



Security And Interoperability in Next Generation PPDR
Communication Infrastructures



Project Number: 313296

Deliverable 7.1

SALUS PPDR platform - Intermediate

Scope	Scheduled deliverable
Lead Beneficiary	OneSource (ONE)
Dissemination level	Public
Document creation date	11/ Oct / 2014
Document release date	20 / Mar / 2015
Contractual Date of Delivery	28 / Feb / 2015
Version	1.1
Status	Final

Abstract: Deliverable 7.1 describes the intermediate SALUS PPDR platform components, highlighting the integration and evaluation plans.

AUTHORS

Name	Organisation	Email
Paulo Simões	ONE	psimoes@onesource.pt
Edmundo Monteiro	ONE	edmundo.monteiro@gmail.com
Luis Cordeiro	ONE	cordeiro@onesource.pt
Bruno Sousa	ONE	bmsousa@onesource.pt
Konstantia Barbatsalou	ONE	konstantia@dei.uc.pt
Hugo Fonseca	ONE	htfonseca@onesource.pt
Georgios Mantas	IT	gimantas@av.it.pt
Hugo Marques	IT	hugo.marques@av.it.pt
Jonathan Rodriguez	IT	jonathan@av.it.pt
Luis Pereira	IT	luis.pereira@av.it.pt
Daniel Zerbib	CAS	daniel.zerbib@cassidian
Bert Bouwers	ROH	bert.bouwers@rohill.nl
Peter Wickson	AW	Peter.wickson@airwavesolutions.co.uk
Wilmuth Müller	FhG	Wilmuth.mueller@iosb.fraunhofer.de
Frank Brouwer	FIGO	frank.brouwer@figonet.com
David Jelenc	UL	david.jelenc@fri.uni-lj.si
Jernej Kos	UL	jernei.kos@fri.uni-lj.si
Denis Trček	UL	denis.trcek@fri.uni-lj.si
Geert Heijenk	UTWENTE	geert.heijenk@utwente.nl
Bernd Meijerink	UTWENTE	bernd.meijerink@gmail.com
Rick Hofstede	UTWENTE	r.j.hofstede@utwente.nl
Alexandros Ladas	KU	A.Ladas@kingston.ac.uk
Olayinka Adigun	KU	o.adigun@kingston.ac.uk
Georgios Charalampopoulos	UPAT	giwrgoscharalampopoulos@gmail.com
Panagiotis Galiotos	UPAT	pgaliot@gmail.com
Jérôme Brouet	ALU-I	jerome.brouet@alcatel-lucent.com
Phillipe Lasserre	ALU-I	philippe.lasserre@alcatel-lucent.com
Marc Billemaz	ALU-I	mmarc.billemaz@alcatel-lucent.com
Branko Kolundzija	UB	kol@etf.rs
Yevgeni Koucheryavy,	UBITEL	y.koucheryavy@ubitel.ru

QUALITY ASSURANCE TEAM

Name	Organisation	Email
Dragan Olcan	UB	olcan@etf.rs
Frank Reinert	FhG	frank.reinert@iosb.fraunhofer.de
Vitaly Petrov	UBITEL	vitaly.petrov@gmail.com

EXECUTIVE SUMMARY

This document represents an initial report of WP7 activities, describing the SALUS PPDR Platform and providing details for the integration and an evaluation plan. It further delineates a preliminary demonstration plan for showing how the defined relations between SALUS PPDR platform components can be achieved – focusing on the initial implementation version of components ready to be integrated starting on M18 (Milestone MS13). The presented work builds upon the developments that have already been undertaken in previous Work Packages (especially WP5 and WP6).

The development of a working prototype of the SALUS PPDR Platform started as scheduled and is currently in progress in order to validate the project goals. This prototype considers the use cases defined in WP2 and follows the guidelines provided by the interim System Architecture currently being discussed in WP3.

Resulting from a joint effort of the partners involved in the SALUS PPDR Platform, this deliverable presents and details the integration plan and methodology for each component, resulting and focusing towards thorough and functional prototypes of services and applications for proving the SALUS concept.

Finally, the first version of the integrated prototype is identified, highlighting the main challenges that will be handled by the SALUS Platform. The complexity and innovation of the defined prototype also reinforces the decision of the chosen integration methodology, where a two-step approach has been adopted (e.g., integration and testing), thus guaranteeing an iterative and comprehensive demonstration of each scenario.

TABLE OF CONTENTS

EXECUTIVE SUMMARY	3
TABLE OF CONTENTS	4
TABLE OF FIGURES	6
TABLE OF TABLES	8
1 INTRODUCTION.....	9
1.1 Motivation, Objectives and Scope	9
1.2 Structure of the Document.....	9
1.3 Relation with other Deliverables	10
2 SALUS PPDR PLATFORM	11
2.1 PPDR Platform overview.....	11
2.2 Platform components	14
2.2.1 TETRA.....	14
2.2.2 TETRAPOL	18
2.2.3 LTE	20
2.2.4 Wi-Fi	25
2.2.5 Wireless Sensor Networks	30
2.2.6 Location Devices.....	37
2.2.7 Video System.....	37
2.2.8 Air Surveillance	38
2.2.9 Command and Control Centre.....	40
2.2.10 Mobility Management	41
2.2.11 Security Services	43
2.2.12 SALUS IP Communications Server	48
3 INTEGRATION PLAN	49
3.1 Integration Roadmap.....	49
3.2 Integration Methodology.....	49
3.2.1 Integration Tools	49
3.2.2 Integration Work Flow	51
4 FIRST INTEGRATION OF PROTOTYPE	58
4.1 Integration status.....	58
4.1.1 PMR-over-LTE ecosystem	58
4.1.2 SALUS Mobility Management.....	58
4.1.3 Message Broker.....	60
4.1.4 HUCare -- Integrated bio-sensors and video solution	61
4.1.5 SALUS PKI	61
4.1.6 SALUS IP Communications Server	62
4.1.7 Intrusion Detection Systems.....	62

4.2	Defined Interfaces	63
4.2.1	Web QoS interface	63
4.2.2	CVDP	63
4.2.3	Heterogeneous Handover in OpenEPC	64
4.2.4	Message Broker	64
4.2.5	SALUS PKI	64
4.2.6	SALUS IP Communications Server	65
4.2.7	Intrusion Detection Systems	65
4.2.8	TETRAPOL CCAPI interface	65
4.3	Roadmap and Open Issues	67
5	EVALUATION PLAN	70
5.1	Introduction	70
5.2	Evaluation Areas, Impacts and Indicators	71
5.3	Methodology and Evaluation technique	72
5.3.1	Analysis and Design	72
5.3.2	Implementation and Execution	73
5.4	Roadmap and Open Issues	75
6	Concluding Remarks	76
	Bibliography	77
	Acronyms	80
	Annex A – Integration report	84
	Annex B – Overall Prototype	85

TABLE OF FIGURES

Figure 1 – SALUS PPDR platform overview	11
Figure 2 – TETRA components description [36]	12
Figure 3 – TETRAPOL components description [36]	12
Figure 4 – LTE components description [36][37]	13
Figure 5 – Wi-Fi components description [36][37]	13
Figure 6 – IPv6 Mobility Management components	14
Figure 7 – The TetraNode	14
Figure 8 – LDS Chameleon	16
Figure 9 – LDS Chameleon and dispatcher functionality	16
Figure 10 – TETRA network management station	17
Figure 11 – TeTRA NMS NodeView	17
Figure 12 – TeTRA Broadband Gateway	18
Figure 13 – TETRAPOL Independent Digital Repeater 3G	18
Figure 14 – TETRAPOL Radio Access Gate	19
Figure 15 – TETRAPOL IDR and AG-R and SALUS CCC with CCAPI Client and Server	20
Figure 16 – eNodeB generic architecture	21
Figure 17 – eNodeB deployment options	21
Figure 18 – eCCM-U with GE MDA - Face Plate	21
Figure 19 – Evolved Packet Core Architectural Modules	22
Figure 20 – Micro Core LTE overview	23
Figure 21 – Alcatel-Lucent 9773 LTE Micro Core detail	24
Figure 22 – TETRA Dispatch Station	25
Figure 23 – KU Wi-Fi Rugged device	26
Figure 24 – SALUS Ad-Hoc network as distributed switch [36]	27
Figure 25 – Redundancy in the backhaul [36]	27
Figure 26 – SALUS Ad-Hoc network operation	28
Figure 27 – Coverage extension	28
Figure 28 – Network ranges	28
Figure 29 – Mapping of OpenVPN link	29
Figure 30 – Overview page of CMS	29
Figure 31 – CMS views: top-left: map; top right: statistics; bottom left: setting overview; bottom right: modify settings	30
Figure 32 – The architecture of the first prototype for the "man-down" detection.	31
Figure 33 – Man-down client application screen-shot	32
Figure 34 – Illustration of usage of sensor data (map plotting)	33
Figure 35 – A series of screen captures for the GUI part of the PPDR UE application	34
Figure 36 – The message broker service logging output.	35
Figure 37 – A screen capture of a back-end application	35
Figure 38 – HUCare Control Centre	36
Figure 39 – Indoor Location	37
Figure 40 – ALU-I E2E Video solution architecture based on partner components	38
Figure 41 – Components comprising to the Air Surveillance System for the first prototype	39
Figure 42 – Principle integration of components comprising to the CCC for the first prototype ..	40
Figure 43 – Detailed view on the FhG-IOSB applications and their integration strategy	40
Figure 44 – Network setup for Mobility Management.	42
Figure 45 – Multicast Proxy implementation overview	42
Figure 46 – SALUS Certificate Profile	44
Figure 47 – Integration Roadmap	49
Figure 48 – Integration Workflow and Responsibilities	52
Figure 49 – Unit Base Testing	54
Figure 50 – Key module evolved during mobility management in LTE EPC	59
Figure 51 – SALUS PKI building blocks	61

Figure 52 – SALUS IP Communications Server building blocks (simplified)	62
Figure 53 – SALUS PKI interface	64
Figure 54 – SALUS IP Communications Server interfaces	65
Figure 55 – SALUS IDS interfaces	65
Figure 56 - First prototype integration roadmap.....	67
Figure 57 – Generic Test Process	72
Figure 58 – Test Procedures	74
Figure 59 – Overall Prototype.....	85

TABLE OF TABLES

Table 1 – LTE Micro Core supported features	24
Table 2 – LTE Terminals	25
Table 3 – SALUS common message format for PPDR force tracking applications	41
Table 4 – Known Approaches in Live MF	46
Table 5 – Components and Functionality of the Mobile MF instance	46
Table 6 – Components and Functionality of the Aggregated MF instance	47
Table 7 – SALUS IP Communication Server features.....	48
Table 8 – Sanity Check Steps	54
Table 9 – Items to be delivered before a test procedure begins.....	56
Table 10 – Items needed for performing an Integration Test	56
Table 11 – Integration Test Output	57
Table 12 – CCAPI Control Centre Principles	66
Table 13 – CCAPI interface service names and categories	66
Table 14 – Mapping of functionalities between first and final prototypes.....	68
Table 15 – Evaluation Activities (according to adopted methodology), Gantt Chart	75

1 INTRODUCTION

WP7 builds upon and integrates the outcomes of previous Work Packages of the SALUS Project, including:

- The interim System Architecture currently being discussed in WP3, along with scenarios and use cases defined in WP2, which require adequate validation and evaluation.
- The infrastructure and platform development that is being conducted in the scope of WP5 and WP6, which need to be integrated into a working prototype – the SALUS Public Protection and Disaster Relief (PPDR) Platform.

This integrated prototype shall demonstrate how to provide next generation PPDR communications for mission-critical Personal Mobile Radio (PMR) voice and broadband data services, with enhanced security, privacy, seamless mobility, Quality of Service (QoS) and reliability. This deliverable provides a first report on the WP7 integration, demonstration and evaluation plans and activities, describing the necessary steps to produce the first version of the SALUS PPDR platform. WP7 work is organised such that preliminary results can be obtained at an early stage, providing feedback to the remaining work packages. By conducting a two-stage incremental approach it is possible to improve not only the entire SALUS platform but also the final prototypes being evaluated, reflecting this two-cycle process in most of the tasks from WP7. The final prototype shall be described later on (Month 33), in Deliverable D7.3.

1.1 Motivation, Objectives and Scope

The scope of WP7 is the integration and the evaluation of the SALUS PPDR Platform. In particular, the integration of components/mechanisms developed in WP5 and WP6, according to the architecture defined in WP3. While other work packages are focused on the development process of the SALUS services and components, the purpose of WP7 is to bring these functional building blocks together and, as a whole, demonstrate and evaluate the SALUS PPDR Platform in the distinct SALUS use cases.

The objectives of this document can be summarized as follows:

- Describe the integration plan that leads to the consolidation of the diverse components used within the first SALUS Platform prototype. Such an intermediary prototype, available at M18, will be the foundation for the validation activities of the SALUS Use Cases – more specifically, the city security, temporary protection and disaster recovery Use Cases which are handled in dedicated tasks of WP7;
- Establish an evaluation framework of the SALUS components. This deliverable lays the grounds for the appropriate definition of such evaluation methodologies, that shall not only provide feedback to the development work packages, but also actively contribute to the dissemination activities that take place in WP8, proving the principles and the whole concept of SALUS.

1.2 Structure of the Document

This deliverable is organized as follows. Section 2 addresses the SALUS PPDR Platform by presenting an overview of the prototype and the description of the multiple components. Section 3 introduces the integration plan, describing the roadmap, the methodology and the tools used for integration activities. Section 4 details the status of the integration in the first prototype and the defined interfaces. Finally, Section 5 presents the evaluation plans, while Section 6 concludes the deliverable.

1.3 Relation with other Deliverables

It should be noted that, for sake of conciseness, it is assumed that the reader is already familiar with the SALUS Project and the outcomes of previous Work Packages (i.e. the deliverables already produced in the scope WP2, WP3, WP4, WP5 and WP6).

2 SALUS PPDR PLATFORM

This section introduces the first prototype of the SALUS PPDR platform, presenting an overview of the platform, providing an high-level description of its main components, and giving an overview of the supporting infrastructure. Afterwards, a detailed description of the key components is presented. These descriptions include components from legacy systems (e.g. TETRA and TETRAPOL) and from future systems for PPDR networks, such as Long Term Evolution (LTE) and Wi-Fi.

As already mentioned, this prototype builds upon the requirements, architecture, Use Cases and system components produced in previous Work Packages, and it is assumed the reader is already familiar with corresponding deliverables – for instance Deliverable D5.2 [36], regarding security aspects; Deliverable D6.1 [37], regarding quality of services in PPDR networks, and Deliverable D2.4 [35], regarding user requirements for each of the three SALUS Use Cases (city security, temporary protection and disaster recovery).

2.1 PPDR Platform overview

This section provides an overview of the system technologies (legacy and future) that compose the SALUS PPDR Platform. The prototype integrates the legacy Private Mobile Radio (PMR) systems, Terrestrial Trunked Radio (TETRA) and TETRAPOL, as well as, future systems for PPDR networks, such as private or commercial LTE and Wi-Fi. Figure 1 illustrates the integration of multiple network systems with the SALUS Command and Control Centre (CCC), assisting commanding officers to control tactical units. The general overview of the platform is depicted in Annex B – Overall Prototype.

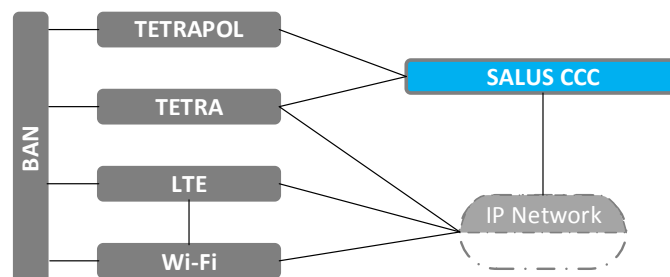


Figure 1 – SALUS PPDR platform overview

The SALUS CCC integrates functional blocks of all the network systems that enable the interconnection between the different PPDR technologies. Figure 2, Figure 3 and Figure 4 illustrate the diverse components and modules of each network system placed in the SALUS CCC and on the terminals of PPDR forces, TETRA, TETRAPOL and LTE, respectively. One common aspect to all the network systems, supporting PPDR operations is the employment of sensors measuring health and location parameters, i.e., the building blocks of Body Area Networks (BAN), to improve the security of PPDR officers or enhance their safety-related functions.

The three SALUS uses cases identified user requirement matrices for Police, Fire, Ambulance and other PPDR forces. Nowadays, most of these PPDR users rely on existing TETRA and/or TETRAPOL networks for mission-critical activities. Nonetheless, the support of advanced features such as high-speed mobile data capability is not supported in TETRA and TETRAPOL [35]. LTE fills this gap and may eventually support mission-critical features commonly assigned to TETRA and TETRAPOL, such as push to talk, group calls, among others. Moreover, the Wi-Fi networks can be employed to allow the communication between PPDR forces in the field, in cases where infrastructure based networks are not able to operate due to power outage or even

due to natural disasters that damage the infrastructure supporting the network. The mobility management introduced in the SALUS PPDR platform enable seamless handovers between heterogeneous technologies, such as Wi-Fi and LTE.

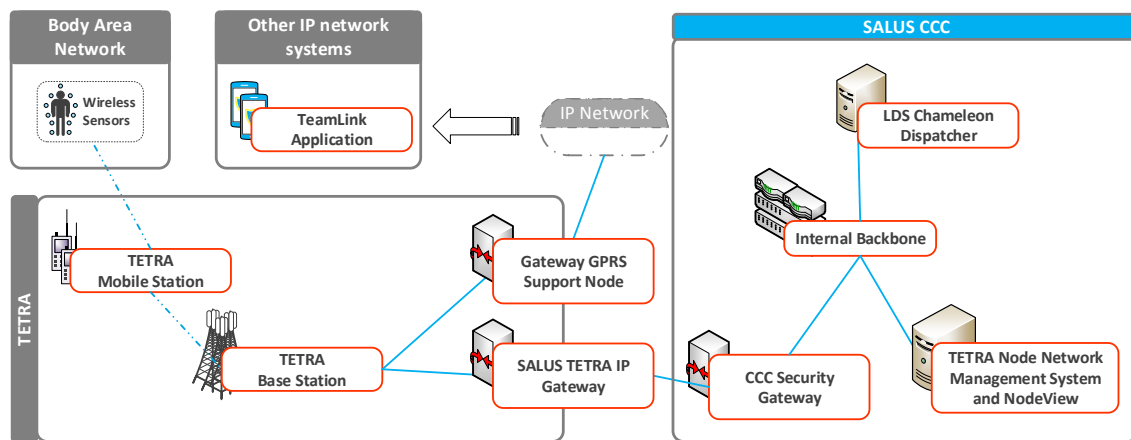


Figure 2 – TETRA components description [36]

Figure 2 presents the TETRA components and the infrastructure that allow the communication with SALUS CCC. As depicted, the components are compliant with the TETRA specification. SALUS introduces the TETRA IP gateway that allows the interworking between TETRA networks (non-IP based) and IP based networks, such as LTE, which are considered as future systems for PPDR networks. The SALUS CCC includes the Network Management System (NMS) and the dispatcher component to monitor the multiple group calls. TETRA components are described in detail later on in section 2.2.1.

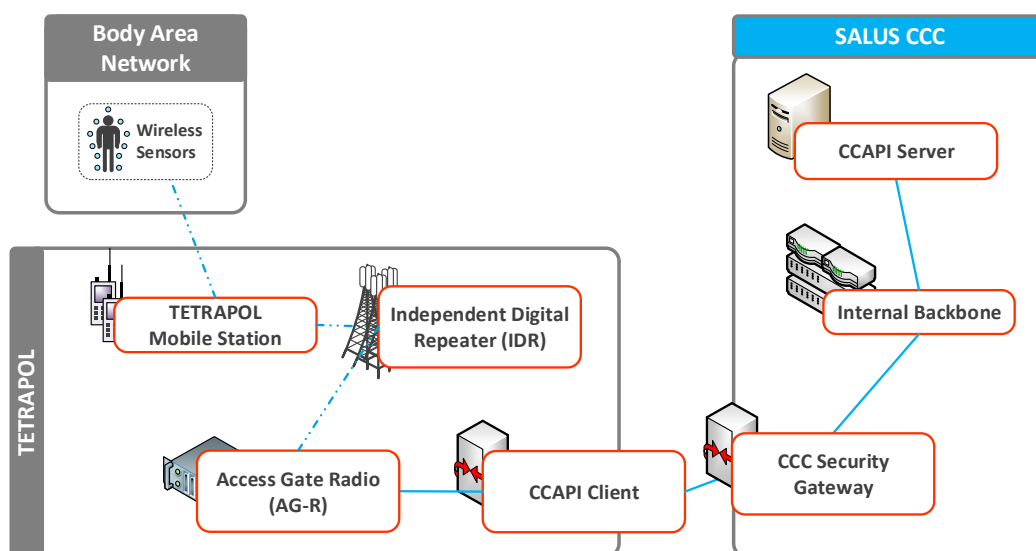


Figure 3 – TETRAPOL components description [36]

Figure 3 presents the TETRAPOL components and infrastructure that allow the communication with SALUS CCC. As in TETRA, the Command Control Application Programming Interface (CCAPI) Client gateway allows the interworking between IP networks and TETRAPOL based networks. Moreover the SALUS CCC includes components to monitor group calls and to manage the TETRAPOL stations, through the TETRAPOL Network Management System and NodeView. TETRAPOL components are detailed in section 2.2.2.

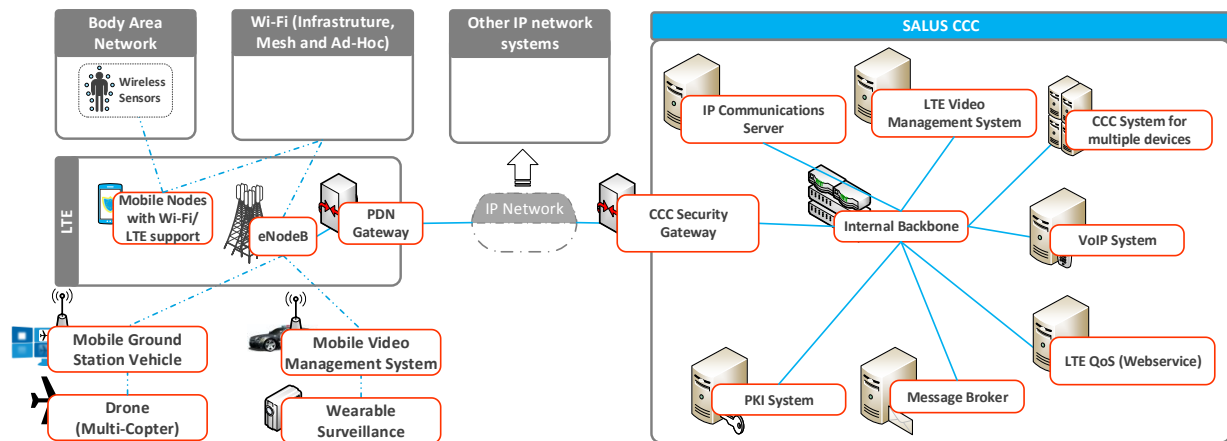


Figure 4 – LTE components description [36][37]

Figure 4 presents the LTE components for PPDR networks to support transmission of images, live video from drones or video cameras (which can be wearable and employed for surveillance or to improve situation awareness in the CCC). In addition, wireless sensors can transmit data related with important bio-signals (e.g. heart rate, blood pressure, position and movement of PPDR forces) through terminals that support both Wi-Fi and LTE.

The SALUS CCC includes the Security Gateway the provides secure access to the CCC internal backbone. The other servers, such as the Public Key Infrastructure (PKI) server, the servers of the Video Management System (VMS) and the security gateway assuring security services in the SALUS Platform, among others, are connected to this backbone. In addition, components introduced by SALUS, such as the Message Broker (MB) that allows alarms from bio-signals to be reported in the CCC are also connected to this backbone. These components have been reported in Deliverable D5.2 [36]. The LTE QoS Webservice allows the configuration of QoS mechanisms in LTE, to meet the requirements of PPDR communications previously identified in Deliverables D2.4 [35] and D6.1 [37].

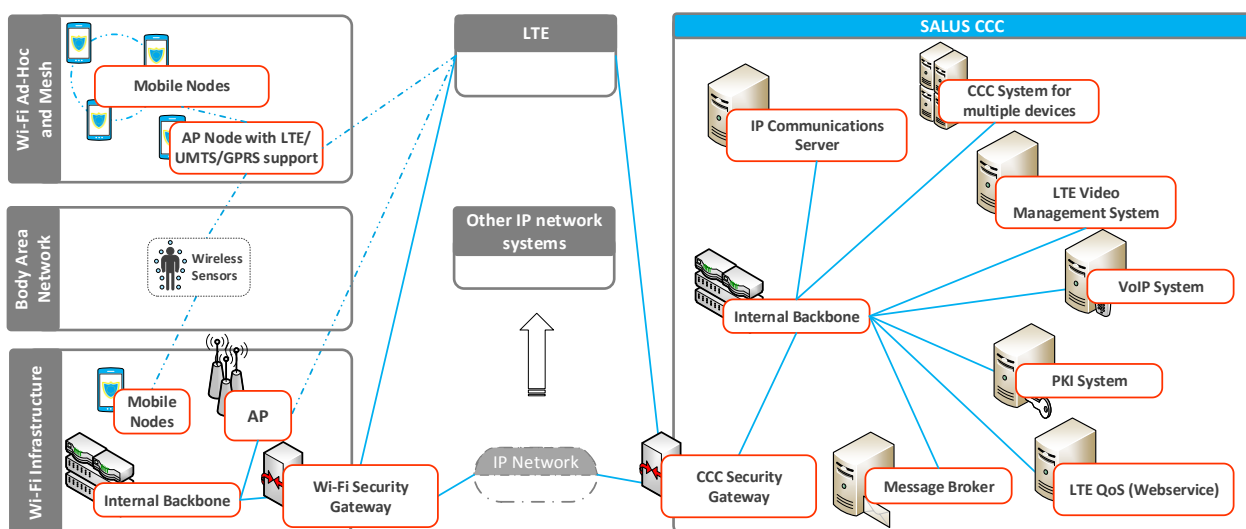


Figure 5 – Wi-Fi components description [36][37]

Figure 5 illustrates the Wi-Fi components, including Ad-Hoc and infrastructure support. The Wi-Fi components connect to the SALUS CCC through IP based networks, such as LTE. Moreover

the Ad-Hoc components are useful in scenarios where Wi-Fi infrastructures are not deployed, such as the Disaster recovery usage scenario.

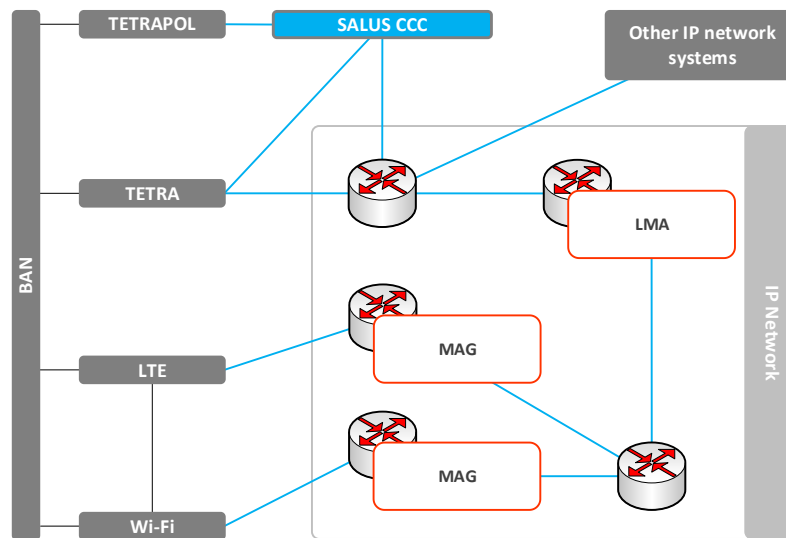


Figure 6 – IPv6 Mobility Management components

Mobility management components are also deployed in the SALUS PPDR Platform to allow handovers within the same network system (i.e., Wi-Fi to Wi-Fi) and/or between heterogeneous network systems, for example from Wi-Fi to LTE, or vice-versa. Figure 6 depicts the elements enabling such handovers. The Mobility Access Gateway (MAG) and the Local Mobility Anchor (LMA) components manage mobility in a technology independent fashion, for IPv6.

2.2 Platform components

This subsection details the components of the PPDR platform.

2.2.1 TETRA

The Terrestrial Trunked Radio (TETRA) is employed in all the SALUS use cases for mission critical services, such as group calls, late entry and emergency calls. Moreover, the TETRA components include also security features, specified in the SALUS Security Architecture [36], to enable the registration of devices, the mutual authentication or even, the air interface encryption.

The TETRA network consists of a rapid deployable TetraNode Mobile system, illustrated in Figure 7.



Figure 7 – The TetraNode

The TetraNode Mobile system includes both the TetraNode Core soft switch and two TETRA Base Station transceivers. The TetraNode Core soft switch runs on a single CompactPCI

platform for industrial applications. This open-standards based hardware platform is designed for industrial, aerospace and military applications, offering high tolerance against electrical and environmental disturbance. The choice of the Linux operating system further helps to achieve high availability and real-time operation. Because the TetraNode Core software, the computing platform, base station carriers and RF distribution system are all identical as for the TetraNode Industrial platform that is designed for mid-size TETRA networks, end-users of TetraNode Mobile benefit from the scalability, performance and economy of scale of these standardised solutions.

