



CONNECARE

WP5 – EVOLUTIONARY INTEGRATION

D5.3: FINAL RELEASE OF THE GENERIC CONNECARE SYSTEM

H2020-EU.3.1: Personalised Connected Care for Complex Chronic Patients

Project No. 689802

Start date of project: 01-04-2016

Duration: 45 months

Project funded by the European Commission, call H2020 – PHC - 2015	
PU	Public
PP	Restricted to other programme participants (including the Commission Services)
RE	Restricted to a group specified by the consortium (including the Commission Services)
✓ CO	Confidential, only for members of the consortium (including the Commission Services)

Revision: 01

Date: 23-12-2019



Document Information

Project Number	689802	Acronym	CONNECARE
Full title	Personalised Connected Care for Complex Chronic Patients		
Project URL	http://www.CONNECARE.eu		
Project officer	Birgit Morlion		

Deliverable	Number	3	Title	Final Release of the generic CONNECARE system
Work Package	Number	5	Title	Evolutionary Integration

Date of delivery	Contractual	MONTH 45	Actual	MONTH 45
Nature	Prototype <input checked="" type="checkbox"/> Report <input type="checkbox"/> Dissemination <input type="checkbox"/> Other <input type="checkbox"/>			
Dissemination Level	Public <input type="checkbox"/> Consortium <input checked="" type="checkbox"/>			

Responsible Author	Eloisa Vargiu	Email	eloisa.vargiu@eurecat.org
Partner	EURECAT	Phone	+34 932 381 400
Participants	Matti Karagach (eWAVE)		

Abstract	In this document, the final release of the generic CONNECARE system is presented. Its customization to each site of the project is documented in deliverables D5.5, D5.7, and D5.9.
-----------------	---

This deliverable reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains. (Art. 29.5)

CONNECARE project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 689802 (Art. 29.4)



Table of contents

EXECUTIVE SUMMARY	4
1. THE GENERIC CONNECARE SYSTEM	7
1.1 ARCHITECTURAL OVERVIEW.....	7
1.2 UPDATED REQUIREMENTS.....	8
1.3 MAIN COMPONENTS.....	9
1.3.1 SACM.....	10
1.3.2 SMS.....	10
1.3.3 Authentication manager.....	11
1.3.4 Queue Manager.....	11
1.3.5 Digital Health Framework / Patient Information Adapter.....	11
1.3.6 Access Control and Data Security.....	12
2. DEPLOYMENT OF THE FINAL RELEASE	13
3. TECHNICAL MANAGEMENT	14
4. PENETRATION TEST	18
4.1 INITIAL PENETRATION TEST.....	18
4.2 FINAL PENETRATION TEST.....	19
5. CUSTOMIZATION TO THE 4 SITES AT A GLANCE	21
6. CONCLUSIONS AND A VISION TO THE FUTURE	23
7. REFERENCES	26
8. ANNEXES	27
8.1 ANNEX 1.....	27
8.2 ANNEX 2.....	46



Executive Summary

WP5 was aimed at defining and implementing the overall CONNECARE system according to an evolutionary approach. According to the co-design approach defined and implemented in WP2 that relies on several cycles and iterations, also the integration phase was evolutionary and done in accordance with requirements changes and feedback from the implementation studies.

This deliverable documents the final release of the CONNECARE system that take into account needs and requirements of the project and that is the main ICT outcome of CONNECARE.

According to the CONNECARE philosophy and for the sake of scalability and transferability, the overall CONNECARE system has been defined and developed as a generic system, totally independent of the specificity of the 4 sites. Following this approach, the CONNECARE system has been, then, customized according to the specific requirements of each of the sites. In fact, the real integration is strictly dependent on the actual situation of each of the involved sites. Moreover, being the integration site-dependent, different solutions have been taken in each site. Thus, the integration is separately processed for each site and only the partner from that site has been involved; deliverable D5.5 “Final Release of the Catalan CONNECARE system”, D5.7 “Final Release of the Israeli CONNECARE system”, and D5.9 “Final Release of the Groningen CONNECARE system” present those customizations.

The rest of the deliverable is organized as follow. In Section 1, the generic architecture of the CONNECARE system is presented together with the updated requirements and its main components. Let us note that each component is described in a specific deliverable: the DHF in the D5.1 “Collaborative digital health framework” (submitted at M6, September 2016); the final SACM in deliverable D3.6 “Final Smart Adaptive Case Management” (submitted at M43); and the final SMS in the deliverable D4.7 “Final Self-Management System” (to be submitted at M45). For the sake of completeness, this deliverable briefly recalls each subsystem; for a detailed description of each, please refer to the corresponding document. Section 2 lists the deployment characteristics of the production environment, whereas Section 3 focuses on technical details on the deployment of the final release. In Section 4, the results and taken actions done at security level with the penetration test performed first by EURECAT and subsequently by an external company in Israel are reported. Section 5 sketches the customization done in the 4 sites. Finally, Section 6 ends the document with conclusions and a vision to the future.

Overall, the work summarized in this document is based on the work made by EURECAT and eWAVE in WP5, together with the support of all technical partners and the supervision of the clinical partners (especially from WP2). The work presented in this deliverable is strictly related with the overall work made in WP5 but also to work made in WP3 and WP4. It updates, advances, and improves the deliverable D5.2 “Study Release of the generic CONNECARE system”, submitted at M18. Moreover, these previous deliverables are highly recommended to be read:



Number	Title	Description
D2.1	Cook-book for project development	The document provides an overall view of the CONNECARE project, and describes the procedures for its development. The deliverable indicates the different phases of the project, with an emphasis on how PDSA cycles will be structured. Overall, the CONNECARE project does not aim at a rigid integrated care solution that needs to be adopted by all potential deployment sites but to a flexible solution that has high potential for generalization at the EU level. In this sense, innovative methodologies involving both global and local stakeholders have been adopted.
D2.5	PDSA process and final design of the CONNEARE system	This deliverable provides a complete view of the Plan Do Study Act (PDSA) methodology used through-out the project, including the main objectives, methods and outcomes for each cycle and how this iterative strategy allowed to shape the CONNECARE system. Moreover, it provides a summary of the final design of the system, with a focus on the functional and non-functional requirements that fostered the development and improvement of the system and how these requirements were tackled.
D3.6	Final Smart Adaptive Case Management System	This deliverable goes with the final release of the Smart Adaptive Case Management system (SACM) by TUM and ADI, integrated to the SMS by EURECAT and the contribution of UNIMORE for the clinical decision support systems.
D4.7	Final Self-Management System	This deliverable goes with the final release of the Self-Management System (SMS) by EURECAT and the contribution of UNIMORE for the recommender system.
D5.1	Collaborative Digital Health Framework	This deliverable describes the collaborative DHF that includes the interoperability model and the communication protocols.
D8.13	Exploitation plan	This deliverable describes how the CONNECARE consortium proposes to exploit the connected care service after completion of the project

Finally, the deliverables on customization and integration in each site, which will be submitted at M45 (December 2019) as the current one, are recommended to be read:

Number	Title	Description
D5.5	Final Release of the Catalan CONNECARE system	The deliverable described the customization and integration of the generic CONNECARE system at Catalan level and to the Hospital Clínic in Barcelona. This deliverable extends the preliminary work documented on D5.4 "Study Release of the Catalan CONNECARE system". The release of the final generic CONNECARE system described in D5.3.
D5.7	Final Release of the Israeli CONNECARE system	The deliverable described the customization and integration of the generic CONNECARE system to Israel. The release of the final generic CONNECARE system described in D5.3.



D5.9	Final Release of the Groningen CONNECARE system	The deliverable described the customization and integration of the generic CONNECARE system to Groningen. The release of the final generic CONNECARE system described in D5.3.
------	---	---



1. The Generic CONNECARE System

The aim of the CONNECARE project is to co-design, develop, deploy, and evaluate a novel, smart, adaptive integrated care system for chronic care management. The purpose of this is to reduce costs and improve patient outcomes by improving the integration of long term care for those chronically sick with more than one long term condition. This section describes the final release (Final Release) of the overall CONNECARE system.

1.1 Architectural Overview

The CONNECARE system is a federation of subsystems each devoted to provide a set of goal-oriented functionalities, whose main components are the Self-Management System (SMS) and the Smart Adaptive Case Management system (SACM). Based on the concept of microservices, the SMS provides intelligent tools to monitor patients (i.e., physical activity, sleeping, health status, drug adherence, simple rehabilitation tasks, and self-checked questionnaires) and to autonomously interact with them through engagement, rewards, and warnings through a recommender system. The SACM has extended functionalities for case modelling and execution, specifically tailored to the healthcare domain. Additionally, the SACM includes a Decision Support System to show patients in a map and to create routes for better organizing visits. The SMS and SACM interact each other through the CONNECARE Queue Manager which connects both subsystems and orchestrates their communication and provides an integration framework to link CONNECARE services to specific Electronic Health Records (EHR) and regional Personal Health Folders (PHF) in each site [1]. Figure 1 sketches the architecture of the final CONNECARE system.

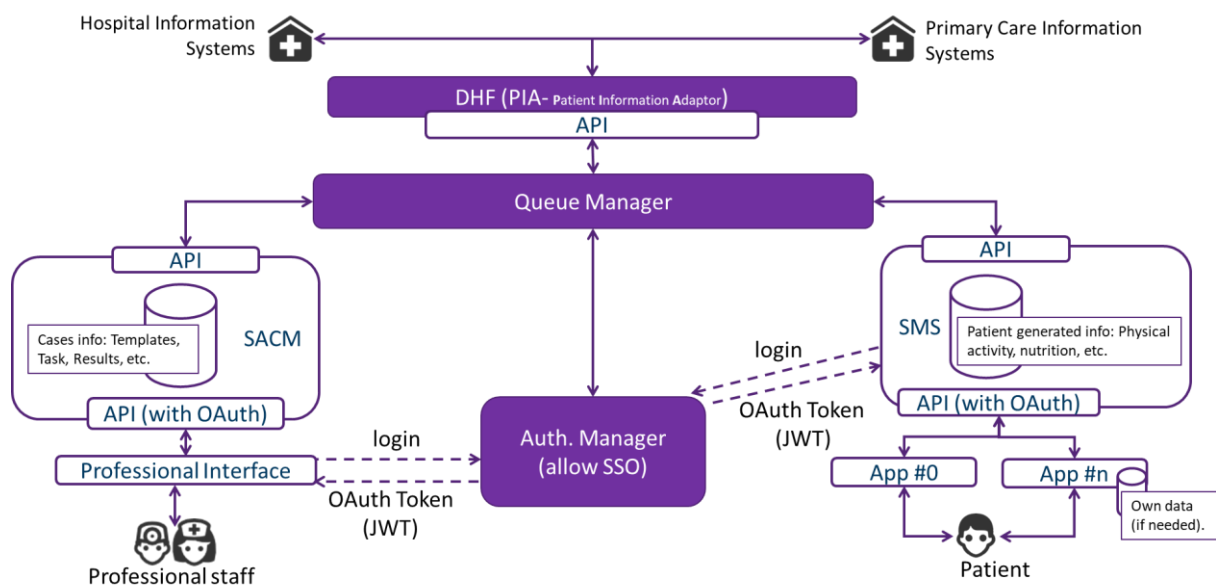


Figure 1 - The architecture of the final release of the CONNECARE system.

Summarizing, the final release of the CONNECARE system contains the following subsystems:



- **SACM** – Smart Adaptive Case Management system, that includes the **mapping DSS**
- **SMS** – Self-Management System, that includes the **recommender system**
- **Authentication Manager** – user management
- **Queue Manager** – message broker
- **DHF** – Digital Health Framework / **PIA** – Patient Information adapter

Figure 2 sketched the architecture zooming in the SACM and SMS to give the full picture.

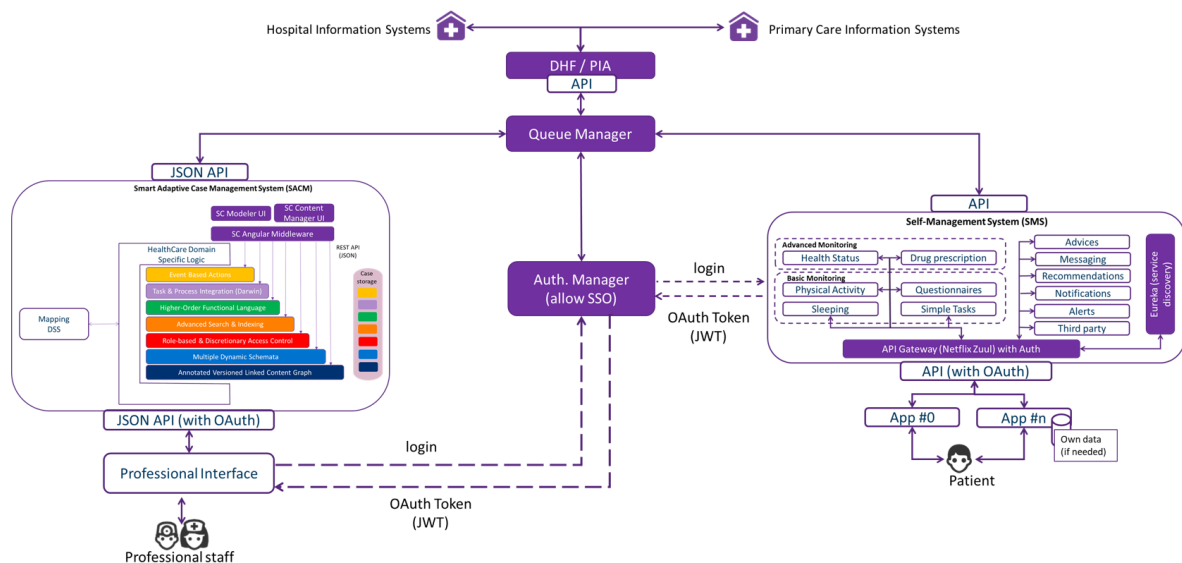


Figure 2 - Final architecture with details on the main subsystems: SACM and SMS.

1.2 Updated Requirements

The CONNECARE system requirements are a set of requirements from all the systems implied in it. During the project, requirements have been continuously updated according to the iterative approach and the evolution of the PDSA cycles.

The initial functional and non-functional requirements are listed in the deliverable D5.2. Here, we report those related to the last 2 periods of the project. In D3.6 “Final Smart Adaptive Case Management System” and D4.7 “Final Self-Management System” the full list of requirements and updates made after the Study Release as feedback from the implementation studies have been deeply reported. For the sake of completeness, they are listed below:

- **SACM**
 - case team management;
 - show the generated linked-data structure of a case with a hierarchical representation;
 - simplify third party integration;



- support correcting tasks at any time even after task completion;
 - allow to add work-plan tasks multiple times and even execute them in parallel;
 - support auto-check questionnaires;
 - support exporting case instances for scientific data analytics and evaluation.
- SMS
 - integrated activity trackers;
 - simple rehabilitation tasks;
 - profile picture in the setting screen;
 - advices with external links and videos;
 - media auto-download.

1.3 Main Components

As already said, the CONNECARE system consists of a federation of subsystems. Thus, one of the critical points is the synchronization of the data and the responsibility to keep the coherence of the information into the system. For this reason, the core systems are responsible to share this information and coordinate the subsystem. Table 1 shows the responsible of the information related to the users: authentication, authorization, and management.

Table 1 - User's responsibilities.

	Auth Management	SACM	SMS	MAPPING DSS
Authentication	Master <i>token creation</i>	Slave <i>token validation</i>	Slave <i>validates token</i>	Slave <i>validates token</i>
User Management (CRUD Operations)	Master <i>user basic fields</i>	Slave <i>basic user fields and additional user fields</i>	Slave	Slave
User Role Management	Master	Slave	Slave	Slave
Authorization	Master	Master	Master	Master

Let us note that, even if it is fully integrated in the SACM, in the table we considered separated the Mapping DSS as a CONNECARE subsystem. This is due to the evolution it had at integration time. Initially it was expected to be a presentation layer of the SACM. Nevertheless, considering its usefulness, clinicians changed the requirements. Thus, to provide all the required functionalities, the Mapping DSS has its own backend and can be considered as a further CONNECARE subsystem.



1.3.1 SACM

The SACM is responsible to manage the clinical process in an adaptive way. Let us recall here its main functionalities:

- Manage users (creating/editing);
- Add a new case;
- Access to the list of my cases;
- Check the summary of a case;
- Manage the stages of a process: Case Identification, Case Evaluation, Workplan definition, Workplan execution, and Discharge;
- Manage the tasks of each stage;
- Access and review the data of a case;
- Manage the team;
- Read and accept notifications;
- Send/receive messages from the team members;
- Send/receive messages from the patients;
- Write/read notes to/from the rest of the team.

The whole SACM functionality and architecture can be found in D3.6 “Final smart adaptive case management system”.

