Digital Security: Cybersecurity, Privacy and Trust
DS-02-2014 Access Control

European Commission | Horizon 2020 European Union funding for Research & Innovation

From Real-world Identities to Privacy-preserving and Attribute-based CREDentials for Device-centric Access Control

ReCRED

WP7 – Large Scale Pilots and End User Experience Assessment

Deliverable D7.2 "Campus-wide Wi-Fi and web services access control pilot set up"

Editor(s):	Ionut Florea, Savvas Zannettou
Author(s):	CSGN: Nicolae Ghibu, George Gugulea, Ionut Florea, Armand Ropot, Mihai Togan CNIT: Alberto Caponi, Claudio Pisa CUT: Michael Sirivianos, Savvas Zannettou UPCOM: Vangelis Bagiatis UC3M: Ruben Cuevas VERIZON: Steven Gevers, Stefan Bocken, Arsalan Najwani
Dissemination Level:	**Public**
Nature:	
Version:	1

ReCRED Project Profile

Contract Number	653417
Acronym	ReCRED
Title	From Real-world Identities to Privacy-preserving and Attribute-based CREDentials for Device-centric Access Control
Start Date	May 1st, 2015
Duration	36 Months

Partners

University of Piraeus	University of Piraeus research center	Greece
Telefónica Investigación y Desarrollo	Telefonica Investigacion Y Desarrollo Sa	Spain
verizon	Verizon Nederland B.V.	The Netherlands
certSIGN BY UTI	certSIGN SA	Romania
wedia	Wedia Limited	Greece
EXUS	EXUS Software Ltd	U.K.
upcom Bringing business and IT together	Upcom Bvba (sme)	Belgium
THE PRODUCTIZERS	De Productizers B.V.	The Netherlands
	Cyprus University of Technology	Cyprus
	Universidad Carlos III de Madrid	Spain
cnit	Consorzio Nazionale Interuniversitario per le Telecomunicazioni	Italy
BAKER & MCKENZIE	Studio Professionale Associato a Baker & Mckenzie	Italy

Document History

Version	Date	Author	Remarks
0.1		George Gugulea	Initial Table of Contents
0.2		Ionut Florea	Merge contributions
0.3		Savvas Zannettou	Update document
0.4-1.0		Ionut Florea	Update document, merge final contributions

Executive Summary

The purpose of this document is to describe the first pilot, Campus-wide Wi-Fi and web services access control, to be deployed in CUT and UC3M premises.

The design of the pilot was realized in WP6 and it combines the work realized in WP2, WP3, WP4, and WP5. The initial version of the pilot allows the consortium to have a first integrated version of the work performed in different work packages. The missing functionalities and the updated security posture will be addressed in the next version.

The document is structured as follows:

- Introduction - briefly presents the pilot
- Components Overview - describes the major pilot components that play an important role in the device authentication
- Hardware Architecture - describes all the hardware components that are part of the pilot and their role, their configuration and the reason for choosing them. This is a hardware view of the pilot.
- Software Architecture – describes all the software components of the pilot, including the OS and OS Tools. This is the software view of the pilot
- Security Considerations – describes the weaknesses of the pilot: spoofing attacks.
- Future Work / Conclusions – describes the evolution of the pilot and how the security risks will be mitigated

Table of Contents

List of Figures

List of Tables

1 Introduction

The Wi-Fi Campus access addresses and upgrades the access to a campus-wide Wi-Fi network and corresponding campus-only web-services. Administrators will define policies that take into account a user’s ID attributes. Students and professors will be able to connect to the Wi-Fi network presenting a minimal set of trustworthy attributes and they will be able to grant access to the Wi-Fi to roaming users.

Deployment of the pilot is realized over the existing networking and identity management infrastructure of the Universities, augmenting them in order to be able to use attribute-based authentication and FIDO. This nonintrusive approach allows a better adoption, especially from the perspective of the IT departments of the Universities.

When the students and professors register with their university, the university’s IT services create and store relevant identity attributes within their infrastructure (such as name, role, classes taken or taught, etc.). With these identity attributes the user can access the campus Wi-Fi and other resources (such as access to research network, moodle, etc.). Specifically, the users will be able to access the various campus resources by revealing a set of required identity attributes, which will be verified by the ReCRED infrastructure. Furthermore, users will authenticate by using biometric solutions such as fingerprint instead of regular username/password scheme. To this end, we will use a ReCRED-developed version of OpenID Connect/oAuth2 that is integrated with FIDO UAF.