The TetraNode Mobile system is equipped with two TETRA Base Stations, offering a capacity of one control channel and up to seven speech and data channels. The system is prepared for the 410 - 430 MHz band for private TETRA applications, whereby the operating frequencies can be assigned freely within the range of 420 - 425 MHz (downlink) and 410 - 425 MHz (uplink). The output power can be configured between 1 and 25 Watts per channel, which can be decreased from 10 mW up to 250 mW by means of a 20 dB attenuator.

TetraNode Mobile provides the following external connections:

- Power Supply 230 VAC nominal, maximum 600 Watts;
- Transmit / receive antenna, output of 2 x 25 Watt maximum;
- Second receiver antenna (optional);
- IP / Ethernet for management, dispatch, Session Initiation Protocol (SIP) telephony and system interconnection (optional).

TetraNode Mobile supports all generic TETRA functionality, including:

- Registration and de-registration;
- Group attachments, including group selection and group scanning;
- Group Call, including Talking Party Identification (TPI) and Late Entry (LE);
- Individual Call, both duplex and semi-duplex;
- Telephony Call, both individual and group;
- Call Identification (CI);
- Priority Call (PC), Pre-emptive Priority Call (PPC) and Emergency Call;
- Speech Item Priority;
- Broadcast Call;
- Status and text messaging;
- Location reporting;
- Dynamic Group Number Assignment (DGNA);
- Discreet Listening (DL);
- Ambience Listening (AL);
- Barring of Incoming and Outgoing Call (BIC, BOC);
- Call Authorised by Dispatcher (CAD);
- Call Forwarding Unconditional (CFU);
- Call Forwarding on Busy (CFB), on No Reply (CFNRy) and Not Reachable (CFNRe);
- Include Call (IC);
- Call Transfer (CT);
- Authentication and Mutual Authentication;
- Air Interface Encryption (AIE);
- End to End Encryption (E2EE);
- Enable, Disable and Permanent Disable.

2.2.1.1 TETRA Dispatch Station

The Line Dispatch Station "Chameleon" (LDS-Chameleon) offers the capability to monitor activity on multiple group calls, as well as the capability to participate in these group calls. In addition, a wide range of features are available to receive and initiate individual calls, receive and transmit text and status messages, and to access specific TETRA features such as Ambience Listening (AL) and Enable / Disable.



Figure 8 – LDS Chameleon

The LDS Chameleon runs on a standard PC using the Microsoft Windows 7 operating system, as illustrated in Figure 8. The scope of delivery includes a PC with monitor, keyboard and mouse (or laptop), Rohill USB Dispatch Microphone (UDM) and a speaker set.

The typical application of the LDS Chameleon is to oversee operations of public safety agencies from the SALUC Command and Control Centre (CCC). The dispatcher can observe the communication of up to 16 groups simultaneously, whereby the talking users are identified by means of a TETRA radio identity and name alias, and each of the groups can be made audible in order to hear the communication. At any time, the dispatcher is able to interrupt the communication in order to talk back to TETRA radio and LTE device users, even while these users keep the Push-To-Talk (PTT) pressed. In addition, the dispatcher is able to patch two or more groups together in order to let them operate as a single group. This supports scenarios whereby different teams or agencies need to work together.

The LDS Chameleon operator is able to communicate with both TETRA radios and LTE device users, as depicted in Figure 9.

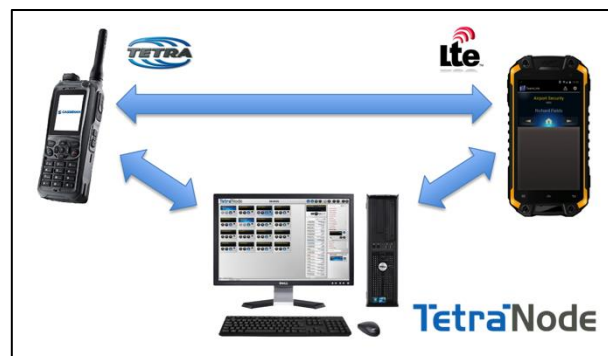


Figure 9 – LDS Chameleon and dispatcher functionality

Both TETRA radios and LTE devices can be provisioned to become interoperable within a single group, or provisioned as different groups that can be patched by the LDS Chameleon operator to behave as a single group by means of a tactical patch.

2.2.1.2 Network Management Station

The **Network Management Station (NMS)** provides the interface between the network and the network manager. It allows the network manager to configure, control, monitor, authorise and maintain the network and its subscribers.

The server of the network management system is integrated in the TetraNode eXchange. The network manager accesses the server by a network management station. The network management station is a Microsoft Windows 7 based PC running the TetraNode Network Management Station application, as depicted in Figure 10.



Figure 10 – TETRA network management station

TetraNode offers fully distributed network management. The databases are located on every TetraNode eXchange to achieve a maximum level of resilience, moreover reference locations are assigned to specific database records to maintain database consistency. Multiple Network Management System clients can access each of the TetraNode eXchanges using HyperText Transfer Protocol (HTTP) and HyperText Transfer Protocol Secure (HTTPS).

The NMS client application offers easy and intuitive navigation through system and node specific parameters and subscriber databases by using the Object Browser. Operation of this Object Browser is very similar to the Microsoft Windows Explorer, making it simple for operators to become acquainted with this part of the NMS. Furthermore, the windows, dialogs and toolbars have a similar and familiar appearance as common Windows applications such as Microsoft Office.



Figure 11 – TeTRA NMS NodeView

NodeView is an application to monitor the real-time performance of the TetraNode network, as illustrated in Figure 11. The application is typically used by a network operator, but can also be used by personnel in the SALUS CCC to obtain a view of TETRA network resources.

The NodeView application offers powerful yet simple-to-use features for network monitoring. It provides both performance monitoring and alarm reporting capabilities. The real-time traffic and event information to the NodeView application is provided by the TetraNode eXchange. The

network manager accesses the NodeView functionality by a workstation consisting of a Microsoft Windows PC, running the TetraNode NodeView application.

The NodeView application provides real-time information about the traffic load and performance of the TetraNode network components and interfaces. Per base station site information is provided about the slots in use, the called, calling and talking parties, the loading of the control channel, the number of calls placed in the queue, among other metrics.

In addition, major failures are shown that might result in loss of capacity. Similar information is shown for all other sites and devices present in the TetraNode system.

2.2.1.3 Broadband Gateway

The Broadband Gateway of TETRA is based on the Dispatch Application Platform (DAP) and is illustrated in Figure 12, on which the CVDP Relay and Gateway applications are running. These applications "talk" the Critical Voice and Data Protocol (CVDP) between each other and the TeamLink applications on top of Android smartphones.

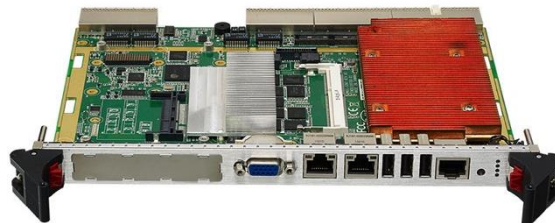


Figure 12 – TeTRA Broadband Gateway

The DAP is a generic framework for server-based applications that runs on the Linux system. Moreover, the DAP in SALUS is installed on a CompactPCI processor board for industrial applications, and inserted in an empty slot of the CompactPCI rack available within the TetraNode Mobile system (cf. Section 2.2.1). The DAP provides generic features for application control and monitoring, such as a generic web server for application installation and upgrades, configuration control and monitoring, logging capabilities, fault reporting using Single Network Management Protocol (SNMP), databases and redundancy operation.

2.2.2 TETRAPOL

TETRAPOL is employed in all the SALUS use cases for mission critical services, as some countries use TETRAPOL instead of employing TETRA. The mission critical services are common to TETRA and include group calls, late entry and emergency calls, among other features. The TETRAPOL equipment consists of a rapid deployable TETRAPOL Independent Digital Repeater (IDR) 3G, illustrated in Figure 13 and an Access Gate Radio AG-R illustrated in Figure 14. The IDR enables the capacity to create an independent tactical bubble and can be deployed indoors, outdoors or in a vehicle, for instance for the Disaster Recovery use case.



Figure 13 – TETRAPOL Independent Digital Repeater 3G

The Access Gate Radio AG-R solution is designed to provide the SALUS Command Control Centre access to TETRAPOL radio communications to mission critical services, such as group, IDR or direct mode, private, or emergency communication. Unlike line connected access, such as the line connected terminal, which requires a physical connection between the command control centre and the network infrastructure, the AG Radio only needs to be under the radio coverage of a network cell to allow the dispatcher or radio operator to participate in on-field communications.



Figure 14 – TETRAPOL Radio Access Gate

The AG-R solution can be used either as a permanent solution to give access to the dispatcher in the SALUS CCC or to give access to an on-field wireless dispatch, or as a backup solution for wired access to the control room. The flexibility of deployment of the TETRAPOL equipment enables its use in the different SALUS use cases. Coupled with the IDR solution the AG-R allows the control room to participate in IDR mode communications under the tactical cell coverage. In addition, the AG-R enables data communications in IDR mode up to the SALUS CCC, namely the Short Message Service (SMS), the operational status and geo-location data exchange.

The AG-R power supply can be remotely managed through IP link, mainly through the CCAPI interface. With this common interface, the radio switch of the command control centre can manage the same way the wired access and the AG-R on the TETRAPOL network.

In the SALUS PPDR Platform, the IDR 3G is associated to the AG-R enables TETRAPOL IDR radio communications and a radio-connected interface to the command control centres, as illustrated in Figure 15. The working frequency can be adapted, according to the range of available frequencies.

IDR 3G description:

- Rack 19" / 3U;
- Power supply: 100-240V AC or 10-36V DC, no battery;
- 5 MHz window in a 380-430 MHz frequency band (NB02, LU12 and LU08 available);
- Compatible with all TETRAPOL terminals MC 9600;
- Open Channel in half-duplex mode, channels dedicated to IDR mode;
- Communications independent from the main communication network;
- 4 power level on user choice: 15 W, 10W, 3 W or 1.5 W;
- Cyphering TETRAPOL.

AG Radio description:

- Rack 19" / 2U;
- Power supply: 110-220V AC (remote controlled power supply through IP link);
- Consumption: 160W (220V);
- Embedded encryption component (ASIC);
- End-to-end encryption of voice and data;

- 380-430 MHz with 10 or 12.5 kHz spacing;
- Maximum power at transmitter output: 10W.

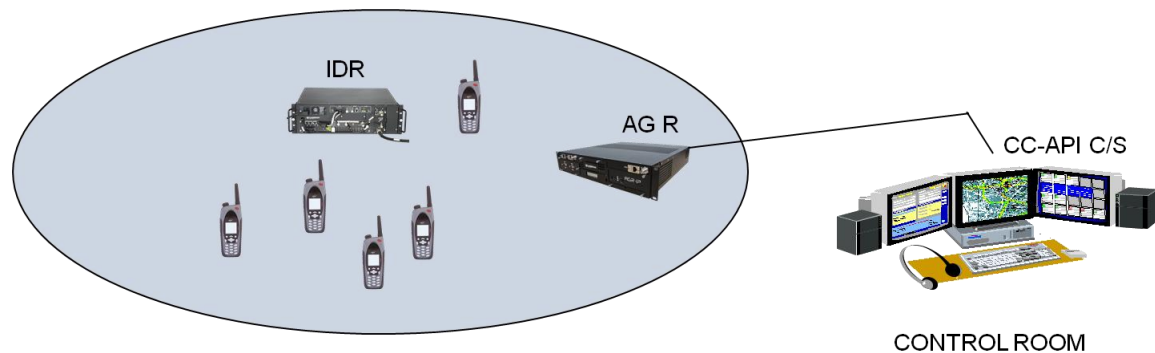


Figure 15 – TETRAPOL IDR and AG-R and SALUS CCC with CCAPI Client and Server

2.2.2.1 TETRAPOL Control Room

The TETRAPOL Control Room, or Command Control Centre interface is the only interface available on TETRAPOL to have access to basic services on the TETRAPOL network, that is access to mission critical services, such as Direct Mode Operation (DMO), group calls, PTT, priority calls, among others. The CCAPI interface enables these functionalities.

The CCAPI Server allows the management of the Access Gate Radio (AG-R) from the Command Control Centre, while the CCAPI Client communicates with the CCAPI Server to exploit the link toward the TETRAPOL Network.

2.2.3 LTE

The SALUS security and privacy architecture besides defining all the components for security also includes details for the migration roadmap of PPDR technologies. In particular, LTE initially is used in cooperation with LTE commercial operators for non-mission critical interworking and afterwards as a LTE network with full mission critical services support.

To enable LTE as a future PPDR network, SALUS considers two main components the LTE Evolved Node (eNodeB) to provide a working network infrastructure and the Open Evolved Packet Core (EPC) with Media Independent Handover (MIH) support for mobility management between heterogeneous technologies.

2.2.3.1 LTE Evolved Node B

The evolved Universal Terrestrial Radio Access Network (eUTRAN) is part of the improved and simplified LTE Radio Access Network (RAN) providing wireless interface to the User Equipments (UE).

The eNodeB, illustrated in Figure 16, is composed of the following elements as shown below:

- The digital part known as the Base Band Unit (BBU): Alcatel-Lucent 9926 D2U composed of Channel Element Module (CEM), Core Control Module (CCM) and rack. The following paragraphs detail the information of each building block and the available configurations provided are shown below.
- The radio part in a compact solution with Transmitter/Receiver Duplexer Unit (TRDU) or distributed with Remote Radio Head connected to antennas.

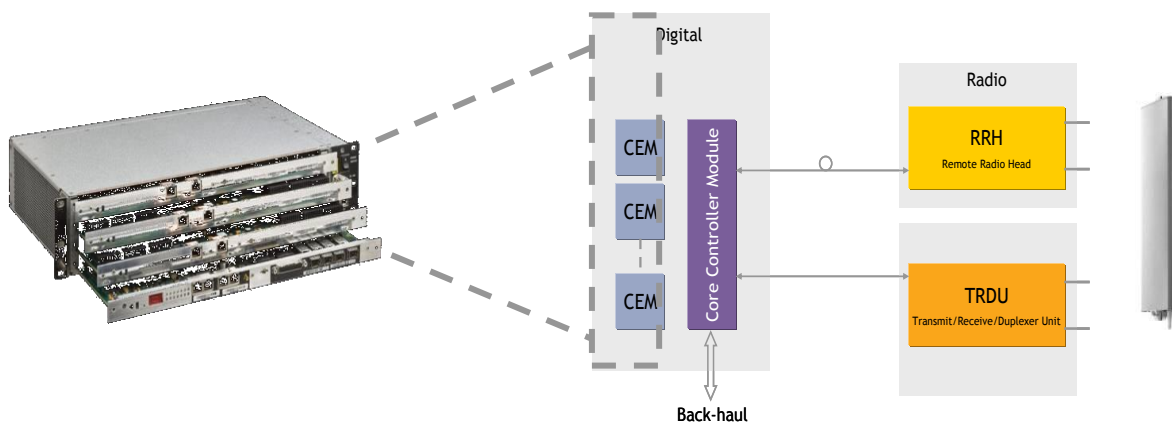


Figure 16 – eNodeB generic architecture

The eNodeB has multiple deployment options, as depicted in Figure 17. In both cases of distributed and macro solution:

- The separation between digital and RF processing is ensured through the Common Public Radio Interface (CPRI) interface,
- The digital processing is ensured by the Base Band Unit, which has the same architecture for macro (compact) and distributed eNodeBs.

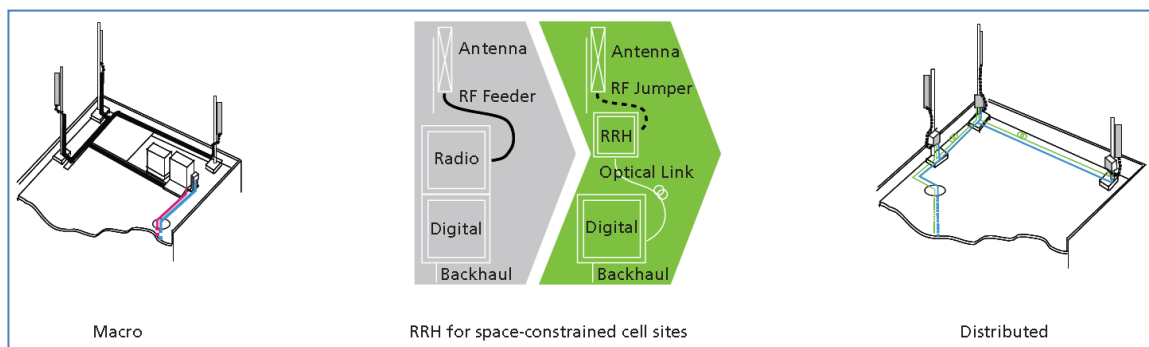


Figure 17 – eNodeB deployment options

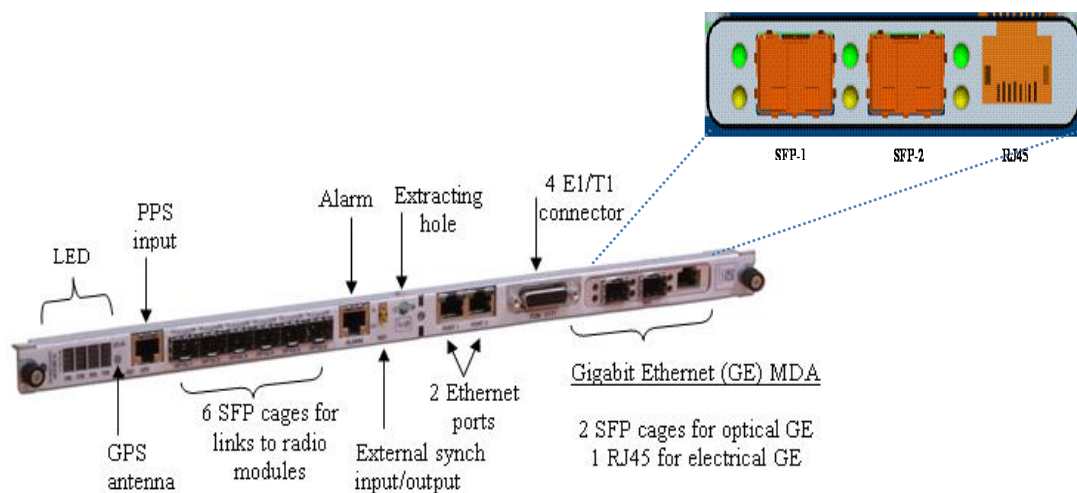


Figure 18 – eCCM-U with GE MDA - Face Plate

The faceplate of the eCCM-U, as illustrated in Figure 18, provides the diverse connectors and LEDs, which correspond to:

- LEDs providing the current status of the module;
- One port for connection to external GPS antenna (in case of integrated GPS receiver) via a Sub-Miniature B (SMB) connector;
- One RJ45 connector used for PPS input from an external GPS receiver; this connector also can provide one RS232 port for control of external equipment;
- One block of 6 Small Form-factor Pluggable SFP format connectors for connectivity to the radios (Remote Radio Head (RRH), TRDU); each SFP connector is equipped with two light indicators; they support 6 CPRI links or 3 CPRI + 3 High Speed Serial Link (HSSL) (HSSL only for UMTS);
- One RJ45 connector used for a 1-wire interface (connectivity to external alarm module, commissioning...).
- One dual SMP connector providing input for 10MHz/15MHz input and 15MHz output.

2.2.3.2 LTE Evolved Packet Core

This subsection details the OpenEPC module (provided by UPAT) and the Micro core solution (provided by ALU-I). OpenEPC is employed to integrate MIH to enable advanced mobility management.

2.2.3.2.1 OpenEPC

3GPP EPC is a key part of the SALUS heterogeneous wireless access implementation. This section presents the OpenEPC platform used to support seamless connectivity that has been developed by Fraunhofer FOKUS [19]. The key elements of EPC that have been configured within the context of SALUS are illustrated in Figure 19.

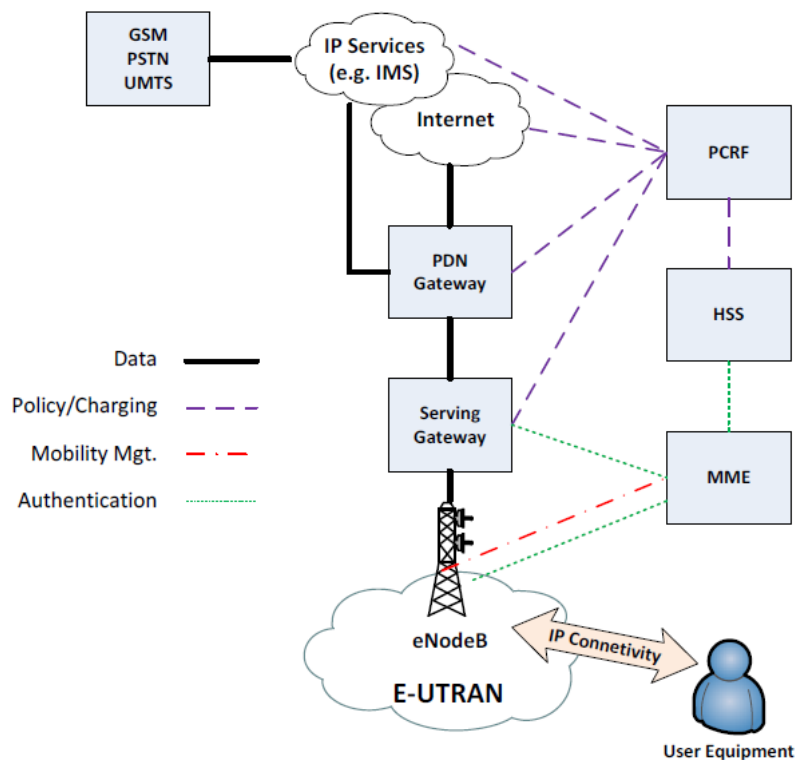


Figure 19 – Evolved Packet Core Architectural Modules

The OpenEPC main modules are the following:

- **Home Subscriber Server (HSS):** This is the main database of the EPC, responsible for storing subscriber's information that may include profile description, within related information, and information regarding the subscriber's established sessions.
- **Mobility Management Entity (MME):** This is the module that performs the mobility management and mobility functions between 3GPP access networks (i.e. eUTRAN and GERAN/UTRAN) by selecting the Serving Gateway (S-GW) and the Packet Data Network Gateway (PDN-GW), as well as, the Service GPRS Support Node (SGSN) during handovers to 2G or 3G 3GPP access networks.
- **Serving Gateway (S-GW):** It is the gateway, where the interface towards eUTRAN terminates. A User Equipment (UE) associated with the EPC is served by a single Serving-GW. In terms of implementation both the S-GW and the PDN-GW may be dwelling in a single physical component.
- **Packet Data Network Gateway (PDN-GW):** It terminates the interface towards the packet data networks (e.g. IP Multimedia System (IMS), Internet, etc.). In case where the UE is connected with different IP services, several PDN-GW are associated with this UE. It performs UE IP allocation, policy enforcement, packet filtering per user and packet screening.
- **Policy and Charging Rules Function (PCRF):** It performs QoS control, access control and charging control. Its function is policy based and authorizes bearer and session establishment and management.
- **Evolved-UTRAN:** It is a simplified radio access network architecture that is composed only with evolved NodeBs (eNodeBs). In terms of functionality Evolved UMTS Terrestrial Radio Access Network (eUTRAN) supports Radio Resource Management (RRM), selection of MME, is responsible for routing user data to the S-GW and for performing scheduling, measurements and reporting.

2.2.3.2.2 Micro Core

This subsection details the Micro Core solution of LTE from ALU-I, based on an all-in-one platform as detailed in Figure 20.

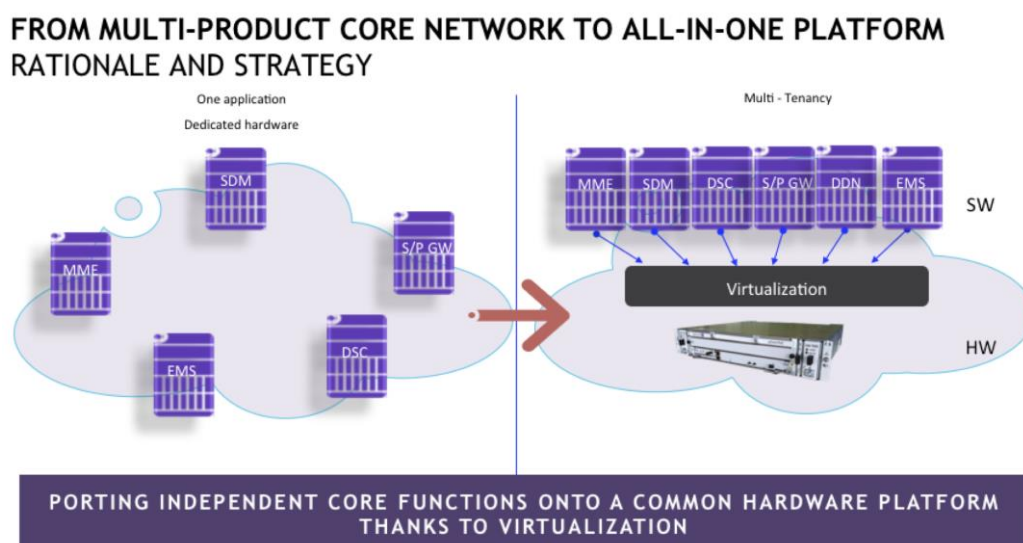


Figure 20 – Micro Core LTE overview

Alcatel-Lucent 9773 LTE Micro Core is the next generation technology platform using the latest virtualization techniques (NFV approach) to integrate in a single and very compact form factor all the different mobile telecom functions of an LTE Core Network, EPC, augmented with HSS and OAM capabilities, as illustrated in Figure 21. It represents a technology leap in the way LTE products are designed and offers numerous advantages over more traditional products based on dedicated hardware/platforms.



Figure 21 – Alcatel-Lucent 9773 LTE Micro Core detail

Based on x86 based platform and leveraging Alcatel-Lucent expertise in LTE and virtualization techniques, the LTE Micro Core delivers superior performance and high reliability. Targeted as small and rapid LTE network deployments, it can fulfil the requirements of a wide variety of uses cases ranging from fix deployments to on the wheel network infrastructures. It also offers a very cost-efficient and future-proof solution for players seeking a simple to deploy and easy to operate small LTE Core Network solution for private networks.

Alcatel-Lucent 9773 LTE Micro Core is a new component of Alcatel-Lucent virtualized mobile solutions portfolio, which leverages virtualization to run telecom applications over standard hardware based platforms and targeted at evolving mobile broadband networks toward cloud-based architectures. The general features supported by the LTE Micro Core are summarized in Table 1.

Table 1 – LTE Micro Core supported features

Features	Performance
Number of eNBs	Up to 22 (first release)
Max Number of Users	2000 active users (first release)
QoS	QCI 1 to 9
Interfaces	S1, S6a, S11, SGi, Gx, Rx, S13, S5
UL + DL Throughput	1Gb/s
3GPP SW Alignment	3GPP R10

ALU-I provides a compact LTE ePC and eUTRAN “micro-core” with one 2.6GHz sector, which is composed by one 9773 LTE Micro Core (LMC) and one eNodeB (BBU2).

2.2.3.3 LTE Network Monitoring System

The LTE provided by ALU-I is monitored by ALU-I 5620 Service Aware Manager (SAM) product, with on RRH in band 20 – 800MHz.

2.2.3.4 LTE Dispatch Station

The Rohill LTE dispatch station consists of a rugged Dell laptop running the Rohill LDS Chameleon on top of the Microsoft Windows 7 operating system, as illustrated in Figure 22. The laptop is connected to the LTE infrastructure by means of a LTE “dongle”. The communication is based on an Unicast UDP/IP connection with the TetraNode Mobile system using the TetraNode IP Gateway version 2 (TIGv2).



Figure 22 – TETRA Dispatch Station

For more information on the capabilities and applications of the LDS Chameleon refer to Section 2.2.1.1. The LTE dispatch station can be used as a mobile command and control solution that is close to the incident.

2.2.3.5 LTE Wi-Fi Terminal

The SALUS PPDR platform is composed by diverse LTE terminals, which are summarized in Table 2.

Table 2 – LTE Terminals

Partner	Terminal	Supported Interfaces	Additional Information	OS
ALU-I	USB dongle 4G band	Band 20 800 MHz	for equipments remote access	Windows driver and session manager
	Samsung Galaxy S3 4G		for hand-held applications	Android 3.2
	Samsung GalaxyTab 4G			Android 4.1
IT	LG G2	LTE; Wi-Fi 802.11 b/g/n; Bluetooth; NFC		Android 4.2.2
ONE	ONE Plus One	LTE; Wi-Fi 802.11 b/g/n/ac; Bluetooth; NFC	built-in sensors for measuring acceleration, proximity, gyroscope and compass	Android 4.4.4
KU	LG Nexus 5	LTE; Wi-Fi 802.11 b/g/n/ac; Bluetooth; NFC	built-in sensors for measuring acceleration, proximity, gyroscope and compass	Android 4.2
UL	Samsung Galaxy Ace 3 (GT-S7275R)	LTE; Wi-Fi 802.11 b/g/n; Bluetooth ¹ ; NFC	built-in sensors for measuring acceleration and proximity	Android 4.4.4
UB	Nokia Lumia 720, Samsung	LTE; Wi-Fi 802.11 b/g/n;	Built-in movement sensors (accelerometer, gyroscope, magnetometer)	Windows Phone 8.1 & Android 4.2

2.2.4 Wi-Fi

The Wi-Fi network is an important part of the various communication technologies expected to interoperate in the SALUS PPDR platform. The SALUS platform benefits from the range of advantages that the Wi-Fi technology offers such as larger bandwidth than the conventional PMR technologies, fast deployment and interoperability with other technologies like LTE,

¹ To allow the connection of Zephyr HxM Heart-Rate monitor

Bluetooth. In this way, high data rate multimedia services with their appropriate Quality of Service (QoS) requirements can be supported.

