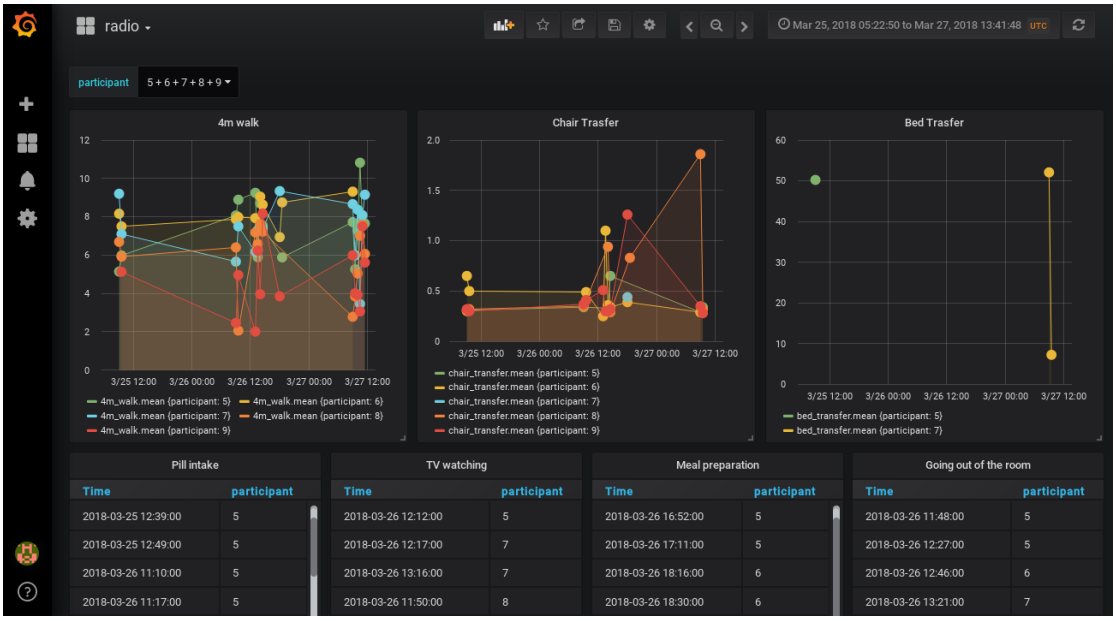

Figure 5: Samples of the visualizations available to the formal caregiver

- Pill intake. There is nothing measurable for this event. The existence of the row corresponds to a successfully recognized event at the specific date and time.
- TV watching. There is nothing measurable for this event. The existence of the row corresponds to a successfully recognized event at the specific date and time.
- Meal preparation. There is nothing measurable for this event. The existence of the rows corresponds to a successfully recognized event at the specific date and time.
- Going out of the room. There is nothing measurable for this event. The existence of the rows corresponds to a successfully recognized event at the specific date and time.

The functionality of the database is exposed as an HTTPS endpoint that accepts queries and returns the appropriate results in JSON format. As an example, consider the following query:

```
SELECT time, participant, value
FROM chair_transfer
WHERE time >= now() - 7d
```

The query above that selects all the events of the last seven days stored in the `chair_transfer` time-series in the database that corresponds to a particular subject.

Notice that the queries are formulated in an SQL-like language[3] [4] and can contain more complex filtering predicates.

3.2 Visualizations

The HTTP API provided by the database is accessed by visualization tools that present the data in a more concise and informative way. The visualization tool used in our case is Grafana[5], an open source user interface focused on visualizing time-series in various ways. The data can be visualized as graphs or tables over time depending on the measurable quantities.