The users will be able to use their mobile device in order to access the campus Wi-Fi and other resources, through the Campus Access mobile app. Initially, the user can open the Campus Access mobile app and see a list of all the available resources. Afterwards, the user asks for access to one or more resources and the Campus Access app informs the user of the identity attributes that will need to be revealed to the Authorization Server (acts as the Relying Party), according to the selected resources. The user grants access to those attributes using (oAuth2) and the app asks for a biometric authentication (FIDO UAF). An Authentication Server (acts as the Identity Provider) authenticates the user and an Authorization Server receives the user's identity attributes, along with the requested resources, and grants access to those resources (according to access control policies that will be defined by the network administrators). Finally, the app displays a message to the user (whether he was successfully authenticated and granted access to the requested resources).

In the rest of the document we will provide a detailed description of the Wi-Fi pilot running on CUT premises and the Wi-Fi pilot running on UC3M premises.

2 Components Overview

Figure 1 demonstrates a general overview of the pilot’s components and the steps for the authentication/authorization of users to CUT’s Wi-Fi. The black components are components that are already deployed at CUT’s premises and they will require none or minimum modifications.

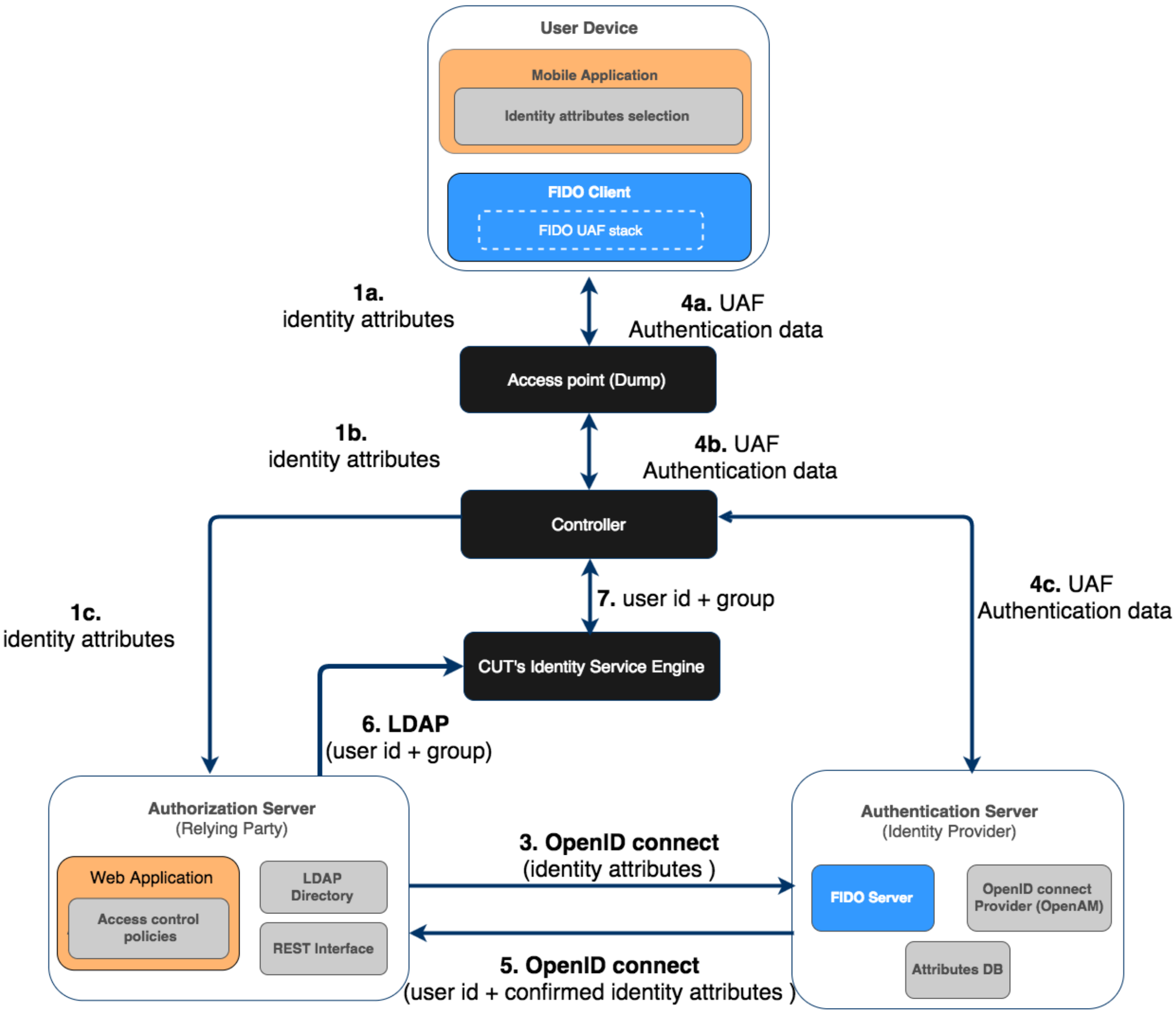


Figure 1: Overview of the pilot’s components

Registration Flow:

The user goes with his device to the registrar, which performs the FIDO registration (private key generated on the device), keeps the public key on the local ID provider attribute DB, and maps attributes to user. It’s a web application for FIDO registration. Here we emulate a registration and credential-issuance procedure. The user goes to the registrar (or logs in somehow securely through the universities VPN, which is outside of the scope of this pilot) so that a person there at the registrar knows physically who this person is. Once he has done the FIDO coupling, the administrator accepts that user with his username and his ID attributes. It also stores the Public key of the FIDO coupling. This is functionality related to the credential management module.