2.2.4.1 Wi-Fi Terminal

A number of Wi-Fi terminals will be used in the SALUS PPDR Platform. The majority of these devices will run different versions of Android operating systems. Brief descriptions of the terminals are presented below.

Rugged Devices

In the SALUS disaster recovery use the Ad-Hoc Wi-Fi connectivity allows PPDR forces to perform rescue operations when no communication infrastructure can be provided. In this sense the SALUS PPDR platform includes rugged devices that have higher protection for water, dust and shock. The HW-T18 Ultra-Rugged smartphone device runs the ChaMeLeon routing protocol on Android. It supports Ad-Hoc Wi-Fi connectivity and is waterproof, dustproof and shockproof. This smartphone is built to withstand the hazards of a typical emergency operational environment.



Figure 23 – KU Wi-Fi Rugged device

Other Android Devices

As mentioned in Table 2, the Nexus 5 device has the capability of connecting to multiple radio access technologies, which can serve as a medium to provide connection to the external networks.

2.2.4.2 Wi-Fi Ad-Hoc

The Wi-Fi Ad-Hoc as a decentralized wireless network will play a vital role for communication between first responders when pre-existing infrastructures are destroyed. This is relevant with all considered scenarios within the SALUS project and dominant in the Disaster Recovery use case. Routing of packets is an important mechanism within Ad-Hoc networks. Every node in an Ad-Hoc network has the capability to serve as a router. The reliability and efficiency of this routing protocol determines the quality of services that can be maintained within the network. Therefore, the SALUS PPDR platform includes support for ChaMeLeon Routing Protocol (CML) [34]. CML is an adaptive and hybrid routing protocol that was designed to operate in Mobile Ad-Hoc Networks supporting emergency communications.

The main characteristic of CML is the adaptability of its routing mechanisms towards changes in the physical and logical state of a Mobile Ad-Hoc Network (MANET).

The SALUS Ad-Hoc Network also includes support to connect to LTE networks, as depicted in Figure 24. The overall system can be regarded as a layer 2 switch. Client devices can connect to the SALUS Ad-Hoc Network both wired (Ethernet) and wireless (Wi-Fi). Any device that connects to the network can communicate with another client device in the network, both where connections to the infrastructure are possible, and in cases when this is not possible.

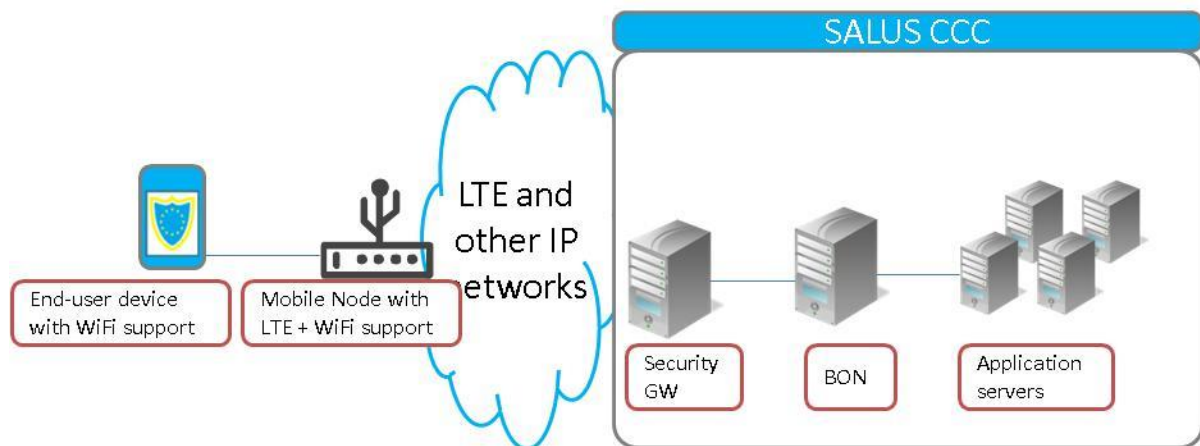


Figure 24 – SALUS Ad-Hoc network as distributed switch [36]

The SALUS Ad-Hoc network consists of multiple nodes; the ones on-field are called Mobile Nodes (MN), while the node in the infrastructure is called the Back Office Node (BON). The Mobile Nodes are physical units, whereas the Back Office Node is a virtual machine on a server platform. The BON serves as a VPN gateway for the MNs and provides transparent access from end devices to the application servers in the CCC.

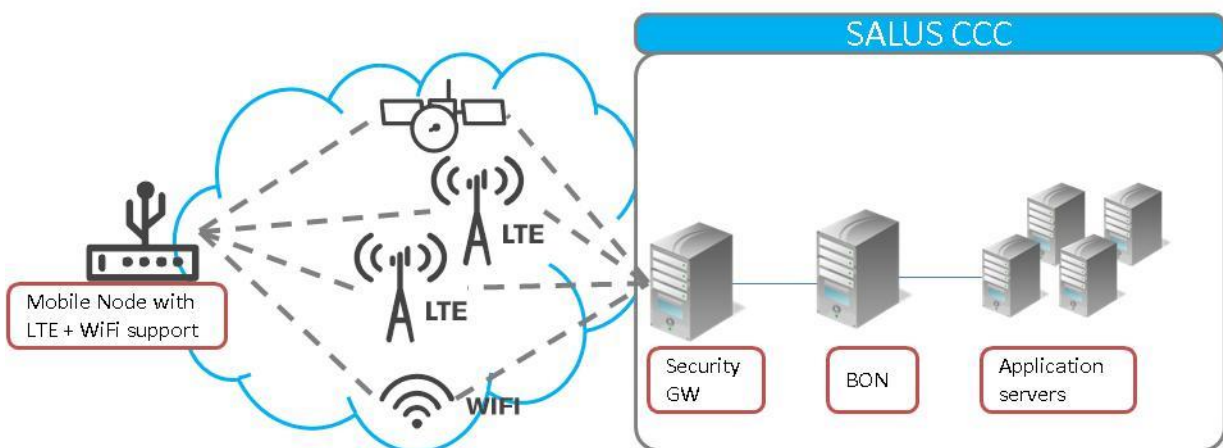


Figure 25 – Redundancy in the backhaul [36]

Figure 25 shows how the SALUS Ad-Hoc creates reliability. Every network has its own reliability, where coverage, congestion and network failure are the main reasons for unavailability. The SALUS Ad-Hoc network is based on the FIGO system, where the FIGO MN can connect to different LTE networks, as well as Wi-Fi infrastructures or through external modems like SatCom (not included in the SALUS PPDR platform). The MN and BON jointly determine the best way to connect, and will hand over the connection where required. Switching infrastructure connection has no impact on the IP addresses for the client devices.

When multiple MNs enter each other's radio coverage, they can create an Ad-Hoc network. Figure 26 shows that communication between clients that connect to MNs in the Ad-Hoc network will flow only through the Ad-Hoc network. No communication will occur through the infrastructure in such case. The Ad-Hoc network is a multi-hop network, where nodes can be repeaters for nodes that do not have a direct link. Figure 26 indicates this with the red arrow.

The above features of the redundant backhaul and the Ad-Hoc network operation are combined. Figure 27 shows how this leads to coverage extension. Suppose LTE operator X has an outage, whereas an MN that is part of the Ad-Hoc network also has a connection through a WLAN network, all MNs in the Ad-Hoc network can communicate using this WLAN connection.

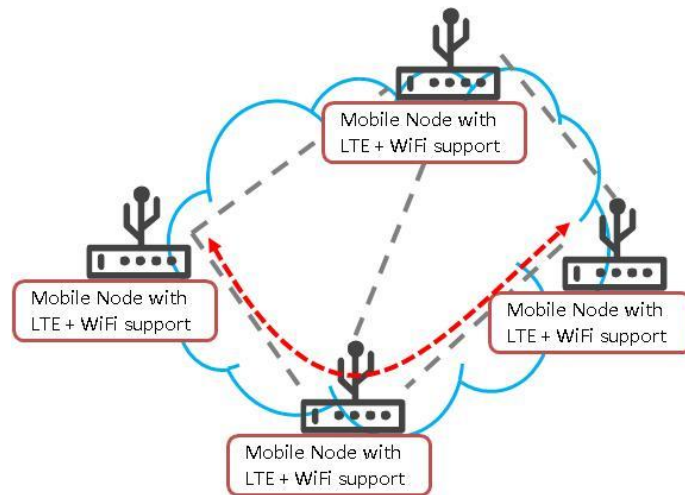


Figure 26 – SALUS Ad-Hoc network operation

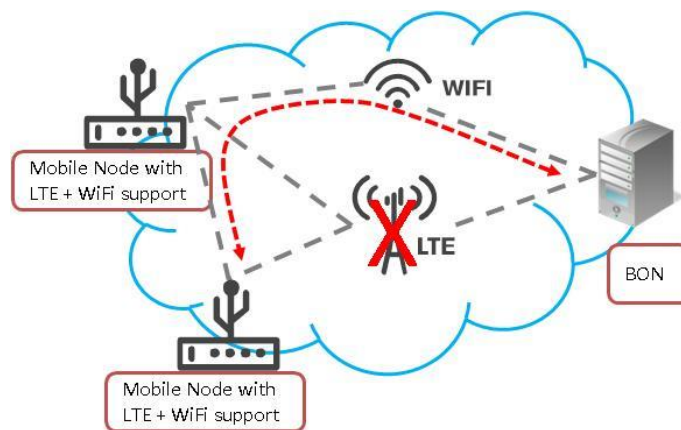


Figure 27 – Coverage extension

The SALUS Ad-Hoc network operates as a distributed switch. Figure 28 shows an example of IP addressing scheme.

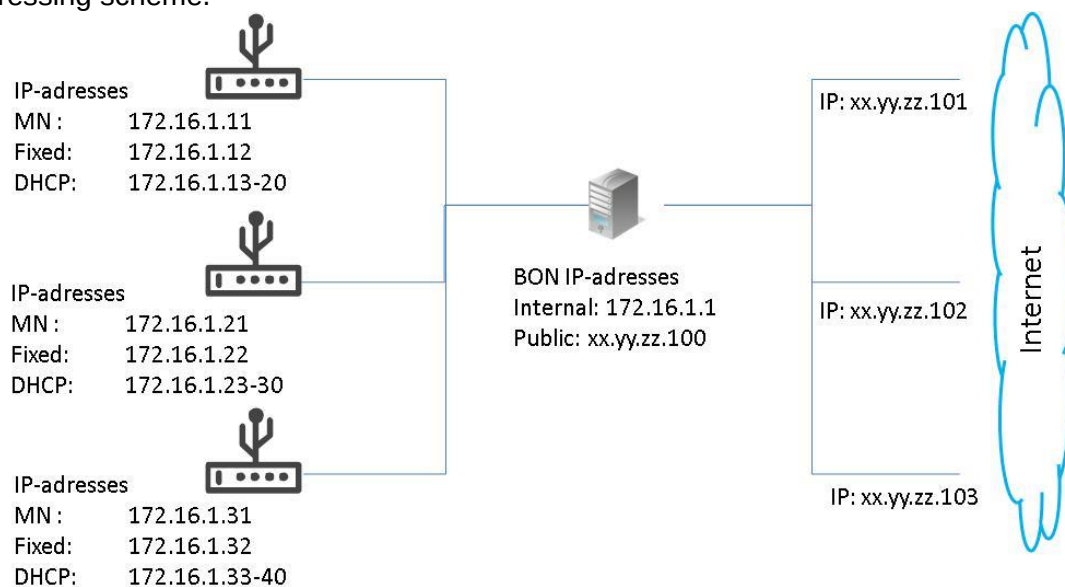


Figure 28 – Network ranges

One FIGO network is a /16 network, supporting 65536 IP addresses. Each MN has a unique IP address and can hand out a predefined set of IP addresses. When a MN is handed over from one MN to another, the client IP address doesn't change, so handing out the IP addresses is independent from where these addresses are used. The IP addressing scheme is an internal scheme. The connection between MN and BON will use the IP addresses of the network operator in use. This external scheme is only used by the MNs and BON. Figure 29 shows how the internal addresses are transported over the external network through a Virtual Private Network, using OpenVPN [40].

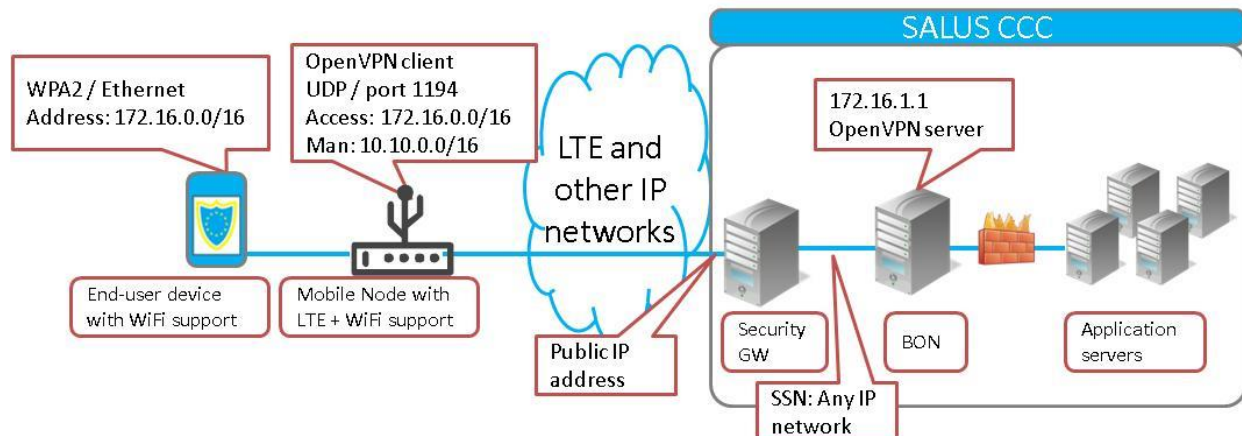


Figure 29 – Mapping of OpenVPN link

2.2.4.3 Wi-Fi Network Monitoring System

The FIGO system has a Network Monitoring System (NMS), which is called the Central Management System (CMS). In addition, the system manager can connect locally at each node to the node specific controls through a local web interface. For illustration, Figure 30 shows the overview page of CMS. It provides a first overview of the status of the overall system.

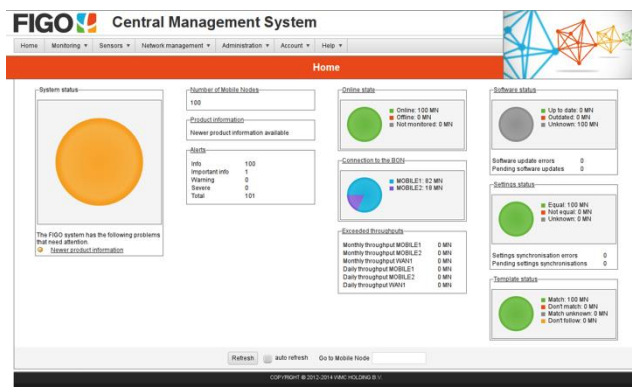


Figure 30 – Overview page of CMS

CMS allows the operator to add nodes, modify settings for all or specific nodes, view the statistical behaviour of nodes or connections and see the location of nodes. For illustration, Figure 31 – CMS views: top-left: map; top right: statistics; bottom left: setting overview; bottom right: modify settings shows a number of sample views of CMS, such as map, statistics views or settings and modification overviews.

A user in the field can manage the node it is connected to using a local web interface. This web interface provides a subset of the functionality, mainly focussing on management of settings.

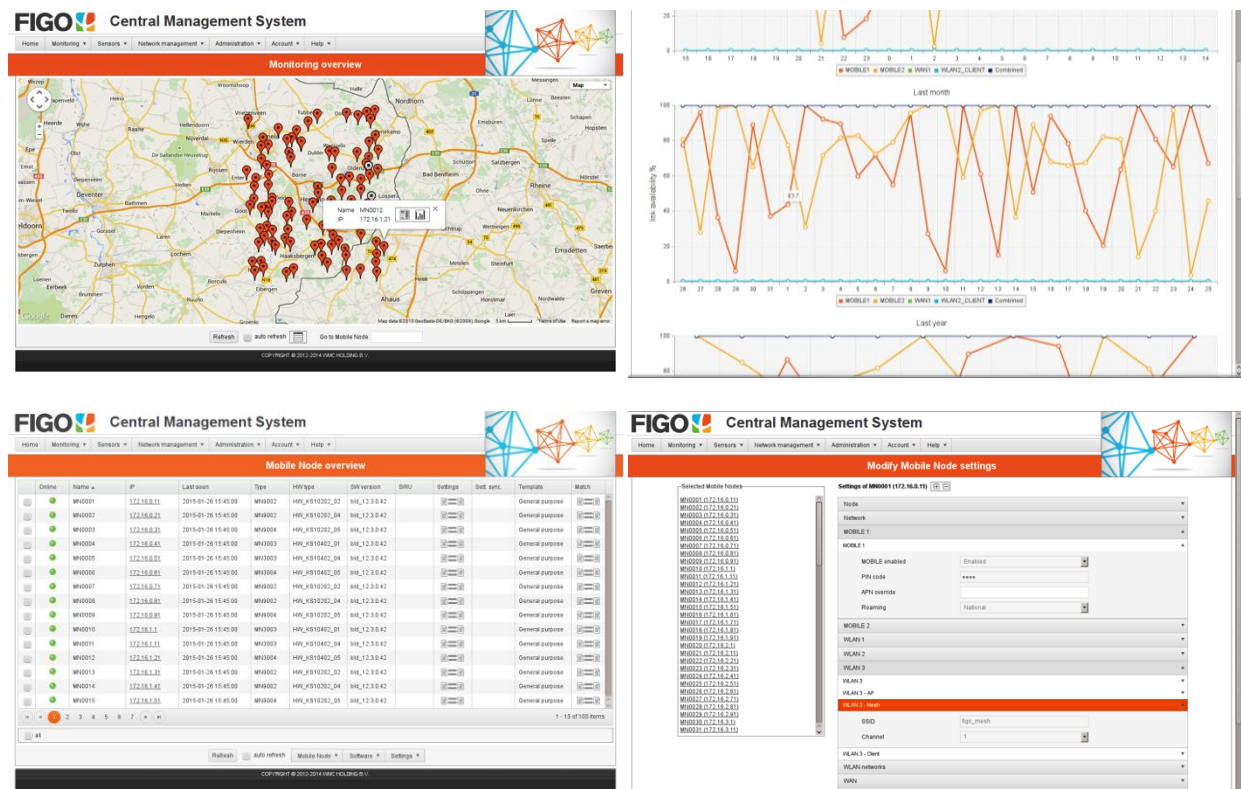


Figure 31 – CMS views: top-left: map; top right: statistics; bottom left: setting overview; bottom right: modify settings

2.2.5 Wireless Sensor Networks

Wireless sensor networks are an integral part of SALUS. They allow collection of information from the environment that improves situation awareness of PPDR forces. The first iteration of the SALUS prototype shall demonstrate a couple of sensor-based applications that are described in the following sections. It should be noted, that these solutions are not yet fully integrated and they often provide dedicated application specific components to achieve required goals. However, the work on a unified architecture is on-going and is also presented in this section. We envision the full integration of all components into proposed architecture in the second prototype iteration.

2.2.5.1 Sensors API Architecture

The Sensor API architecture denotes the components that facilitate the collection, transport and processing of sensor data. Sensor data is usually collected by sensors attached to the bodies of in-field deployed PPDR personnel and then sent via their hand-held terminals (UEs) to back-end application for processing. Since the sensors are of various kinds, the components needed to support such process are also numerous. The goal of the sensor API architecture is to provide a unified solution.

The architecture that SALUS proposes consists of a network element called the Message Broker (MB) that acts as an application level router. It represents a single network point of attachment for all devices that send sensor data and for all back-end application that process it. The concept of the message broker is introduced and described in Section 2.7 of SALUS deliverable D5.2 [36].

The message broker actually entails two components: the message broker service which is a network service to which sensor readings are sent and then forwarded to back-end applications for processing, and the message broker API, which is a software library that allows developers of UE applications and the developers of back-end applications to exchange messages with the help of the message broker service. These two components are discussed in Section 4.1.3.

2.2.5.2 Movement Sensors

Most smart-phones used today have integrated movement sensors. For that reason and the fact that smart-phones have built-in many modern communication interfaces (2G/3G /LTE, Wi-Fi, Bluetooth) smart-phones are used as movement sensors within SALUS prototype.

Movement sensors, within smart-phones, communicate to a server with public IP and preinstalled server application in the first prototype. The communication is done using the HyperText Transfer Protocol (HTTP), by exchanging messages in HyperText Markup Language (HTML) and Extensible Markup Language (XML) formats.

The first prototype of the "*man-down*" detection is based on HTML5/JavaScript/PHP code and works as follows. The architecture of the application is shown in Figure 32.

The publicly visible webpage contains HTML5/JavaScript code for movement sensors reading and "*man-down*" detection. The PPDR user who wishes to provide such data to SALUS opens the webpage, which starts the sensor reading from the user terminal, displays the readings on the terminal screen and communicates back to the server. The app can send raw sensor data or "*man-down*" alerts (or both), which can be configured in the code. The data and alerts are streamed to the server via Internet in XML form, in the background.

The server keeps the data, filters the data if necessary and communicates with higher system layers through an API. The server accepts HTML/PHP inputs, and provides output in XML format depending on the request. The output can be raw sensor data of one or more active users, user's history, or just currently active "*man-down*" alerts. Also, the server can be used in the development phase to show the possible usage of collected data to plot the user's position and sensors data or alert(s) on a map.

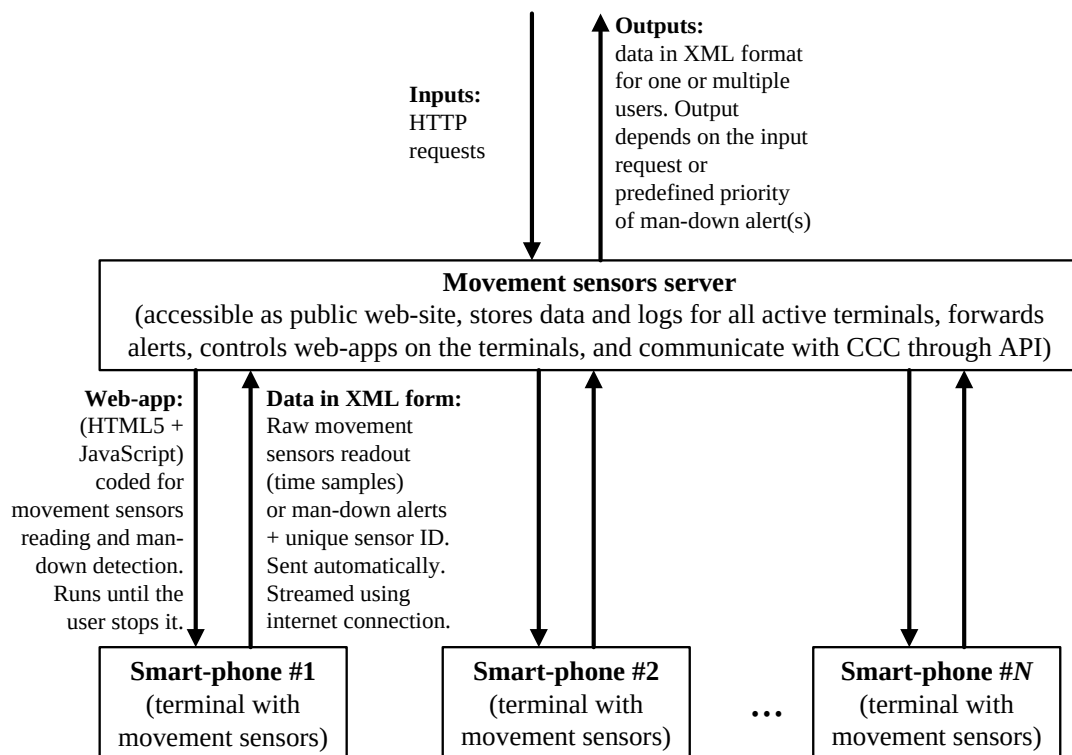


Figure 32 – The architecture of the first prototype for the "*man-down*" detection.

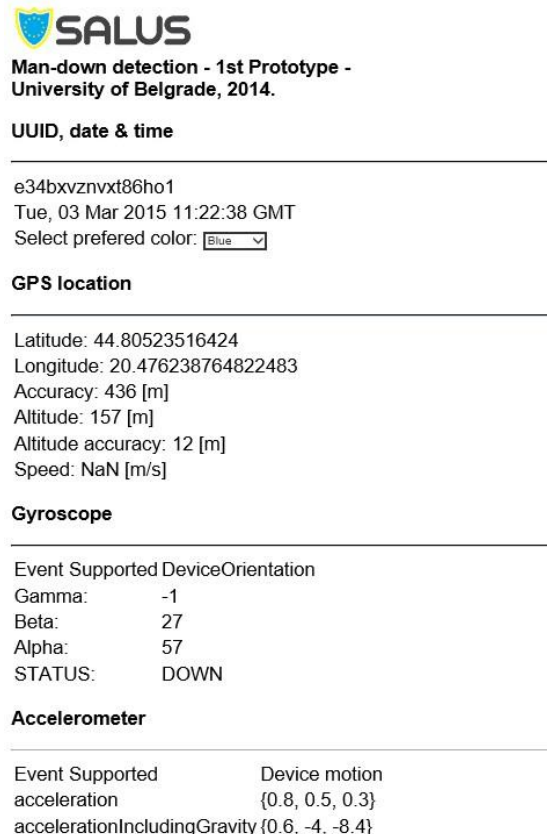
2.2.5.2.1 Prerequisites for using the application

In order to use the first working prototype of the application, the user needs a smart-phone with:

- Integrated movement sensor(s), most smart-phones on the market do have them;
- Working internet connection;
- Location Global Position System (GPS) enabled;
- Web-browser that is capable of handling HTML5/JavaScript.

2.2.5.2.2 Client-side Application

The user starts the application on his/her smart-phone by visiting the webpage, as depicted in Figure 33.



SALUS
Man-down detection - 1st Prototype -
University of Belgrade, 2014.

UUID, date & time

e34bxvznvxt86ho1
Tue, 03 Mar 2015 11:22:38 GMT
Select preferred color: Blue ▼

GPS location

Latitude: 44.80523516424
Longitude: 20.476238764822483
Accuracy: 436 [m]
Altitude: 157 [m]
Altitude accuracy: 12 [m]
Speed: NaN [m/s]

Gyroscope

Event Supported DeviceOrientation
Gamma: -1
Beta: 27
Alpha: 57
STATUS: DOWN

Accelerometer

Event Supported	Device motion
acceleration	{0.8, 0.5, 0.3}
accelerationIncludingGravity	{0.6, -4, -8.4}

Figure 33 – Man-down client application screen-shot

The application interface is divided into four parts, all of them visible simultaneously:

- UUID date & time;
- GPS location;
- Gyroscope;
- Accelerometer.

The first part, **UUID date & time**, contains information for Unique User Identifier (UUID), date and time of the last sensor reading, and allows the user to select the preferred colour for the presentation on the map. The UUID is a randomly generated sequence of 16 alphanumerical characters. Its main purpose is to uniquely distinguish one user from another. It is generated when the user visits the client webpage and is active as long as the user stays at the page (i.e., as long as the user provides the sensor readout). In the first part, the user can only change the preferred colour, among 10 predefined, for display at the map. This is provided for the development and testing purposes primarily.

The second part, **GPS location**, contains data for geolocation of the user/sensor, namely: latitude, longitude, accuracy, altitude, altitude accuracy, and speed. Those data are not necessary for the functioning of the "man-down" application but are needed in case that the sensor will be plotted at the map.

The third part, **Gyroscope**, contains data obtained from built-in gyroscope of the smart-phone. This data is essential for correct working of "man-down" application. The data consist of three angles: alpha, beta and gamma. From those data, the OK/DOWN status of the device is calculated. In the present version of the application, the device needs to stay down for 5 seconds in order to trigger the detection of "DOWN" position, while the "OK" status is triggered immediately. Those are variables in the code, and can be tuned if needed. There is a line, STATUS, which indicates the detected position of the device.

The fourth part, **Accelerometer**, contains data obtained from built-in accelerometer of the smart-phone. Those data are not currently used in "man-down" application, but are typical part of movement sensors present in smart-phones today. The accelerometer data can be used to determine is the officer (with the sensor) is standing, walking, or running for example, if needed.

2.2.5.2.3 Server-side Application

In order to demonstrate the possible usage of the "man-down" application, an HTML page that consumes the data from sensors and plots info on a map was created, as shown in Figure 34.