1.3.2 SMS

The SMS is responsible to give support to patients and carers for empowerment and engagement. Let us recall here its main functionalities:

- Monitor physical activity (steps and level of activity) and health measurements (i.e., blood pressure, temperature, weight, heart rate, and oxygen saturation);
- Manually add health measurements;
- Perform tasks and follow-up them: questionnaires; simple rehabilitation tasks, drugs intake and adherence;
- Consult advices automatically generated or sent by professionals and personalised for the given patient;
- Send/receive messages from the team of professionals in charge;
- Read and accept notifications;
- Set up the app: language, notifications, configure the devices, manage the profile picture, manage auto-downloading of received content.

The whole SMS functionality and architecture can be found in the “D4.7 “Final smart adaptive case management system”.



1.3.3 Authentication manager

The authentication manager is responsible to manage: the CONNECARE users; their role in the CONNECARE system; relationships between users; and the rights of each user (i.e., which professionals can access to which patients and what data can be accessed by the professionals). Its main functionalities are: single sign on (SSO), creation and management of the authentication tokens; and login / logout to the system.

The authentication manager did not change with respect to the implementation of the Study Release. The full description is given in the deliverable D5.2 “Study Release of the generic CONNECARE system”.

1.3.4 Queue Manager

The Queue Manager acts as a message broker and is responsible to interconnect the two main subsystems of CONNECARE (SACM and SMS) with the DHF (also called, PIA) that is the connector to the external system (e.g., hospital information systems, electronic health record). Its main functionalities are: manage a queue of messages for each system being fully transparent and guarantee asynchronous communication between SACM and SMS.

1.3.5 Digital Health Framework / Patient Information Adapter

The Digital Health Framework (DHF) presented in the D5.1 “Collaborative digital health framework”, submitted at the very beginning of the project (M6), evolved according to the requirements of each of the involved sites and it is now also called Patient Information Adapter (PIA).

The PIA is responsible to connect the clinical information systems to the CONNECARE system and transfer patient’s data to CONNECARE. It transforms messages to/from an external format (e.g., HL7) from/to the one adopted in CONNECARE and the adopted protocol (e.g., C-CDA) to REST. Working at data level, the PIA is aimed at adapting the data from each kind of protocol and system to one generic protocol of the CONNECARE API. In so doing, it is generic and may be adapted to any clinical information system and site.

Figure 3 sketches the PIA and its interactions with the rest of the CONNECARE system.

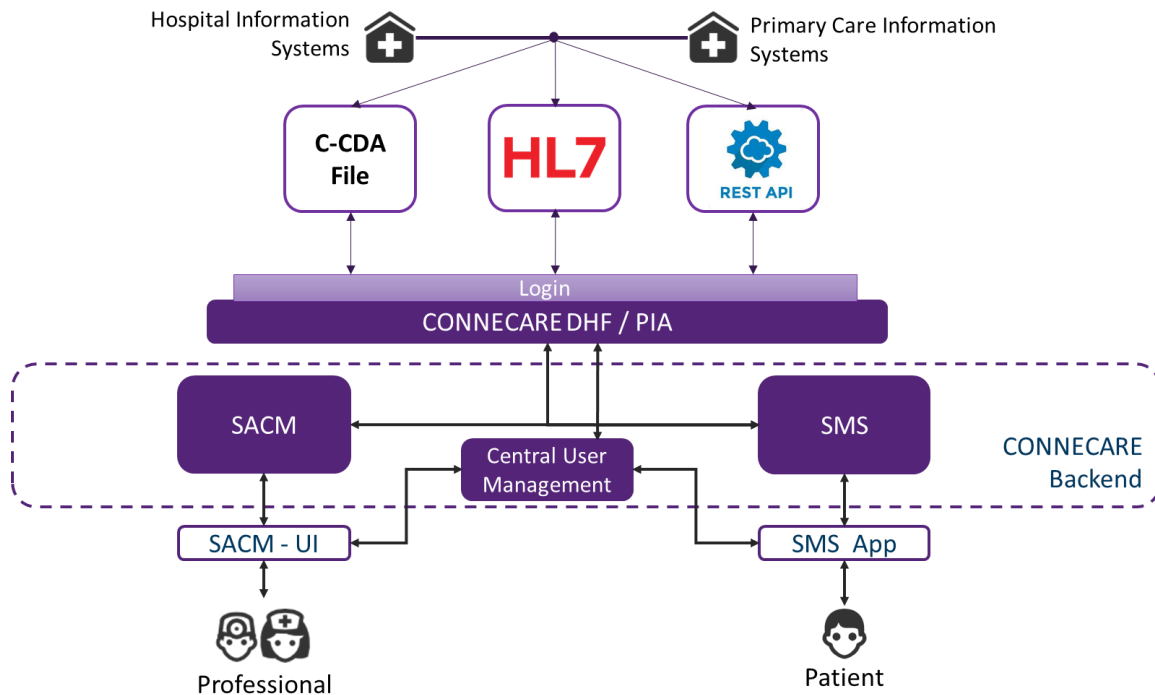


Figure 3 - The PIA and its interaction with the CONNECARE system.

1.3.6 Access Control and Data Security

The access control and data security did not change with respect to the implementation of the Study Release. Their full description is given in the deliverable D5.2 “Study Release of the generic CONNECARE system”.



2. Deployment of the Final Release

As reported in D5.2 “Study Release of the generic CONNECARE system”, two instances in Amazon Central Europe have been reserved. Table 2 details the Amazon environments contracted at the beginning of the project.

Table 2 - Initial configuration of the Amazon environments

	Integration Environment	Production Environment
Provider	Amazon (Europe Central)	Amazon (Europe Central)
Type of instance	m4.xlarge	m4.xlarge
RAM	16 GB	16 GB
HDD Space	100 GB (EBS Volume)	100 GB (EBS Volume)
SO	Ubuntu 16.04.1 LTS	Ubuntu 16.04.1 LTS
Domain	test.connecare.eu	system.connecare.eu
Open ports	22, 80, 443, 8084, 8085 (to Internet)	443 (to internet) 22 (to selected partners)

Due to the incorporation of new components (as for instance the PIA in Groningen and the integration of mapping DSS with its own backend), serious performance problems have been experienced in the production environment. Sometimes it caused collapses at specific times of the day and the system stopped work. All the incidences have been immediately “manually” resolved by the EURECAT team (i.e., restarting the servers and the services). To avoid this kind of problems, EURECAT decided to upgrade the Amazon production environment, the new characteristics are listed in Table 3.

Table 3 - New characteristics of the production environment

	Production Environment
Provider	Amazon (Europe Central)
Type of instance	m4.2xlarge
RAM	32 GB
HDD Space	120 GB (EBS Volume)
SO	Ubuntu 16.04.1 LTS
Domain	system.connecare.eu
Open ports	443 (to internet) 22 (to selected partners)

After the server update, the performance issues in the production environment were reduced in a very important way and the system did not stop again.



3. Technical Management

One of the main goals of WP5 was the collaboration between the different technical teams (i.e., EURECAT, TUM, ADI, UNIMORE, and eWAVE) and specificities of the sites in order to deliver the final CONNECARE system. After the 1st review meeting, to improve the management of the whole project (versions, cycles, and list of actions), eWAVE started working as a Product Owner of the overall CONNECARE system. The tasks, user stories, and bugs are reported on the unique JIRA project log that eWAVE opened for the project. The first workflow was defined on 16/1/18 and since then, CONNECARE technical teams have worked accordingly. eWAVE led the maintenance of the technical aspects in the project implementation log file. Each new bug was reviewed by eWAVE and opened in the JIRA if needed. Furthermore, eWAVE led the "end-to-End" QA cycles as part of the evolutionary integration task.

To better follow-up the project, a weekly meeting was scheduled every Wednesday at 10:30 (CEST) held through gotomeeting, participated by eWAVE, EURECAT, TUM, and ADI from the beginning¹ and joined also by UNIMORE when the integration of the intelligent tools (recommender system and mapping DSS) started. Each meeting was aimed at reviewing the current status of the work, rising the open issues, and solved technical issues with the collaboration of all the partners. After every meeting, the product owner distributed a meeting summary including also the tracked issues. Figure 4 shows an example of the meeting held on December 6th, 2018.

The open issues are:

- CON-566 and CON-565 - **Fixed and working fine**
 - Patient questionnaires – **few new bugs were opened:**
 - Should be checked again by **Mauricio** with the specific case ID that Ariel sent,
 - Also **Matti** will test again with the new APK that **Eloisa** will send
- Drugs and advices – **new issue for Jak was opened** – should be fixed and delivered today, **Jak** please update us

Final release date has not changed:

Version Number/scope	Description	Re Release to TEST	Release to PROD	Responsible	Comments
4.3	1. Manual Advices 2. Drugs prescriptions 3. Bugs fixes 4. Adding new "patients questionnaires" : For Assuta: <ul style="list-style-type: none"> • 4 simple questions For UMCG CS2: <ul style="list-style-type: none"> • NPS (English + Dutch) • SUS (English + Dutch) • VAS • Autocheck health status For UMCG CS: <ul style="list-style-type: none"> • CCQ (Dutch Only) • ACQ (Dutch Only) For Lleida – already done	22/11/18 5/12/18	? 19/12/18	Felix + Mauricio + Jak+ Matti	SACM Backend is ready in different branch SMS advices UI infrastructure is ready backend of the SMS is ready. CON-566 - will be ready by Jak till the 29/11/2018 - Jak CON-565 - will be ready by Jak till the 29/11/2018 – Jak CON-585 – will be ready by Jak till the 5/12/2018 – Jak CON-578 – Kitiara CON-590 – Jak

Figure 4 - Example of summary of a technical meeting

¹ IPHEALTH also participated until their termination.

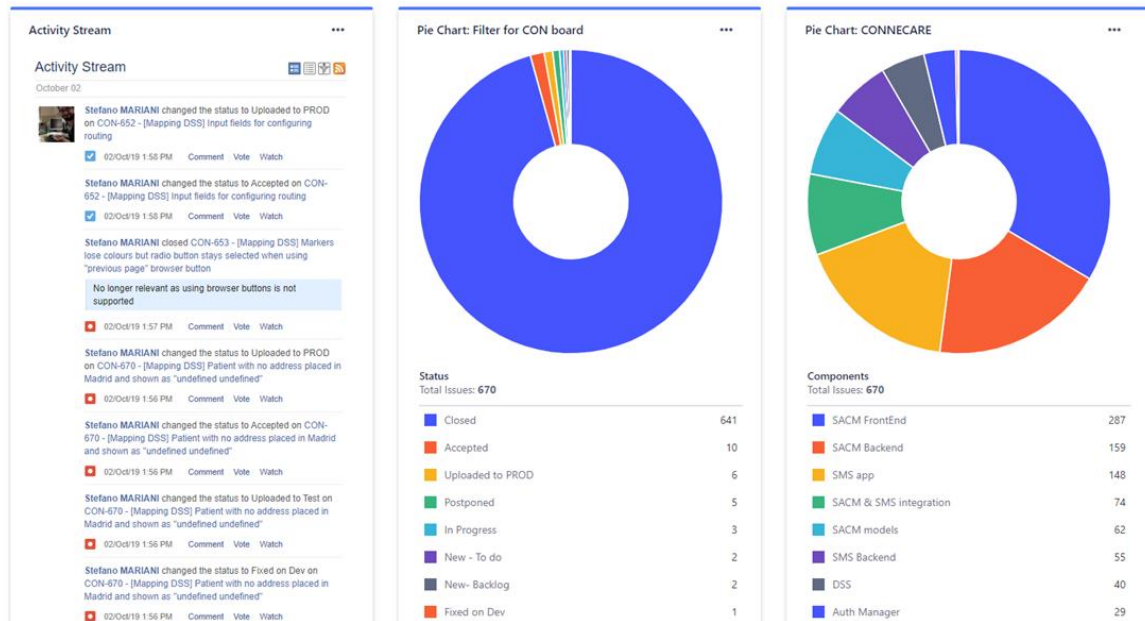


Figure 5 - CONNECARE JIRA dashboard

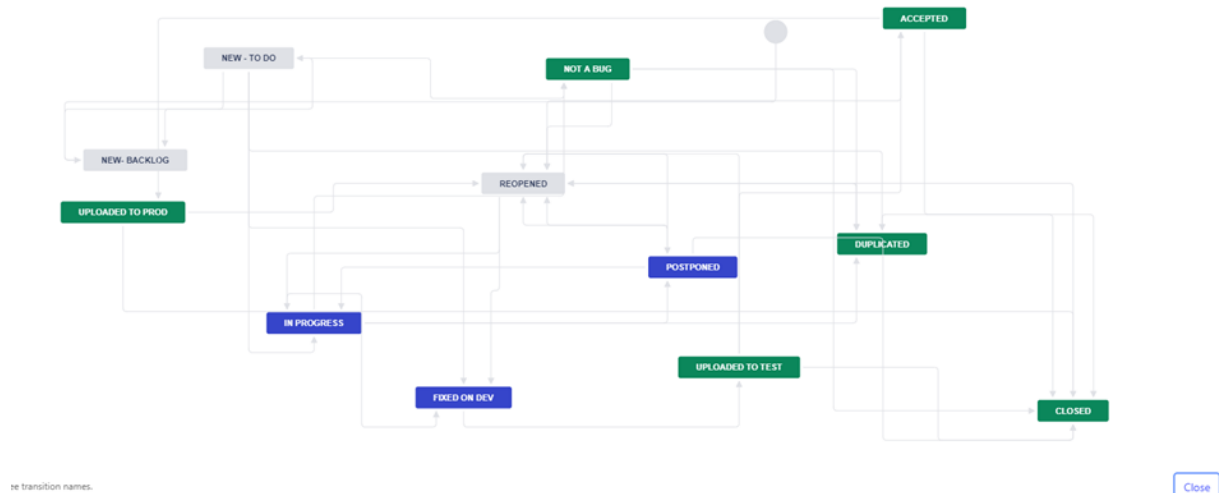


Figure 6 - CONNECARE JIRA workflow

To make more agile the coordination and management of the overall development we worked with JIRA as our main tasks, bugs and versions management tool:

- We opened a CONNECARE project in the JIRA;
- We defined roles and permissions to each technical partner;
- We defined our specific workflow in the JIRA;
- We defined epics, tasks, and bugs;
- We defined sprints and versions and assign ticket to each sprint;



- We managed and set our priorities to the open issues.

Figure 5 shows the main dashboard of the CONNECARE project in the JIRA, Figure 6 sketches the workflow, and Figure 7 the final list of issues by component. Let us note that for both SACM and SMS the backend and the frontend have been considered separately. Moreover, the DSS (that refers to the mapping DSS) and the recommender system appear separated to the corresponding subsystem that integrate them (SACM and SMS, respectively) because the JIRA issues refer to the integration tasks.

The number of issues for each component are not necessarily pointing at number of bugs but they are the result of number of change requests, technical complications, and process of the product development for each component.

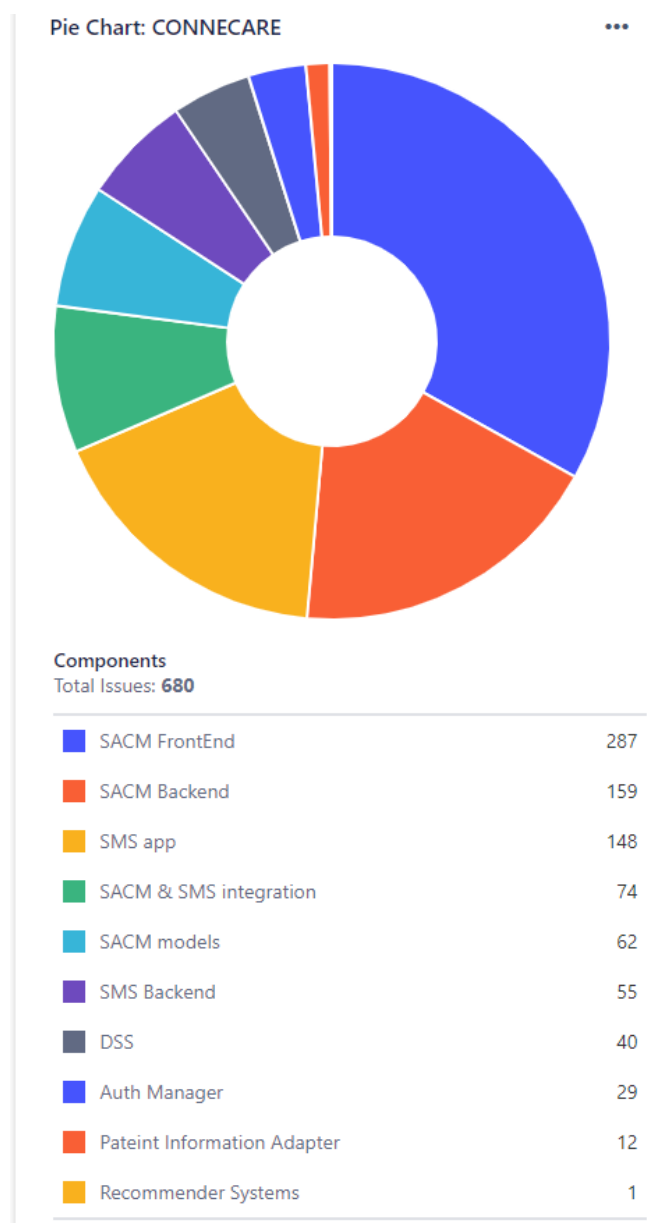


Figure 7 - Final list of issues by component



In order to get feedbacks from the clinical partners for each new version or feature that was released to production, Google Document was opened for bugs reporting² (Figure 8 shows an extract):

- Bugs were reported by the clinical partners (usually the case manager) into the document;
- Critical bugs and stoppers was also accompanied with email;
- The product owner gathered the feedbacks and opened appropriate tickets in the Jira according to priority;
- Critical and high bugs were fixed and released in “hot fixes” versions.