Pilot Flow:

Initially, the user should visit the ReCRED-developed smart-phone application. From there he will choose a set of identity attributes that will be used for the authentication/authorization. Then he connects to the nearest access point where he is redirected to the Authorization Server (Steps 1a-1c) through the Access Point and the Controller. The next step, involves the initiation of the Authorization/Authentication procedure. The Authorization Server, which acts as the Relying party, will contact the Authentication Server, which acts as the Identity Provider, via the OpenID connect protocol and it will transfer the user’s set of identity attributes (Step 3). From there the authentication server will perform a FIDO UAF authentication by asking the user to provide his fingerprint (the authentication is performed as is documented in the FIDO UAF Authentication Specification) (Steps 4a-4c). After that the Authentication Server will check the provided set of identity attributes against the already stored identity attributes in the attributes database that it maintains. When the authentication is completed, the Authentication Server will inform (again via the OpenID connect protocol) the Authorization Server regarding the outcome of the authentication (Step 5). The Authorization Server will ask the Authentication Server to provide the identity attributes for the user. From there, the Authorization Server will use the already pre-defined access control policies to decide which resources will grant to the user. For example, a user would be able to access the research network and use BitTorrent. Finally, the Authorization Server will update his LDAP directory with the authorization decision. The authorization decision involves the assignment of a group to a user and the update of the LDAP directory with this decision (the Authorization Server maintains a mapping between network resources and groups. A group may consist of one or several resources). Using this LDAP directory, the CUT’s Identity Service Engine will be responsible to provide the granted resources according to the assigned group (Step 6) (Note that in a typical scenario, the LDAP directory on the Authorization Server is not required. This requirement was made by CUT’s IT department because the CUT’s Identity Service Engine requires having access to an LDAP directory). Finally, the CUT’s Identity Service Engine will communicate with the network controller in order to provide an IP address to the user for the granted resources according to the assigned group (Step 7).

2.1 Mobile Application

On the user’s device we will deploy the Campus Access mobile app, which will be used so users can gain access to the campus Wi-Fi and other resources. The application will employ two modalities for attribute selection by the user. In the first modality, the user selects the identity attributes that he wants to reveal and the authorization/authentication flow starts. In the second modality, the user selects a network resource that he wants to gain access to and the mobile application asks the Authorization Server to fetch which identity attributes must be revealed. Consecutively the mobile application shows to the user which attributes will be used and if he agrees then the authorization/authentication initiates with these identity attributes. Within the application, biometric authentication will be used. Thus, the user’s device will run the underlying FIDO stack. The app initiates FIDO registration and authentication and sends the attributes for a specific request.

Overall, the mobile device offers the following functionality:

- It allows the user to see all the resources that are available at any given time and request access to one or more of those resources. For example, the user can see that he can request access to the campus Wi-Fi, as well as a Moodle platform and/or a Bittorent client.
- It guarantees the transparency of the procedure and preserves the privacy of the end user. The user knows exactly which attributes of his identity need to be revealed in order to gain access to a specific resource.
- It initiates the required authentication and authorization processes, so that it is always verified that the user is who he says he is and that he has the required roles and/or permissions in order to gain access to a requested resource.
- It provides various useful information regarding the purpose of the application, the registration process, answers to frequent questions, etc.

2.2 Access Point

The mobile device access to the network is realized through the Wi-Fi network. The system contains an access point that the user can easily access without requiring authentication. The mobile phone that the user uses to access the Wi-Fi network has no knowledge about any secrecy that the existing network devices know. Therefore, the access to the physically Wi-Fi network is open and so the access point creates an “open” Wi-Fi. Nevertheless, the user cannot access anything else but the ReCRED software stack and the mandatory services required to function (e.g. DHCP). The further access to the rest of the resources is restricted by the means of the Network Controller and therefore is not the role of the Access Point to restrict the user access (as we see in the conventional Wi-Fi network architectures). Being an “open” access point leaves the traffic between the user device and the rest of the network unencrypted and therefore open to different type of attacks but we will not treat this in the current version of the pilot (see chapters 5 and 6 for further discussions on this matter).

For the CUT pilot, we will be use the dumb access points that are already deployed at CUT’s premises. No additional modifications will be made to access points. All traffic coming from unauthenticated users will be redirected to the network controller.