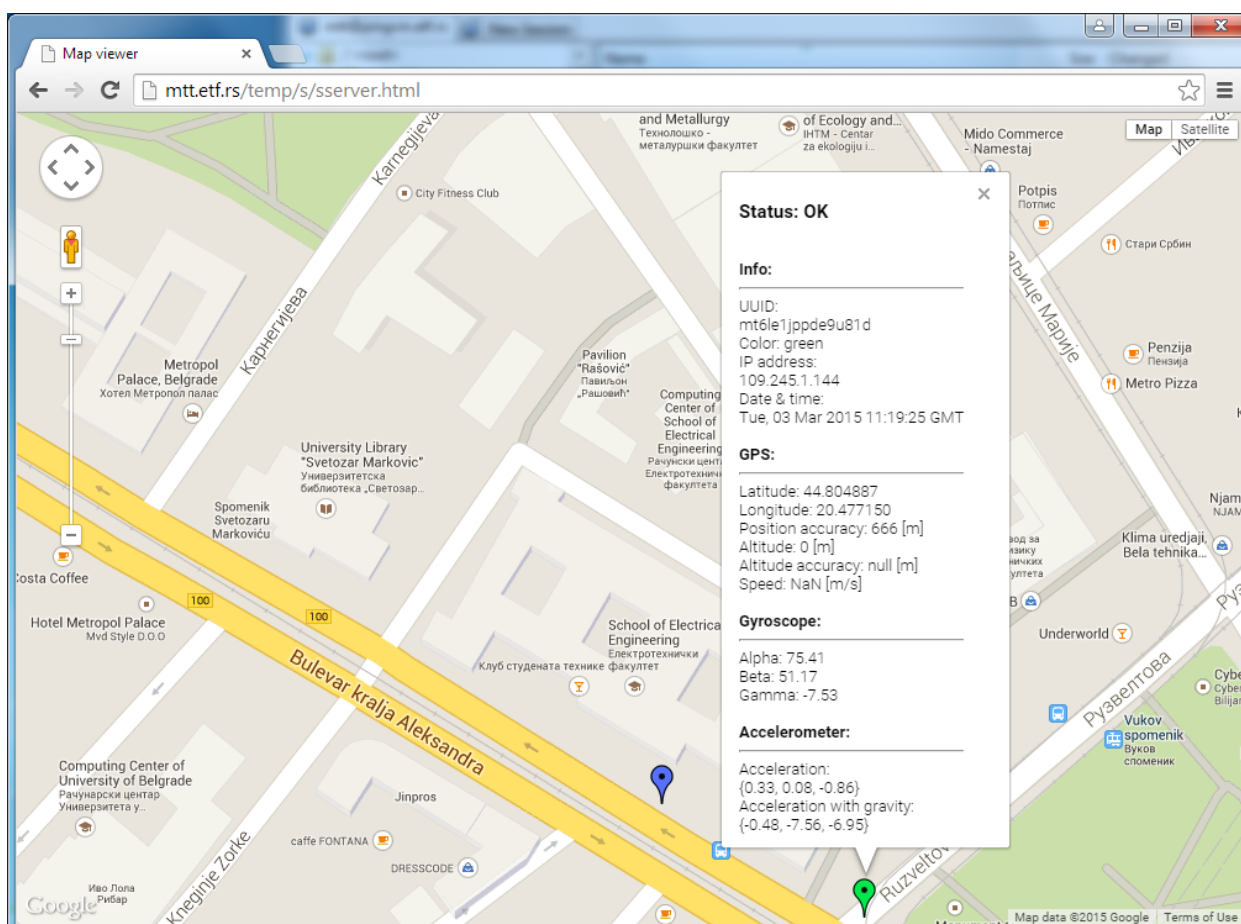


Figure 34 – Illustration of usage of sensor data (map plotting)

One pin at the map represents one sensor. There can be multiple sensors presented at the map. The colour of the pin can be chosen by the user, the default colour is green. If the sensor status is "down" the pin colour changes to red.

By clicking the pin, the map viewer can read all the data provided from the sensor to the server.

2.2.5.3 Bio-signals Sensors

This subsection provides details regarding sensors providing bio-signals, such as Heart Rate, blood pressure, temperature, or other kind of information like location updates.

2.2.5.3.1 Heart rate and location updates

As an example of a bio-signal monitoring application, UL shall demonstrate the streaming of heart rate signal and the GPS location of the PPDR user to the back-end application. (While the sending of location update is not a bio-signal application, we present them together since they are part of the same UE application.) The solution consists of the following components:

- A heart rate sensor (HxMTM Smart heart rate monitor from Zephyr™);
- A PPDR UE device (Samsung Galaxy Ace 3 running Android described in Table 2) with an application that uses the message broker API;
- The message broker service;
- A back-end application that shows the location of PPDR users on a map and, upon requests, readings from other sensors.

The heart rate sensor is connected to the UE device via the Bluetooth wireless connection. The connection is secured with a pairing procedure that should be performed as part of the set-up.

A dedicated application has been developed that runs on the PPDR UE device, collects readings from sensors (heart rate and GPS), and sends them to the back-end application via the message broker service. Since the PPDR UE runs the Android operating system, the developed application is also an Android application.

The Android application consists of two components: a service that runs in the background and a GUI that allows the user to interact with the application. The service represents the application core and does most of the work: it periodically queries sensors for readings, in configurable intervals; it packages readings into messages and forwards messages to the back-end application; and it responds to incoming requests from the message broker and can thus change the settings of the PPDR UE. All communication with the message broker service is performed with the help of the message broker API. The GUI is used only to allow the user to configure various settings and start or stop the background service, as depicted in Figure 35.

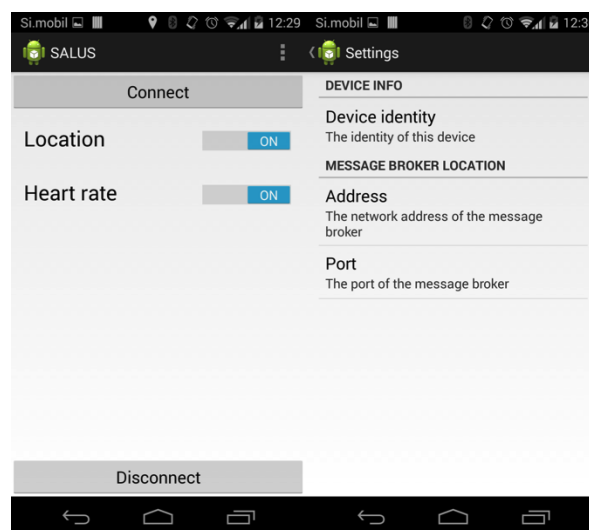


Figure 35 – A series of screen captures for the GUI part of the PPDR UE application

The exact format of messages that carry sensor readings has not been finalized yet. As an intermediate solution, all messages are in JavaScript Object Notation (JSON) format. Figure 36

features a screen capture of the logging console at the message broker service. The image shows the JSON format of messages that carry location information and the heart rate signal.

```
2015-01-29 11:39:04,624 [DSF] ['david#location', '{"timestamp":1422515490,"lng":14.48785922,"lat":46.0448745}']
2015-01-29 11:39:05,078 [DSF] ['david#heart_rate', '{"timestamp":1422527945,"rate":78}']
2015-01-29 11:39:05,662 [DSF] ['david#location', '{"timestamp":1422515490,"lng":14.48785922,"lat":46.0448745}']
2015-01-29 11:39:06,116 [DSF] ['david#heart_rate', '{"timestamp":1422527946,"rate":78}']
2015-01-29 11:39:06,620 [DSF] ['david#location', '{"timestamp":1422515490,"lng":14.48785922,"lat":46.0448745}']
2015-01-29 11:39:07,152 [DSF] ['david#heart_rate', '{"timestamp":1422527947,"rate":77}']
2015-01-29 11:39:07,827 [DSF] ['david#location', '{"timestamp":1422515490,"lng":14.48785922,"lat":46.0448745}']
2015-01-29 11:39:08,132 [DSF] ['david#heart_rate', '{"timestamp":1422527948,"rate":77}']
2015-01-29 11:39:08,838 [DSF] ['david#location', '{"timestamp":1422515490,"lng":14.48785922,"lat":46.0448745}']
2015-01-29 11:39:09,141 [DSF] ['david#heart_rate', '{"timestamp":1422527949,"rate":77}']
```

Figure 36 – The message broker service logging output.

The back-end application is intended to show the location of PPDR users on a map and, upon request, to sensibly visualize readings from various sensors. In the first prototype the back-end application is implemented as a stand-alone application, while in the second iteration of the prototype the application shall be integrated in the command and control centre. Currently, the back-end application is implemented as a Meteor application and runs in a web browser.

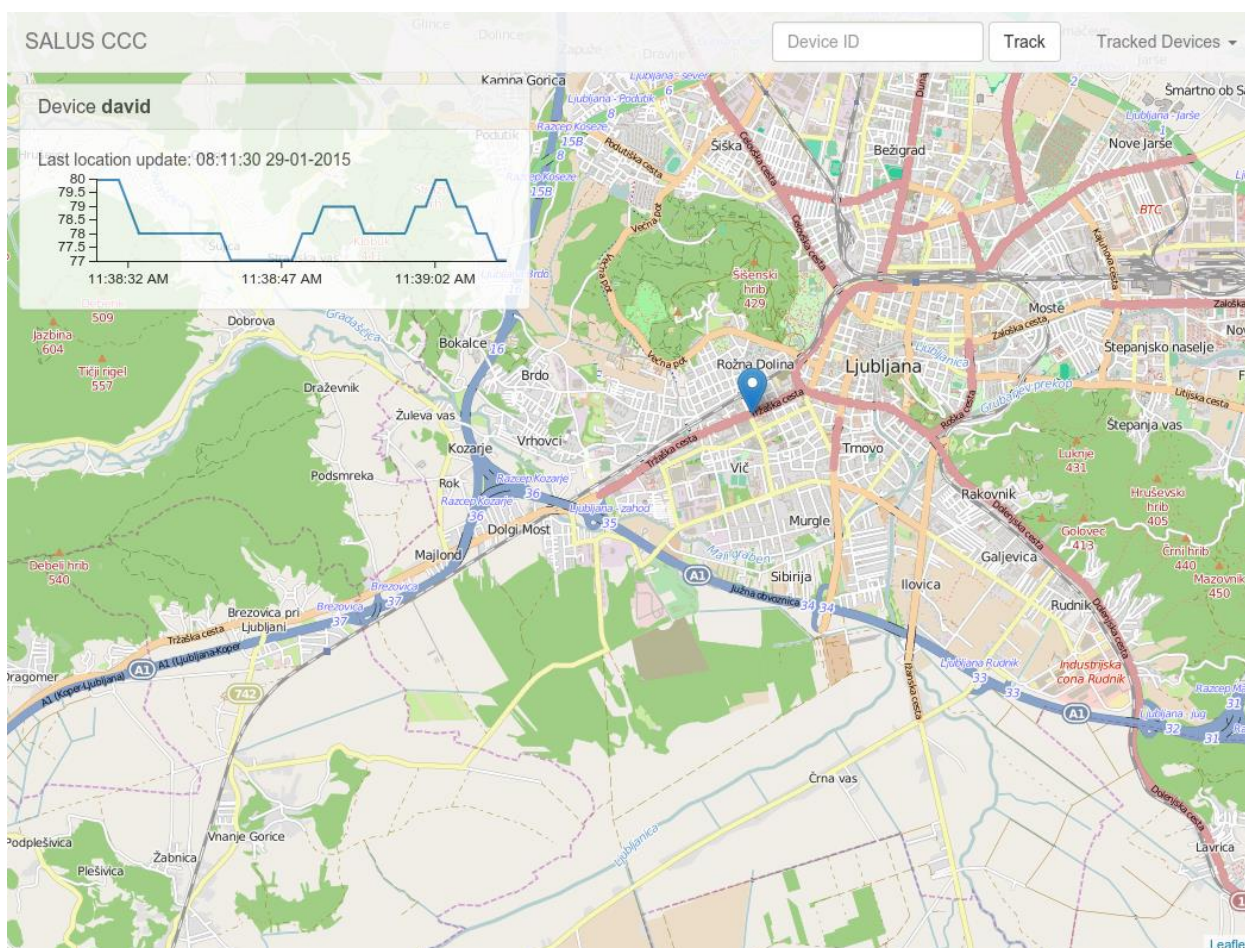


Figure 37 – A screen capture of a back-end application

2.2.5.4 HUCare – Integrated biosensors and video solution

Using medical approved biosensors, ONE aims at obtaining a myriad of body signals in order to interpret current user health state, location and position (e.g., up, lying down, sitting). These sensors are currently being integrated into an embedded system platform whose purpose is to collect and aggregate input from multiple data sources. Attached to this embedded platform there is a bespoke H.264 hardware video acquisition system, a Global Positioning System

(GPS) unit and a communication dongle prepared for 4G and wireless networks. This platform is capable of producing readings at the rate of 1KHz, reportedly more than enough to perform real medical evaluations [50]. Prior to sending out this information, a light and fast analysis is performed locally on the data, searching for critical events (e.g. heart attack, falls of users).

All the information produced by biosensors is collected through the embedded platform and promptly assembled and stored in a centralized system, which will be performing long term storage of the information, as well as a more thorough analysis of that information – e.g. searching for abnormal patterns in the readings employing concepts of Big Data handling, like the use of Map-Reduce paradigms. The centralized system will also be responsible for employing *data sharding* algorithms due to the amount of sensor information that will be stored, performing this separation taking into account, for instance, the age of the stored data.

The Control Centre is a centralized system performing close monitoring of all the biosensor embedded systems (i.e., from different users). All the configured devices will show in the application, where readings will be updated sporadically. There is also the possibility to select a specific user, and perform close evaluation of the bio-signals being received at the time, as illustrated in Figure 38. Through the use of a live video feed, and both ways audio communication, there is the possibility to exchange information using this platform.

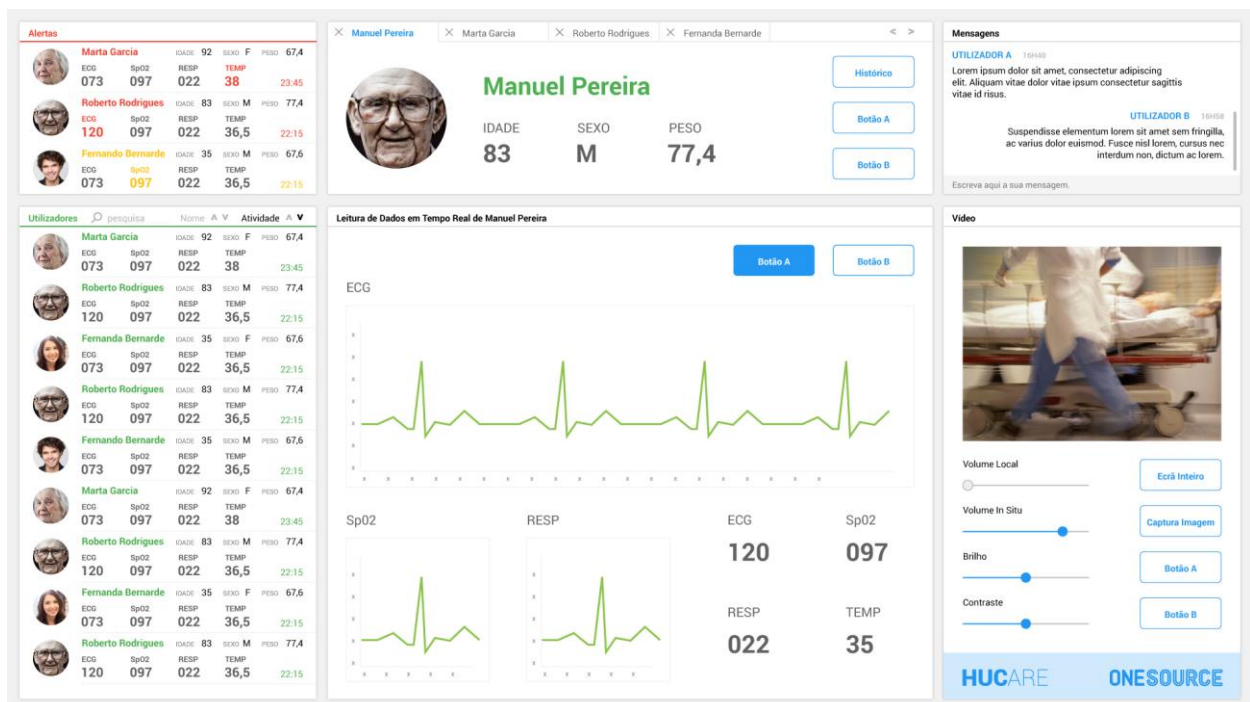


Figure 38 – HUCare Control Centre

All the communications between the intervenient of the platform implement the Diameter protocol. Moreover, all the bio-signals and signalling communication is encrypted using Transport Layer Security (TLS), through the negotiation of self-signed certificates generated at the time of connection. As for the video and audio communication, its encryption scheme is still subject to research, where several UDP encryption standards are being evaluated according to the security mechanisms specified in SALUS, as per D5.2 [36].

In an effort to reduce message serialization/deserialization times, as well as network usage, several studies regarding different proposals for messaging schemes for network communications exchange were performed. The adopted protocol was Protobuf, developed by Google [51]. It revealed to be quite light on the processing requirements side of the biosignals embedded platform, while reducing the overall transmission packet size.

2.2.6 Location Devices

This section describes the location services/devices that are integrated into the PPDR platform.

2.2.6.1 Indoor Location

The indoor location service comprises a set of smart electricity plugs with built-in low-power Wi-Fi access points deployed in the building, as depicted in Figure 39. Each of these devices periodically transmits a Beacon frame with a unique Service Set Identifier (SSID) to identify these Access Points (APs).

Wi-Fi-capable mobile device on the user side has an active application that periodically (once per second/minute/etc., adjustable depending on the node mobility) collects the list of SSIDs currently in communication range and rank them according to the distance towards the AP based on received signal strength/SNR/etc. values. This list is periodically transmitted to the location server application on the server side, for further processing. The server aggregates these lists from a number of mobile devices and processes the data in these lists to estimate the mobile device positioning with respect to other devices with maximum available precision. In addition, the server transmits, in almost real-time, this information in machine-readable format (e.g., XML) to the Operator's terminal, where it could be visualized.

Such systems may be used for indoor positioning, where the accuracy of satellite-based positioning is inefficient.

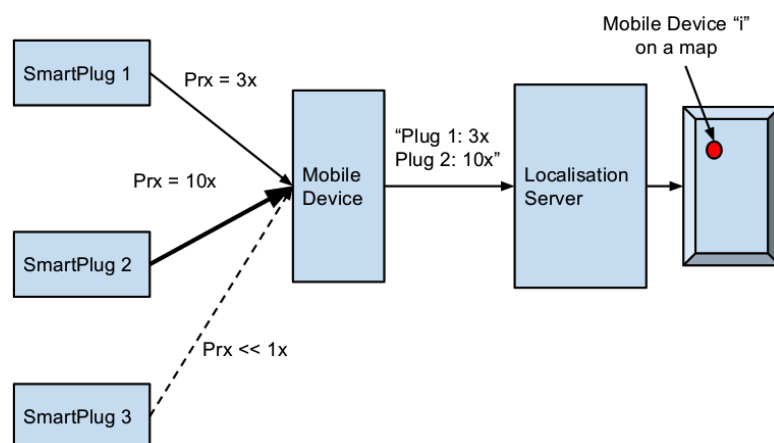


Figure 39 – Indoor Location.

2.2.7 Video System

Video is the key application driving the need for wireless broadband public safety networks. It brings additional tools and information for better and faster decision making while enhancing operational effectiveness:

- By getting mobile live (or recorded) videos coming from the field:
 - The dispatcher in the command and control centre can get a better understanding of the situation (“when an image is worth 1,000 words, a video is worth 10,000!”),
 - The First Responders in the field can focus on their task (especially in a stressful situation) and avoid lengthy voice descriptions of the scene.
- By dispatching live (or recorded) mobile videos to first responders assigned to respond to an incident, first responders get better informed in real-time of the actual situation they are about to face.

Figure 40, below, identifies the architecture of the E2E video system in PPDR.

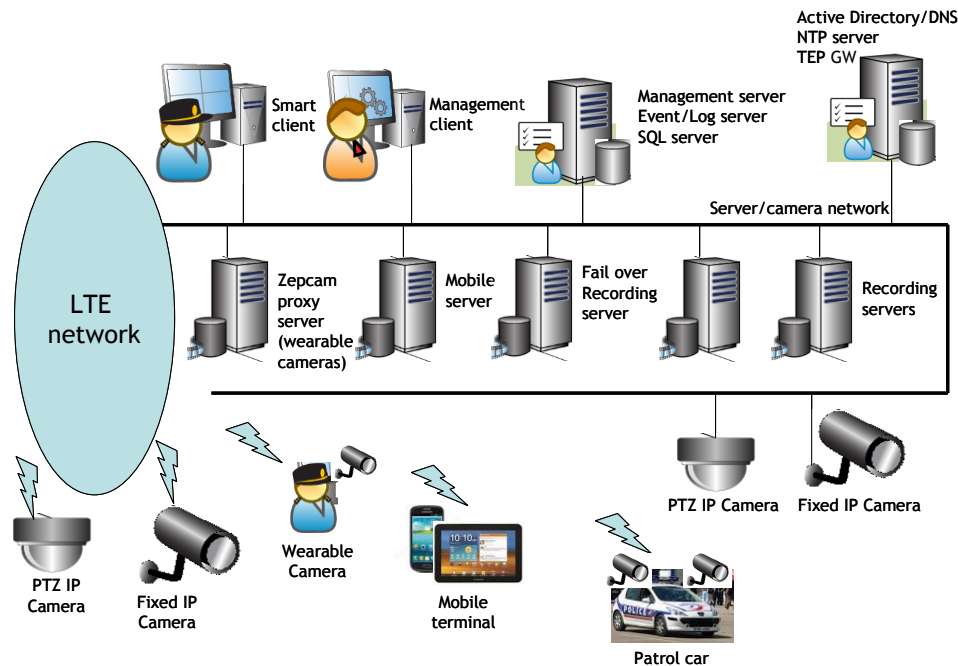


Figure 40 – ALU-I E2E Video solution architecture based on partner components

The end-to-end video system integrated by ALU-I is based on industry proven solutions and consists of:

- Milestone VMS solution (Xprotect Corporate), designed for large multi-site and multi-server installations with centralized management of devices, servers and users. It includes management server and clients, recording servers (with failover), mobile video capabilities and smart clients to provide operators instant access to live cameras and/or recorded videos and the alarm manager²;
- Axis Communications IP cameras (such as the M3114-R for vehicle mounted applications or other models for fixed / temporarily fixed applications),
- In-vehicle video solution built around a rugged PC with video storage capabilities, local in-vehicle VMS – based on XprotectGo – and protocols to transparently connect multiple IP cameras (up to 5) to a central VMS over LTE,

Personal video camera solutions (integrated with the Milestone VMS framework) using the cameras embedded in smartphones/tablets using Milestone Mobile³) that enable the use of the terminal as a Milestone client and the ability to push live video from the cameras⁴.

2.2.8 Air Surveillance

The Air Surveillance System consists of the following three main parts:

- Mobile Ground Control Station (MGCS);
- Sensors;
- Assets carrying the sensors.

The sensors can be deployed stationary or on moving platforms (assets) such as Unmanned Air Vehicles (UAVs), Unmanned Ground Vehicles (UGVs) or aerostats. The system includes means

²Advanced features such as Federation and High Availability of the VMS will be industrialized in the next release of AOFR but are supported by the Milestone product.

³ Milestone Mobile is available on Android and iOS platforms.

⁴ It was also planned to use wearable cameras from Zepcam (Zepcam TI Live) but as of the DR4, these have not been successfully integrated into R1.

to control different kinds of mobile platforms and to direct them to potentially interesting locations especially in areas with no prior sensor equipment. The actual MGCS system is highly mobile and portable. It can be taken to and implemented at any location with relative ease.

The Mobile Ground Control Station (MGCS) is capable of controlling and coordinating a customizable set of sensors and sensor carriers. In addition, there are interfaces to external exploitation stations and control centres. The ground control station is an adaptable prototype system for managing data acquisition with various sensors, mobile Ad-Hoc networks and mobile sensor platforms. The main task of the ground control station is to work as a data integration hub between multiple sensors and a super-ordinated control centre. The MGCS provides a comprehensive situation picture in complex environments. MGCS offers a number of support services related to surveillance and reconnaissance tasks like resource planning, sensor data exploitation and personal training.

For the first prototype a micro UAV is used as sensor platform in combination with a video camera. The camera delivers proprietary video streams over a proprietary wireless protocol and a data link using the spectrum of 433 MHz, 868 MHz, 2,4 GHz or 5,8GHz. The streams will be converted into H.264 video streams by the video server component, as illustrated in Figure 41. This video server also allows the dispatching of the encoded video streams to dedicated receivers by streaming the data to IPv4-based endpoint addresses of the clients. Actually only unicast is supported. This capability will be used for the SALUS first prototype to dispatch video data to the Command and Control Centre. From there these streams can be analysed and further dispatched according to operational needs.

The MGCS provides various additional components for position tracking of platforms, control of sensors and platforms, situation display, among others. These components are summarized under the concept of “GCS Components” in Figure 41 and are attached to an internal communication bus from where they receive and will send their data to. The communication throughout the MGCS runs on top of an Ethernet-based IP network (LAN).

For the first prototype several of these components will be replicated to the CCC like the situation display, UAV Tracking and a Video Display. The communication between the MGCS and other nodes (for example the CCC node) is established via an LTE-based Air Interface, which in turn requires an already established cellular based LTE communication infrastructure. An LTE capable router will route the IP-packets to the dedicated IP endpoints. For receiving these data over the LTE Air interface a corresponding LTE capable router is required. All application communication runs on top of the IPv4 protocol.

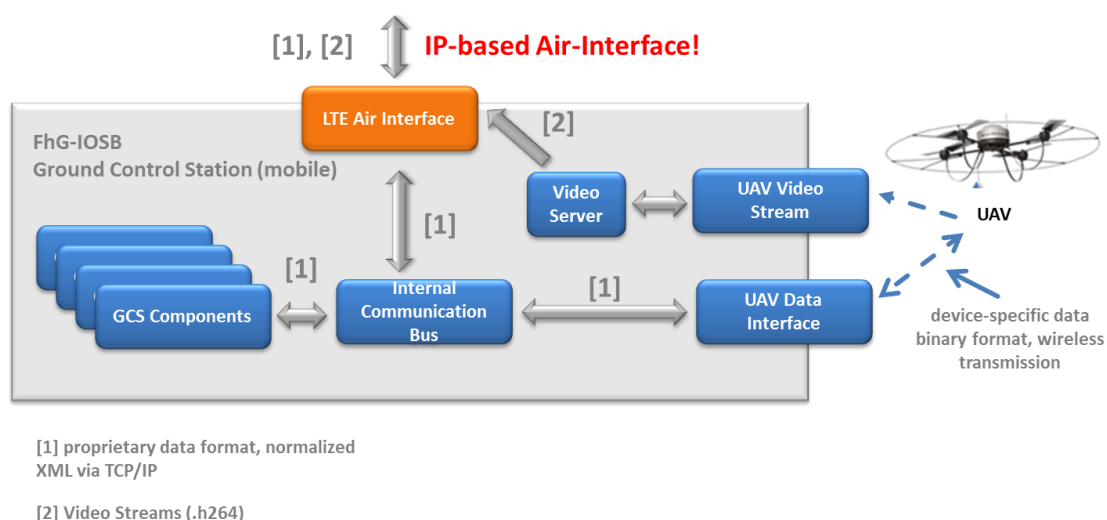


Figure 41 – Components comprising the Air Surveillance System for the first prototype.

2.2.9 Command and Control Centre

Within the first prototype the Command and Control Centre (CCC) represents the operational Decision-making node at the tactical level. It consists of various applications collocated in one location (for example dedicated control room), connected via an IP-based LAN. The principle integration of FhG-IOSB components for the first prototype is shown in Figure 42.