Technical problems with SMS&SACM										
Issue Log										
Issue (Step in the process)	CS1	Assuta	Description	Priority (H, M, L)	Modeling	Reported By (role and name)	Assigned To (role and name)	Fixed	Date Resolved	Resolution/ Comments
I00011	CS1	Assuta	All automatically calculate fields (such as BMI) require that the form will be saved and re-entered and only then the calculated score can be viewed. It was not like that at first, and it's makes very difficult to work.	High	SACM	Reut Ron	Felix	Fixed		CON-538
I00012		Leida	[10099] The time of the measurements shown in the graph is not correct.	Medium	Integration	Mireia Massip	Mauricio	Fixed		CON-539
I00013		Leida	[10147] I have added a manual measure by mobile app but it does not work (null timeslot)	High	Integration	Mireia Massip	Juanma	Fixed		
I00014		Leida	[10149] The automatic alert shows in SACM is not correct	Medium	Integration	Mireia Massip	Juanma	Fixed		
I00015	CS2	UMCG	One patient used all his data (from the provider) was not connected to Wi-Fi) using the FitBit and ConneCare app, leading to an increase of his telephone company bill. I was wondering if it might be possible to install the app in a certain way that they only work using Wi-Fi (because otherwise this will lead to cost increase for the elderly if they are not able to connect to Wi-Fi everytime and they use there 4G network).	Low	Other	Mathijs Plas	Juanma	In Progress		Low priority, CON-540

Figure 8 - Extract of the Implementation Log

At the beginning of the development phase of the CONNECRAE project, the product owner together with all the technical team and the consensus by the clinical partners defined a versioning plan. Starting from that, the JIRA sprints was defined and the plan was fully delivered during the project (see Figure 9).

Version number	Date	Scope
V1	January 2018	Partly SACM working with initial SMS version
V2	March 18	<ol style="list-style-type: none"> 1. CS1 for all sites 2. Work plan 3. Summary page 4. Walking Activity + HR +Blood pressure (FITBIT)
V3	April 18	<ol style="list-style-type: none"> 1. CS1 + CS2 in all sites 2. Questionnaires 3. Monitoring prescription 4. Simple tasks
V4	May 18	<ol style="list-style-type: none"> 1. Messages 2. Notifications 3. Reminders – Push notifications
V5	July 18	<ol style="list-style-type: none"> 1. Drugs prescriptions 2. Manual Advices
V6	January 19	<ol style="list-style-type: none"> 1. Additional Questionnaires
V7	February 19	<ol style="list-style-type: none"> 1. PIA for UMCG
V8	March 19	<ol style="list-style-type: none"> 1. Recommendation System
V9	September 19	<ol style="list-style-type: none"> 1. Mapping DSS
V10	September 19	<ol style="list-style-type: none"> 1. PIA for Assuta

Figure 9 - Versioning of the CONNECARE system

² <https://docs.google.com/spreadsheets/d/13G7bnZgRYKugzOd7fvjchG9ZSLdUFmhWmcMwl5yK0GI/edit#gid=808141279>



4. Penetration Test

4.1 Initial Penetration Test

On April 2018, the Cybersecurity Unit in EURECAT performed the penetration test of the current version of the generic CONNECARE system running in the production environment.

Table 4 - Main results of the penetration test performed by EURECAT

OWASP Top 10			
	Title	Times	Criticism
1	Injection	40	Medium
2	Broken Authentication and Session Management	0	Not applicable
3	Cross-Site Scripting (XSS)	2	Medium
4	Insecure Direct Object References	0	Not applicable
5	Security Misconfiguration	0	Not applicable
6	Sensitive Data Exposure	0	Not applicable
7	Missing Function Level Access Control	0	Not applicable
8	Cross-Site Request Forgery (CSRF)	0	Not applicable
9	Using Components with Known Vulnerabilities	0	Not applicable
10	Invalidated Redirects and Forwards	0	Not applicable

Table 5 - Found vulnerabilities by the penetration test performed by EURECAT

Vulnerabilities			
	Title	Times	Criticism
1	HTML code injection	40	Medium



2	XSS	2	Medium
---	-----	---	--------

The audit found several vulnerabilities in the way the parameters supplied by the users of the application are treated. These vulnerabilities may allow to alter the behaviour of the application in a malicious way and to be a potential vector for attacks with greater impact. The summary of the results is given in Table 4. The found vulnerabilities are listed in Table 5. The full document of the penetration test performed by EURECAT is presented in the Annex 1.

All the issues with “medium” criticism have been solved by EURECAT with the support of TUM.

4.2 Final Penetration Test

On March 2019, the ASSUTA Medical Center asked for performing a penetration test to certify the security of the CONNECARE system and its adoption at the hospital. The penetration test started on May 2019 and was performed by the company BUGSEC Cybersecurity (<https://bugsec.com/>).

They found that the system security risk level was high. In particular, during the test, they found many issues with the protection of sensitive information and things that could harm application users.



Figure 10 - Summary of the findings by BUGSEC Cybersecurity

Figure 10 summarizes the main findings, Table 6 reports the issues marked of high impact. The full document delivered by BUGSEC Cybersecurity is given in the Annex 2.

EURECAT solved the issues classified as High (no Critical ones were identified), reported in Table 6, and in October 2019 the penetration test passed without any open issues, The second tests were performed by “Bugsec” a third party security company in Israel. Thus, the ASSUTA Medical Center certificated the system.



Table 6 - List of issues with high risk level

Item	Test Type	Risk Level	Component	Topic	General Explanation	Status
4.1	Applicative	High	Application	Unrestricted File Upload	The Unrestricted File Upload vulnerability describes a situation where the server does not adequately restrict the type and content of files uploaded to the server by users. An attacker could exploit this weakness in order to cause a Denial of Service or even to gain control of the server.	Vulnerable
4.2	Applicative	High	Application	Insecurely Designed Component	The Insecurely Designed Component vulnerability describes a situation where a certain component is designed or implemented in an improper way which constitutes a security risk.	Vulnerable



5. Customization to the 4 Sites at a Glance

From the very beginning, when the technical partners started gathering the requirements, many differences among the sites arose. Thus, the overall CONNECARE system has been defined and implemented in a very generic way and, at the same time, the “local” technical partners worked together with the clinical partners to customize the solution according to the specific needs:

- EURECAT worked with IRBLL and IDIBAPS;
- eWAVE worked with ASSUTA and, also, with UMCG (after the termination of IPHEALTH).

Details on the customizations are given in D5.5 (Catalonia), D5.7 (Israel), and D5.9 (Groningen), Table 7 summarizes the main differences categorized as “clinical information system integration”, “process”, “functionalities”, “third party”, and “further adopted ICT tools”.

Table 7 - Main customizations in the 4 sites

	ASSUTA	UMCG	IRBLL	IDIBAPS
Clinical information system integration	<p>In Israel the main clinical system in Assuta Hospital is the “Kameleon system” which can work with web services API to get patient data.</p> <p>The PIA can connect to the API and get patient information by ID.</p>	<p>The clinical system in UMCG is the “EPIC” system</p> <p>The only access to the patient’s information in the EPIC system was through CCD-A files that actually can be downloaded from the patient portal.</p> <p>The PIA can import CCD-A file, convert the patient information into the CONNECARE format, and send the patient’s information to the CONNECARE system.</p>	<p>The CONNECARE DHF will be connected to the Shared Electronic Health Record (HC3) defined at Catalan level.</p>	<p>The DHF is composed of the Fast Healthcare Interoperability Resources (FHIR) connected to the Hospital Information System (HIS) and a proxy server.</p> <p>Each time a new patient is introduced into the HIS her/his information passes to the proxy server. The proxy server selects the information to be sent to CONNECARE, it authenticates as a trusted user, receives a valid token, and, then, the patient is created into the CONNECARE system.</p>
Process	<p>Once the Case Evaluation stage has been done, a new Case Evaluation can be added. The process may be repeated infinitely.</p>		<p>CS2 relies on two workplan definitions: before and after the surgery</p>	<p>Case Evaluation tasks are optional.</p>
		<p>Once a questionnaire has been filled, the corresponding score is highlighted according to a traffic light criteria (green if in the given range of values, yellow if it overpasses the given range but still in a “normal” range, and red if it overpasses the range and generates an alarm).</p>		



Functionalities	Sedentary activity is monitored and a maximum of suggested minutes prescribed.		In the CS2, high level activity is not monitored.	Physical activity is monitored only in terms of number of steps.
	The Recommender System has been tested with selected patients.	The Recommender System has not been used.	The mapping is available and running.	The Recommender System has not been used.
	Simple rehabilitation tasks can be prescribed to the patient.	A graphical view of a human body is used in CS2 to show the place of the surgery.	A graphical view of a human body is used to show data from the Charlson. Moreover, in CS2 also the place of the surgery is show.	
			The notes screen in the SACM has been implemented as a "wall" to leave messages.	
Oxygen saturation is also prescribed in the SACM and measures manually added in the SMS by the patients.				
Suitable videos for educational purposes, specifically made for CONNECARE have been used as advice				
Third party	Due to CE mark problems, Withings/Nokia devices have not been used. Patients put manually the monitored measures.			The LifeVit wristband has been used instead of Fitbit. Medical devices have not been used.
Further ICT tools				Besides the overall CONNECARE system, further ICT tools have been experimented.



6. Conclusions and a Vision to the Future

The CONNECARE system was defined at the very beginning of the project according to the initial requirements of all the sites and the expectation of the project, as reported in the Document of Actions. First the overall architecture was defined, subsequently a generic solution for the integration with the clinical sites was searched for and the DHF defined (described in the D5.2 “Study Release of the generic CONNECARE system”). eWAVE implemented the integration layer called Patient Information Adapter (PIA) and in order to get and adapt information from the hospitals and integrate them to CONNECARE, the layer can get patients data from different formats and convert them into the CONNECARE format. The integration is different in Catalonia, Groningen, and Israel due to the different site requirements and limitations (described in the D5.5 “Final release of the Catalan CONNECARE system”, D5.7 “Final release of the Israeli CONNECARE system” and D5.9 “Final release of the Groningen CONNECARE system”).

As reported in the D3.3 “First Smart Adaptive Case Management system” and D3.6 “Final Smart Adaptive Case Management system”, the SACM is based on the SocioCortex, a social information modelling platform previously developed by the TUM. The SocioCortex has been adapted to healthcare scenario (see D3.1 “SocioCortexfor healthcare”) by TUM. On top of the SocioCortex, ADI implements the frontend according to the co-design approach of CONNECARE, the requirements, and the feedback by the clinicians all along the project. Regarding the SMS, the initial idea was to use the VitalinQ solution by IPHEALTH as a baseline. Thus, from the very beginning of the project the EURECAT and the IPHEALTH worked together to define the best solution for the backend (as reported in D4.1 “First Self-Management System”). After the IPHEALTH termination, the overall backend, called xCARE, has been developed by EURECAT (see D4.7 “Final Self-Management System”). Moreover, EURECAT has been in charge of developing the frontend of the SMS (i.e., the app) for both Android and iOS smartphones. All the work concerning the integration and communication between the SACM and the SMS (including the Queue Manager) was performed by EURECAT with the support of TUM and started once a preliminary version of both backends was available. The authentication manager and the access control and data security were implemented from the very beginning of the project by EURECAT. Both the mapping DSS and the recommender system were investigated and implemented by UNIMORE once the requirements were gathered and the clinicians started using the overall system. Their integration into the SACM and the SMS has been performed by UNIMORE with the support of EURECAT. During all the phases, eWAVE was in charge of performing the quality tests and managing the overall work.

Summarizing, we identified the starting and ending TRL of each of the components, including the overall system (a detailed table is given in D8.12-813), as depicted in Figure 11. Let us note that the DSS mapping ends with a TRL=5 instead of 6 as the other subsystem, because it was required and adopted only in Lleida and tested outside the clinical studies by the team of professional from the IRBLL (details are given in D3.4 “Stratification and mapping DSS”).

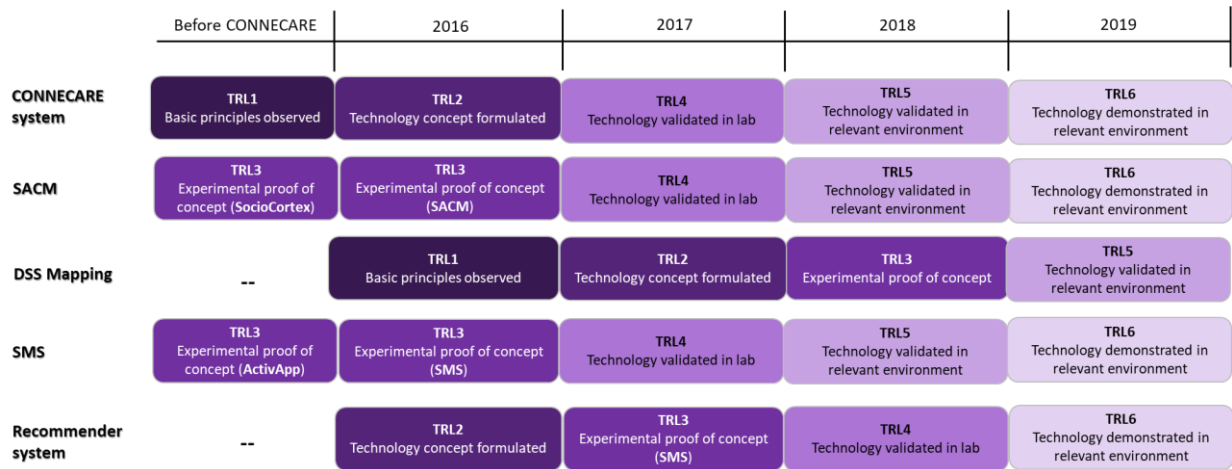


Figure 11 - TRLs of the CONNECARE system and each components

Considering the CONNECARE system as a whole, we may summarize the evolution of the system as in Figure 12.

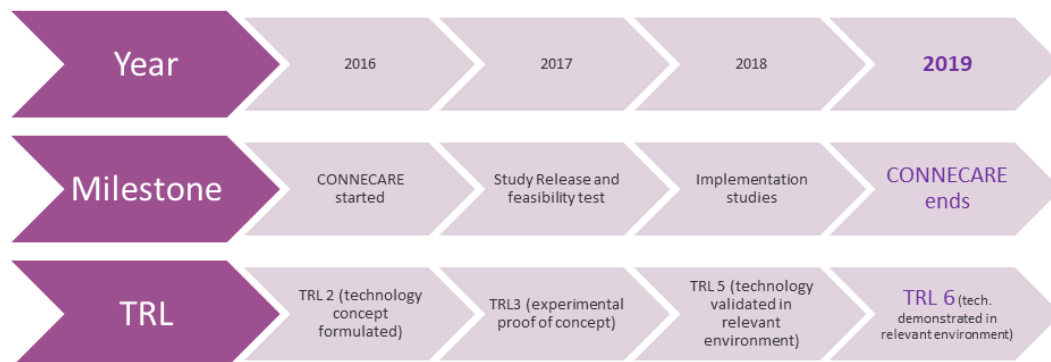


Figure 12 - Evolution of the CONNECARE system

As for the future directions, we envisage that, taking into account all the requests of improvement coming from the clinical sites (as reported in the implementation log and in D3.6 “Final Smart Adaptive Case Management system”, for the SACM, and D4.7 “Final Self-Management System”, for the SMS), the system is ready to be deployed at large-scale covering the following dimensions:

1. **Territorial:** implementation of the integrated-care programmes in further sites;
2. **Population:** extension of the integrated care programme at regional level in all the sites: the whole Region of Catalonia, The Netherlands, and in the Assuta Medical Centers’ network, providing the service to complex chronic patients from all over the countries.
3. **Diseases management:** implementation in different chronic diseases and different case types (e.g., heart disease, pulmonary disease, anaemia, cognitive impairment, metabolic syndrome).



In so doing, we will prove: continuity; increasing number of final users (both professionals, patients, and carers); replicability and flexibility of the solution; and sustainability in terms of costs of the involved health and social care systems.

Strategies for business model and exploitation plan of the CONNECARE system are out of the scope of the current deliverable. The corresponding information is part of D8.13 “Exploitation plan” submitted at M44.