Figure 5 depicts an overview of the visualizations implemented for the formal caregiver's data access. The events that contain measurable quantities (for example, chair transfer, bed transfer, etc) are depicted

[3] cf. `https://docs.influxdata.com/influxdb/v1.5/tools/api/#query`
[4] cf. `https://docs.influxdata.com/influxdb/v1.5/query_language/data_exploration/`
[5] https://grafana.com/

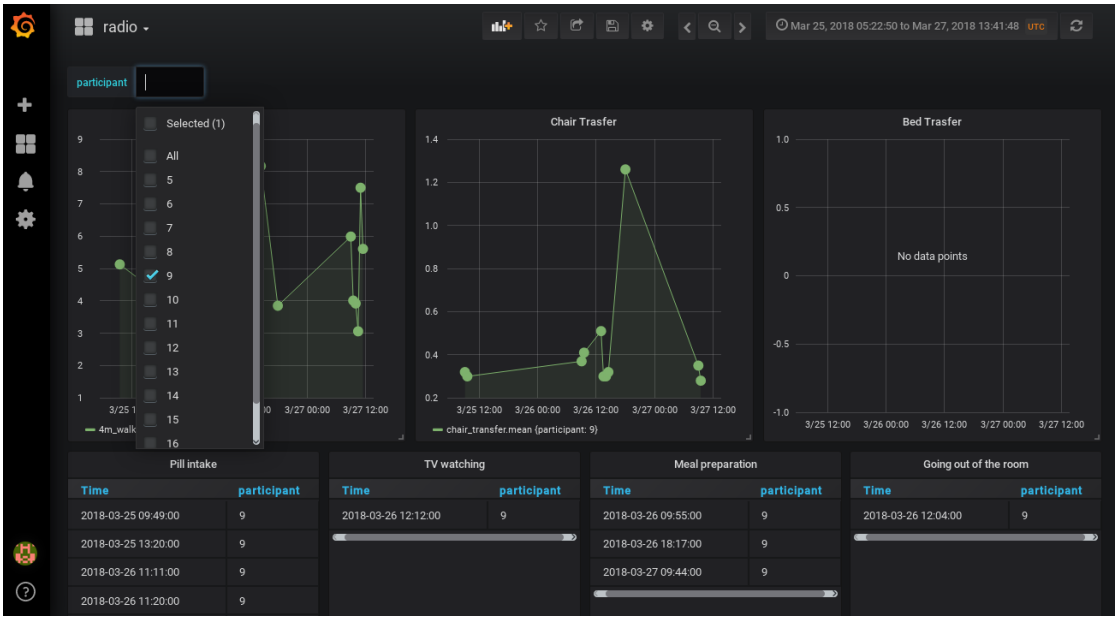

Figure 6: The formal caregiver selects the participants to visualize from the participant list that accessible to them.

as line graphs over time while the other events are depicted as tables.

3.3 Authentication and Authorization

Each of the aforementioned systems (i.e, the backend InfluxDB and the visualization tool Grafana) implement their own authentication and authorization mechanisms. The RADIO ecosystem utilizes both mechanism to achieve the desired privacy protection: data is written by RADIO Home accounts into their respective InfluxDB databases, and data is read by medical personnel by their Grafana accounts.

More specifically, authentication is performed in both systems using a username and password mechanism. Each user has their own pair of authentication tokens, provided to the system over an encrypted channel (more specifically, using the HTTPS protocol). Account credentials are created and distributed out-of-band by IT administration personel, who also maintain the mapping from Grafana users (i.e., medical personnel) to appropriate subsets of InfluxDB databases (i.e., primary users).

Grafana is the main entry point for accessing the data. The main users of Grafana are the formal caregivers that have access to view graphs of events from specific databases (people) on a need-to-know basis. Formal caregivers can select which database's data to use in the graphs they view, but only among those their authentication allows. The graphs can also be parameterized by, for example, filtering on the time interval that is visible or the subjects that contribute to the graph. Figure 6 shows how a formal caregiver can select the desired parameters.

Apart from the main users, there is also another group of users, "IT administrator" that is responsible to create new visualizations and manage the permissions of the main users.

The database management system has its own authentication and authorization mechanism. Users such as formal caregivers cannot access directly the database. The main users of the backend database are (a) the data providers and (b) the data consumers. The data provider in this case is the main controller of the corresponding RADIO House and is granted *write-only* permissions. It follows that each main controller has access to exactly one database. Moreover, the main controller cannot modify or delete already written data. The data consumer on the other hand is granted permission to *read-only* permissions.