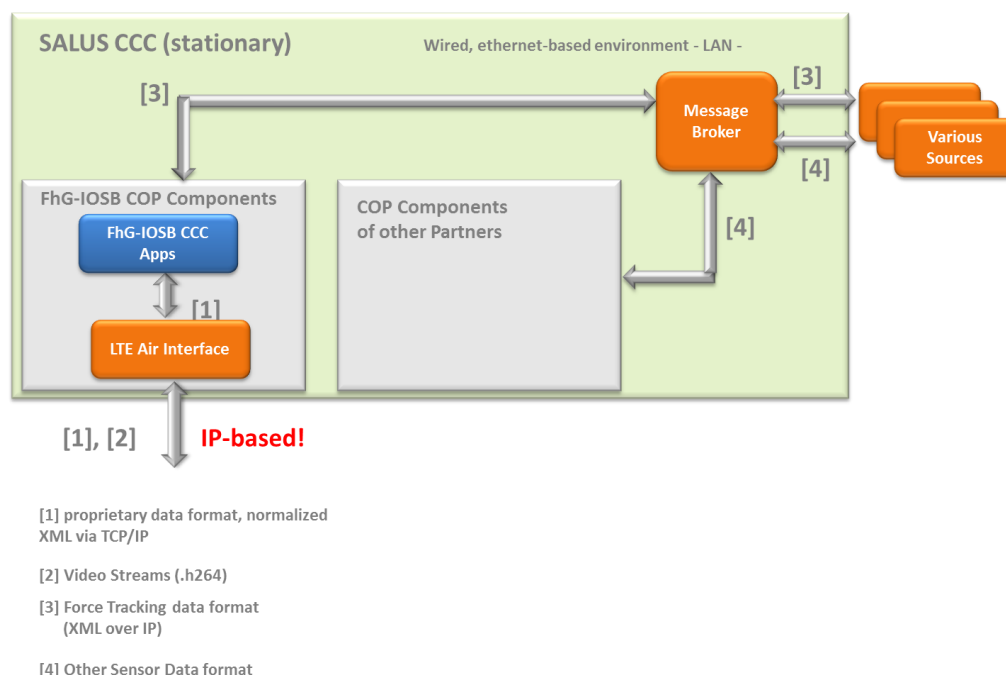


Figure 42 – Principle integration of components comprising to the CCC for the first prototype

Figure 43 provides a more detailed view on the FhG-IOSB applications and their integration strategy for the CCC. As described in the previous chapter the Video Display and the Situation Display for the sensor carriers (drones) will be replicated from the MGCS. Again these applications are connected to the internal communication bus, which is in turn connected to the MCGS internal communication bus via the LTE Air Interface. An additional application – the PPDR Force Tracking application – will be provided and integrated for the final prototype. This application allows the tracking of PPDR-forces (their status on location) in the field in space and time. This functionality relies on a common (XML-based) message format with a SALUS-wide defined syntax and semantics. A preliminary message format is defined, see [3] in Figure 42. This format will be refined according to the results achieved on the first prototype and according to the end-user feedback. In Table 3 the top-level message elements are shown along with their semantics.

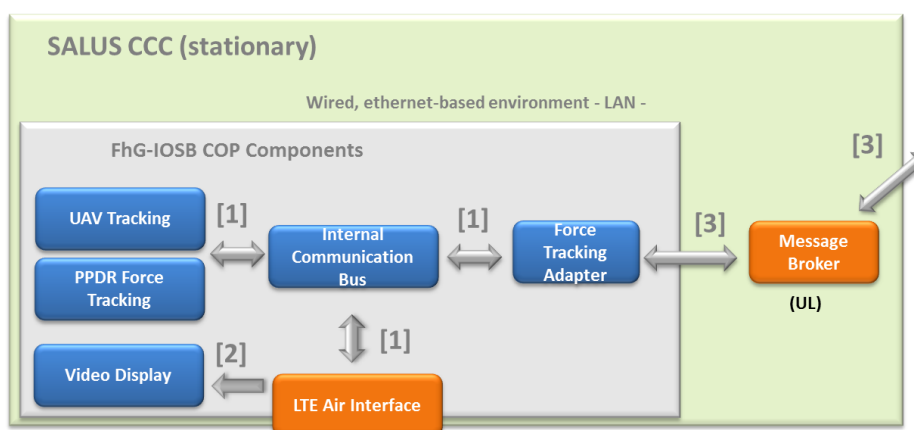


Figure 43 – Detailed view on the FhG-IOSB applications and their integration strategy

Table 3 – SALUS common message format for PPDR force tracking applications

Message Element	Semantics
Heartbeat	If true this Item represents a Heartbeat signal. The default value is "false"; normally, if within a predefined time interval (defined as an operational rule as the Minimum delay in seconds between two consecutive updates) there are no data collected the track message will not be sent at the interval expiration. If the heartbeat is "true" the data provider must always send a message (also if empty) at the interval expiration time (again this requires an operational agreement)
Resource Position	Element contains a set of elements which all together constitute positional data of a resource in space and time. Every element type is defined separately.
Resource Identification	Element contains a set of elements which all together constitute the identification of the resource that is associated with data in the Resource Position element. Usually needed for a standard situation display. Every element type is defined separately.
Resource Operational Status	Element provides information on operational status of the PPDR resource associated with data in Resource Position section. Various pre-defined status definitions are available.

PPDR force tracking messages originate from various sources and will be managed via a dedicated Message Broker instance. For the CCC an instance of the Message Broker is running within the CCC's network. It receives messages from publishers and forwards them to subscribed consumers. The PPDR Force Tracking application acts as a message consumer. In order to reuse pre-existing software capabilities the IOSB PPDR Force Tracking application is integrated into the pre-existing IOSB software infrastructure. This requires a message transformation between the SALUS common message format for PPDR force tracking and the internal message structure used within the IOSB software environment (Internal Communication Bus). This transformation is realised by dedicated transformation software, the Force Tracking Adapter shown in Figure 43. For the Message Broker based integration, the Message Broker API is used by this adapter. Currently only one-way transformation is supported.

ALU-I CCC solution is provided by Rohill (Tetra Node with Chameleon Dispatcher) and is pre-integrated with ALU-I LTE components.

2.2.10 Mobility Management

Mobility management plays an essential role in the routers in the backbone of the SALUS PPDR platform. Proxy Fast Mobile IPv6 (PFMIPv6) will be used to support mobility of devices between IPv6-based networks. PFMIPv6, as used in SALUS provides extensions to standard Proxy Mobile IPv6. The main extensions to be used in SALUS and implemented in this version of the SALUS PPDR Platform are the use of signalling and tunnelling for fast and seamless handover, for both unicast and multicast traffic. The architecture of mobility management in SALUS, the concept of PFMIPv6, and the extensions have been described in SALUS Deliverable 6.1, especially section 5.1 [37]. The following paragraphs provide an overview of the prototype implementation, and describe the implementation of the extensions.

The main components of the prototype network set-up are the Local Mobility Anchor (LMA), and a number of Mobility Access Gateways (MAGs), see Figure 44. The MAG provides to the rest of the network a fixed IPv6 address with which a Mobile Node (MN) is reachable. The MAGs provide the same fixed IPv6 address to the MN as it is moving from one subnetwork to another subnetwork, from Access Point (AP) to AP. Tunnelling between LMA and MAGs is used to deliver packets to the MAG where the MN is connected. Additionally, to provide seamless handovers, tunnels between MAGs are used. For providing seamless handovers for multicast traffic, also MAG-MAG tunnels are used.

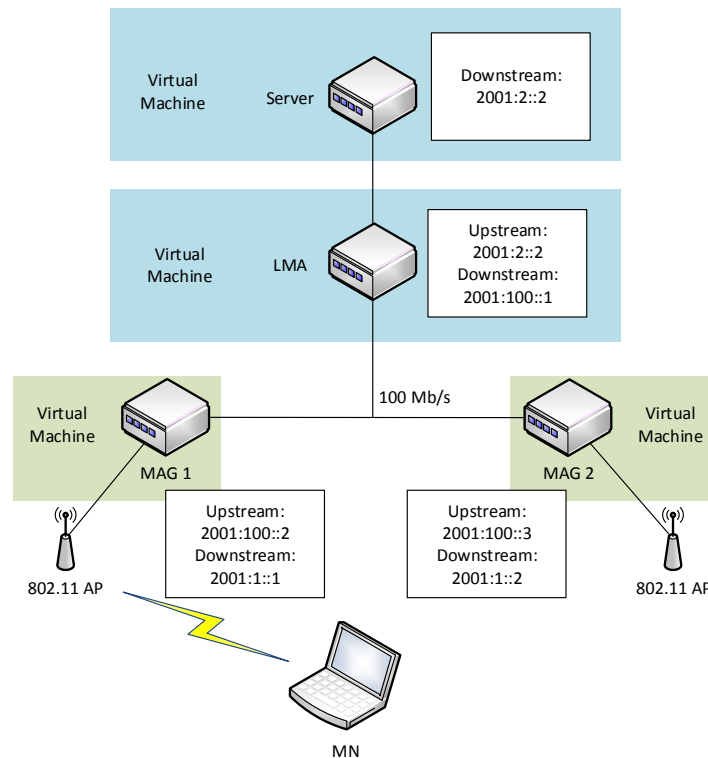


Figure 44 – Network setup for Mobility Management.

The implementation of the multicast for PFMIPv6 system was built on top of an existing MIPv6 implementation called UMIP [31] extended by open-air interface to support PMIPv6 operations [32]. The decision to extend existing software was made to focus on the implementation of fast handovers and multicast support instead of building the entire system from the ground. In addition, the PFMIPv6 system with multicast support runs on top of a standard Linux operating system. The Fast handover functionality was implemented mostly by extending existing C files and methods. The relevant message structures for fast handovers and ways to send, receive and parse them were added to the software. The MAG-to-MAG tunnels were implemented by using the existing methods to set up tunnels between the MAG and LMA. The MAG-to-MAG tunnel is built on demand when a Handover initiation request is received at the New MAG (NMAG) and its acknowledgement is received at the Previous MAG (PMAG). To implement multicast functionality a completely new part was written for the PFMIPv6 software implementing multicast proxy functionality as standardised in RFC 6224 [33]. This allows multicast clients to connect to the PFMIPv6 network and subscribe to multicast streams from their home network. An overview of these extensions can be seen in Figure 45.

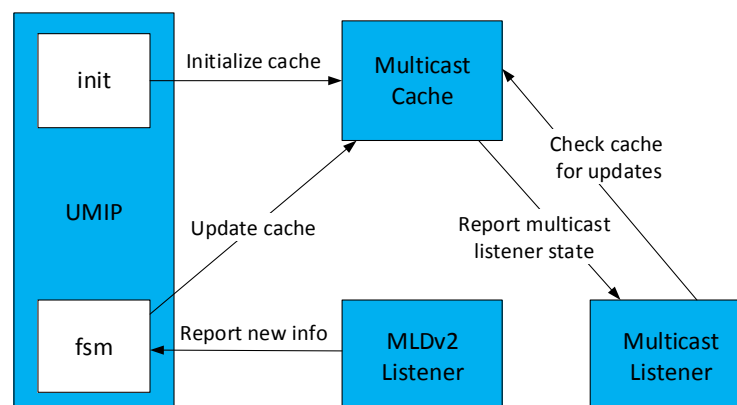


Figure 45 – Multicast Proxy implementation overview

The core of the multicast component is the *Multicast Cache*, which contains information on all the multicast subscriptions a MN (in the case of a MAG) or MAG (in case of a LMA) has on the systems downstream interfaces.

The *MLDv2 listener thread* listens for incoming MLDv2 messages on the systems downstream interfaces. When a group membership request is received this thread will update the Multicast Cache with the new information (adding or removing membership) and forward the request to the correct upstream interface (a tunnel to the MNs LMA in case of the MAG). Multicast functionality is important for SALUS as group communication in IP networks is more efficient using multicast compared to unicast traffic.

To set up the actual multicast forwarding the *Multicast listener* thread is used. This thread listens on all upstream interfaces (physical or tunnel) for incoming multicast traffic. It then checks if there are downstream devices with subscription for the specific multicast packet and updates the multicast forwarding table accordingly. This construction might seem more complicated than needed but is a result of the way multicast forwarding entries must be passed to the Linux kernel. The multicast router will only accept entries as a pair with source and group. This leads to the need of knowing the source of a multicast packet for the kernel to be able to forward it. Therefore, it is required a checking for all incoming multicast packets to see if we need to make a forwarding entry for it.

Efficiency of the implementation is increased by filtering for multicast UDP packets on the kernel level and passing them to the program, afterwards. The first step in the program is to check if we actually have downstream nodes with subscriptions to this group. If this is the case we check if we need to make or alter a forwarding entry for this packet.

2.2.11 Security Services

This section introduces details regarding the components providing security services, such as the Public Key Infrastructure (PKI), Intrusion Detection Systems (IDSes) and the Mobile Forensics (MF).

2.2.11.1 SALUS Public Key Infrastructure

The SALUS Public Key Infrastructure (PKI) was first introduced in the SALUS Deliverable 5.2 [36]. It is a component of the SALUS Authorization and Authentication Centre (AAuC) and provides functions for the management of digital certificates, which are needed to enable secure communications among SALUS communication devices. These devices can be PPDR terminals - such as LTE User Equipment (UE), Wi-Fi Mobile Nodes (MN), updated revisions of TETRA or TETRAPOL Mobile Stations (MS) - or infrastructure communication devices – such as the SALUS CCC Gateway, Wi-Fi Gateway and the FIGO MN.

The SALUS PKI also provides means to revoke digital certificates (e.g., where the associated private key has been compromised) and to download Certificate Revocation Lists (CRL).

The issued digital certificates use the X.509 v3 certificate template, compliant with RFC 3280 [38]. Figure 46, depicts a sample digital certificate issued by the SALUS Certification Authority (CA) for terminal authentication, for a full explanation of the certificate fields, please refer to RFC 3280. Currently, the SALUS CA can issue certificates for Server authentication, Client authentication and IPsec authentication.

Sections 4.1.5 and 4.1.6 further detail the current implementation status of the SALUS PKI and its interface.

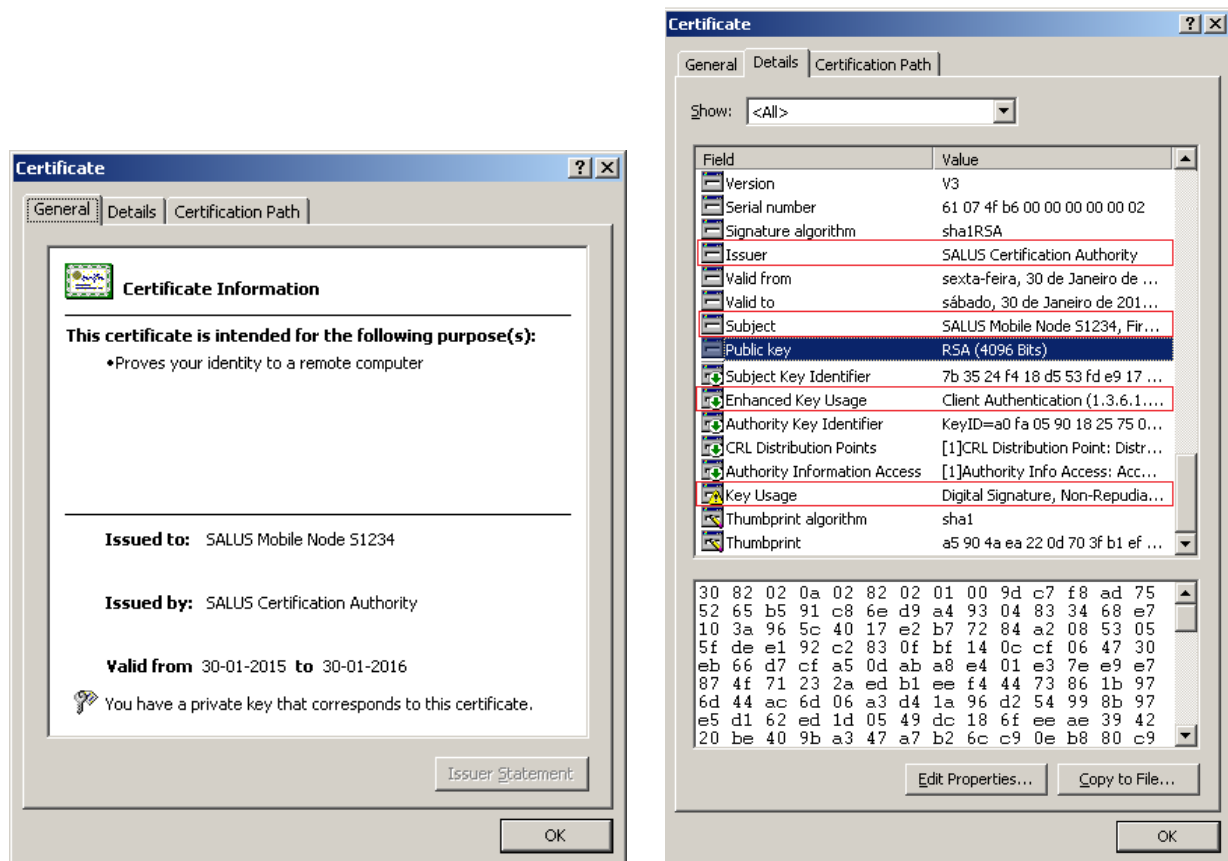


Figure 46 – SALUS Certificate Profile.

2.2.11.2 Intrusion Detection Systems

SALUS Intrusion Detection Systems (IDSes) are deployed in all networks of the SALUS Security Architecture and are responsible for detecting intrusions in critical infrastructure components. The IDS architecture consists of a number of sub-components, as specified in Deliverable D5.2 [36]:

- **Flow exporter** – Exports network traffic flows in a standardized fashion using Cisco's NetFlow [48] or IPFIX [49].
- **Flow collector** – Receives flow data from flow exporters and performs pre-processing and storage, such that the data can be used by analysis applications at a later stage.
- **IDS agent** – The agent performs the actual intrusion detection logic within the network area it resides. When hooked up to a flow monitoring system (consisting of flow exporters and collectors), for example, the agent performs intrusion detection based on the flow data it retrieves from the flow collector. Another example of agent deployment is on mobile (handheld) devices, where they can be used for MF.
- **Global/local Security Manager** – Security Managers coordinate all incident reports received by the various IDS agents in the field. As such, they have a wider overview of the security state of the network than the agents and as such can take preventive measures against intrusions, for example.

In order to be as robust against possible intrusions as possible, several types of intrusion detection are featured within the SALUS Security Architecture (in the form of IDS agents):

- **Host-based intrusion detection** – an host-based IDS analyses log files, file system modifications, processes, network traffic, etc. This type of IDS needs to be deployed on end systems and generally has access to operating system internals.

- **Network-based intrusion detection** – typically performed by devices within the network (so not on end hosts), and analyses network traffic of a large number of hosts. As such, a complete overview of ingress and egress traffic is typically available and links with high throughputs can be monitored.

2.2.11.3 Mobile Device Forensics

Mobile Forensics (MF) is “the science of recovering digital evidence from a mobile device under forensically sound conditions using accepted methods” [1], [2], [4] and has become a routine task for investigators after the increase in use and capabilities of mobile devices.

Evidence acquired during a forensic investigation can serve for various purposes, including auditing and recovery and other fallback solutions. Not only are investigators capable of tracking and identifying entities, with which the device owner interacted, but also to use the retrieved evidence, extracting information about potential malicious activity occurring in the devices after recognizing anomaly patterns or recovering malware signatures.

There are two fundamental approaches on how data acquisition from a mobile device is performed. The first approach is older and it takes place after a target device is seized. Also known as **post-mortem** acquisition, it requires isolation of the device from every networking source to prevent from any intentional or unintentional alteration of its state. The second approach is the **live acquisition** that monitors the device status and provides information on the processes/applications running on the mobile device. However, these methods used in order to retrieve a memory image differ in significant parts, making use of certain attributes of the device, affecting it in a different scale and returning divergent results. MF acquisition methods are classified to Physical, Logical and Manual, with the latter one not being capable of extracting a bitwise memory image:

- **Physical Acquisition** interacts with the physical storage medium of the target device. Since the procedure requires hardware involvement, there is a high risk of irreversible device damage.
- **Logical Acquisition** interacts with files and directories that reside inside a means of logical storage and retrieves a bitwise copy of them. It is mainly associated to providing contextual information such as file system location and date-time stamps, for the formerly mentioned entities.
- **Manual Acquisition** despite the fact of not retrieving a bitwise memory copies, it has its contribution as a “first-line responder” of acquisition, since it corresponds to whichever attribute an individual is capable to acquire through direct interaction with the device.

Post-mortem acquisition is capable of retrieving a wide range of forensically important data. By acquiring a whole image of a system, useful audit data can be extracted, such as device behaviour, modified assets and their associated timestamp. Nevertheless, post-mortem acquisition cannot be applicable for every existing data type.

Once a device is shut down, volatile information is permanently lost. This fact led to the development of the second main forensic method approach, the one of **live acquisition**. Especially in mobile devices, where, due to storage limitations, data such as browsing history and instant message exchange threads is possible to be stored in the volatile memory, thus not being accessible for a post-mortem investigation [4]. A representative example is Volatility [5], a tool performing memory analysis and contains add-ons that scavenge twitter posts and Facebook related data. Live acquisition is also capable of providing contribution in protection of mobile devices. Taking into consideration that live acquisition methods focus on monitoring of the devices state, information such as running processes and changes associated to them, can serve as variables to anomaly detection techniques.

In Table 4, some of the most significant research works on MF, as malicious activity identification, are presented. Since the specific discipline is relatively new, there are still a small number of them.

Table 4 – Known Approaches in Live MF

Reference	Platform	Description
[4]	Android	Access to the <i>procfs</i> virtual file system by the running processes directory; (<i>/proc</i>) and performance of a dump on processes representing messaging applications.
[5]	Android	Live forensic tool with learning capabilities. Identification is based on applications permissions and traces
[6]	N/A	High-level framework of forensic acquisition with IDS and cloud contribution with an intermediate proxy.
[7]	N/A	Implementation of a fake GSM/GPRS network that filters the entire network traffic from and to a potentially compromised mobile device and examines the traffic records.

Post-mortem and live acquisition modules are incorporated in the SALUS MF architecture, which can be found in the SALUS Deliverable D5.2 [36]. A summary of its functionality can be found in the following paragraphs, so as to provide better co-ordination between the components.

The SALUS MF instance consists of two main modules. The mobile MF module, functioning on LTE mobile terminals and the aggregated module, which is hosted at the security manager of the SALUS Intrusion Detection System (IDS). In each section there exists a dedicated table where the functionality of each subsystem component is explained and technical details are presented. The functionalities of the SALUS MF instance are summarized in Table 5.

Table 5 – Components and Functionality of the Mobile MF instance

Component	Functionality	Tools Used
Post-mortem Acquisition Unit	- Activation upon compromising of non-volatile data assets - Interaction with the device file system, partial or full memory acquisition	AFLogical acquisition suite [27]
Live Acquisition Unit	- Activation upon compromising of volatile data assets - Partial or full memory capture in Android LTE devices	LiME – Linux Memory Extractor [24] and Volatility framework [25]
Local Database Instance	- Container of smaller instances of policies and security countermeasures - Container of malware signatures, upon request	SQLite [26]
Data Traffic Distribution Manager	- Activation and deactivation of the Acquisition and detection modules - Parsing of acquired data and distribution to the appropriate detection unit - Optional tools for data encryption/decryption	Python scripts [28] REST API
Signature-based Module	Testing retrieved data for malware signatures	Custom Android
Behavior-based Module	- Recording of training phases in a non-contaminated environment with custom [36] or generated memory images - Search for malicious activity patterns in the system after an attack	Custom Android

Mobile MF Module

The MF module is an application for mobile devices running the Android OS. It consists of an application interface for user control and background services for the post-mortem and Live acquisition Units. Moreover, local, private SQLite database containing instances of the policies and countermeasures found in the aggregated version. Upon receiving an alert from the Host Intrusion Detection System (HIDS) and the correlation engine, the Data Traffic Distribution Manager triggers the operation of either the post-mortem or the live acquisition Unit. If the compromised asset concerns volatile data, the Live Acquisition Unit is triggered; otherwise, the post-mortem Acquisition Unit starts operating.

Considering that physical acquisition demands interaction with hardware and usually sets the mobile device in a non-operational state, option that is excluded because of the need for continuous service provision, logical acquisition will be conducted. Logical is considered as a more appropriate candidate because it does not require a system reboot (cause for volatile data loss) and is also flexible in acquiring either specific parts of the file system or the whole copy of it if needed. The post-mortem Acquisition Unit is responsible for the procedure mentioned above. On the other hand, the Logical Acquisition Unit accesses components such as process lists and log files and retrieves evidence associated to the compromised asset.

Aggregated MF Module

In the aggregated instance of the SALUS MF module, the post-mortem and Live Acquisition Managers can be found, to perform several functionalities, as summarized in Table 6. Those entities, after receiving input about reoccurring attack patterns, are responsible for creating and updating rule sets about which acquisition corresponds to the matching incident. Meanwhile, the Update Manager is responsible for creating insertions of the newly discovered threat activity in the Central and Local Databases.

Table 6 – Components and Functionality of the Aggregated MF instance

Component	Functionality	Tools Used
Post-mortem Acquisition Manager	Activation upon compromising of non-volatile data assets Interaction with the device file system, partial or full memory acquisition	Custom Java code
Live Acquisition Manager	Activation upon compromising of volatile data assets Partial or full memory capture in Android LTE devices	Custom Java code
Database	Container of smaller instances of policies and security countermeasures Container of malware signatures, upon request Contains authorized user credentials	MySQL Server [29]
Update Manager	Activation and deactivation of the Acquisition and Parsing of acquired data and distribution to the appropriate detection unit	RESTful services for requests in XML format
Integrity Control Unit	Receives data from the mobile instances and creates an integrity calculation formula, according to inputs deriving from the IDS, the correlator and system and network logs. Performs additional hashing on retrieved data Notifies the Update Manager and the SALUS IDS	Custom Java code with REST API support
Identification Module	Posts verified acquisition results to the database	Custom Java code with REST API support

2.2.12 SALUS IP Communications Server

The SALUS IP Communications server enables voice and video communication over IP networks. Table 7 describes its features.

Table 7 – SALUS IP Communication Server features

Feature	Description
Point-to-point voice or video call	Allows IP enabled PPDR terminals to be configured with extension numbers and to establish voice or video calls just by dialling the extension number. Alternatively, calls can also be completed by using aliases such as the name of the user to which the terminal is associated – this is a configurable feature.
Group voice call	Allows IP enabled PPDR terminals to establish a group voice call by dialling predefined numbers. For instance, a PPDR terminal may participate in a specific group call just by dialling that group call number. Additionally, group calls may be protected by a pin number, for security reasons.
Multiple codec support	To cope with dynamic network conditions the SALUS IP Communications Server is configured to support multiple codecs (different bitrates). The following codecs are supported: G.711 (64 Kbps) G.726 (16, 24, 32, or 40 Kbps), G.729A (8 Kbps), GSM (13 Kbps), iLBC (13.3, 15.2 Kbps), Speex (between 2.15 and 22.4 Kbps) and G.722 (64 Kbps).
Outside connectivity	It enables IP enabled terminals to establish voice calls with external networks, such as the Public Switch Telephone Network (PSTN) or the Integrated Services Digital Network (ISDN). This feature is not yet fully functional for the SALUS first system prototype.
Automatic Call Distribution (ACD) Queues	Enables the SALUS IP Communication Server to provide a dispatcher similar service. It can queue up incoming calls from PPDR users and deliver them to a single or multiple dispatchers. When a dispatcher becomes available, the highest-ranked caller (or the first one, depending on server configuration) in the queue is delivered to that dispatcher, and everyone else in the queue moves up a position.
Automated attendant	Makes available a voice menu system to automatically answer incoming calls. It allows callers to choose amongst multiple options such as: transfer to extension, transfer to voicemail, transfer to a queue, play message(s), connect to a submenu or connect to a dispatcher. This functionality, in conjunction with the ACD Queues (previous row) and the IVR services (next row) can also be used by SALUS to demonstrate how a user can report an emergency and be redirected to the appropriate emergency service.
Interactive Voice Response (IVR) services	Similar, but not equal to the automated attendant, it allows to take input from a caller, perform an action based on that input and return a result to the caller. The IVR services implies a more complex mechanism than the automated attendant, since the caller needs to understand what the IVR expects from him, a method of receiving input from the caller, logic to verify that the caller's response is a valid input, logic to determine what the next step of the IVR should be, and a storage mechanism for the responses.

Sections 4.1.5 and 4.2.6 further details the current implementation status of the SALUS PKI and its interface.

3 INTEGRATION PLAN

This section describes the different steps of integration plan phases, giving an overview and description of the integration methodology, followed in the next subsections with the description the SALUS specific integration roadmap, integration tools and integration methodology.

3.1 Integration Roadmap

In this section a high-level view of the integration methodology and roadmap is presented. The roadmap addresses the complete integration of the different functional elements of the SALUS PPDR platform as described in Section 2.