7. References

- [1] E. Vargiu, J.M. Fernández, F. Miralles, S. Nakar, V. Weijers, H. Meetsma, S. Mariani, M. Mamei, F. Zambonelli, F. Michel, F. Matthes, J. Kelly, J. Eaglesham, R. Kaye. Patient Empowerment and Case Management in CONNECARE. Global Conference on Integrated Care (GCIC 2018). Singapore, February 1-3, 2018.



8. Annexes

8.1 Annex 1



CONNECARE

WP5 – Evolutionary Integration

Security Audit

H2020-EU.3.1: Personalised Connected Care for Complex Chronic Patients

Project No. 689802

Start date of project: 01-04-2016

Duration: 42 months

Project funded by the European Commission, call H2020 – PHC - 2015	
PU	Public
PP	Restricted to other programme participants (including the Commission Services)
RE	Restricted to a group specified by the consortium (including the Commission Services)
CO	Confidential, only for members of the consortium (including the Commission Services)

Revision: 01

Date: 04-04-2018 (circulated 15-01-2019)



Table of contents

1. INTRODUCTION	29
2. ANALYZED VULNERABILITIES	30
3. SUMMARY OF THE RESULTS	36
4. RECON	38
4.1 OPEN PORTS	38
4.2 DETECTED HOSTS	38
5. ENCOUNTERED VULNERABILITIES	39
5.1 HTML CODE INJECTION.....	39
5.1.1 <i>Affected URLs</i>	43
5.1.2 <i>Recommendations</i>	43
5.2 XSS	44
5.2.1 <i>Affected URLs</i>	46
5.2.1 <i>Recommendations</i>	46



1. Introduction

This document is the result of the audit process in “white box and black box” mode made to the host <https://system.connecare.eu>.

To carry out the audit, several widely used methodologies have been taken into account, such as:

- OSSTMM <http://www.isecom.org/research/osstmm.html>
- OWASP https://www.owasp.org/index.php/OWASP_Testing_Guide_v4_Table_of_Contents
Expanded and adapted by our team to make them more effective.

In all the identified vulnerabilities, the tests carried out, their objective and the impact of the findings classified with two metrics will be detailed:

- Criticality; it defines the degree of damage that can be inflicted on a system in case an attacker obtains profit of it.

Low: The impact of the vulnerability does not imply compromise of the system, but it does mean that an attacker forces the system to behave in a non-predefined way.

Medium: The impact of the vulnerability does not directly imply the commitment of the system, but it can put at risk the manner of operate of some system components, affect third parties in the use of the infrastructure, or be the starting point to a total commitment of the system or application.

High: The vulnerability's impact supposes the commitment of the system or data that the application handles. confidentiality, as well as the possibility of causing an error that generates an interruption in the provision of the service.

- Priority; it establishes the priority that must be assigned to the correction of the vulnerability found. Try to define the cost in resources of applying a corrective measure and its impact on the infrastructure.

Low: The application of the corrective measure involves a major change in the infrastructure (update an operating system, change the software version or add new functionality to an application) to correct a problem that does not pose a high risk.

Medium: The application of the corrective measure involves altering the configuration of a service that must be previously tested in pre-production or correcting an application by modifying its source code.

High: The corrective measure involves a change in the configuration of some service without impact on the operation of the services.



2. Analyzed Vulnerabilities

OS command injection
SQL injection
SQL injection (second order)
ASP.NET tracing enabled
File path traversal
XML external entity injection
LDAP injection
XPath injection
XML injection
ASP.NET debugging enabled
HTTP PUT method is enabled
Out-of-band resource load (HTTP)
File path manipulation
PHP code injection
Server-side JavaScript code injection
Perl code injection
Ruby code injection
Python code injection
Expression Language injection
Unidentified code injection



Server-side template injection
SSI injection
Cross-site scripting (stored)
HTTP response header injection
Cross-site scripting (reflected)
Client-side template injection
Cross-site scripting (DOM-based)
Cross-site scripting (reflected DOM-based)
Cross-site scripting (stored DOM-based)
JavaScript injection (DOM-based)
JavaScript injection (reflected DOM-based)
JavaScript injection (stored DOM-based)
Path-relative style sheet import
Client-side SQL injection (DOM-based)
Client-side SQL injection (reflected DOM-based)
Client-side SQL injection (stored DOM-based)
WebSocket hijacking (DOM-based)
WebSocket hijacking (reflected DOM-based)
WebSocket hijacking (stored DOM-based)
Local file path manipulation (DOM-based)



Local file path manipulation (reflected DOM-based)
Local file path manipulation (stored DOM-based)
Client-side XPath injection (DOM-based)
Client-side XPath injection (reflected DOM-based)
Client-side XPath injection (stored DOM-based)
Client-side JSON injection (DOM-based)
Client-side JSON injection (reflected DOM-based)
Client-side JSON injection (stored DOM-based)
Flash cross-domain policy
Silverlight cross-domain policy
Cross-origin resource sharing
Cross-origin resource sharing: arbitrary origin trusted
Cross-origin resource sharing: unencrypted origin trusted
Cross-origin resource sharing: all subdomains trusted
Cross-site request forgery
SMTP Header Injection
Cleartext submission of password
External service interaction (DNS)
External service interaction (HTTP)
Referer-dependent response



X-Forwarded-For dependent response
User agent-dependent response
Password returned in later response
Password submitted using GET method
Password returned in URL query string
SQL statement in request parameter
Cross-domain POST
ASP.NET ViewState without MAC enabled
XML entity expansion
Long redirection response
Serialized object in HTTP message
Duplicate cookies set
Input returned in response (stored)
Input returned in response (reflected)
Open redirection
Open redirection (DOM-based)
Open redirection (reflected DOM-based)
Open redirection (stored DOM-based)
SSL cookie without secure flag set
Cookie scoped to parent domain



Cross-domain Referer leakage
Cross-domain script include
Cookie without HttpOnly flag set
Session token in URL
Password field with autocomplete enabled
Password value set in cookie
File upload functionality
Frameable response (potential Clickjacking)
Browser cross-site scripting filter disabled
HTTP TRACE method is enabled
Cookie manipulation (DOM-based)
Cookie manipulation (reflected DOM-based)
Cookie manipulation (stored DOM-based)
Ajax request header manipulation (DOM-based)
Ajax request header manipulation (reflected DOM-based)
Ajax request header manipulation (stored DOM-based)
Denial of service (DOM-based)
Denial of service (reflected DOM-based)
Denial of service (stored DOM-based)
HTML5 web message manipulation (DOM-based)



HTML5 web message manipulation (reflected DOM-based)

HTML5 web message manipulation (stored DOM-based)

HTML5 storage manipulation (DOM-based)

HTML5 storage manipulation (reflected DOM-based)

HTML5 storage manipulation (stored DOM-based)

Link manipulation (DOM-based)

Link manipulation (reflected DOM-based)

Link manipulation (stored DOM-based)

Document domain manipulation (DOM-based)

Document domain manipulation (reflected DOM-based)

Document domain manipulation (stored DOM-based)

DOM data manipulation (DOM-based)

DOM data manipulation (reflected DOM-based)

DOM data manipulation (stored DOM-based)

Database connection string disclosed

Source code disclosure

Directory listing

Email addresses disclosed

Private IP addresses disclosed

Social security numbers disclosed



Credit card numbers disclosed
Private key disclosed
Robots.txt file
Cacheable HTTPS response
Base64-encoded data in parameter
Multiple content types specified
HTML does not specify charset
HTML uses unrecognized charset
Content type incorrectly stated
Content type is not specified
SSL certificate
Unencrypted communications
Strict transport security not enforced
Mixed content

3. Summary of the Results

The audit found several vulnerabilities in the way the parameters supplied by the users of the application are treated. These vulnerabilities allow to alter the behavior of the application in a malicious way and to be a potential vector for attacks with greater impact.

OWASP Top 10			
	Title	Times	Criticism
1	Injection	40	Medium
2	Broken Authentication and Session Management	0	Not applicable



3	Cross-Site Scripting (XSS)	2	Medium
4	Insecure Direct Object References	0	Not applicable
5	Security Misconfiguration	0	Not applicable
6	Sensitive Data Exposure	0	Not applicable
7	Missing Function Level Access Control	0	Not applicable
8	Cross-Site Request Forgery (CSRF)	0	Not applicable
9	Using Components with Known Vulnerabilities	0	Not applicable
10	Invalidated Redirects and Forwards	0	Not applicable

The following vulnerabilities have been founded:

Vulnerabilities			
	Title	Times	Criticism
1	HTML code injection	40	Medium
2	XSS	2	Medium



4. Recon

4.1 Open Ports

PORT	SERVICE	VERSION
22/tcp	SSH	SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.4
80/tcp	HTTP	nginx/1.12.1
443/tcp	HTTPS	nginx/1.12.1
8084/tcp	?	?

4.2 Detected Hosts

Using DNS configuration analysis, the following host related to the audited page have been detected:
redcap.connecare.eu

```
DNS Servers
-----
dns1.nominalia.com.      81.88.57.102      AS39729 Register.it SpA
  🗄️ 🌐 🔄 🛡️ 📡 🟢      dns1.nominalia.com      Italy
-----
dns2.nominalia.com.      81.88.63.48       AS39729 Register.it SpA
  🗄️ 🌐 🔄 🛡️ 📡 🟢      dns2.nominalia.com      Italy
-----
MX Records ** This is where email for the domain goes...
-----
10 mail.nominalia.com.   195.110.124.132   AS39729 Register.it SpA
  🗄️ 🔄 📡 🟢      mail.register.it        Italy
-----
TXT Records ** Find more hosts in Sender Policy Framework (SPF) configurations
-----
"v=spf1 include:spf.webapps.net ~all"
-----
Host Records (A) ** this data may not be current as it uses a static database (updated monthly)
-----
connecare.eu             95.142.152.194    AS198047 UK Webhosting Ltd
  🗄️ 🌐 🔄 📡 🟢      HTTP: Apache            United Kingdom
  🗄️ 🌐 🔄 📡 🟢      HTTPS: Apache
-----
redcap.connecare.eu      35.157.100.86     AS16509 Amazon.com, Inc.
  🗄️ 🌐 🔄 📡 🟢      ec2-35-157-100-86.eu-central-1.compute.amazonaws.com
  🗄️ 🌐 🔄 📡 🟢      HTTP: Apache            United States
  🗄️ 🌐 🔄 📡 🟢      HTTPS: Apache
```

This host hosts an app for database management.



Log In

Please log in with your user name and password. If you are having trouble logging in, please contact [CONNEXARE Consortium](#).

Username:

Password:

[Forgot your password?](#)

What is relevant about this host is that this database management app reveals the email address of a person related to the project.

ⓘ CANNOT RESET PASSWORD FOR "root"

The username "root" cannot be reset due to one of the following reasons: 1) It is not a valid REDCap username, 2) You have not yet set up a security question for your REDCap account, or 3) The password for this user is not able to be reset in REDCap because it can only be reset using an outside resource at your institution.

If you are not sure what to do now or are not sure where or how to reset your password for this account, please contact your [REDCap administrator](#).

<mailto:eloisa.vargiu@eurecat.org> ←

This email account should be a generic account not linked to a specific person since, from that e-mail, a malicious attacker could attempt a social engineering attack on the person to obtain information or access to the infrastructure.

5. Encountered Vulnerabilities

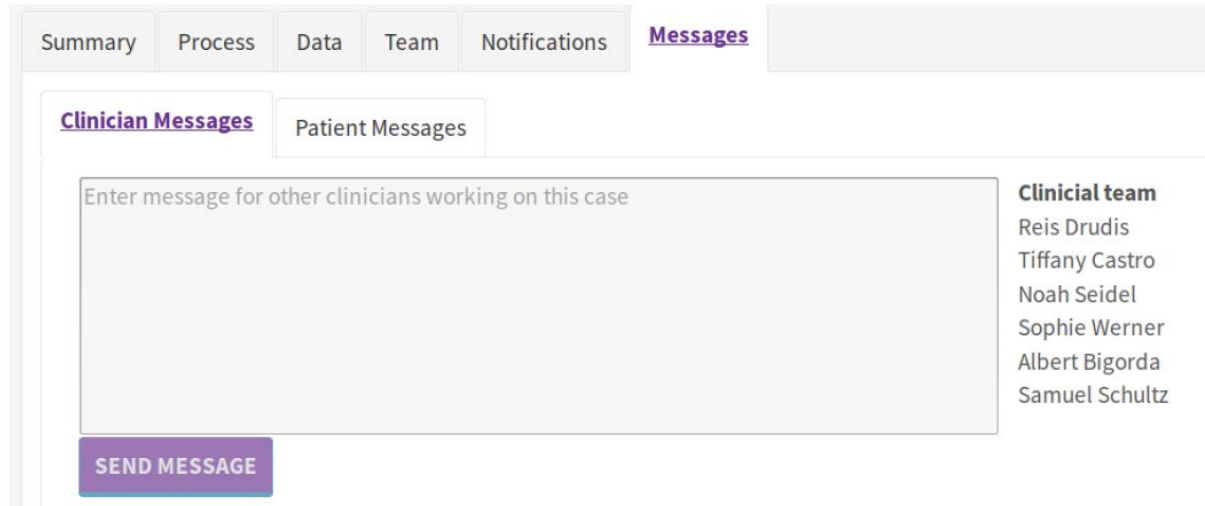
5.1 HTML Code Injection

Criticism	Medium	Number of occurrences	42
Probability	Not applicable	Priorities	Medium
Require authentication	Yes	Exposition	Medium/Intranet

During the audit it was found that in multiple parts of the application it is possible to inject HTML code that is then returned as part of the original application. This vulnerability allows altering the composition of the



website both aesthetically (include external content) and maliciously (malware download, spam, phishing ...). As an example, let us consider the following screen:



This messaging interface allows leaving messages to the rest of the team that supports a patient. This interface freely admits that the messages contain HTML which distorts the functionality and allows, for example, to load external content indiscriminately. Using the following HTML sentence as a message ``. It causes the image to load as part of the message.




Clinician Messages Patient Messages

Enter message for other clinicians working on this case

Clinical team
Reis Drudis
Tiffany Castro
Noah Seidel
Sophie Werner
Albert Bigorda
Samuel Schultz

SEND MESSAGE

Sophie Werner
Mar 23, 2018 13:16



Another vector of this attack may be the inclusion in seemingly legitimate links that lead to the downloading of malware. Using the following HTML sentence ` Updated Radiographs ` can add a link to any site where malware has been deposited and try to persuade the user to run it.



Clinician Messages Patient Messages

Enter message for other clinicians working on this case

Clinical team
Sophie Werner

SEND MESSAGE

Sophie Werner
Mar 23, 2018 13:31
Hola, tenemos nuevas radiografías que es muy urgente que sean revisadas, por favor, den su opinión

[Radiografías actualizadas](#)

It can also be linked using HREF to an external site that simulates belonging to the original infrastructure in which the access credentials are requested, giving rise to a Phishing attack.