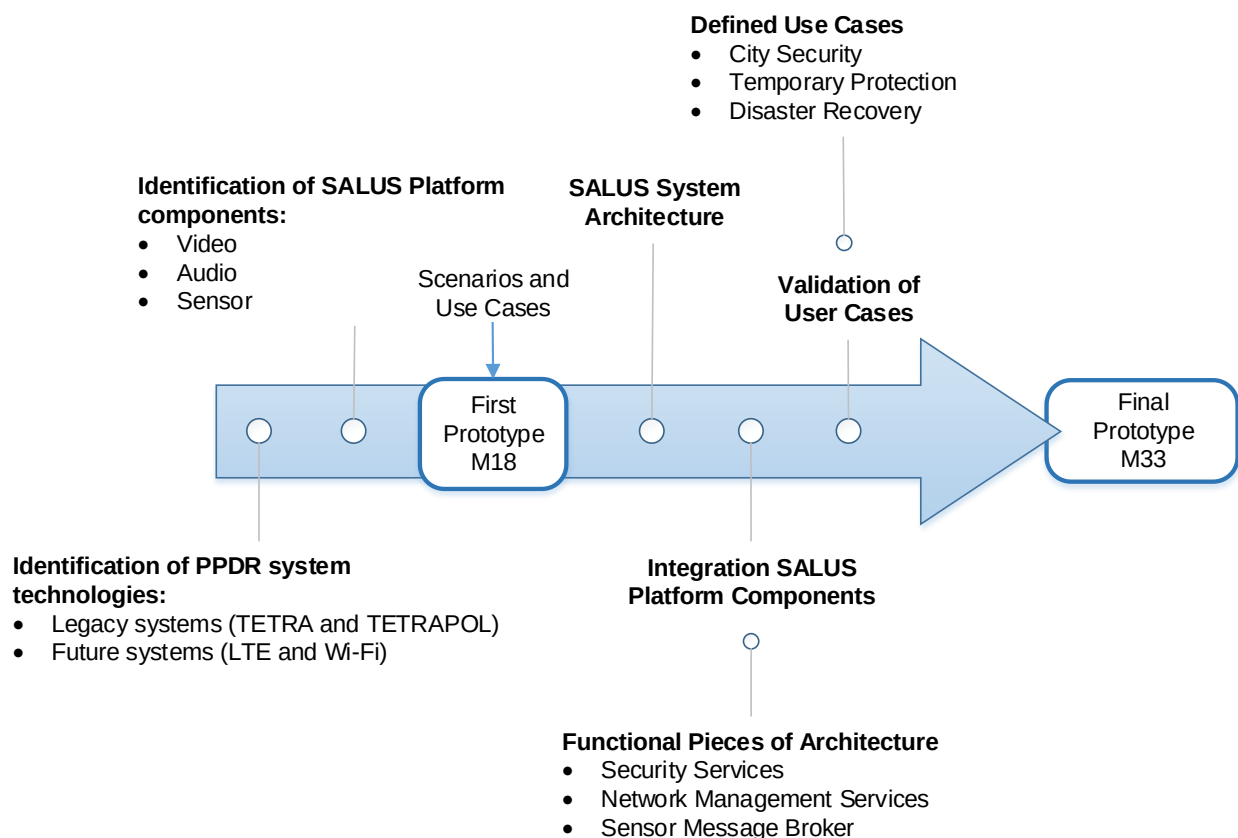


Figure 47 – Integration Roadmap

Figure 47, there are several high level tasks included in the roadmap according to the three phases committed in the SALUS Description of Work. Note that for simplicity, the figure does not depict the unit and integration testing which is planned at each of the integration phases.

3.2 Integration Methodology

This subsection describes in detail the potential integration tools, focusing in the control version software, bug tracking and collaborative tools, and the specific integration work flow, that are used to perform integration activities.

3.2.1 Integration Tools

At the present date, the development teams no longer work in isolation but collaboratively. Typically, this means that information is distributed over several tools, which can be dislocated in different parts of the group all over the world. Additionally, project managers require

knowledge about the project state and notifications about the development process. Integration tools were proposed to address this challenge.

3.2.1.1 Control Version Software

Up to now, no universally control version software exists for application development. To select the most suitable software tool, it is essential to define the needs and goals for the development, as well as, correctly identify all the limitations. The tool selection may depend on different factors such as: application size, number of team members, the level of detail, visual representation, etc. We focus on tools for version control that are well-known, and they are differentiated by operating system environments:

- *Concurrent Version System (CVS)* [9] – is a client-server version control system for UNIX and Windows operation system. It provides source and document changes recording and flexible modules database.
- *Subversion* [10] – is a centralized version control system, which is part of the Apache Software Foundation, and it provides reliable, simple, and safe environment.
- *Git* [11] – is a distributed version control tool available for multiple platforms (Linux, Windows, OS X, and Solaris). It provides cheap local branching, convenient staging areas, and multiple workflows.
- *Mercurial* [12] – is a distributed version control tool, very similar to Git. Was original developed to compete with Git, however it seems less successful.

Currently, Subversion is one of the most used server based control version software. It has all the features of the CVS and improves, in addition to a wide range of usage and software maturity. CVS is an older version control system and it is rarely used today. For the faster distributed solutions, Git has a clear advantage over similarly tools. Once it is learned, it increases the speed of production and leads to better branch management. In addition, there are proprietary and open source tools for version control that can be used in the diverse operating systems. Git is the chosen tool for control version due to its integration with the bug tracking software, namely the redmine tool [18].

3.2.1.2 Bug Tracking Software

A software-testing tool allows the discovery of bugs in minimal time and effort and is used in the testing phase of the software development life cycle. Moreover such type of tools allow to complete certain tasks automatically, improve efficiency of testing, and discern issues that would be hard to locate using manual review. This section focuses on common bug tracking tools. The primary goal is to analyse the features supported by these testing tools, which help in reducing coding resources and increasing efficiency.

- *MantisBT* [13] – is an open source free bug tracking system, web-based. Apart from being a bug tracking system, it can also manage documents shared among users and integrate scheduling system, in order to organize work allocation and assignment for the workforce.
- *Trac* [14] – is a free project management and bug tracking system, web-based. It has an advanced wiki which brings reporting features for software projects, allowing easy bug tracking, and a timeline showing the overall project events (current and past).
- *Jira* [15] – is commercial software that is licensed for the number of users and is very commonly used for bug tracking and project management. It has a web interface that makes it easy to deal with and report errors and bugs.

Mantis Bug Tracker and Trac, despite providing fewer features than Jira, offer mostly what is required in common projects and are the two most chosen tools. This is essential because they are provided as open source tools, which helps to reduce some of the cost. Those who use Jira focus primarily on management features, and for that reason Jira has many features, which

are not really useful for bug tracking system. The chosen bug tracking tool relies on the redmine tool [18] that besides including support for git, also includes bug-tracking features.

3.2.1.3 Collaboration Software

Collaboration software leads to the share of ideas and information among co-workers and across an organization. It's easy to understand that collaboration software can help connect individuals and information when it's needed most. Collaboration software enables communication, but there are many other benefits to consider: improved project management, better workflow, better management of invoicing and knowing that employees and partners have immediate access to critical business information and documents when they need it. There are five types of collaborative software:

- *Project collaboration tools* – Team members are engaged on a daily basis through shared workspaces and activities. Individuals are responsible for sharing content, performing tasks with dependencies on each other and collectively will be rewarded for positive outcomes of their projects. In selecting project collaboration tools, it must consider those that can serve an enterprise, and implement by department, group, and project basis. One particular project collaboration tool, Podio [16] provides ready-built and customizable apps that teams can immediately take advantage to support uses as varied as marketing projects, human resource recruiting, and partner-supplier relationships.
- *Social software platforms* – are primarily designed for building an open network culture. In a true enterprise social network, people are connected to freely access resources within the organization. Resources for subject matter experts and other business critical functions are not only discoverable but personalized recommendations through profiles, search mechanisms, and news feeds and blogs with linked commenting systems.
- *Innovation management tools* – provide the impetus for crowdsourcing ideas and collaborative exchange of expertise for co-creating products and services. The innovation processes are managed through presentation of challenges that create business value within an organization.
- *Web Conferencing Tools* – Web conferencing tools are integral social technologies for communication and collaboration across enterprise social networks near or far. Regional offices in geographically dispersed locations are bridged easily with voice and video, as well as cloud based file storage for use in conducting presentations GM [17] is one particular example among others offering enterprise-class collaboration tools.
- *Wiki Tools* – wiki tools are practical for building knowledge bases and maintaining documents. Wikis may help to manage decision support systems to update and distribute service and support to a global customer base. They are typically integrated in social technology toolsets, including team workspaces that will show business benefits for on-going collaborative work activities.

The collaboration software chosen in SALUS considers the redmine tool [18] with support for multiple projects, per project forums and wiki. It also includes support for bug tracking system and file management, as mentioned earlier. Redmine is also an open source tool.

3.2.2 Integration Work Flow

The goal of the integration workflow is to describe the specific actions, which have to be executed in order to integrate two different software components. The proposed workflow is illustrated in Figure 48. For simplicity, the integration team feedback that is expected to happen in all the integration phases is not depicted.

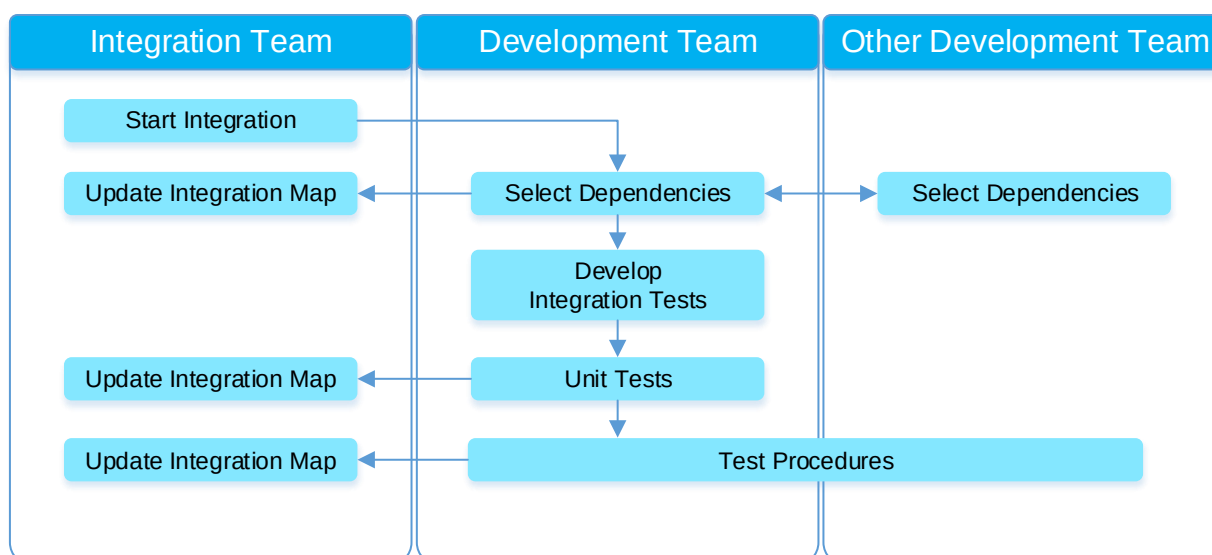


Figure 48 – Integration Workflow and Responsibilities

The integration workflow presumes the collaboration between three distinct teams, which are not monolithic and that should interact accordingly to the previous figure:

- *Integration team* – the integration team is the one responsible for controlling the integration workflow, maintaining the integration map and deciding when the different stages of the integration process start. Additionally the integration team will ensure the underlying testbed for realizing the integration will be available. The integration team is composed of integration task members and by testbed administrators;
- *The development team* – the team bringing a specific component to the integration. The team is in charge of adapting their own components to be integrated with the other components according to the common agreed interface. Additionally, the team is in charge of the internal functionality of the components, as well as, of the development of the specific Integration Tests proving that the component is robustly build and it does what it is claimed. The team is composed of selected developers of the components to be integrated;
- *Other development teams* – the other development teams are bringing their own components. The other development teams are in charge of adapting their own components according to the common agreed interfaces. The other development teams are composed of selected developers of the components to be integrated. There is no difference between the implementation teams except the trust in the component which is under integration.

After the iterative execution of the workflow described in Figure 48 for components interacting with each other the result of the workflow will be an integrated working prototype. The results are specified in an integration report that includes the output of the integration activities and issues found in the integration process. This integration report can be based on the model presented in Annex A – Integration report. The two development teams, whose components need to be integrated, have to develop test cases and sanity checks for their own components and have to do the testing procedures together.

The process can start when components have core functionalities developed and are at stable status. The timing of the different procedure steps depends almost completely on the maturity of the different components. With the first developments the core functionalities are provided for the majority of components. After that, the specific dependencies, the specific integration

interfaces and the Unit Tests can be realized, depending on the needs for providing meaningful partial prototypes and their demonstration. The integration process includes the following steps.

3.2.2.1 Step 1: Dependencies Selection/Prototype Interface Definition

The first step of integration corresponds to the selection of dependencies, which are selected in agreement between the two development teams, involved into the two sides of the dependency interface and the integration team. A dependency or an interface is characterized by:

- Transport level protocol – the protocol carrier for the specific interface (e.g., Internet Protocol (IP) for Generic Routing Encapsulation (GRE) interfaces, User Datagram Protocol (UDP), Transport Control Protocol (TCP).
- Protocols, such as GRE, HyperText Transfer Protocol (HTTP).
- Communication Parameters – the format of the transported information (e.g. Attribute-Value Pair (AVP) for Diameter interfaces, etc.)
- Parameters Value Ranges – which ranges are covered by the transported information (e.g. character strings of max 255 characters, etc.)
- Data flows and procedures with initial state, procedure output and effect on the internal components state.

A common list of dependencies is created in agreement by the different development teams. The integration team will have to bring these dependencies to the integration map and update it.

3.2.2.2 Step 2: Integration Tests Definition

The development team has to develop the test cases that will have to be executed, against the components that are being integrated. The test cases developed by each team for its own component are designed to test the component itself and are performed in the unit tests.

The integration test cases should settle the dependencies parameters based on the dependency characterization parameters, defined in the previous section, in a clear and consistent manner.

The results are meant to decide if the component is functioning correctly on the tested interface and that the two elements that have to be integrated are fit to be used together. These tests will also be useful afterwards, in case of internal modifications that can be made to any of the components, in order to determine if the overall functionality has been altered or not.

3.2.2.3 Step 3: Tests

The integration tests cover the communication between the two elements that have to be integrated and are applied to the interface between them. However before running them, another step is necessary to ensure that these tests can be carried out in a controlled way and are not extensively error prone due to the missing functionality on the integrated components side. For this reason, unit tests have to be designed and implemented in the form of test clients, by the development team. The tests will be executed together with the integration team, ensuring that the promised features are delivered with the quality required for the integration.

There is no limitation in number of Unit Tests per component or size of a Unit Test. However, it is expected that the tests will cover all the significant functionality and the functionalities of the specific component, through this ensuring that the quality level required for an integration phase is reached. Each development team has to provide a set of Unit Tests for their own component, in order to verify that the functionality is in accordance with the requirements. They do not require a detailed verification of the functionality, just to check in a fast way that this functionality works, herewith not affecting the functionality of other software.

To perform a successful Unit Test, a test program has to be implemented based on the specification of the use cases. A Unit Test must describe the complete behaviour of the component and is based on the described scenarios. The robustness and the functional evaluation roles can be concluded in the following two lines:

- What is not available in the unit test program is considered not implemented in SALUS (Unit Tests related to background software may be missing).
- What cannot be automatically installed, configured and tested multiple times in a row by using the test program is not ready for integration. By automatic installation, it is considered that there are a limited number of scripts to be called for the overall system and one for each component.

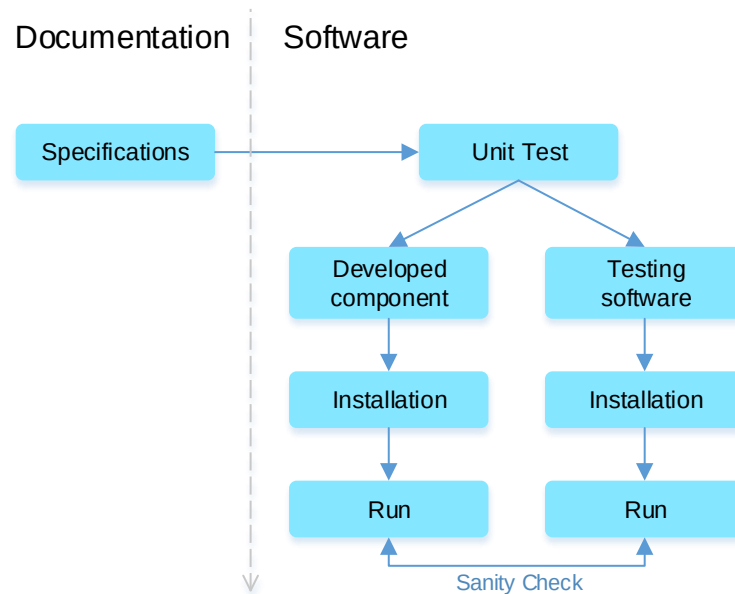


Figure 49 – Unit Base Testing

The development of a Unit Base Testing is illustrated in Figure 49. Starting from the specification, a set of unit tests is defined. Having the unit test description, two developments will succeed in parallel, for the component itself and for the test program. Both programs have to be installed and run into the same environment. Running the test program will result in a sanity check of the developed component. The integration task makes no assumption on the communication between the component and the test program.

For a sanity check procedure to be considered completed the several items have to be fulfilled, as detailed in Table 8.

Table 8 – Sanity Check Steps

Step	Description	Observations
1	Unit Tests are available	A Unit Test describes a portion of behaviour of the component; Based on the architecture and scenarios; Functionalities tested in the Unit tests should be detailed;
2	Test program / Stub Components are available	Basic principle: What is not available in the test program is not implemented; A suite of test programs/stub components should be able to generate the pre-conditions;
3	Software can be installed and run on local testbeds	Software can be installed and run on local testbeds or on an integration testbed for the prototypes.
4	Running the sanity checks	Running the test program resulting in success for multiple times; A 1-page report is generated with the running test program.

As depicted in Figure 49, starting from the unit test description two software entities have to be developed: one is the component that has to run the functionality needed to support the use case and the second is the test program or stub interface. The last one is able to simulate the conditions in which the first one will perform. Together these two components can perform a sanity check operation. For example, a component, which has to be integrated, provides an HTTP interface, so for sanity checking, a stub HTTP client has to be developed. Running them together and establishing a successful HTTP communication, between them, means that the sanity check was passed for the HTTP interface.

Another example would be a component that uses Diameter interface and has to perform a sanity check. The procedure would involve another Diameter stub, which provides the state machine, the protocol stack and an Application Programming Interface (API) accessible via commands. The Diameter stub which provides the remote end of the Diameter interface will be integrated within the other component. In this way, for interfaces using protocols with complex state machines and encoding/decoding mechanisms, both ends of the interface are developed by a single development team, thus not requiring two different implementations (one for each end of the interface) and interoperability between these implementations.

3.2.2.4 Step 4: Integration Testing Procedure

In order that two software components fulfil their purpose in an effective and relevant way, several conditions must be met:

- They must both be able to be installed and then ran on an integration or local testbed, that means the ability to use the same infrastructure as the other components;
- They must both provide an automatic installation process with as few user interaction as possible;
- The human participation in the installation is not exceeding 20 minutes (time spent except the running of the automatic scripts). It is recognized that multiple minor upgrades of the software will happen during the integration phase. This limitation enables the reduction of the time required for passing from an update to a next test.

If these conditions are met the sanity check must be performed several times and a 1-page report must be generated, showing the successful outcome of the tests. In the case of failure, the team must get back to the developing stage and improve the software. After making sure that the sanity checks are passed, the integration team has to update the integration map to include that the specific component passed the checks and that can be considered robust enough for an integration test. At this point half the integration workflow is planned and what are still missing are the test procedures.

3.2.2.4.1 Preconditions

Prior to the testing stage there is a set of conditions that have to be fulfilled in order to proceed to the testing design and activity, as summarized in Table 9.

The first step refers to the documentation and represents a list of functions that are involved in the test. This list must be accompanied by the sanity check report for each of them. In this way, the development team presents what has to be integrated, along with the sanity checks that guarantee the functionality. With the documentation part being covered, then the software part needs to be addressed. Here the development team has to provide the functions involved in the test along with the test programs, or stub interfaces used for sanity checking, and has to specify the input data which will be used during the integration test. At this point the prerequisites phase has ended and the design of the testing procedures has to be done.

Table 9 – Items to be delivered before a test procedure begins

Step	Description	Observations
1	A formalization document of the testing	Components involved in the test; Sanity check test reports of the functions involved;
2	Software	The components involved; The test programs/stubs for the functions involved; Input data for this specific integration test; Automatic mechanism for generating initial conditions;
3	Testbed	A testbed on which the components can be installed;
4	People	The development teams for the components that are integrated; At least one member of the testbed deployment and of the integration team to address specific items appearing during the integration.

3.2.2.4.2 Test Conditions

Before a test case can be started, the following items, presented in Table 10, have to be prepared.

Table 10 – Items needed for performing an Integration Test

Step	Description	Observations
1	Design items	Designing the integration test case; Designing a unified unit test; Designing input test data (if it was not made at the unit test);
2	Environmental Conditions	Setting up the appropriate system; Applying to the environment the functions involved, the unified unit test and input test data; Realizing the test pre-conditions;
3	Expected output	Expected output of the test case.

The development teams are responsible for the design of the integration test case. This has to contain the purpose of the test, what are the expected results, the procedures that are involved and the input test data, as well as, the initial conditions and testbed reproducible conditions. If no input data is available from the unit-testing step then it needs to be specified.

Another step is preparing the environment, namely installing the required OS, installing all the prerequisites, setting up the interfaces, etc. Using this environment, the components have to be deployed along with the unified unit test and the input test data. If there are any pre-conditions that have to be met, like getting the component into a certain state then this will have to be performed.

At this point all the required elements have been gathered and the integration test case can be performed.

3.2.2.4.3 Test Output

After the tests are concluded an integration test report has to be produced by the integration team. It will contain the integration test output data and a problem report if necessary. A simple model that can be followed is presented in Annex A – Integration report, considering the steps in Table 11.

The test results have to be documented and the result has to be identified if the test ended as expected (i.e., if they passed or failed). The report has to mention if all the aspects of the functionality of the component (input, output, monitoring) have passed the test. Test logs may be voluminous and have to be compressed to have their contents prepared for a quick overview and reference.

Table 11 – Integration Test Output

Step	Description	Observations
1	Output	Matching the expected output with the output;
2	Integration test report	A very short report on the executed procedures and their success;
3	Problem report (if necessary)	A documented report on the problems discovered including as detailed as possible the recognized missing functionality, the bug or the misbehaviour due to un-alignment between the components as well as possible reasons why problems appeared.

The report has to be analysed by the integration team and in case of the integration testing has not been concluded successfully then appropriate feedback must be provided to the development teams, in order to correct the misbehaving of the components. If the procedure was successful the integration map will be further updated.

4 FIRST INTEGRATION OF PROTOTYPE

This section describes the integration status of the first prototype, the integration plan and the roadmap towards the final PPDR prototype. Interfaces between the different components are also highlighted.

4.1 Integration status

This subsection details the integration status of the SALUS components in the first prototype of the PPDR platform.

4.1.1 PMR-over-LTE ecosystem

The first iteration for the SALUS application platform with CVDP relay and gateway, as well as the first iteration of the TeamLink app on Android, are available for integration.

The CVDP relay application includes the subscriber database for authentication of devices, as well as mobility services for roaming and handover of calls between different wireless networks, including private and public LTE networks and Wi-Fi networks.

The CVDP relay currently supports the following functionality:

- Registration and de-registration;
- Group attachments, including group selection and group scanning;
- Group Call, including Talking Party Identification (TPI) and Late Entry (LE);
- Individual Call, semi-duplex;
- Call Identification (CI);
- Priority Call (PC), Pre-emptive Priority Call (PPC) and Emergency Call;
- Speech Item Priority;
- Broadcast Call;
- Location reporting.

Note that the CVDP relay currently supports IP unicast signalling, voice and data only, while the mechanisms to control IP multicast are not defined yet. The CVDP gateway connects the CVDP relay to the TetraNode Mobile system. The CVDP gateway supports the TetraNode IP Gateway version 2 (TIGv2) for connection to the TetraNode Mobile system. By means of this gateway, calls can be established that include both TETRA radios and LTE devices. Also the LDS Chameleon can reach both radios in the TETRA and LTE domain. The same functionality is applicable as for the CVDP relay application.

The TeamLink application can be installed on any recent Android smart phone. This TeamLink application provides typical Private Mobile radio (PMR) features in a user-friendly manner. Again, the same functionality is applicable as for the CVDP relay application. The TeamLink application also supports Bluetooth connected speaker-microphones⁵ with physical PTT button to allow for better voice quality and improved PTT operation.

4.1.2 SALUS Mobility Management

The Mobile Independent Handover (MIH) for the SALUS PPDR platform is already implemented for integration. The SALUS MIH solution provides a heterogeneous handover between LTE to Wi-Fi using PMIPv4 and currently the LTE access is emulated over Ethernet links. For the SALUS MIH Solution the Packet Data Network Gateway (PDN-GW) is developed in OpenEPC to terminate the interface towards the packet data networks (e.g. IMS, Internet, etc.).

⁵ Supporting the Advanced Audio Distribution Profile (A2DP)

The following modules are implemented by the OpenEPC platform to support mobility management in LTE EPC based on the specification of the aforementioned entities. Figure 50 depicts the entities, where modules are located as well as the interfaces over which the modules interact.

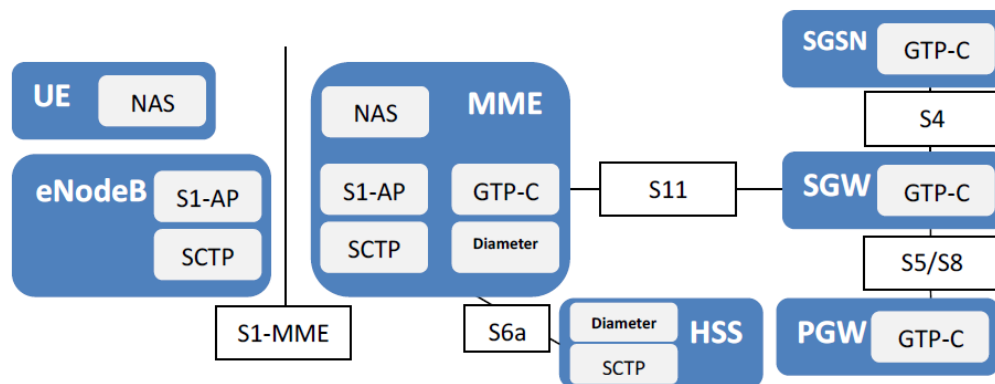


Figure 50 – Key module evolved during mobility management in LTE EPC

- **S1 Application Protocol (S1AP) Module [20]:** The S1AP specifies the eUTRAN radio network layer signalling protocol for the S1 interface. The S1AP supports the functions of S1 interface by signalling procedures such as mobility management, radio resource management, location reporting, UE context Modification or Release function.
- **GPRS Tunnelling Protocol (GTP) Module:** This module implements the GTP stack for both version 1 and version 2. The GTP for the control plane (GTP-C) sub-module is the protocol that tunnels signalling messages between the S-GW and MME through S11 interface, SGSN through S4 and PDN-GW through S5/S8.
- **Non-Access Stratum (NAS) Module [21]:** This module implements the NAS stack between the UE and the MME on the network side (outside of eUTRAN). NAS is an additional layer of abstraction to protect important information like key and security interworking between 3GPP and non-3GPP network.
- **Stream Control Transmission Protocol (SCTP) Module [22]:** This module implements the SCTP protocol.
- **Addressing Module:** This module is used to determine the appropriate SGW for the client's attachment. Several addresses can be configured, which will then be used in a round-robin order.

The following OpenEPC modules are also configured for the heterogeneous handover between LTE and Wi-Fi access networks.

- **lma module:** The main module of the LMA component contains the logic for handing the requests coming from the MAGs of the various Access Network Gateways (ANGWs).
- **mag module:** The main module of the MAG component contains the logic for handling requests coming either directly from the UEs, or from other access network components, like 3GPP components (MME, SGSN). The module also uses the *gw_bindings* module to store the current state of the attached UEs. Currently, the MAG can work in two ways:
- **ANGW:** in this case the module will receive the attach/detach requests directly from the UEs. This is done using the Enhanced Host Configuration Protocol (EHCP) protocol, which is a modified version of the DHCP protocol developed for OpenEPC. The MAG

will forward the request to the LMA and after the request is processed, the module replies to the UE using the same EHCP protocol.

- S-GW: when the MAG works at an SGW, instead of the *ehcp* module, the *sgw_s11s4* module is loaded, along with the required *gtp* module. This module handles the requests coming from MMEs and SGSNs.
- *gw_bindings* module: The *gw_bindings* module stores the current state of the attached clients. It is used by both the MAG and the LMA. The information is stored in a hash table of configurable size. The module supports database communication by pushing the binding state to the database every time there is a change in the state. The module stores all the information about a user in one structure, which contains a list of all the PDN connections activated for the client.
- *pdn_ops* module: This module does the IP allocation from pools along with DNS procedures. The *pdn_ops* module is required by the LMA to establish or release tunnels based on the UE binding state.
- *routing* module: This is the basic module that incorporates all the routing protocols through their sub-modules. These are the *routing_encap* for tunnelling packets in IP-in-IP or GRE tunnels, the *routing_raw* for receiving and sending packets using raw sockets and *routing_gtp* for receiving and sending packets using gtp tunnels.
- *gtp_s5s8* module: the GTP S5/S8 interface implementation for communication between S-GW and the PDN-GW. This module contains the procedures for 3GPP Accesses with PMIP-based S5/S8. The LMA will then work in a dual way, accepting both PMIP requests and GTP requests, replying to each using the respective protocol.

4.1.3 Message Broker

As stated in section 2.2.5.1, the concept of the message broker entails two components: the message broker service, which is a network service, and the message broker API, which is a software library that application developers use to interface the message broker service. The status of the two components is described separately.

4.1.3.1 Message broker service

The Message broker service is a network service that runs on a well-known network address and to which all devices that want to send and receive messages connect. It is based on the idea of reliable messaging and currently uses TCP as the transport mechanism. (UDP can be supported too.) The service is implemented in Python programming language and based on the Zero Message Queue (ZMQ) library [41].

The implemented message broker service supports both communication channels: the control and the data channel. The data channel carries data messages: messages that contain sensor readings or similar type of *business logic* content. The control channel, analogously, carries control messages: messages that contain instructions, for instance, for changing settings on clients and similar. Further information of the message broker can be found in SALUS Deliverable D5.2 [36].

The current implementation of the message broker service lacks security controls. These are envisioned to be developed by the time of the second prototype.

There is an on-line instance of the message broker service running on the address *mb.sec-salus.eu* address on port 5550. This instance can be freely used by SALUS partners to test their applications. The overall status of the message broker service is ready for integration.

4.1.3.2 Message Broker API

The Message Broker API (MB API) is a software library that runs on a UE or on a back-end computer and provides a programmable interface for communicating with the message broker service. Currently, only Java programming language is supported, while the support for other programming language will be added later. The MB API offers developers several methods for sending and receiving messages. To aid developers, a set of tutorials in the collaborative system redmine system has been set up.

Additionally, three proof-of-concept applications have been developed that use the MB API. First, a back-end application that given a person's GPS coordinate and its heart-beat rate shows the position of the person on a map together with its heart rate. Second, an Android application that streams heart-beat rate signal and GPS location coordinates to the aforementioned back-end application via the message broker service. These applications are overviewed in section 2.2.5.3.1. The third application, as described in section 2.2.5.4, employs simple algorithms to detect outliers in the bio-signals being monitored, promptly throwing alerts into the MB.

As with the message broker service, the MB API is ready for integration, but lacks security controls. These are planned to be implemented in the second iteration of the SALUS prototype.

4.1.4 HUCare -- Integrated bio-sensors and video solution

HUCare is integrated with the Message Broker (MB), though a Java API (described in section 4.1.3.2) in the first prototype. HUCare sends alarms to the MB each time a certain threshold or anomaly is detected (for instance man-down or heart attack events). The alarms information contains the sensor ID, a timestamp and a description of the alert, so that applications subscribing to this kind of alert can process these alarms.

4.1.5 SALUS PKI

The SALUS PKI is being developed in Java and currently operating on a Linux (Ubuntu) operating system. The PKI concept itself is being built using the Bouncy Castle crypto APIs [42] for Certification Authority (CA) creation and associated functionalities such as public/private key generation, digital certificate generation and revocation. The SALUS PKI also contains a secure database, developed using the PostgreSQL [43], for user and key storage. User and device interaction with the PKI is achieved by using a Tomcat web server, which provides a Secure Sockets Layer (SSL) encrypted interface for user/device certificate enrolment and PKI administration (back office). For device direct certificate enrolment the PKI allows Secure Shell (SSH) connections for remote command execution. The building blocks of the PKI Server are depicted in Figure 51.

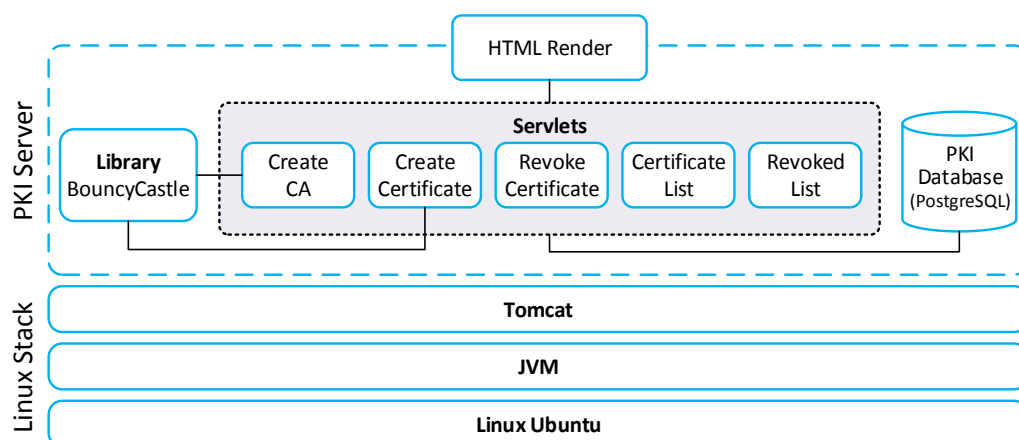


Figure 51 – SALUS PKI building blocks

The interface used to request digital certificates or certificate revocation lists is described in section 4.2.5.

4.1.6 SALUS IP Communications Server

The SALUS IP Communications Server is built using Asterisk [44] version 11. Asterisk is an open source framework for building communication applications; it acts as the underlying platform that enables the SALUS IP communication features described in section 2.2.12, Table 7. As depicted in Figure 52, the entire system runs on top of the CentOS Linux platform.

At the core of asterisk is the dialplan. The dialplan is a form of scripting language that contains instructions that Asterisk follows in response to external triggers. It defines how calls flow into and out of the system. The dialplan can make use of applications that provide call functionality to the system, such as answer a call, play a sound (which is a resource), hang up a call, etc. Asterisk configuration and monitoring can be done through a command line interface or through a manager interface.

At the top of the diagram the channel drivers are depicted. They are responsible to enable communications with external devices and translating a particular signalling or protocol to the core. For the scope of SALUS two channels are enabled, the Session Initiation Protocol (SIP) and the Digium Asterisk Hardware Device Interface (DAHDI). SIP is defined in RFC 3261 [39] and is a well-known application-layer control (signalling) protocol for creating, modifying, and terminating sessions⁶ with one or more participants. DAHDI contains the kernel drivers for telephony adapter cards that enable asterisk to interface with legacy (non-IP) systems, such as the PSTN or ISDN networks (it also includes automatic configuration utilities and test tools).

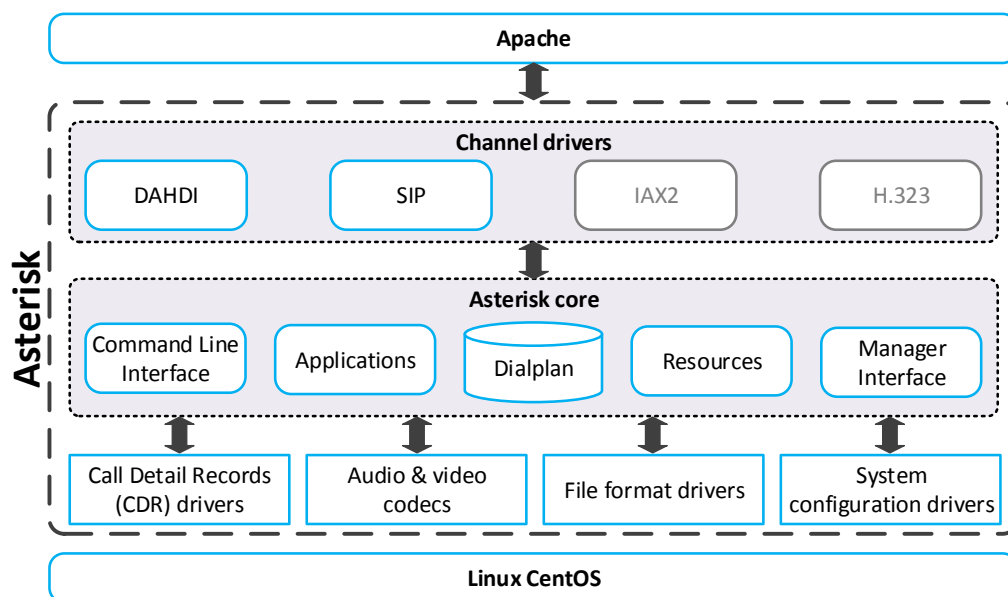


Figure 52 – SALUS IP Communications Server building blocks (simplified)

At the bottom of the diagram, the Call Detail Records (CDR) drivers allow to write call logs to a database, audio and video codecs for encoding and decoding media and file format drivers to save media to disk in a particular file format.

Section 4.2.6 details the SALUS IP Communications Server interfaces.

4.1.7 Intrusion Detection Systems

The various components of the SALUS Intrusion Detection Systems (IDSes) described in Section 2.2.11.2 currently have different implementation/prototyping status:

⁶ These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences.

- Flow exporter, flow collector (UTWENTE) – The flow export and collection components will be based on the (open-source) packages nProbe [45] and NfSen [46], respectively. These components run on any Linux distribution and their source code can be audited, if necessary. The flow exporter package needs to run with super-user permissions such that it can intercept network packets from a specified network interface.
- IDS agent (UTWENTE) – The IDS agent that will be used for the first demonstration is SSHCure [47], a flow-based intrusion detection system for SSH. Like the other flow-based components, SSHCure is an open-source package that is under active development. SSHCure runs as a plugin for the popular flow collector NfSen, and therefore has to run on the same machine. It has been tested on many Linux and BSD distributions.
- IDS agent (ONE) – The IDS agent is still under active development by ONE, especially with regard to the various forensic data acquisition methods; it supports both live acquisition and post-mortem acquisition. The agent runs on mobile (handheld) devices and forms a hybrid IDS by analysing both network traffic and various resources of the mobile device itself.
- Global/local Security Manager (KU) – The Security Manager is the least mature component of the SALUS Security Architecture, as it currently only resides within the design phase. As a consequence, no prototype exists yet that could be demonstrated; this will be part of future demonstrations.

Due to the fact that the global/local Security Manager is not ready for prototyping yet and the Security Manager interconnects all IDS components, no interaction between IDS components can be demonstrated in this part of the project. This part of T5.4 is therefore left out for future demonstrations.

4.2 Defined Interfaces

The diverse components communicate with others, through defined interfaces. This subsection details such interfaces, providing information regarding the messages exchanged.

4.2.1 Web QoS interface

This Web-RTC for Dynamic QoS interface is under development. It relies on a RESTFUL web API and it is not present in the first prototype due to its current status. Note that this interface aims to allow the dynamic configuration of LTE for QoS support of certain applications (e.g., video, pre-emptive services).

4.2.2 CVDP

The Critical Voice and Data Protocol (CVDP) is available as open specification for third parties to interconnect applications within the PMR-over-LTE ecosystem. Within the SALUS project, it will be used to interconnect with Location Services applications.

CVDP is based on XML-over-UDP/IP, although in a later stage other, more compressed, variants will be considered to minimise bandwidth requirements. The IP packets will be carried over LTE using specific QoS parameters that need to be provisioned on basis of Differentiated Services Code Point (DSCP) and/or IP port numbers.

In a later stage, it is expected to integrate support of Mission-Critical Push To Talk over LTE (MCPTToLTE) within the CVDP architecture. This means that the end-user can benefit from both the current and evolving capabilities of CVDP and its ecosystem, as well as potential applications and devices that will become available after standardization of MCPTToLTE.

Because of the common system architecture of CVDG and MCPTTtoLTE, this integration is both feasible and beneficial for the end-user investing early in PMR capabilities on top of LTE.

4.2.3 Heterogeneous Handover in OpenEPC

In case where the UE is connected with different IP services, several PDN-GW are associated with this UE (User Equipment). It performs UE IP allocation, policy enforcement, packet filtering per user and packet screening. Moreover, PDN-GW acts as the mobility anchor between 3GPP and non-3GPP access. It has interfaces to the S-GW, the Evolved Packet Data Gateway (ePDG), which is responsible for interworking between the EPC and untrusted non-3GPP networks, such as a Wi-Fi, interface to the 3GPP AAA server, the trusted non-3GPP access gateway and the PCRF. The MIH functionality (MIHF) it will be located at the S-GW (Serving Gateway) and the A-GW (Access Gateway) in order to monitor the access networks and update the MIH *Information Service* (MIIS) database.

UPAT's immediate future steps are to investigate a full solution for Core Network Mobility Management, transparent to mobile clients and SALUS PPDR platform, and a Client Device Mobility solution, supporting Vertical Handoffs with PMIPv4 and PMIPv6 using the OpenEPC platform. The components of MIH solution for Client Mobility Support will have the role of interacting with the Core Network Mobility Management Components providing an enhanced experience for the SALUS applications running on the client devices. Also, UPAT is currently working on integrating and implementing the 802.21A protocol [23] on the existing MIH platform. The main concept introduced by this protocol, is to secure the MIH messages and reduce the delay imposed by the authentication procedure, by performing a proactive authentication.

4.2.4 Message Broker

The only way UEs and back-end applications can interface with the message broker is through its API. Therefore the only interface that can be used to connect to the message broker service is offered by the message broker API. This means that currently only Java-based applications can connect to the message broker service. But since MB uses the IP stack, the support for other languages can be added later by providing APIs for other languages; in general, any IP-capable device may connect to the message broker service. Once secured, the message broker service will only be reachable over IPSec connections.

The message broker can support arbitrary types of messages and it is not limited to any specific formats. For the second iteration of the prototype the SALUS plans to provide an application-specific message formats.

4.2.5 SALUS PKI

As described in SALUS Deliverable 5.2 [36], see also Figure 53, the SALUS PKI uses the *S-Au-Data* interface, an SSL/SSH encrypted data channel, that can be used by users, Security Gateways, UEs (LTE) or MNs (Wi-Fi) to request digital certificates or certificate revocation lists.



Figure 53 – SALUS PKI interface

The *S-Au-Data* interface can use either HTTP/SSL or SSH connections. PPDR users can interface with the PKI through a secure HTTP over SSL connection (web page). Security Gateways, UEs or MNs can use automated certification enrolment by using a pre-configured SSH connection for remote command execution.

4.2.6 SALUS IP Communications Server

The SALUS IP Communications Server mainly uses a standardized SIP interface. To ensure compatibility with legacy PSTN networks (non-IP based communication network) it also includes a Telephony Interface Card (TIC) that can support Foreign Exchange Office (FXO) and station (FXS) interfaces. Even though not currently supported, through the DAHDI drivers the server can also interface with primary (T1/E1) or basic (BRI) ISDN interfaces.

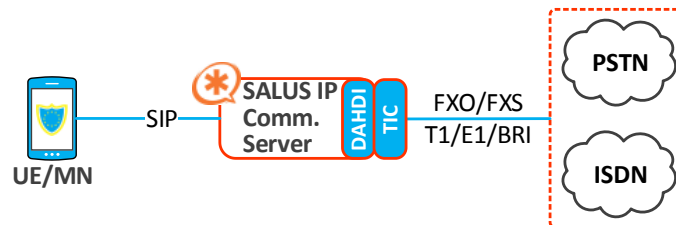


Figure 54 – SALUS IP Communications Server interfaces

4.2.7 Intrusion Detection Systems

The SALUS intrusion detection systems feature two types of interfaces, as shown in Figure 55:

- *S-FE-IDS* – This is the interface over which flow data (exported using Cisco’s NetFlow or IPFIX) is transferred to flow collectors for storage and pre-processing. Since the first demonstration will feature our flow-based IDS, this interface will be demonstrated (implicitly).
- *S-IDS-IDS* – Due to the lack of a prototype of the global/local Security Manager, no inter-IDS communication can be demonstrated yet at this stage of the project. Demonstration of this interface will be part of a future demonstration.

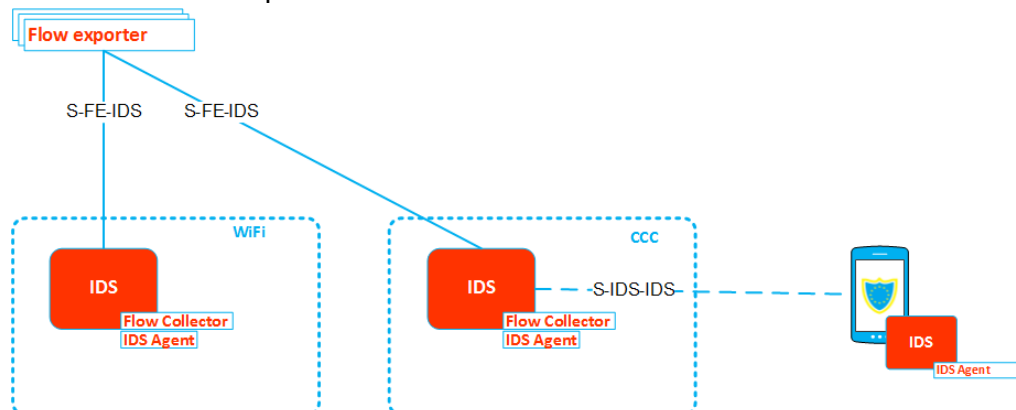


Figure 55 – SALUS IDS interfaces

4.2.8 TETRAPOL CCAPI interface

The CCAPI interface provides service control protocol at application layer:

- Communication control and Access Gate management;
- TETRAPOL system monitoring (through AGs).

In IDR mode, the CCAPI server is connected to a single AG-R interfaced to a tactical cell. A CCAPI client is operating in IDR mode when it is tuned onto an IDR channel to participate in an unrelayed group communication. Once in IDR mode, the CCAPI client can also:

- Send and receive SMS; then the former IDR communication is automatically resumed once the SMS is sent.
- Be notified about direct or IDR mode emergency communication.

Voice and SMS transmissions may be end-to-end encrypted.

The external interface offered towards the Command Control Centres follows diverse principles, as detailed in Table 12.

Table 12 – CCAPI Control Centre Principles

Principle	Characteristic
High-level	The concepts are abstract enough that the Control Centre must not know the TETRAPOL detailed objects if it doesn't need to know them.
Clarity	The language used and the data model of the object enable to easily understand the interface and its behaviour.
Continuity	The objects used for the CCAPI interface are very close to the ones used for the CCIS interface, as well as the technical interface.
Upwards compatible	It is possible to upgrade a network with no Control Centre upgrade. It is possible to upgrade a Control Centre with no network upgrade.

Hereafter the categories of services and data flows managed through CCAPI interface are identified in the system specification under the names as specified in Table 13

Table 13 – CCAPI interface service names and categories

Categories	Description
Local control	Management of specific mechanisms to manage communications between CC-API server and client applications.
Circuit control	AG management from client applications.
System monitoring	Data management for accessibility of communications and related entities such as operational groups and coverage.
Call control	Provides services to setup, release, participate or withdraw of voice communications for client applications.
Call advertising	Provides notifications for activation and deactivation of group communications and private communications.
Voice traffic signaling	Provides notification about PTT requests and releases, voice reception and transmission detection for AG.
Data	SMS service is available at CC-API interface.

The AG-R has different operation modes, it is either line-connected or radio. In both cases, it may work in:

- **Monitoring mode.** Signalling events are reported to the client application to monitor events that occur in the TETRAPOL system. There is only one monitoring AG in a cell.
- **Communication mode.** A client application is participating into a group communication or a private call. When the communication is not active (no call is on-going and the hang-time elapsed), the AG-R is idle.

Only one AG in a cell may be set in monitoring mode. When several CCAPI clients are connected to AGs located in the same cell, only one CCAPI client can set one of its AG-R in monitoring mode. Others can just have communication mode AGs and cannot retrieve the monitoring signalling.

It should be noticed that an AG-R can only deal with one mode. The client application is responsible to select one available AG to process monitoring, but a cell may operate with no radio monitoring AG. A radio monitoring AG may be selected to work in communication mode, but it must be disabled for monitoring first. The CCAPI server only checks if there is no more than one monitoring AG in a cell. When a monitoring AG is in fallback mode, other selected AGs keep the on-going communications. When the monitoring AG is an AG-R, the client application may select another one.

4.3 Roadmap and Open Issues

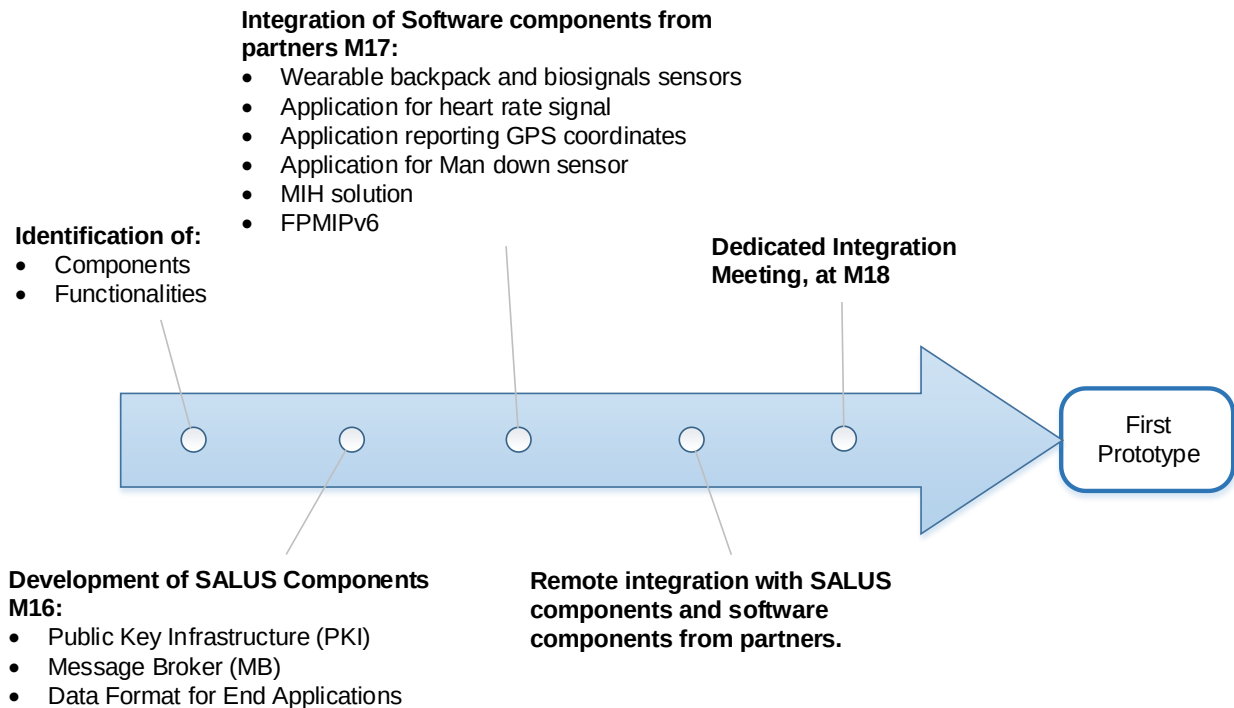


Figure 56 - First prototype integration roadmap

Figure 56 depicts the roadmap to integrate components of the first SALUS prototype. As presented, the first step includes the identification of components and respective functionalities that compose the first prototype. Following this step, deadlines were established regarding SALUS developed components, such as Message Broker and PKI, described in section 4.1.3 and 4.2.5, respectively.

Remote integration activities with the SALUS components and software components from each partner were promoted, for instance, the integration between sensors and the message broker. Such integration activities, besides being performed remotely were also performed in a dedicated integration meeting with the purpose of preparing the demonstration of the prototype.

The integration progress is constantly monitored in the regular WP7 calls where all partners with components are participating.

As presented, the first prototype does not implement all the functionalities supported by the diverse SALUS components.

Table 14, below, summarizes the mapping of functionalities demonstrated in the first prototype and in the final prototype.

Table 14 – Mapping of functionalities between first and final prototypes.

Partners	Functionality	First Prototype	Final Prototype
UL and all with sensors	Integration of sensor information in the SALUS CCC application	Each partner with sensors provides its own CCC application to view sensor information	Sensor information will be integrated in the CCC application for complete situation awareness.
UL	Security in the MB.	The message exchange with the MB is not secured.	MB will implement security mechanisms defined in the SALUS security architecture.
UTWENTE, UPAT	Mobility Management Components	The Mobility Management Components support Vertical and Horizontal Handovers between Wi-Fi and LTE with IPv4	The Mobility Management components are integrated through MIH to allow Vertical Handovers between Wi-Fi and LTE with IPv6
UPAT	Mobility Management Component with secure MIH		Integration of 802.21A in the MIH platform
KU, FIGONET	ChaMeleon integrated with Ad-Hoc Networks system	ChaMeleon routing protocol is deployed in a Wi-Fi Rugged device but not integrated with Ad-Hoc Network System of SALUS	Integration of ChaMeleon with SALUS Ad-Hoc System
UL, FhG and all with sensors	Sensor Message format	Partners using the Message Broker or other mechanism can specify their own message format for sensors. For instance, can be based on JSON, on XML, or other.	A common message format will be defined to be supported by the diverse sensor applications
IT	Outside connectivity in SALUS IP Communications Server	No support for Voice calls with external networks (PSTN or ISDN)	Support calls with external networks.
ROH and ALU-I	PMR-over-LTE support of multicast in CVDP relay	CVDP relay only support unicast signalling	CVDP relay supports multicast signalling
ROH and ALU-I	Integration of Mission-Critical Push to Talk over LTE in CVDP architecture	Available provided update by Rohill on pre-integrated equipments in ALU-I premises	Integration of Mission-Critical Push to Talk over LTE in CVDP architecture
UTWENTE, ONE, KU	Intrusion Detection Systems	The flow-based IDS (UTWENTE) will be demonstrated, together with a first prototype of the Mobile Forensics module.	IDSes by UTWENTE and ONE, featuring inter-IDS communications, based on the local/global Security Manager.
UBITEL	Indoor localisation service for end-user mobile devices	Proof-a-concept prototype with the localisation server, capable of estimating mobile devices localisation in real-time	Localisation service Is integrated with Message Broker and other necessary SALUS security Mechanisms as well as with common operators' visual interfaces
UB	Sensor data: man-down application, detection of man-down situation using motion sensors in smart-phones.	The possibility to detect man-down situation will be demonstrated with additional basic info and plot on the map.	The application will use the common message format for communication with SALUS CCC.
UB	Wi-Fi physical layer jamming		Demonstration of Wi-Fi jamming on physical layer using strong electromagnetic field. Detection and estimation of jamming area using smartphones.

Partners	Functionality	First Prototype	Final Prototype
FhG	Command Control Centre Applications	Situation display, UAV tracking and Video Display	Situation awareness (i.e. integration of status from sensors); PPDR Force Tracking
CAS	TETRAPOL Network connection	Tactical bubble with Control Room interface	Projectable TETRAPOL network, similar to an actual network, with its Control Room interface
ALU-I / IT	RESTFUL API Dynamic QoS	Not ready	To be provided
ALU-I	1x 9773 LMC	ePC micro-core LTE available	ePC micro-core LTE available
ALU-I	1x enodeB	ePC micro-core LTE available	Idem
ALU-I	1x Band 20 RRH	ePC micro-core LTE available	Idem
ALU-I	1x Usb 4G dongle	Available	Available
ALU-I	3x galaxy S3 4G band 20	Available	Idem
ALU-I	3x GalaxyTab 4G band 20	Available	Idem
ALU-I	VMS solution	Based on Milestone Available	Idem
ALU-I	PTT solution	Based on Rohill Available	Idem
ONE	HUCare	Integrated with MB	Supporting security mechanisms defined in SALUS

5 EVALUATION PLAN

This section relates to the experimentation and evaluation process of the SALUS PPDR platform integrated prototype, as a whole, according to the scenarios and use cases defined in WP2 and WP3. Task 7.1 will execute this process.

It is important to highlight that the evaluation work done under the scope of Work Package 7 will be mainly experimental, using: testbeds and prototypes (basic and final) produced by WP5 and WP6. Furthermore, activities (mostly simulations) will take place within the context of Task 7.2, 7.3 and 7.4 for the validation of specific aspects of the scenarios. These activities will naturally provide a valuable input for evaluation activities.

Insights to the evaluation process objectives and methodology to be adopted will be provided, as well as to the roadmap foreseen at the time of writing. Evaluation will be separated in two parts: impact evaluation and process evaluation. Impact evaluation will deal with the measurement of direct effects of the SALUS PPDR solution. Process evaluation will provide insights to the drivers and barriers that may arise during the preparation, implementation and operation of the SALUS PPDR solution.

Note that during the project lifetime the evaluation plan will be considered as an iterative process, eventually periodically updated, in order to cope with possible revisions.

5.1 Introduction

This section defines the strategic evaluation of objectives, according to the requirements defined in WP2, as well as those defined in WP7. These strategic objectives include the use of the appropriate selection of Key Performance Indicators (KPIs) (qualitative and quantitative) suitable for evaluation.

Along with the selection of indicators, it is important to set the qualitative targets and expected results to be achieved per each of the indicators. These targets and expected results provide a clear identification of “success”, otherwise would mean that there is no real notion of whether the effort and investment will bring the desired benefits. Moreover, only with a recognized success there can be a reasonable potential for the adoption of the proposed solution.

The selection of indicators followed the following criteria:

- *Relevance*: an indicator should represent an assessment criterion, i.e. have a significant importance for the evaluation process.
- *Completeness*: the set of selected indicators should consider all aspects of the solution under evaluation.
- *Measurability*: the identified indicators should be capable to be measured objectively or subjectively.
- *Familiarity*: the indicators should be easy to understand.
- *Non-redundancy*: indicators should not measure the same aspect of an assessment criterion.
- *Independence*: small changes in the measurements of an indicator should not impact preferences assigned to other indicators of the evaluation model.

Furthermore, it is necessary to select and test a set of use cases according to their characteristics and requirements. Having accomplished the abovementioned objectives, the impact evaluation and process evaluation is carried out, by comparing the measured outcomes with the initial definition of success.

Finally, this process should then lead to a more deep analysis to the possible adoption of SALUS PPDR platform solutions by the different stakeholders. The following subsections present the areas, impacts and indicators, relevant methodologies and evaluation technique.

5.2 Evaluation Areas, Impacts and Indicators

The three scenarios developed under WP2 place into context certain functionalities required by PPDR responders. Almost any particular functional requirement could be used dependant on the situation presented to PPDR responders. The three scenarios therefore focused on the functionalities deemed most appropriate to that scenario although some are common to every scenario. Group calls for example are so important to the operational effectiveness of the majority of PPDR responders it will be used in any scenario requiring police, fire and ambulance responders jointly attending.