5.1.1 Affected URLs

```
/sacm/api/v1/dualtasks/1ax0ndkbmshm7/duedate [request body]
/sacm/api/v1/dualtasks/1gvn0t3rl6nuo/duedate [request body]
/sacm/api/v1/dualtasks/1s6eowkb7tctl/duedate [request body]
/sacm/api/v1/dualtasks/1s6eowkb7tctl/humanpart/complete [by JSON parameter]
/sacm/api/v1/dualtasks/1s6eowkb7tctl/humanpart/complete [date JSON parameter]
/sacm/api/v1/dualtasks/1s6eowkb7tctl/humanpart/complete [isManualActivation JSON
parameter]
/sacm/api/v1/dualtasks/1s6eowkb7tctl/humanpart/complete [nrAlerts JSON parameter]
/sacm/api/v1/dualtasks/1s6eowkb7tctl/humanpart/complete [request body]
/sacm/api/v1/dualtasks/2pvush5dwvax/humanpart/complete [by JSON parameter]
/sacm/api/v1/dualtasks/2pvush5dwvax/humanpart/complete [date JSON parameter]
/sacm/api/v1/dualtasks/2pvush5dwvax/humanpart/complete [isManualActivation JSON
parameter]
/sacm/api/v1/dualtasks/2pvush5dwvax/humanpart/complete [nrAlerts JSON parameter]
/sacm/api/v1/dualtasks/2pvush5dwvax/humanpart/complete [request body]
/sacm/api/v1/dualtasks/tpmbre0bqrcm/humanpart/complete [by JSON parameter]
/sacm/api/v1/dualtasks/tpmbre0bqrcm/humanpart/complete [date JSON parameter]
/sacm/api/v1/dualtasks/tpmbre0bqrcm/humanpart/complete [isManualActivation JSON
parameter]
/sacm/api/v1/dualtasks/tpmbre0bqrcm/humanpart/complete [nrAlerts JSON parameter]
/sacm/api/v1/dualtasks/tpmbre0bqrcm/humanpart/complete [request body]
/sacm/api/v1/dualtasks/xrlyur6dsyae/duedate [request body]
/sacm/api/v1/dualtasks/xrlyur6dsyae/humanpart/complete [by JSON parameter]
/sacm/api/v1/dualtasks/xrlyur6dsyae/humanpart/complete [date JSON parameter]
/sacm/api/v1/dualtasks/xrlyur6dsyae/humanpart/complete [isManualActivation JSON
parameter]
/sacm/api/v1/dualtasks/xrlyur6dsyae/humanpart/complete [nrAlerts JSON parameter]
/sacm/api/v1/dualtasks/xrlyur6dsyae/humanpart/complete [request body]
/sacm/api/v1/humantasks/18lt81p4grc2w/duedate [request body]
/sacm/api/v1/humantasks/1e2k9l69epd12/duedate [request body]
/sacm/api/v1/humantasks/1mbxj5gqnef84/duedate [request body]
/sacm/api/v1/humantasks/1n0apyevod7fn/duedate [request body]
/sacm/api/v1/humantasks/1tzq4bdlr1eq0/duedate [request body]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [by JSON parameter]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [date JSON parameter]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [externalId JSON parameter]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [isManualActivation JSON parameter]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [next JSON parameter]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [nrAlerts JSON parameter]
/sacm/api/v1/humantasks/k5oybxyxqo2/complete [request body]
/sacm/api/v1/humantasks/schdtm0b594m/duedate [request body]
/sacm/api/v1/messages [case JSON parameter]
/sacm/api/v1/messages [request body]
/sacm/api/v1/messages [text JSON parameter]
```

5.1.2 Recommendations

From the point of view of security, any interaction with the user that involves taking parameters to then build with them something that is reflected in the application, should be analyzed with the utmost caution and thoroughness in search of HTML / Javascript code. For this, there are many 'regular expressions' or

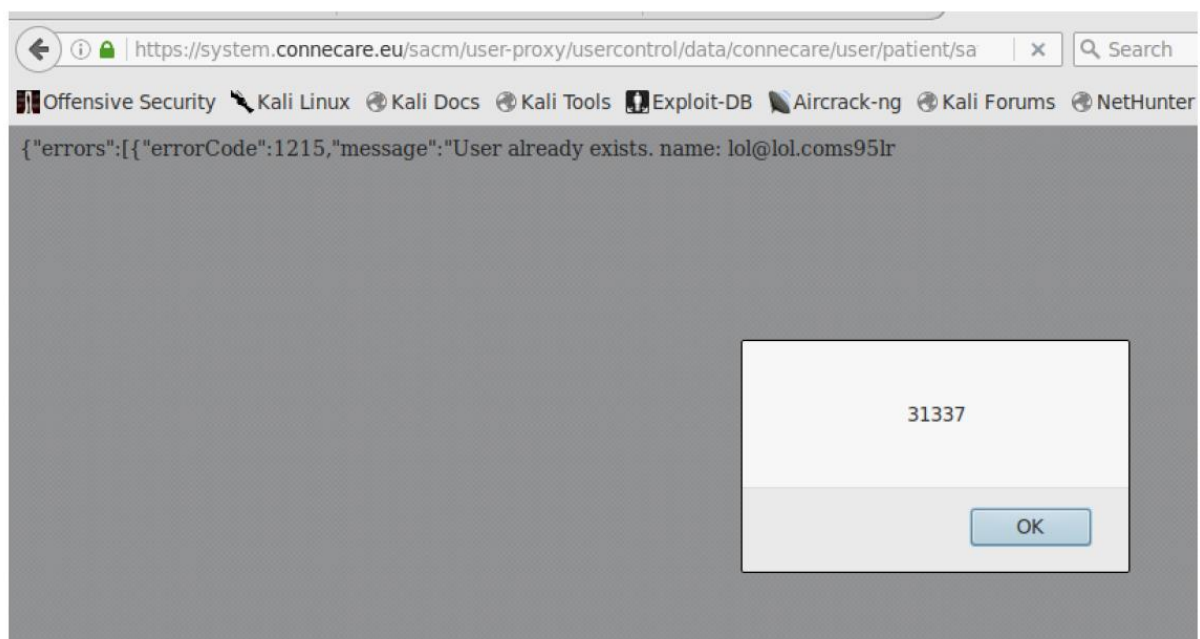


even libraries that allow an input from the user to be analyzed and converted to a format that prevents the obvious limits of the application from being exceeded.

5.2 XSS

Criticism	Medium	Number of occurrences	2
Probability	Not applicable	Priorities	Medium
Require authentication	Yes	Exposition	High/Internet

During the audit it was possible to verify the existence of at least two vulnerabilities of type XSS (Cross-site scripting). These vulnerabilities occur at the moment that an input provided by the user in the form of JavaScript code, is returned without 'trying' (escape) by the application. This causes a malicious user to insert JavaScript code arbitrarily and therefore cause unwanted effects to third users (theft of cookies, alteration of the appearance of the app, etc).



In the case of the audited application, the fact that this vulnerability occurs through JSON requests and not directly through GET / POST calls over a URL, makes the vulnerability less critical due to the added difficulty to be exploited.

Example of a vulnerable JSON call:



```
{
  "id": null,
  "username": "lol@lol.coms95lr<script>alert(31337)</script>j6pft",
  "email": "lol@lol.com",
  "roles": [
    "6c6864f0922811e7bd0c0242ac120002"
  ],
  "data": {
    "patientnr": "34453",
    "firstname": "trest",
    "lastname": "test",
    "birthdate": "1998-03-12T17:53:12.110",
    "gender": "UNKNOWN",
    "maritalstatus": "MARRIED",
    "educationlevel": "8",
    "culturallevel": "HIGH",
    "phone": "45454545",
    "mobile": "454545454",
    "language": "ES",
    "carename": "sdsds"
  }
}
```



5.2.1 Affected URLs

```
/sacm/user-proxy/usercontrol/data/connecare/user/patient/save [username JSON parameter]  
  
/sacm/user-proxy/usercontrol/data/connecare/user/professional/save [username JSON  
parameter]
```

5.2.1 Recommendations

As in the previous vulnerability, from the point of view of security, any interaction with the user that involves taking parameters to then build with them something that is reflected in the application, should be analyzed with the utmost caution and thoroughness in search of HTML / Javascript code. For this, there are many 'regular expressions' or even libraries that allow an input from the user to be analyzed and converted to a format that prevents the obvious limits of the application from being exceeded.

8.2 Annex 2

Penetration Test Report

Assuta

CONNECARE

Table of Contents

- .1 Characteristics 4**
 - 1.1 General 4
 - 1.2 System Details 5
- .2 Executive Summary 6**
 - 2.1 General 6
 - 2.2 Findings Distribution 7
- .3 Findings Summary 8**
- 4. Findings Details 13**
 - Item 4.1 13
 - Item 4.2 16
 - Item 4.3 19
 - Item 4.4 22
 - Item 4.5 24
 - Item 4.6 26
 - Item 4.7 27
 - Item 4.8 31
 - Item 4.9 32
 - Item 4.10..... 34
 - Item 4.11..... 37
 - Item 4.12..... 39
 - Item 4.13..... 41
 - Item 4.14..... 43
 - Item 4.15..... 45
 - Item 4.16..... 47
 - Vulnerability Description 47

.5 Appendices ¡Error! Marcador no definido.

5.1 Methodology **¡Error! Marcador no definido.**

5.2 Findings Classification **¡Error! Marcador no definido.**

1. Characteristics

1.1 General

Report Version	1.0	
Date	29.04.19	
Test Environment	Production / test	
Test Type	Grey Box	
Test Period	March - April 2019	
RE-Test Date	-	
System Components	Type	URL
	Web Application	https://system.connecare.eu/sacm
	Android	pro-connecare.mobile.apk
Vendor	-	
Project team	Name	Title
	Natalie Menachem	Projects Operation Team Leader
	Liran Segal	Head of Offensive Security
	Daniel Rabinowitz	Penetration Tester
Report Writer	Daniel Rabinowitz	
Test Limitation		

1.2 System Details

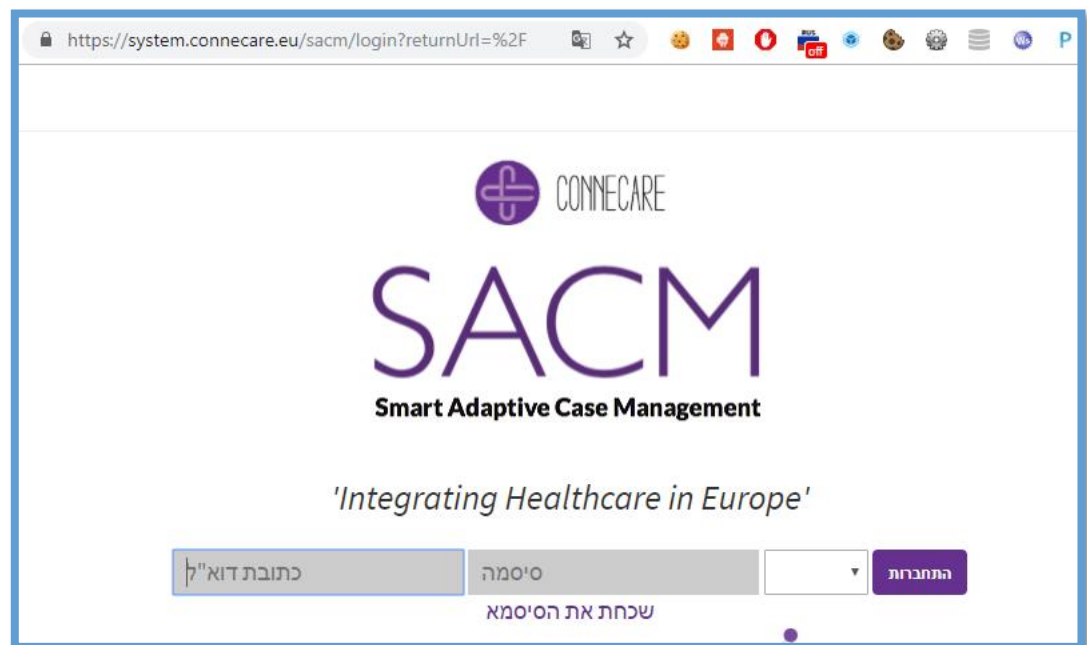
❖ Description

Assuta's Connecare system is a system for the Assuta hospital's patients. The system provides monitoring of biological measurements such as blood pressure, remote support from Assuta nurses in the pre-operation program, and Maccabi nurses for a period of three months after discharge, etc.

❖ Technology

- ❖ Nginx 1.14.0
- ❖ AngularJS 5.2.10
- ❖ Zone.js
- ❖ Lodash 4.17.10
- ❖ TinyMCE 4

❖ Screenshots



2. Executive Summary

2.1 General

This Penetration test was conducted during March-April 2019 against **Connecare system**, in order to ensure the system's ability to withstand attacks and to increase the protection of the data they contain.

Test Summary:

The system's security risk level is **High**, during the test we found many issues with the protection of sensitive information and things that could harm application users.

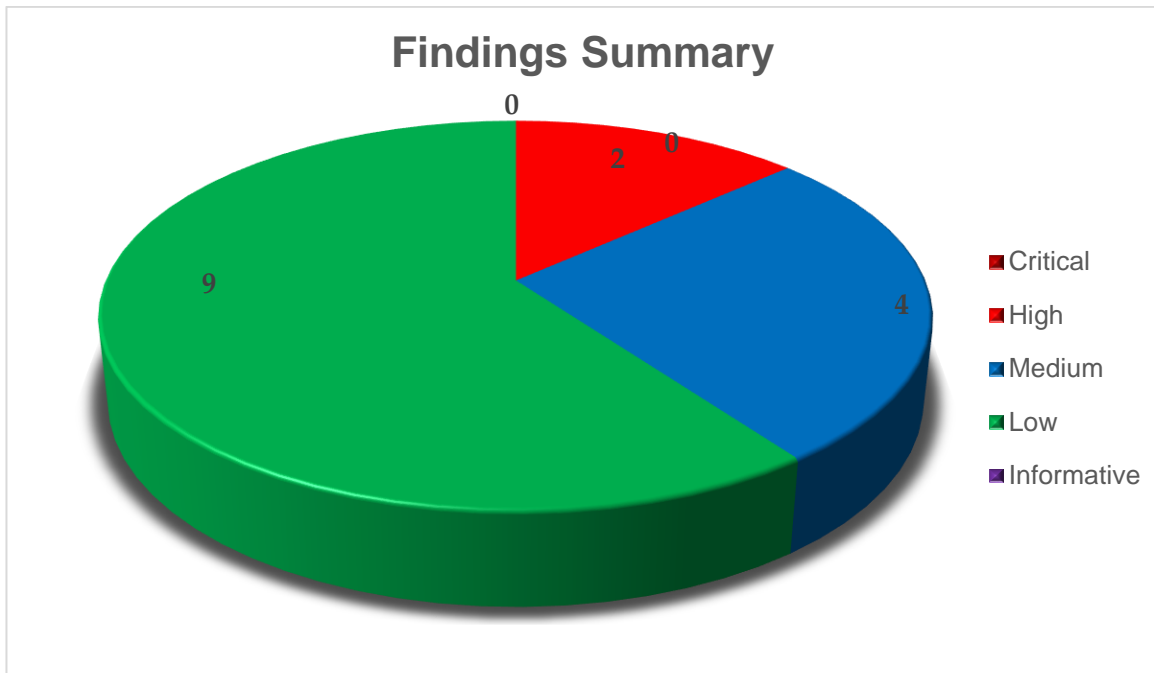
Below are a few examples of the weaknesses we found:

- The system does not prevent the uploading of files with malicious content / extensions that are not defined as legitimate for upload by the system definitions.
- The password reset mechanism is improperly implemented. When a password reset request is sent, the system automatically resets the password to a random password. In this way, any user's password could be reset without them knowing.
- The system does not contain mechanisms to prevent automated attacks, and thus an attacker could send a large number of requests in order to cause a server overload.
- System users can set "weak" passwords that are easy to guess, (e.g. numbers in order).
- The system locks users' accounts after a number of failed login attempts.

An intermediate or above level of technical knowledge is required to exploit most of the vulnerabilities. You are advised to apply corrections and/or controls to compensate for the findings. These can be seen in the list of recommended rectifications for each finding. Below are a number of recommended solutions:

- It is recommended to harden the file upload mechanism so that only authorized file types can be uploaded.
- It is recommended to send an email to the user with a link that redirects the user to a password reset form.

2.2 Findings Distribution



3. Findings Summary

Item	Test Type	Risk Level	Component	Topic	General Explanation	Status
4.1	Applicative	High	Application	Unrestricted File Upload	The Unrestricted File Upload vulnerability describes a situation where the server does not adequately restrict the type and content of files uploaded to the server by users. An attacker could exploit this weakness in order to cause a Denial of Service or even to gain control of the server.	Vulnerable
4.2	Applicative	High	Application	Insecurely Designed Component	The Insecurely Designed Component vulnerability describes a situation where a certain component is designed or implemented in an improper way which constitutes a security risk.	Vulnerable
4.3	Applicative	Medium	Application	Insufficient Anti-Automation	Insufficient Anti-Automation vulnerability allows an attacker to create automated processes to map the system's accounts and passwords or to cause a DoS (Denial of Service).	Vulnerable

Item	Test Type	Risk Level	Component	Topic	General Explanation	Status
4.4	Applicative	Medium	Application	Insecure Password Policy	The Insecure Password Policy vulnerability describes defects in the system's password policy that make it possible to use weak passwords to log into the system.	Vulnerable
4.5	Applicative	Medium	Application	Accounts Lockout	The Account Lockout vulnerability describes a situation where the system locks users' accounts after a number of failed login attempts. A malicious entity could exploit this mechanism in order to lock system users' accounts and thereby cause a Denial of Service.	Vulnerable
4.6	Applicative	Medium	Application	Sensitive Data Cached	The Sensitive Data Cached vulnerability describes a scenario in which an attacker with physical access to a victim's computer can steal sensitive information (e.g.: usernames, passwords).	Vulnerable
4.7	Applicative	Low	Application	Lack of Security Headers	The Lack of Security Headers vulnerability describes the absence of the security headers that provide an extra level of security by helping to reduce attacks and security vulnerabilities.	Vulnerable

Item	Test Type	Risk Level	Component	Topic	General Explanation	Status
4.8	Applicative	Low	Application	Insecure Cross-Origin Resource Sharing	The Insecure Cross-Origin Resource Sharing vulnerability makes it possible for an attacker to redirect site users' requests through himself and thus to steal sensitive information.	Vulnerable
4.9	Applicative	Low	Application	Improper Error Handling	The Improper Error Handling vulnerability describes a situation where the server sends users detailed error messages that contain information about it.	Vulnerable
4.10	Applicative	Low	Application	Insecure Session Policy	The Insecure Session Policy vulnerability describes a situation where the session management mechanism is improperly implemented, which constitutes a threat to system users.	Vulnerable
4.11	Applicative	Low	Application	Sensitive Accessible Services	The Sensitive Accessible Services vulnerability describes a situation where there are open services on the server that provide an attacker with new attack vectors on those open services and even make it possible to expose sensitive information about the system's structure and users.	Vulnerable

Item	Test Type	Risk Level	Component	Topic	General Explanation	Status
4.12	Applicative	Low	Application	Not Obfuscated Code	The Not Obfuscated Code vulnerability makes it possible for an attacker to examine the application's code in order to find weaknesses in the application.	Vulnerable
4.13	Applicative	Low	Application	Undetected Jailbreak or Rooted Device	The Undetected Jailbreak or Rooted Device vulnerability describes a situation where it is possible to install and use applications on a jailbroken device, which increases the attacker's leeway.	Vulnerable
4.14	Applicative	Low	Application	Insecure Data Storage	The Insecure Data Storage vulnerability describes a situation where the application stores sensitive information locally on the device it is installed on..	Vulnerable
4.15	Applicative	Low	Application	Old Application Version	The Old Application Version vulnerability increases the vulnerability to security risks. Security issues are frequently discovered in old software when the manufacturer's	Vulnerable

Item	Test Type	Risk Level	Component	Topic	General Explanation	Status
4.16	Applicative	Low	Application	Information Disclosure	Information Disclosure vulnerability helps an attacker to perform more effective attacks based on the system information.	Vulnerable

4. Findings Details

Item 4.1

Test Type: Applicative

Topic: Unrestricted File Upload

Risk Level: High

Severity: High Exploitation Probability: High

Vulnerability Description

A file upload mechanism enables users to upload files to the server for future use. The Unrestricted File Upload weakness describes a situation where the file upload mechanism can be exploited by an attacker in the following attack scenarios:

Denial of Service (DoS) – when an attacker can upload files onto the server without limitations on file size, content or the number of files that can be uploaded simultaneously, this can help an attacker to use up all of the free storage memory on the server and even to overload the server’s bandwidth, thereby causing a denial of service.

Server Takeover – when an attacker can upload files onto the server without limitation on content (MIME Types, file name etc.), this can help the attacker to upload a malicious file onto the server that will give him control over the server (e.g.: uploading a web shell file).

Vulnerability Details

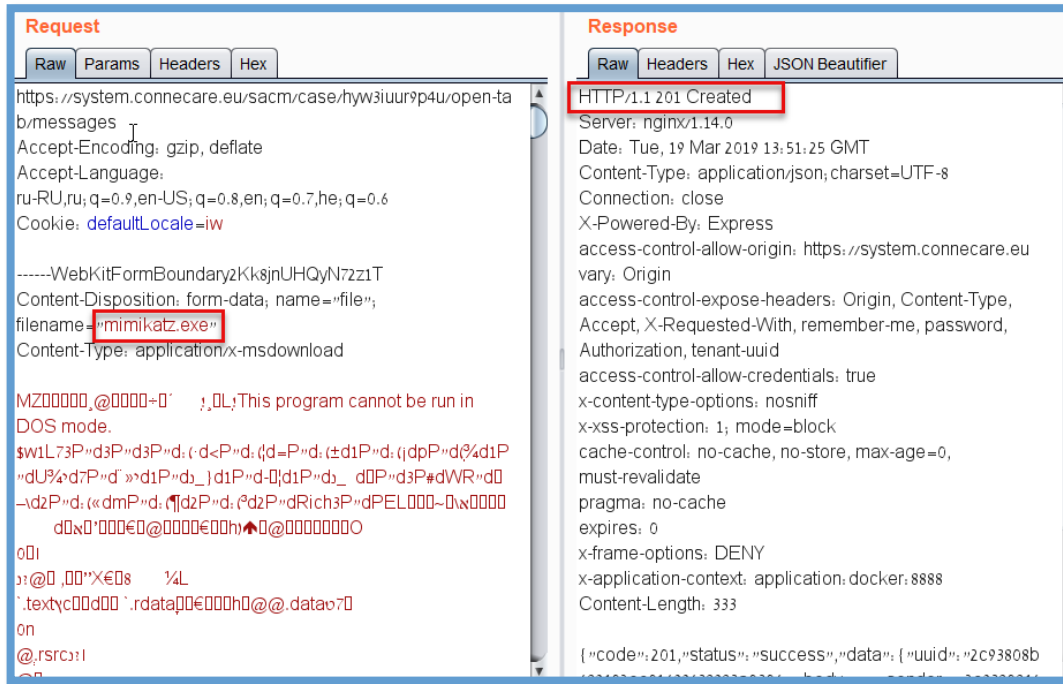
During the test that was done, we found that the system does not prevent the uploading of files with malicious content that have MIME Types that are not defined as legitimate for upload by the system definition. After the file has been uploaded, the system changes that file’s name to a random name and removes the file’s extension (e.g.: .exe, .pdf, .rar). The only thing the system blocked was files with the extensions “.txt, .xlsx, .docx, .doc.”

In addition, we found that when a file with the .csv extension is uploaded, the system changes the name and also adds the .xls extension.

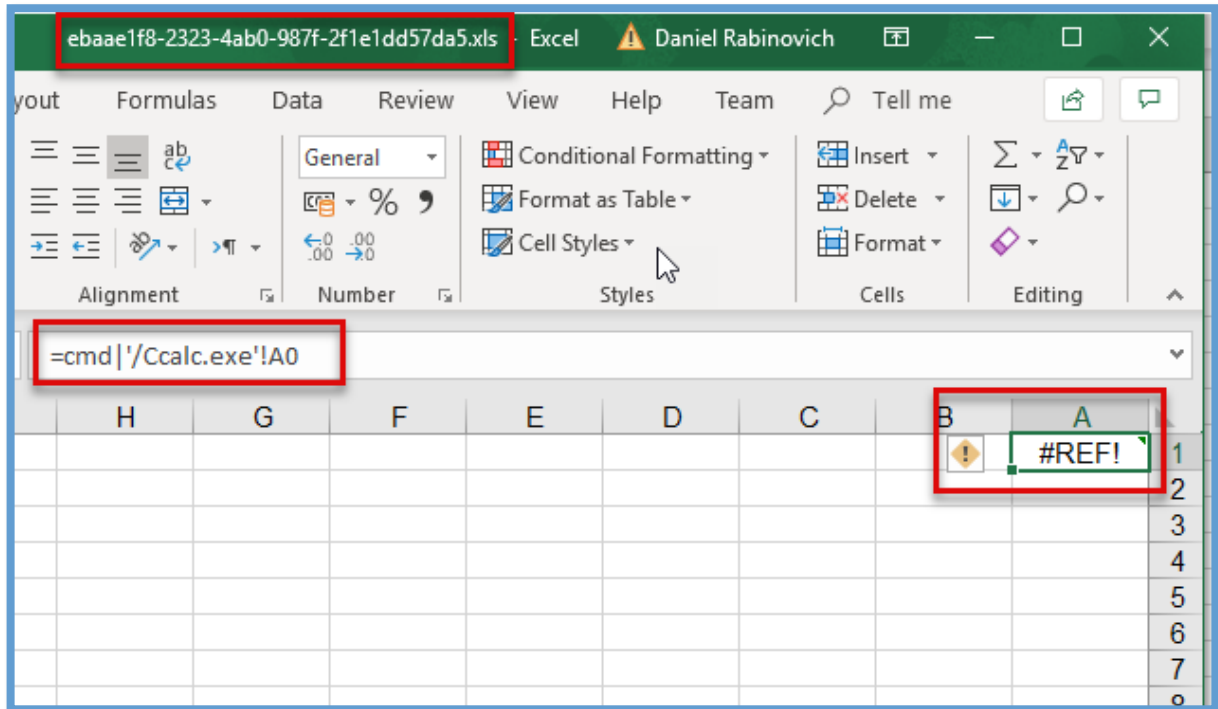
As a result, we were able to upload files with various extensions (e.g. html, exe, csv, php, aspx) that have malicious content (Mimikatz, Webshell, Formula Injection, Cross Site Scripting). An attacker or any other malicious entity could upload files with malicious content, which could help the attacker with future attacks.

Screenshots

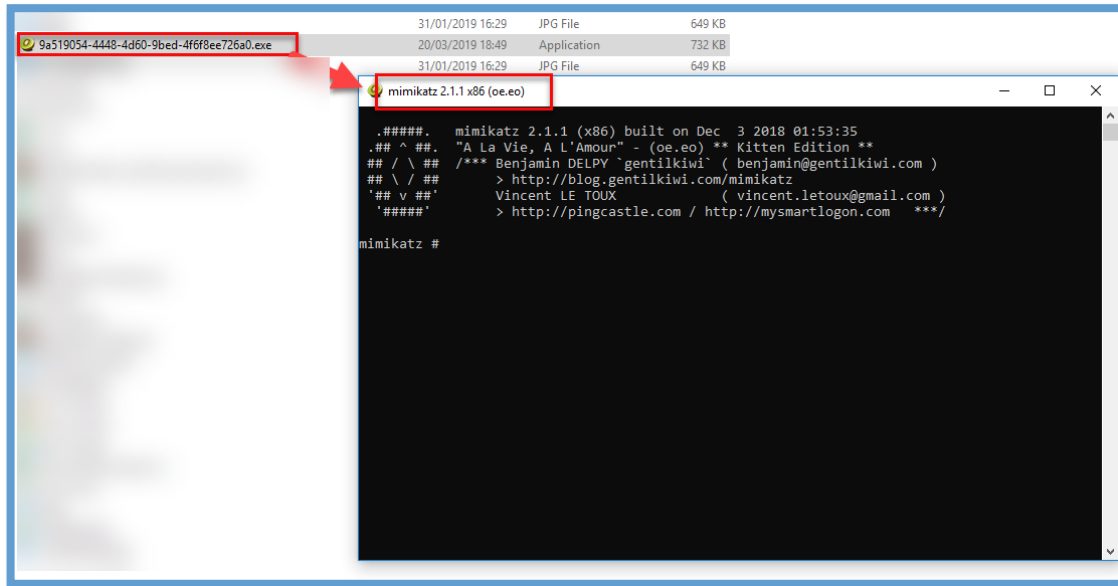
The following screenshot shows a file with malicious content being uploaded to the server:



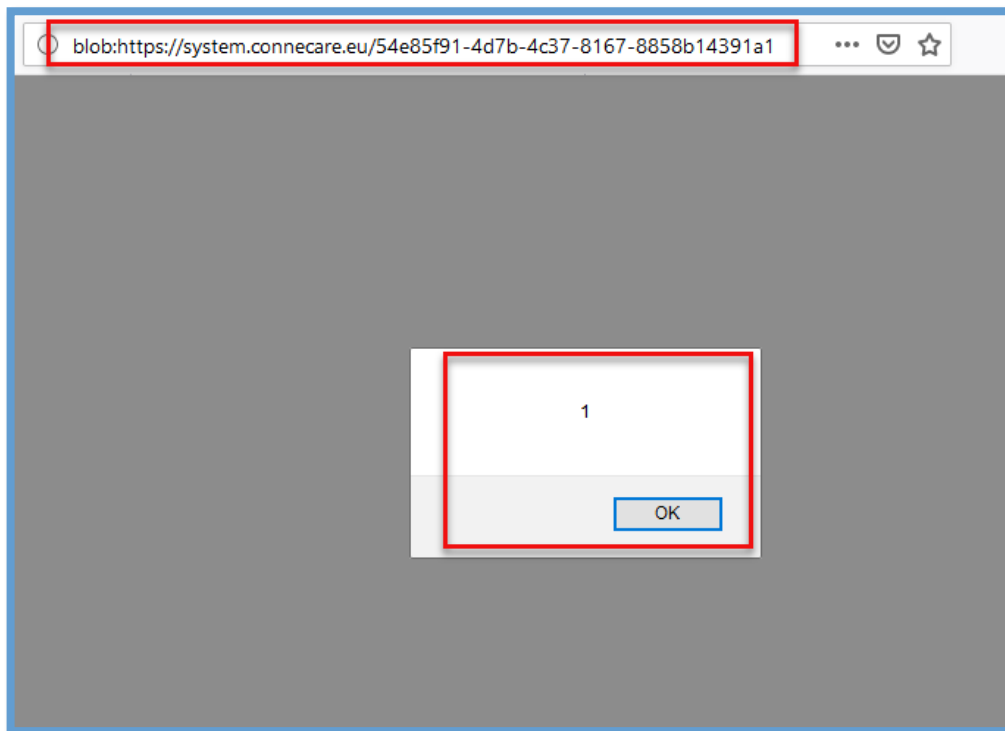
The following screenshot shows a file with malicious content that we downloaded from the system server being opened:



The following screenshot shows the execution of a file after we downloaded it from the system server and added an extension (.exe):



The following screenshot shows the execution of malicious JavaScript code after an HTML file was uploaded to the system server:



Recommended Rectification

It is recommended to harden the file upload mechanism so that:

- The files are stored in a dedicated library and not under the web library in which the site files are stored. A user must not be able to choose the library independently.
- Do not rely on file extensions. Check the Mime type that appears in the files' headers to ensure that the files arrive in the proper format. (This test should be done with the white list method).
- Limit the size of the files that can be uploaded to the system (e.g.: a maximum file size of 15 MB).
- Check the type of file according to its content and magic number identity.
- Store the files with random names. For example, the file Research123.jpg should be stored on the sever with the name sgbrys3f2504.jpg with the key 46811147 (the key can be stored in a database in order to link the original filename to the stored filename). This suggestion is intended to prevent malicious users from attempting to upload malicious files onto the server and then locating them and trying to execute them through the web server.
- Access to the file upload mechanism should only be enabled for authorised users and only after authentication.

You can find additional information on this weakness and the ways to implement solutions to rectify it in the following links:

<http://www.acunetix.com/websitesecurity/upload-forms-threat.htm>

[https://www.owasp.org/index.php/Unrestricted File Upload](https://www.owasp.org/index.php/Unrestricted_File_Upload)

<http://cwe.mitre.org/data/definitions/434.html>

Item 4.2

Test Type: Applicative

Topic: Insecure Design Component

Risk Level: High

Severity: High Exploitation Probability: High

Vulnerability Description

The Insecure Design Component vulnerability describes various situations in which a system component is designed or implemented improperly in a significant way that puts the application or system users at risk and could result in damage to the organization such as:

- System shutdown.
- Unauthorized actions being performed.
- Sensitive information leaking.
- The theft of login details and unauthorized control of user accounts.

The existence of this vulnerability makes it possible for an attacker to carry out a range of attacks against the system that will enable him to perform malicious actions and harm the organization or system users.

Vulnerability Details

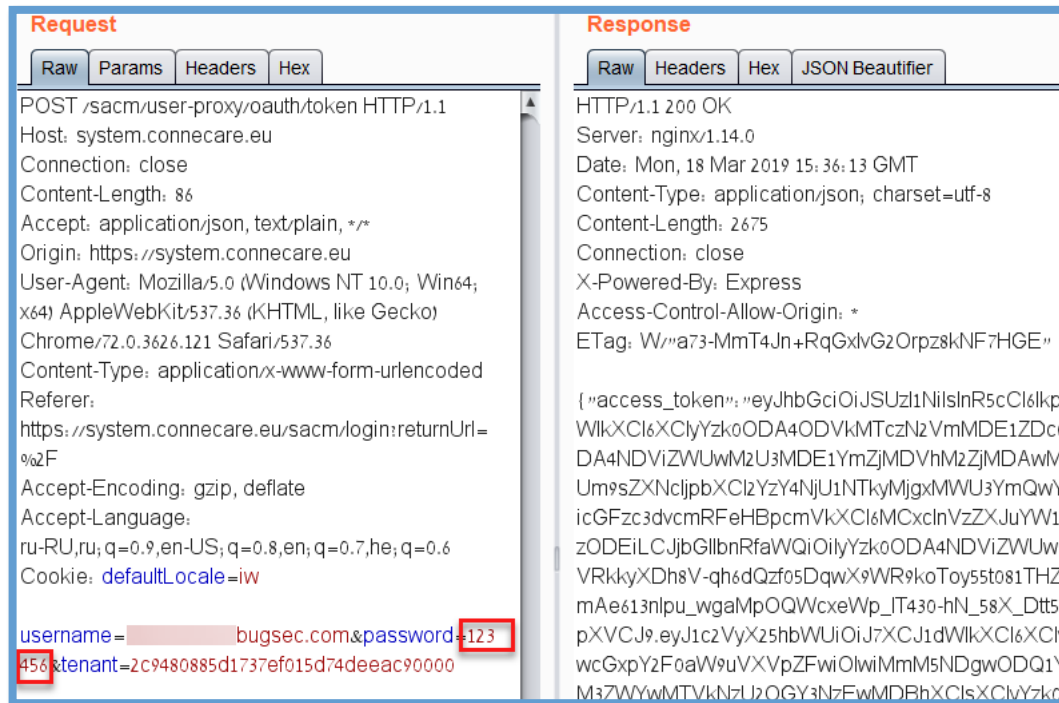
During the test, we found that the password reset mechanism is improperly implemented, and that when a password reset request is sent, the system automatically resets the password to a random password. In this way, any user's password could be reset without his knowledge. The following steps are required to carry out this attack:

- Enter the system login page.
- Click on "Forgotten your password".
- Enter the email address of the user whose password you want to reset.

An attacker or any other malicious entity could exploit this weakness in order to reset the passwords of the users on the system.