At a high level the main requirements can be summarised as follows:

- **Voice** – Users can make a variety of voice call types including group, announcement/broadcast, emergency, individual and telephony interconnect calls. The infrastructure must ensure that emergency calls have priority over other calls, releasing capacity for an emergency call to be connected pre-empting other users where necessary.
- **Video** – Users can send and receive video imaging to all users, specific user groups or individuals, either from a dispatcher to the group or vice versa.
- **Data Based Applications** – Users can have mobile access to various data applications such as messaging services and email, organisation-specific databases and other data-rich applications such as location services, augmented reality and DNA/Fingerprint scanning.
- **Network Interoperability** – Communications are possible between users from different organisation whether operating on a common infrastructure or connected via different technologies or networks, as well as communications between aircraft and users on the ground. Additional network coverage or capacity can be deployed quickly and easily where such a requirement exists, for example in remote locations.
- **Security** – All the communications must include secure mechanisms, such as encryption, and mutual authentication.

The above functionalities will be used as the main focus areas for the testing of the SALUS demonstrators but other innovative functionalities developed under SALUS are not excluded. A key need will be to demonstrate the functionalities across different communications technologies to prove a defined objective for which a clear test objective and expected outcome will need to be defined.

Testing can be a complex, resource intensive and an expensive activity. Practical aspects, such as the location of the test environment, also need to be considered carefully. These are all considerations that when taken fully into account may limit the scope or shape of a particular test.

Any testing must be performed within a controlled environment. It is envisaged that most of the functionalities will be practical testing with end user participation. However there may be some areas that can only be realistically tested within the confines of a laboratory type environment. The latter is particularly relevant where for example loading testing is necessary or there is a need to evaluate performance. An example of the latter would be call set-up times, which could be difficult to measure outside of a laboratory environment. If a laboratory environment is

necessary then a restricted number of end users would still be able to participate or witness a test.

5.3 Methodology and Evaluation technique

The evaluation of the SALUS demonstrator will be based on proving that the specified PPDR functionalities exist end to end whether between terminals, or between terminals and a control terminal. To do this efficiently, the evaluation process will be based on a number of test activities based on a generic test process. This generic test process illustrated in Figure 57 has been adopted from the ISEB/ISTQB Fundamental Test Process.

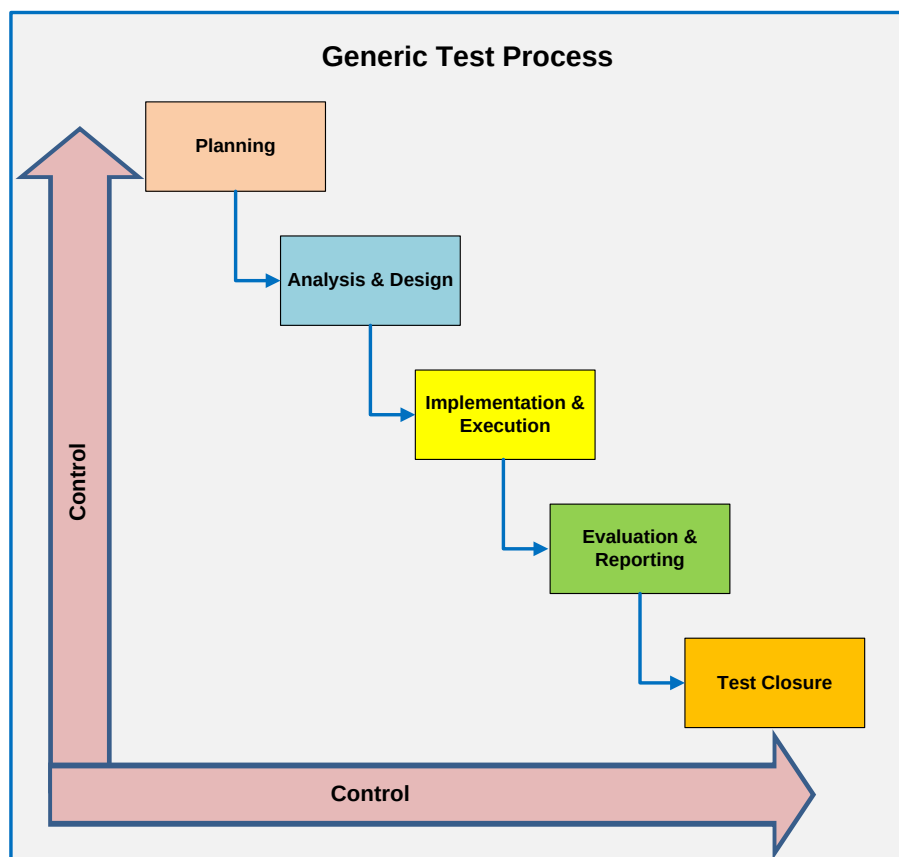


Figure 57 – Generic Test Process

The key element in the Test Process is the Control action. This is present throughout the process because it is an ongoing activity of comparing actual progress against the plan and reporting the status, including deviations from the plan. The control element also ensures that the Test plan is updated as the testing progress to ensure it is a true reflection of the actions taken. The test planning stage is where the risk analysis takes place. This will involve analysis of the requirements and deliverables for a system or software solution to identify risk areas. Once the risks have been identified they can then be classified using ISO 9126 definitions and then prioritised into the order in which they need to be addressed. Once the risks have been identified and prioritised only then can the testing effort required be estimated and subsequently planned.

5.3.1 Analysis and Design

When there is sufficient detail regarding the amount of test effort required the Analysis and Design stage can start. Further analysis of the requirements will lead to defined tests cases, which in turn will allow the test scripts to be developed. Test data is an important part of testing and will need to be representative of the data in the deployed system or software solution. A key

output from this stage will be the Test Schedule because once the tests have been designed the test sequence and subsequently the test schedule can be determined.

5.3.2 Implementation and Execution

This is where the system or a component part of a system to be tested is made available to the test team so that it can be installed onto the test environment ready for testing. Once the release has been installed or setup and configured in the test environment the test execution phase can begin. This is where the test scripts will be run and all results will be recorded on Airwave's Test Management system. Defect management also sits in the Implementation and Execution stage.

5.3.2.1 Evaluation & Reporting

Evaluating exit criteria is the activity where test execution is assessed against the defined objectives. This should be completed for each test level. It is during this stage where we will determine if the testing has been successful in analysing the risks that remain at the end of testing in comparison to the quality level set during the planning stage.

5.3.2.2 Test Closure

Test closure activities collect data from the completed test activities to consolidate experience, testware, facts and numbers. Test closure activities occur at project milestones such as a test project is completed (or cancelled) or a milestone has been achieved. Test closure activities can include lessons learned meetings, test process improvements, closure of test incident reports.

5.3.2.3 Test Control

Test control is the ongoing activity of comparing actual progress against the plan and reporting the status, including any deviations from the plan. It involves taking the actions necessary to meet the objectives of the project under test. In order to control testing, the testing activities should be monitored throughout the project. Test planning takes into account the feedback from the control activities.

5.3.2.4 Test Methodology

Test Methodologies are required within larger testing life cycles. Smaller life cycles patches & corrections or Ad-Hoc testing may require one or two test activities or types of testing only. The introduction, for example, of a new service that requires four new systems to be integrated will definitely require test methodologies.

The test approach defines test activities to address test objectives, there may be multiple test methodologies required or the test activities could be combined into a single test methodology. Therefore it may appear that test methodologies are a "woolly" entity, as from the example above, the one or two test activities or types of testing may be also included in the test methodology of the introduction of a new service.

A test methodology is a documented and repeatable combination of test phases, types of testing and test process to support the test approach. The test methodology also needs to align to the project life cycle or development methodology. For example if a test principle is to address risks of a newly developed software solution, issues would arise if utilising the V-model testing methodology when a prototyping development methodology is being used.

Although there are standard, well known and documented test methodologies, like the V-model, we must remain flexible in our choice of methodology to ensure we do not neglect the test approach by purely following a known methodology. The methodology supports the test approach, not the other way round. Therefore the test methodology will be defined in the test strategy by selecting test methodologies that fit with the test approach and test activities required.

The diagram depicted in Figure 58 shows an overview of how a system is broken down to create test procedures.

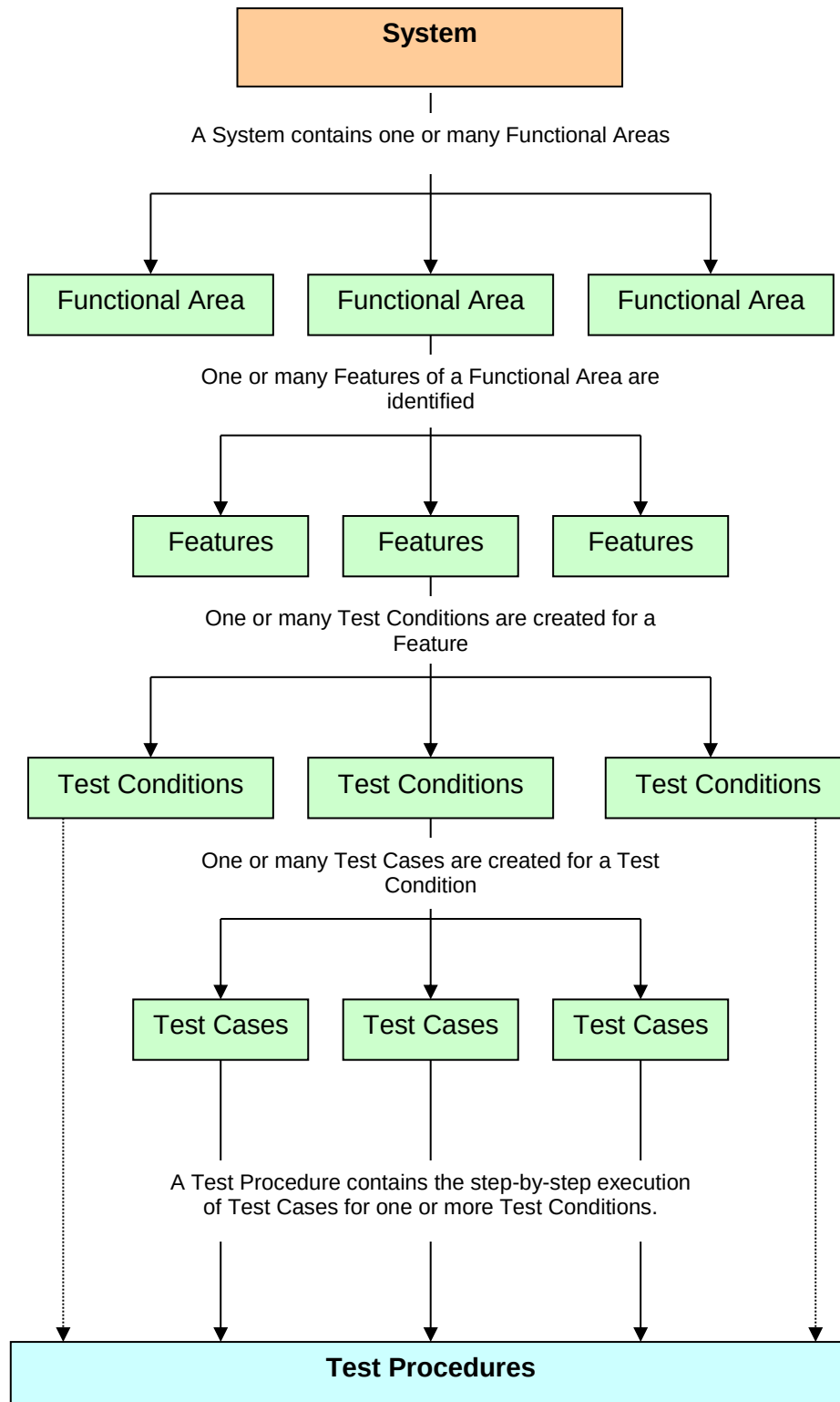


Figure 58 – Test Procedures

The methodology for evaluation of the areas of interest will focus on the end-user experience of the SALUS solution. Per item mentioned above one or more situations will be created that show the gain of the solution.

5.4 Roadmap and Open Issues

This section presents the roadmap for the activities showed in the previous section. Table 15 summarizes the roadmap, identifying the activities to be conducted in order to achieve results from different Task (7.2, 7.3 and 7.4), which will be presented in the deliverables D7.2 and D7.4. Each phase comprises all evaluation process activities. As highlighted before in this chapter, the evaluation process will be iterative.

Table 15 – Evaluation Activities (according to adopted methodology), Gantt Chart

	Dec-14	Jan-15	Feb-15	Mar-15	Apr-15	May-15	Jun-15	Jul-15	Aug-15	Sep-15	Oct-15	Nov-15	Dec-15	Jan-16	Feb-16	Mar-16	Apr-16	May-16	Jun-16	Jul-16	Aug-16
Week#	1 6	1 7	1 8	1 9	2 0	2 1	2 2	2 3	2 4	2 5	2 6	2 7	2 8	2 9	3 0	3 1	3 2	3 3	3 4	3 5	3 6
Methodology definition																					
Review Requirements																					
Definition of strategic evaluation objectives																					
Definition of evaluation areas and impacts																					
Selection of indicators																					
Definition of success and targets																					
Definition of evaluation technique																					
Data Collection, Processing and Analysis																					
Adoption analysis																					

6 CONCLUDING REMARKS

This deliverable follows the purpose of reporting the current Integration and Evaluation plans that strive to highlight the added value of the SALUS PPDR platform. This is achieved by defining applications and services in the SALUS PPDR platform that are relevant for the PPDR community and demonstrating them in realistic and tangible scenarios.

The approach towards the definition of appropriate demonstration scenarios was defined to match the longevity of the project and the scheduled tasks to be completed throughout it. An iterative process is considered and several independent goals were established to promote integration of components and respective evaluation towards a complete and fully functional SALUS PPDR platform, with advanced features.

Finally, due to the wide-range of interests embraced by the SALUS PPDR project, the expected outcome of the all the proposed applications, prototypes, scenarios and metrics, varies according to their final purpose. Not only are these considered for exhibiting and presented in generic public events, but also for specific target audiences. Moreover, several actions are also considered for internal validation and performance evaluations, being the obtained results confidential and private.

BIBLIOGRAPHY

- [1] Jansen, W. & Ayers, R. P., 2007. SP 800-101. Guidelines on Cell Phone Forensics. Gaithersburg, MD, United States: National Institute of Standards & Technology.
- [2] ASTM International, 2009. Forensic Science Standards. [Online] Available at: <http://www.astm.org/Standards/forensic-science-standards.html>
- [3] ISO/IEC, 2012. Guidelines for identification, collection, acquisition, and preservation of digital evidence, First Edition
- [4] Thing, V. L. L., Ng, K.-Y. & Chang, E.-C., 2010. Live memory forensics of mobile phones. Digital Investigation, Volume 7, pp. 74-82.
- [5] Volatile Systems, 2011. The volatility framework: volatile memory artifact extraction utility framework. <https://www.volatilesystems.com/>
- [6] DiCerbo, F., Girardello, A., Michahelles, F. & Voronkova, S., 2010. Detection of malicious applications on Android OS. s.l., Springer-Verlag Berlin Heidelberg, pp. 138-149.
- [7] Houmansadr, A., Zonouz, S. A. & Berthier, R., 2011. A cloud-based intrusion detection and response system for mobile phones. Washington, DC, USA, IEEE Computer Society, pp. 31-32.
- [8] Schutz, P., Breuer, M., Hofken, H. & Schuba, M., 2013. Malware proof on mobile phone exhibits based on GSM/GPRS traces. pp. 89-96.
- [9] CVS, 2006. CVS - Concurrent Versions System. [Online] Available at: <http://www.nongnu.org/cvs/>
- [10] Subversion, 2014. Apache Subversion. [Online] Available at: <https://www.subversion.apache.org/>
- [11] GIT, 2014. GIT - Free and open source distributed version control system. [Online] Available at: <http://www.git-scm.com/>
- [12] Mercurial, 2014. Mercurial - Distributed source control management tool. [Online] Available at: <http://www.mercurial.selenic.com/>
- [13] MantisBT, 2014. Mantis Bug Tracker - Open source issue tracker. [Online] Available at: <http://www.mantisbt.org>
- [14] Trac, 2012. Trac - Open Source Project. [Online] Available at: <http://www.trac.edgewall.org/>
- [15] Jira, 2014. Jira - Tracker for teams planning and products. [Online] Available at: <http://www.atlassian.com/software/jira>
- [16] Podio, 2014. Podio Software. [Online] Available at: <http://www.podio.com>
- [17] GM, 2014. GlobalMeet – Web Conferencing Services. [Online] Available at: <http://www.globalmeet.com>
- [18] Redmine, 2015, Flexible Project Management, <http://www.redmine.org>.
- [19] Fraunhofer FOKUS OpenEPC toolkit, <http://www.openepc.net/>
- [20] 3GPP TS 36.413: "Evolved Universal Terrestrial Access Network (E-UTRAN); S1 Application Protocol (S1AP)".
- [21] 3GPP TS 24.301: "Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3.
- [22] IETF RFC 4960: "Stream Control Transmission Protocol".

- [23] Marin-Lopez, R.; Bernal-Hidalgo, F.; Das, S.; Lidong Chen; Ohba, Y., "A new Standard for securing media independent handover: IEEE 802.21A," Wireless Communications, IEEE, vol.20, no.6, pp.82,90, December 2013.
- [24] LIME – Linux Memory Extractor, 2015, <https://github.com/504ensicsLabs/LiME>
- [25] The Volatility Framework, 2015, <https://code.google.com/p/volatility/>
- [26] SQLite, 2015, <http://www.sqlite.org/>
- [27] AFLogical OSE, 2015, <https://github.com/viaforensics/android-forensics>
- [28] Guido, M., Ondricek, J., Grover, J., Wilburn, D., Nguyen, T., Hunt, A., 2013. Automated identification of installed malicious android applications. Digital Investigation 10, Supplement (0), S96 – S104, Proceedings of Thirteenth Annual (DFRWS) Conference.
- [29] MySQL, 2015. <http://www.mysql.com/>
- [30] Mobile HTML 5, 2015, <http://mobilehtml5.org/>
- [31] UMIP, UMIP - Mobile IPv6 and NEMO for Linux, <http://www.umip.org/>,
- [32] Open Air Interface, Open air interface proxy mobile IPv6 (OAI PMIPv6), <http://www.openairinterface.org/openairinterface-proxy-mobile-ipv6-oai-pmipv6>
- [33] Schmidt, T and Waehlich, M and Krishnan, S; Base deployment for multicast listener support in Proxy Mobile IPv6 (PMIPv6) Domains; IETF RFC 6224, April 2011.
- [34] Panaousis, Emmanouil A., Ramrekha, Tipu A., Millar, Grant P. and Politis, Christos (2010) Adaptive and secure routing protocol for emergency mobile Ad-Hoc networks. International Journal of Wireless and Mobile Computing, 2(2), pp. 62-78.
- [35] SALUS Deliverable 2.4, "User Requirements Definition – Final", December 2014.
- [36] SALUS Deliverable 5.2, "PPDR Security Architecture, end-to-end security, privacy mechanisms and intrusion detection approach - Intermediate", January 2015
- [37] SALUS, Deliverables 6.1, "Connectivity, QoS provisioning and Seamless Mobility for Hetnets – Intermediate", November 2014.
- [38] R. Housley et al, "RFC 3280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", April 2002
- [39] J. Rosenberg et al, "RFC 2543: SIP: Session Initiation Protocol", June 2002.
- [40] OpenVPN Technologies Inc., "OpenVPN" [Online] <https://openvpn.net/2015>.
- [41] ZeroMQ Code-Connected, [Online] <http://zeromq.org/>, 2015.
- [42] Bouncy Castle Crpto APIs, [Online] <https://www.bouncycastle.org/>, 2015
- [43] PostgreSQL, [Online] <http://www.postgresql.org/>, 2015
- [44] Asterisk, [Online] <http://www.asterisk.org/>, 2015
- [45] nProbe, [Online] <http://www.ntop.org/products/nprobe/>, 2015
- [46] NfSen, [Online] <http://nfsen.sourceforge.net/>, 2015
- [47] SSHCure, [Online] <https://github.com/sshcure/sshcure/>, 2015
- [48] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), IETF, October 2004. [Online]. Available: <http://www.ietf.org/rfc/rfc3954.txt>
- [49] IPFIX: B. Claise, B. Trammell, P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011 (Internet Standard), IETF, September 2013. [Online]. Available: <http://www.ietf.org/rfc/rfc7011.txt>.
- [50] BioSignalsPlux, Biosignals wearable body sensing platform [Online]. Available: <http://www.biosignalsplux.com/>.

- [51] Google, Protocol Buffers – Google’s data interchange format, [Online]. Available: <https://code.google.com/p/protobuf/>.

ACRONYMS

#	
3GPP	3 rd Generation Partnership Project
A	
AAuC	SALUS Authentication and Authorization Centre
AIE	Air Interface Encryption
AG-R	Access Gate Radio
AL	Ambience Listening
ANGW	Access Network Gateway
AP	Access Point
API	Application Programming Interface
B	
BAN	Body Area Networks
BBU	Base Band Unit
BIC	Barring of Incoming Call
BOC	Barring of Outgoing Call
BON	Back Office Node
C	
CA	Certification Authority
CAD	Call Authorised by Dispatcher
CCAPI	Command Control Application Programming Interface
CCC	Command and Control Centre
CCM	Core Control Module
CDR	Call Detail Records
CEM	Chanel Element Module
CFB	Call Forwarding on Busy
CFNRe	Call Forwarding on Not Reachable
CFNRy	Call Forwarding on No Reply
CFU	Call Forwarding Unconditional
CI	Call Identification
CML	ChaMeLeon Routing Protocol
CMS	Central Management System
CPRI	Common Public Radio Interface
CRL	Certificate Revocation List
CT	Call Transfer
CVDP	Critical Voice and Data Protocol
D	
DAHDI	Digium Asterisk Hardware Device Interface

DAP	Dispatch Application Platform
DGNA	Dynamic Group Number Assignment
DL	Discreet Listening
DMO	Direct Mode Operation
DSCP	Differentiated Services Code Point
E	
EHCP	Enhanced Host Configuration Protocol
E-UTRAN	Evolved UMTS Terrestrial Radio Access Network
E2EE	End to End Encryption
F	
FXO	Foreign Exchange Office
FXS	Foreign Exchange Station
G	
GPRS	General Packet Radio Service
GPS	Global Position System
GTP	GRPS Tunnelling Protocol
H	
HIDS	Host Intrusion Detection System
HSS	Home Subscriber Server
HSSL	High Speed Serial Link
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
I	
IC	Include Call
IDR	Independent Digital Repeater
IDS	Intrusion Detection System
IMS	IP Multimedia System
ISDN	Integrated Services Digital Network
IVR	Interactive Voice Response
J	
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
L	
LDS	Line Dispatch Station
LE	Late Entry

LMA	Local Mobility Anchor
LMC	LTE Micro Core
LTE	Long Term Evolution
M	
MAG	Mobility Access Gateway
MANET	Mobile Ad-Hoc Network
MCPTTtoLTE	Mission-Critical Push To Talk over LTE
MDA	Multifunction Digital Adapter
MF	Mobile Forensics
MGCS	Mobile Ground Control Station
MIH	Media Independent Handover
MME	Mobility Management Entity
MN	Mobile Node
MS	Mobile Stations
N	
NAS	Non-Access Stratum
NMAG	New MAG
NMS	Network Management Station
O	
OS	Operating System
P	
PC	Priority Call
PCRF	Policy and Charging Rules Function
PDN-GW	Packet Data Network Gateway
PFMIPv6	Proxy Fast Mobile IPv6
PKI	Public Key Infrastructure
PMAG	Previous MAG
PMR	Personal Mobile Radio
PPC	Pre-emptive Priority Call
PPDR	Public Protection and Disaster Relief
PSTN	Public Switch Telephone Network
PTT	Push to Talk
Q	
QoS	Quality of Service
R	
RRH	Remote Radio Head
RRM	Radio Resource Management

S	
SAM	Service Aware Manager
SALUS	Security and Interoperability in Next Generation PPDR Communication InfrastructureS
SCTP	Stream Control Transport Protocol
SFP	Small Form-factor Pluggable
SIP	Session Initiation Protocol
SGSN	Service GPRS Support Node
SMB	Sub-Miniature B Connector
SMS	Short Message Service
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SSID	Service Set Identifier
SSL	Secure Socket Layer
T	
TETRA	Terrestrial Trunked Radio
TIC	Telephony Interface Card
TIGv2	TetraNode IP Gateway version 2
TLS	Transport Layer Security
TPI	Talking Party Identification
TRDU	Transmit/Receive/Duplexer Unit
U	
UAV	Unmanned Aerial Vehicle
UDM	USB Dispatch Microphone
UE	User Equipment
UGV	Unmanned Ground Vehicle
UMTS	Universal Mobile Telecommunication System
UUID	Unique User Identifier
V	
VPN	Virtual Private Network
VMS	Video Management System
X	
XML	Extensible Markup Language
W	
WP	Work Package
Z	
ZMQ	Zero Message Queue

ANNEX A – INTEGRATION REPORT

This annex describes a model that can be used to report the status of the integration of two components.

Test case ID	# Identification of Test Case (number)		
Test Items	Components being integrated		
Description	Description of the test integration		
Input Specification	How integration is triggered? For instance how component A calls component B (e.g. via Restfull APIs)		
Output Specification	Achieved output.		
Environmental Needs	Conditions to perform integration. Which services should be running.		
Test Step Details			
Step N	Details	Input Data	Expected Result
# Identification of step	Components	What is the data that is exchanged	The expected result in the integration process.
Integration Issues			
Issue N	Description	Possible Solution	
# Identification of issue	Description of issue, when it occurs	Possible solutions to solve the issue.	

ANNEX B – OVERALL PROTOTYPE

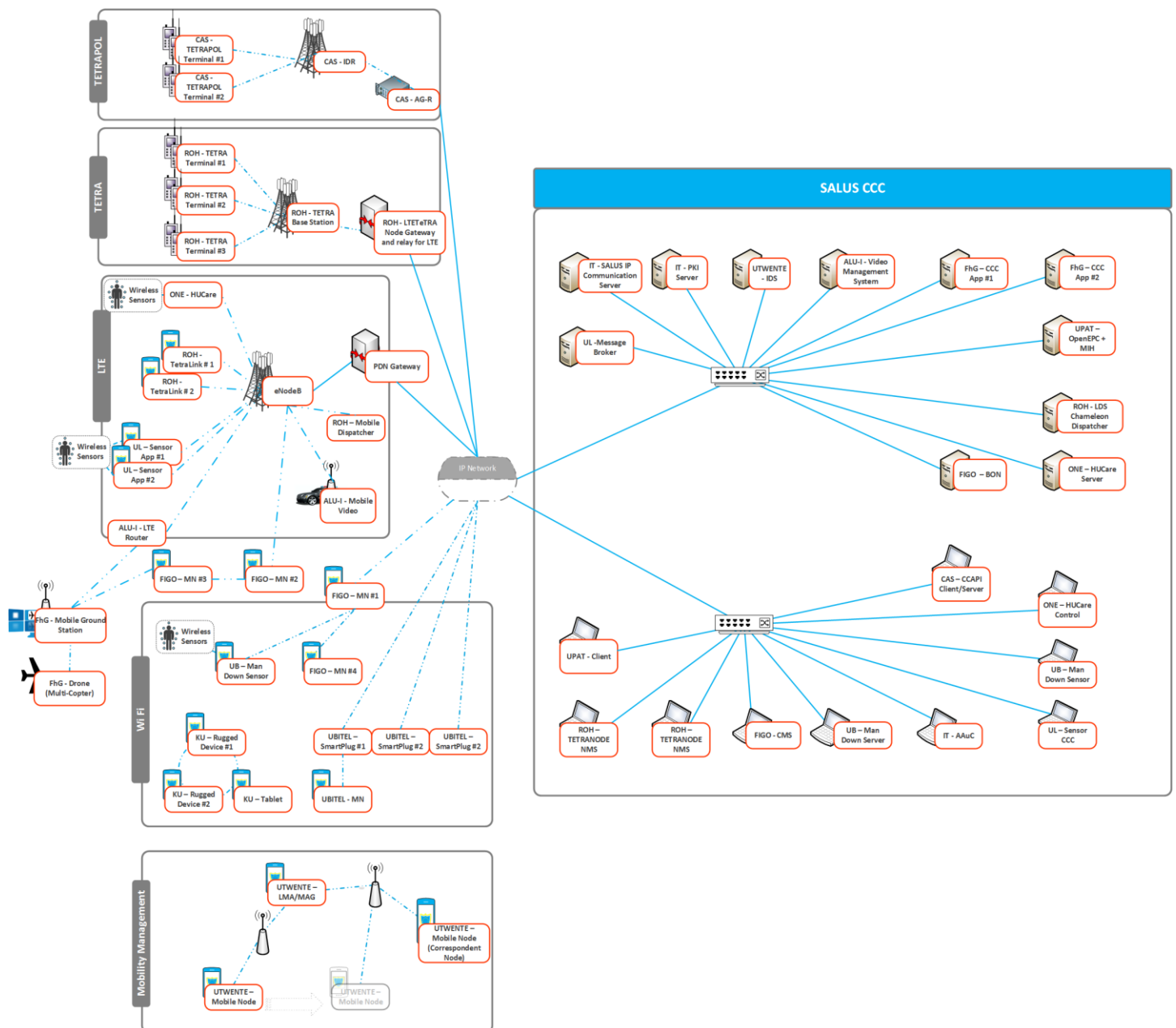


Figure 59 – Overall Prototype