Screenshots

The following screenshots show the password before and after sending the reset request:

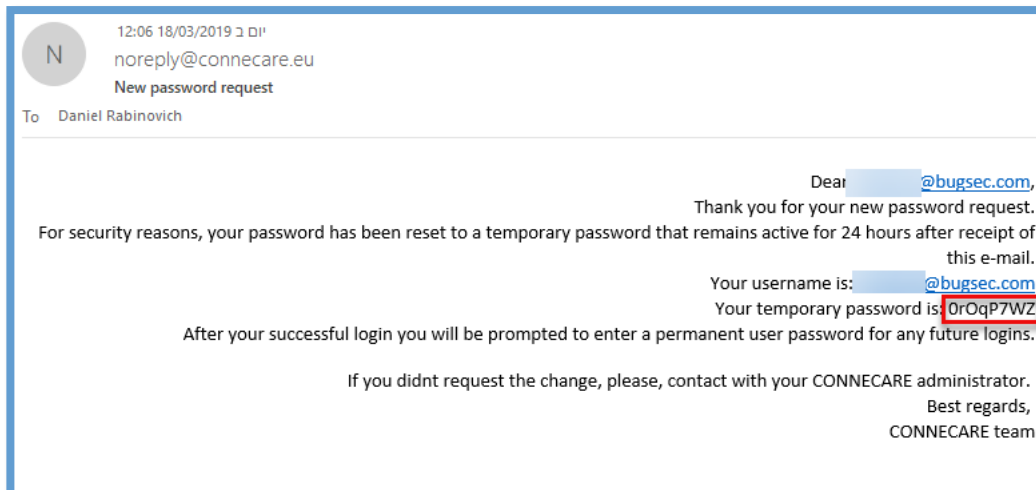


The screenshot displays an HTTP request and response. The request is a POST to `/sacm/user-proxy/oauth/token` with a `password` parameter highlighted in red. The response is an application/json containing an `access_token`.

```
Request
Raw Params Headers Hex
POST /sacm/user-proxy/oauth/token HTTP/1.1
Host: system.connecare.eu
Connection: close
Content-Length: 86
Accept: application/json, text/plain, */*
Origin: https://system.connecare.eu
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.121 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://system.connecare.eu/sacm/login:returnUrl=%2F
Accept-Encoding: gzip, deflate
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7,he;q=0.6
Cookie: defaultLocale=iw
username=bugsec.com&password=123456
tenant=2c9480885d1737ef015d74deeac90000

Response
Raw Headers Hex JSON Beautifier
HTTP/1.1 200 OK
Server: nginx/1.14.0
Date: Mon, 18 Mar 2019 15:36:13 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 2675
Connection: close
X-Powered-By: Express
Access-Control-Allow-Origin: *
ETag: W/"a73-MmT4Jn+RqGxlvGzOrpz8kNF7HGE"

{"access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bmludXkiOiJ1bmludXkiLCJpdiI6IjE4MDYyMzYyMjE1IiwiaWF0IjoiMTUyMzYyMjE1MjE1In0="}
```



The screenshot shows an email notification from `noreply@connecare.eu` titled "New password request" sent to Daniel Rabinovich. The email body contains the following text:

Dear @bugsec.com,
Thank you for your new password request.
For security reasons, your password has been reset to a temporary password that remains active for 24 hours after receipt of this e-mail.
Your username is: @bugsec.com
Your temporary password is: OrOqP7WZ
After your successful login you will be prompted to enter a permanent user password for any future logins.
If you didn't request the change, please, contact with your CONNECARE administrator.
Best regards,
CONNECARE team

Recommended Rectification

- It is recommended to send the user an email with a link that will redirect him to the password reset form.

Additional information:

https://www.owasp.org/index.php/Forgot_Password_Cheat_Sheet

Item 4.3

Test Type: Applicative

Topic: Insufficient Anti-Automation

Risk Level: Medium

Severity: Medium Exploitation Probability: Medium

Vulnerability Description

Insufficient Anti-Automation vulnerability occurs when a web form does not have an efficient protection against automated requests and Brute-Force attacks. This vulnerability might expose the system to two main attack vectors: The first one is a Denial of Service by repeatedly requesting for resources from the server and the other vector is a Brute-Force attack which is an attempt to discover passwords by systematically trying every possible combination of letters, numbers and symbols until you discover the correct combination.

Vulnerability Details

During the test that was done, we found that the system does not contain a mechanism for preventing automated attacks for the following forms:

- The password reset service: <https://system.connecare.eu/sacm/login?returnUrl=%2F>
- The system login page: <https://system.connecare.eu/sacm>

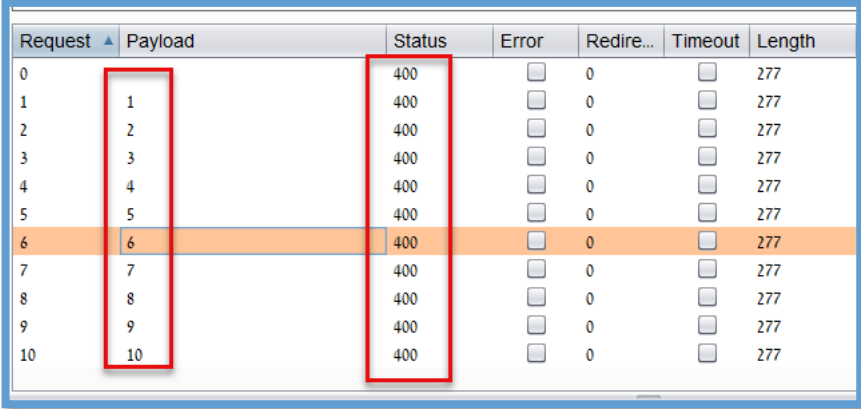
These are the steps that need to be taken in order to exploit the weakness in the login page:

- Enter the main page
- Catch the request using a proxy server (e.g.: Burp Suite).
- Define the “email” parameter that will change with each request according to the file of email addresses that we have created in advance.
- Send multiple requests to the server.

As a result, an attacker or any other malicious entity could exploit this weakness to overload the server by creating multiple requests through the login mechanism.

ScreenShot

The following screenshot shows the process of sending messages automatically:



Request	Payload	Status	Error	Redire...	Timeout	Length
0		400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
1	1	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
2	2	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
3	3	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
4	4	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
5	5	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
6	6	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
7	7	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
8	8	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
9	9	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277
10	10	400	<input type="checkbox"/>	0	<input type="checkbox"/>	277

Recommended solutions

- It is recommended to limit users to up to 5 requests in a period of time and after 5 requests, present them with a secure Captcha mechanism. Such a mechanism is present on the main login screen in the new version of the system. Another example of an effective, well-known mechanism is Google's reCaptcha.

For additional information: <https://www.google.com/recaptcha/intro/index.htm>

- It is recommended to create and apply a bandwidth throttling mechanism on the users who are logged into the system which does not allow any individual user to create more than X sensitive requests within a period of time Y (for example, a mechanism that does not allow a user to send more than 5 file upload requests within a period of 5 minutes).

Recommended Rectification

It is recommended to harden the password policy on the system so that weak passwords that do not meet the company's set policy cannot be used:

- A correct, secure password policy can be set using the following recommendations:
- Password length of at least 10 characters
- The password must contain a combination of uppercase and lowercase letters, numbers and special characters.
- The user's password should not be valid for more than 3 months.
- The system should not allow reuse of passwords that have already been used in the past.

Item 4.5

Test Type: Applicative

Topic: Account Lockout

Risk Level: Medium

Severity: Medium Exploitation Probability: Medium

Vulnerability Description

An Account Lockout occurs when an attacker can lock user accounts using a Brute Force attack on the system login page, providing the username that he wants to lock and additional incorrect login details. An attacker who exploits this vulnerability in the system could cause a Denial of Service with respect to logging in.

Vulnerability Details

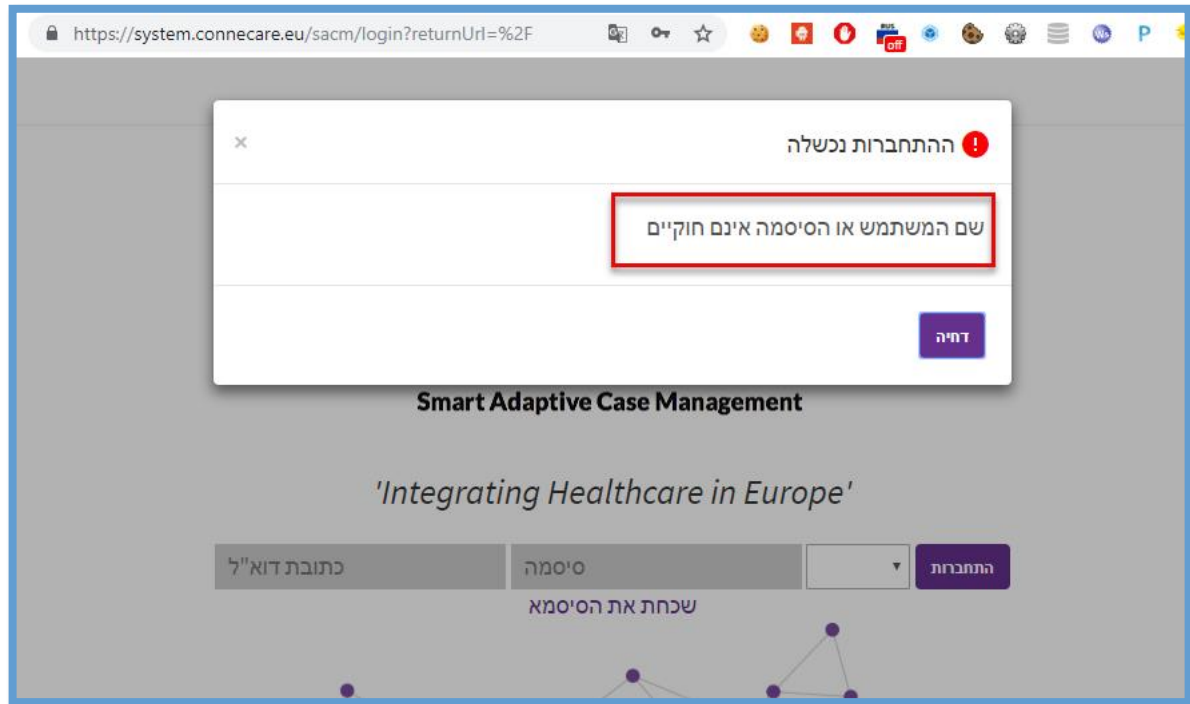
During the test that was done, we found that the system locks user accounts after a number of incorrect login attempts. When entering the “system login” page, the user must enter system login details, and if a user enters incorrect details, the user’s account will be locked after 3 attempts. To lock a user out, the following steps need to be taken:

- Access the “system login” page
- Enter incorrect login details 3 times
- The user’s account is locked

An attacker could exploit this weakness in order to arbitrarily block system users from accessing their accounts, ultimately leading to a denial of service.

ScreenShot

The following screenshot shows a system user's account being locked after 3 incorrect login attempts:



Recommended Rectification

- It is recommended not to block the accounts of system users.
- If there is a requirement to block user accounts, ensure that there is a time limit on this blocking (around 20 minutes)
- It is recommended to limit users to up to 5 requests in a period of time, and after 5 requests, to present them with a secure Captcha mechanism. Such a mechanism is present on the main login screen in the new version of the system. Another example of an effective, well-known mechanism is Google's reCaptcha.

For additional information: <https://www.google.com/recaptcha/intro/index.html>

Item 4.6

Test Type: Applicative

Topic: Sensitive Data Cached

Risk Level: Medium

Severity: Medium Exploitation Probability: Medium

Vulnerability Description

When activating the program, there are processes that could contain sensitive information such as usernames, passwords, database connection certificates, etc. This vulnerability exists when sensitive data about the servers / system users is stored without encryption in the client's cache memory (e.g. login passwords). This sensitive information could be read by a malicious entity with physical access to the client's computer.

Vulnerability Details

During the test, we found that when accessing the cache memory of the program's processes, one can see sensitive information such as: system users' email addresses. As a result, an attacker with physical access to a workstation could exploit this vulnerability in order to find out system users' email addresses.

In order to exploit this vulnerability, we took the following steps:

- Open the application manager and access the processes tab.
- Create a dump file for the process named "firefox.exe" (shown in the screenshots below)
- Download the "Strings" program from Microsoft from the following link:
<https://docs.microsoft.com/en-us/sysinternals/downloads/strings>
- Convert the dump file we have created to a regular text file using the following command:
- strings.exe firefox.DMP > dump.txt
- Open the resulting text file and search for keywords. In our case, we searched for "@".

ScreenShot

The following screenshot shows sensitive information in the dump file:

```
muchnik_t@mac.org.il
muchnik_t@mac.org.il
danielra@bugsec.co.il
danielra@bugsec.co.il
efratmatti@gmail.com
efratmatti@gmail.com
ischak_ma@mac.org.il
ischak_ma@mac.org.il
avivbe@assuta.co.il
avivbe@assuta.co.il
daniel1@bugsec.com
daniel1@bugsec.com
neta.rrr@gmail.com
neta.rrr@gmail.com
hadarro@assuta.co.il
hadarro@assuta.co.il]
victoriap@assuta.co.il
victoriap@assuta.co.il
maximre@assuta.co.il
maximre@assuta.co.il
danan_ir@mac.org.il
danan_ir@mac.org.il
daniel@bugsec.com
daniel@bugsec.com
reutro@assuta.co.il
```

Recommended Rectification

- It is recommended not to store sensitive information in the cache
- If possible, implement a mechanism such as JWT (the most recent version) or a similar mechanism, which will encrypt the content that is sent between the client and the server using a unique signature.

For more information:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control>

Item 4.7

Test Type: Applicative

Topic: Lack of Security Headers

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

HTTP headers are only fields, encoded in plain text, that constitute part of HTTP requests and response headers. They are designed to provide information about the connection that has been established, the requested resource, as well as the returned resource itself. HTTP security headers provide an additional layer of security by helping to reduce attacks and security vulnerabilities. The following is a list of HTTP headers linked to security:

- X-Frame-Options improves the protection of Web applications against clickjacking attacks. It makes a declaration about the policy that is transmitted from the host to the client browser about whether the browser should display the transported content in frames on other websites.
- X-XSS-Protection – this mechanism enforces the use of the XSS attack filtering which is a default setting in modern browsers.
- X-Content-Type-Options prevents the browser from interpreting files as something other than what is declared by the type of content in the HTTP headers.
- Content-Security-Policy – this is a security header that requires the careful adjustment and precise definition of the policy. If this option is activated, CSP has a significant influence on the way in which the browser processes pages (e.g. JavaScript is dropped as a default option and must be explicitly allowed in the policy). CSP prevents a wide range of attacks, including XSS and other injections on the site.
- X-Permitted-Cross-Domain-Policies – this is an XML document that gives a Web client, such as Adobe Flash Player, permission to handle data on domains. When clients request content stored in a certain source domain, and this content creates requests directed to a domain that is not its own, the remote host needs to host an inter-domain policy that gives access to the source domain, and enables the client to

continue the transaction. Generally the meta-policy is declared in the primary policy file, but for those that cannot write to the root library, they can also be declared on the meta-policy using the HTTP response header X-Permitted-Domain-Policies.

- Referrer-Policy – an HTTP heading that regulates which referrer information, sent in the Referer heading, should be included with requests made.

Vulnerability Details

During the test, we found that the server does not include the following headers in its responses:

- X-Frame-Options
- X-XSS-Protection
- X-Content-Type-Options
- Content-Security-Policy
- X-Permitted-Cross-Domain-Policies
- Referrer-Policy

The lack of these security headers could increase the risk and the likelihood of a range of attacks being carried out through a variety of attack scenarios against system users. As a result, an attacker could gain access to their accounts or to sensitive information on the system.

Screenshot

The following screenshot shows the lack of security headers:

Request

Raw Params Headers Hex

POST /sacm/user-proxy/oauth/token HTTP/1.1
Host: system.connecare.eu
Connection: close
Content-Length: 86
Accept: application/json, text/plain, */*
Origin: https://system.connecare.eu
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.121 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://system.connecare.eu/sacm/login:returnUrl=%2F
Accept-Encoding: gzip, deflate
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7,he;q=0.6
Cookie: defaultLocale=iw

username=[REDACTED]@bugsec.com&password=123456&tenant=2c9480885d1737ef015d74deeac90000

Response

Raw Headers Hex JSON Beautifier

HTTP/1.1 200 OK
Server: nginx/1.14.0
Date: Mon, 18 Mar 2019 11:02:30 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 2675
Connection: close
X-Powered-By: Express
Access-Control-Allow-Origin: *
ETag: W/\"a73-OlWjaUk57odKRLjSMnA0i6iFBjs\"

```
{ "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXLTJ5IiwiaWF0IjoiYXNjaUk57odKRLjSMnA0i6iFBjs\"
```

Recommended Rectification

- It is recommended to implement the relevant security headers in the server response.
For more information about security headers, see the following link:

https://www.owasp.org/index.php/OWASP_Secure-Headers_Project#xpcdp

Item 4.8

Test Type: Applicative

Topic: Insecure Cross-Origin Resource Sharing

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

Cross-Origin Resource Sharing is a mechanism responsible for informing a user's browser which domains should be allowed access to resources on the domain we are browsing, that is, from which domains (that have redirected the user) it can receive requests.

When the header Access-Control-Allow-Origin (a mechanism responsible for sharing the server's resources) is set to * (wildcard), a loophole is created which effectively lets any domain access resources on the server.

Vulnerability Details

During the test, we found that the header Access-Control-Allow-Origin contains the value * (wildcard), and as a result, the server makes the following scenario possible (for example):

1. An attacker sends an email to one of the site's users with the address of his malicious site – the link itself looks innocent, but in fact, when it is accessed, the attacker causes the user to send a request to the system server.
2. Behind the scenes, the attacker has ensured that after the client sends his requests to the system server, his browser sends the attacker the details that were sent in the request.

The steps above will result in every item of data that is sent in a POST request between the client and server actually passing through the attacker's server. As a result, the attacker will uncover all of the sensitive data sent in the requests, e.g. personal information.

ScreenShot

The following screenshot shows the modified request and the server response which includes Access-Control-Allow-Origin:

The screenshot displays a network request and response in a browser's developer tools. The request is a POST to the URL `/sacm/messaging/v1/conversation/case/hyw3iuur9p4u/file` with an `Origin` header set to `https://bugsec.com`. The response is an `HTTP/1.1 500 Internal Server Error` with an `Access-Control-Allow-Origin` header set to `https://bugsec.com`. Both the `Origin` header in the request and the `Access-Control-Allow-Origin` header in the response are highlighted with red boxes.

Recommended Rectification

- It is recommended to advised not to enable the transfer of login details in the header
Access-Control-Allow-Credentials: <https://www.html5rocks.com/en/tutorials/cors/>
- It is recommended to define only authorized domains in the Access-Control-Allow-Origin header.

For additional information, see the following link:

<https://www.geekboy.ninja/blog/exploiting-misconfigured-cors-cross-origin-resource-sharing/>

Item 4.9

Test Type: Applicative

Topic: Improper Error Handling

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

The Improper Error Handling vulnerability describes a situation where detailed error messages enable an attacker to gather information and learn about the system and its components.

An attacker could use the information on the system's technology that he has gathered in order to plan an attack on the server.

Vulnerability Details

During the test, we found that the server returns a system error which includes information about the server's technologies when the request is changed to methods that are not enabled on the server side. For example, when we change the way the request is sent from the GET method to a DEBUG request, the server returns error messages that expose information about the system server's technologies. An attacker or any other malicious entity could exploit this weakness in order to gain information about the server's technologies and to focus his attack by searching online for the known vulnerabilities of the server version.

Screenshot

The following screenshot shows the error message returned by the server that contains technical information about it:

Request				Response				
Raw	Params	Headers	Hex	Raw	Headers	Hex	HTML	Render
<pre> DEBUG/sacm/assets/translations/locale-en.json HTTP/1.1 Host: system.connecare.eu User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0 Accept: application/json, text/plain, */* Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://system.connecare.eu/sacm/ Connection: close Cookie: defaultLocale=en If-Modified-Since: Wed, 13 Mar 2019 07:27:47 GMT If-None-Match: W/"3b52-16975f33538" </pre>				<pre> HTTP/1.1 502 Bad Gateway Server: nginx/1.14.0 Date: Thu, 21 Mar 2019 12:18:58 GMT Content-Type: text/html Content-Length: 173 Connection: close <html> <head><title>502 Bad Gateway</title></head> <body bgcolor="white"> <center><h1>502 Bad Gateway</h1></center> <hr><center>nginx/1.14.0</center> </body> </html> </pre>				

Recommended Rectification

- It is recommended, when the system creates error messages, do not display detailed error messages to system users, and instead document these error messages in a dedicated internal error log.
- It is recommended to redirect system users to a generic error page which does not reveal details about the system.

Item 4.10

Test Type: Applicative

Topic: Insecure Session Policy

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

The Insecure Session Policy vulnerability describes a situation where the session management mechanism is improperly implemented and conceals security risks that could enable an attacker to gain control over system users' accounts.

Below are a number of examples of security flaws that could be found in the session management mechanism:

- The lifespan of the session ID, which determines the time after which an inactive session is destroyed on the server side, is not set, or is set to an extremely long time.
- The session ID is not random and/or is easy to guess.
- The session ID does not change and remains fixed with each new login.

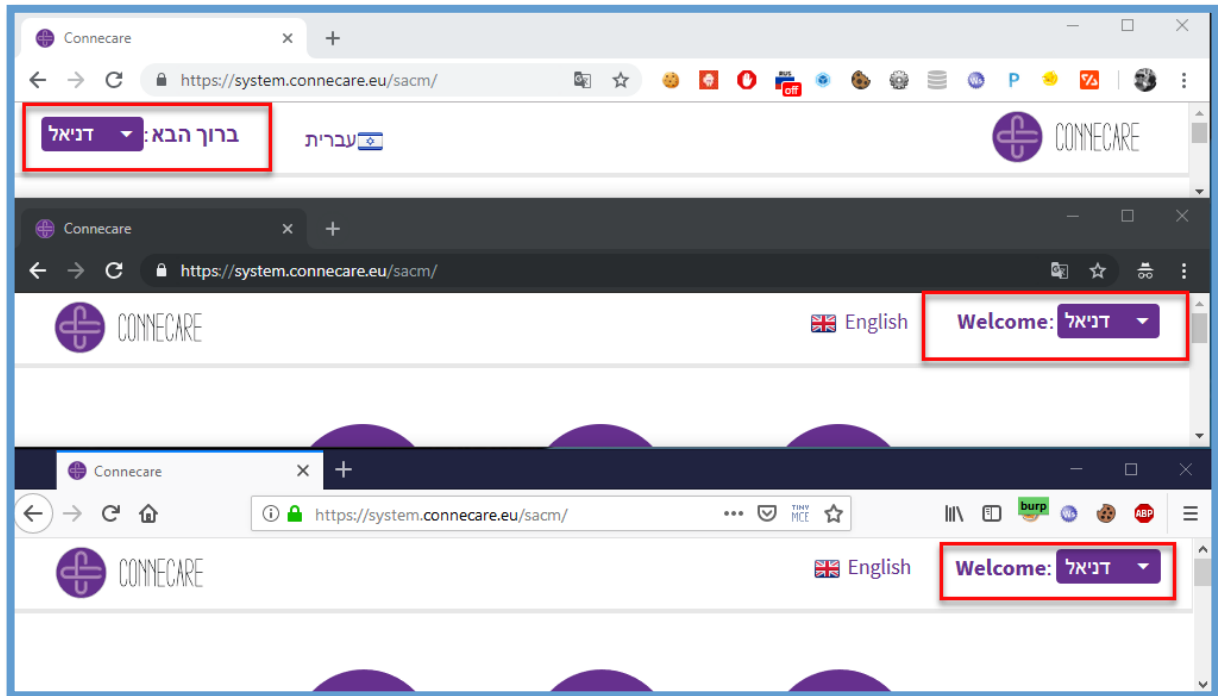
These flaws could enable an attacker to discover system users' session IDs and use them to take control of their accounts, to take actions on their behalf and to view information he is not authorized to see.

Vulnerability Details

During the test, we found that the application does not implement a Session Timeout mechanism. In the test, we found that it is possible to log in a number of times simultaneously, without receiving any indication that there are a number of logins for a specific user. The failure to use a Session Timeout could lead to a situation where a client has no indication from the system that there has been another simultaneous login to his account. As a result, an attacker or any other malicious entity could exploit this weakness to log into users' accounts without their knowledge if he manages to obtain account login details.

ScreenShot

The following screenshot shows the same account being logged into from three different browsers:



Recommended Rectification

- It is recommended, when the application is logged into from a new source while there is an active session, it is advisable to allocate a new session ID to the new login source and destroy the previous session on the server side.
- It is recommended to show users the last web address from which their accounts were logged into.

Item 4.11

Test Type: Applicative

Topic: Sensitive Accessible Services

Risk Level: Low

Severity: Medium Exploitation Probability: Low

Vulnerability Description

The Sensitive Accessible Services vulnerability describes a situation where the system server exposes open ports. This information could help an attacker to learn about the system server and the software installed on the system server in order to carry out future attacks.

Vulnerability Details

During the test, we found that the system server exposes details about itself and the services running on it. An attacker or any other malicious entity who uncovers this information could use it to refine future attacks on the system.

ScreenShot

The following screenshot shows a survey of the ports and services that are running on the system server:

Port	State (toggle closed [1] filtered [0])	Service	Reason	Product	Version	Extra info
22	tcp open	ssh	syn-ack	OpenSSH	7.2p2 Ubuntu 4ubuntu2.8	Ubuntu Linux; protocol 2.0
80	tcp open	http	syn-ack	nginx	1.14.0	
443	tcp open	http	syn-ack	nginx	1.14.0	
8084	tcp open	http	syn-ack	Node.js Express framework		
8085	tcp open	http	syn-ack	Node.js Express framework		

Recommended Rectification

- It is recommended to block access to services that are not being used.
- It is recommended to limit access to sensitive services on the system to dedicated IP addresses only, through Firewall Filtering.
- It is recommended to enable only secure connections to the server (TLS1.3) using the HTTPS protocol, starting from the system login screen, avoid redirecting system users through a protocol that is not encrypted (e.g. through the HTTP protocol).

Item 4.12

Test Type: Applicative

Topic: Not Obfuscated Code

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

The Not Obfuscated Code vulnerability describes a situation where an attacker who comes into contact with the application's code could search it for security vulnerabilities such as: Buffer Overflow, Hard-Coded Credentials, and other vulnerabilities that could endanger application users.

Vulnerability Details

During the test, we found that the application's source code is not obfuscated. An attacker could locate the application's apk file, analyze it (during the test we used "JD-GUI" to view the code) and gain a better understanding of the application's structure, map out its interfaces, locate potential vulnerabilities, and search for sensitive parameters in the code.

Screenshot

Below is a screenshot showing part of the application's code:

```
package md50cb2783eefe60cb86ffc2ef027fb39c;
import android.app.Activity;
import android.os.Bundle;
import java.util.ArrayList;
import mono.android.IGUserPeer;
import mono.android.Runtime;
import mono.android.TypeManager;

public class ActivityLifecycleCallbacks implements IGUserPeer, android.app.Application.ActivityLifecycleCallbacks {
    public static final String __md_methods = "n_onActivityCreated(Landroid/app/Activity;Landroid/os/Bundle;)V;getOnActivityCreated_Landroid_app_Activity_Landroid_os_Bundl
    private ArrayList refList;

    private native void n_onActivityCreated(Activity activity, Bundle bundle);
    private native void n_onActivityDestroyed(Activity activity);
    private native void n_onActivityPaused(Activity activity);
    private native void n_onActivityResumed(Activity activity);
    private native void n_onActivitySaveInstanceState(Activity activity, Bundle bundle);
    private native void n_onActivityStarted(Activity activity);
    private native void n_onActivityStopped(Activity activity);

    static {
        Runtime.register("Acr.UserDialogs.Infrastructure.ActivityLifecycleCallbacks, Acr.UserDialogs", ActivityLifecycleCallbacks.class, __md_methods);
    }

    public ActivityLifecycleCallbacks() {
        if (getClass() == ActivityLifecycleCallbacks.class) {
            TypeManager.Activate("Acr.UserDialogs.Infrastructure.ActivityLifecycleCallbacks, Acr.UserDialogs", "", this, new Object[0]);
        }
    }

    public void onActivityCreated(Activity activity, Bundle bundle) {
        n_onActivityCreated(activity, bundle);
    }

    public void onActivityDestroyed(Activity activity) {
        n_onActivityDestroyed(activity);
    }
}
```

Recommended Rectification

- Obfuscate the application's source code (you can use third party software such as [ProGuard](#)). For additional information, see the following link:
<https://riis.com/blog/android-obfuscation>

Item 4.13

Test Type: Applicative

Topic: Not Obfuscated Code

Risk Level: Low

Severity: Low **Exploitation Probability:** Low

Vulnerability Description

The Undetected Jailbreak or Rooted Device vulnerability refers to the lack of a mechanism for identifying and alerting users when an attempt is made to access the application from a device with root privileges. Using the application from such a device increases the risk of sensitive information leaking, because the process running on the device can access any part of the application using root privileges.

Vulnerability Details

During the test, we found that the application can be activated on a device with root privileges without any indication to the user. If the device is rooted (jailbroken), there are ways to get around the operating system's security mechanisms. As a result, an attacker with physical access to the device could access any information stored on it. Furthermore, in such a situation, malicious applications can perform read/write operations on the private information of other applications on the same device and violate the sandbox principle.

Screenshot

The following screenshot shows the application being run on a device that has strong (root) privileges:

```
root@OnePlus2:/ # cd data/local/tmp/
root@OnePlus2:/data/local/tmp # ./frida-server
C:\Users\user\Downloads>frida-ps -Uai
  PID  Name                               Identifier
-----
15276  ANI HAL Service                     com.dsi.ant.server
15276  Android System                      android
15946  Black Hole                           com.android.galaxy4
24309  Connecare                            connecare.mobile
16413  Contacts Storage                     com.android.providers.contacts
```

Recommended Rectification

- It is recommended, when the application is activated, display a warning message to the user explaining the possible risks. Consider a use in [SafetyNet Attestation API](#) to implement protection.

Item 4.14

Test Type: Applicative

Topic: Undetected Jailbreak or Rooted Device

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

The Insecure Data Storage vulnerability occurs when the application's keys assume that users will not have access to the storage systems of the devices the application is installed on, and therefore store sensitive information on them. This information could contain usernames, passwords, cookie details, geographical information, etc. By rooting, it is possible to connect the Android device to a computer / to connect through SSH and view that information using dedicated software that is available across the internet.

Vulnerability Details

During the test, we found that the application stores sensitive information (e.g.: RefreshAccessTokenKey, PushNotificationTokenKey, AccessKey) locally on users' devices. The following location exposes sensitive information:

- /data/data/connecare.mobile/shared_prefs/connecare.mobile_preferences.xml

As a result, an attacker with physical access to a device could access sensitive information and perform operations on the user's behalf without his knowledge.

Item 4.15

Test Type: Applicative

Topic: Old Application Version

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

The Old Application Version vulnerability increases the vulnerability to security risks. Security issues are frequently found for old software when the support provided by the manufacturer is no longer up-to-date.

Vulnerability Details

During the test conducted, we found that the server version Nginx 1.14.0 is not updated to the latest version released by the manufacturer. The manufacturers usually release updates when security breaches are discovered. Failing to update the system could weaken the level of system security and make it easier for potential attackers to find vulnerabilities in the system.

The following is a link to the known vulnerabilities of the system in version 1.14.0:

[Nginx 1.4.0 Vulnerabilities.](#)

Screenshot

The following screenshot shows the version of the Nginx server:

The screenshot displays the network tab of a browser's developer tools, showing an HTTP request and its corresponding response. The request is a POST to the endpoint `/sacm/user-proxy/oauth/token`. The response is an HTTP 200 OK. Two headers in the response are highlighted with red boxes: `Server: nginx/1.14.0` and `X-Powered-By: Express`. The request headers include `Host: system.connecare.eu`, `Accept: application/json, text/plain, */*`, `Origin: https://system.connecare.eu`, `User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.121 Safari/537.36`, `Content-Type: application/x-www-form-urlencoded`, `Referer: https://system.connecare.eu/sacm/login:returnUrl=%2F`, `Accept-Encoding: gzip, deflate`, `Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7,he;q=0.6`, and `Cookie: defaultLocale=iw`. The request body contains `username=` followed by a redacted value, `bugsec.com&password=123456&tenant=2c9480885d1737ef015d74deec90000`. The response body is a JSON object containing an `access_token`.

Recommended Rectification

- It is recommended to ensure that there is an orderly process for updating the system server, its components and all the technologies it uses, in order to prevent known vulnerabilities from being used against the system and its users. For additional information on the new versions, see the following link: <https://nginx.org/>

Item 4.16

Test Type: Applicative

Topic: Information Disclosure

Risk Level: Low

Severity: Low Exploitation Probability: Low

Vulnerability Description

Information Disclosure vulnerability is a misconfiguration issue that allows users to view information about technologies used by the application. This information mostly appears in server responses, errors, or in broken functionality.

Response headers reveal the server's type, version, and maybe other technologies in use, which may help an attacker in finding vulnerabilities and plan his attack on the system.

Vulnerability Details

During the test, we found that the system server exposes information and details about itself (and about the technologies the system uses) in the headers sent to users in server responses.

The following is a list of all the headers that contain information about the server's technologies:

- *X-Powered-By: Express*
- *Server: Nginx 1.4.0*

A malicious entity who discovers information about the system details (and the technologies implemented in the system) could exploit this when planning a future attack on the system server by using known weaknesses that can be found across the internet.