2.3 Network Controller

The network controller is primarily responsible for regulating network traffic of unauthenticated users. In order to perform Authorization and Authentication of a user, communication should be established between the user’s device and the ReCRED’s Authorization/Authentication components. The network controller is responsible for providing a communication channel between the different components that are used for Authorization/Authentication and the user’s device. Specifically, the network controller receives network traffic from the access points and distributes the traffic to either the Authorization Server or the Authentication Server or the CUT’s Identity Service Engine.

After the completion of the Authentication and Authorization procedures, the network controller has the responsibility to allow access to a network resource. Specifically, the CUT’s Identity Service

Engine notifies the network controller about the Authorization outcome of a user’s request. The network controller sends a request to a DHCP server so that the user will gain an IP address to the appropriate network.

Note that the network controller performs a lot of other operations that are outside of the scope of ReCRED thus are not mentioned in this Section. Here we concentrate on the basic operations that are performed by the network controller in order to provide network access to users.

2.4 Authorization Server

The Authorization Server is an important component for this pilot as it acts as a Relying Party. Within the Authorization Server, the network administrator should be able to set access control policies that will define the granted resources for user’s requests. Furthermore, this component will have a Machine Learning (ML) back-end that will be responsible to provide suggestions for new access control policies.

Once the authentication and authorization is completed, the Authorization Server will update the LDAP directory that is deployed on the server so that CUT’s Identity Service Engine can read it and provide the appropriate network resources to users.

Furthermore, the Authorization Server will communicate with the Authentication Server and the user’s device via the OpenID connect protocol. During these interactions, user’s attributes and user’s id will be exchanged.

The Authorization Server will also provide a REST API so that the Campus Access mobile app can read the available resources and the identity attributes required for each resource.

2.5 Authentication Server

The Authentication Server is the other important component implemented for the pilot as it acts as the Identity Provider. The Authentication Server is responsible for authenticating users and identity attributes. To achieve that, a FIDO server will be deployed so that FIDO UAF authentication will be performed. Furthermore, the Authentication Server will maintain an attributes database that will provide a mapping between a user’s id and his respective confirmed identity attributes. By using this database, the Authentication Server will be able to confirm the user’s identity attributes. The Authentication server includes the gateSAFE module for FIDO/OpenID connect authentication. Furthermore, the Authentication server will act as an OpenID connect provider and it will be responsible to provide Authentication and end-user info to the Authorization Server.

gateSAFE provides a single-sign-on implementation such that users authenticate once; subsequent authentications, if necessary, will take place without user intervention. gateSAFE has a modular architecture, allowing the parallel operation of more servers, answering to a big number of connections.

At the beginning of ReCRED project gateSAFE provided:

- implementation of the users’ authentication mechanisms in a unique access point (Single Sign On – SSO);

- user authentication based on X509 digital certificates;
- integration with advanced security services: on-line validation of digital certificate status, directory services and time stamp services;
- secured connections between the client and the server over unsecure communication lines;
- user access management;
- monitoring of the connections, the users’ traffic and activity tracing;

Within ReCRED, gateSAFE features were extended to include FIDO authentication functionalities.

2.6 CUT’s Identity Service Engine (ISE)

The CUT’s Identity Service Engine is the main component that is currently used at CUT premises for the Authentication, Authorization and Accounting needs of the university. It offers a centralized network access policy management to provide consistent and secure access to users. From the administrator point of view, the ISE provides great visibility and accurate device identification. This is very important as it minimizes the unknown endpoints and possible threats in the network.

Currently, the CUT’s ISE authenticates the users with a username/password combination and then it applies a pre-defined university’s authorization policies in order to provide access to network resources to the users. When a user authenticates, it is assigned a group (such as student, staff, etc.). The ISE has a pre-defined set of access rules for each of these groups. For example, a staff member should be able to access the research LAN whereas a student should not. The ISE reads from an LDAP directory in order to acquire a user’s group and apply the pre-defined set of access rules.

For the purposes of this pilot, this component will be modified, by CUT’s IT department, in order to meet ReCRED needs. Specifically, all the Authentication and Authorization procedures will be carried away by the ReCRED components (Authentication and Authorization Server). The CUT ISE will be only responsible for verifying and confirming the authorization rules. Also, the Accounting part, that already takes place in the ISE will be used without modifications. Due to the fact that the ISE needs to use an LDAP directory we will deploy a LDAP directory at the Authorization Server. This constraint is set by the deployed infrastructure at CUT and in a typical scenario the extra LDAP directory is not needed. In a typical scenario, the component that will enforce the Authorization decision will call the user info endpoint at the Authentication Server via the OpenID connect protocol in order to acquire the user’s group and proceed with the completion of the user’s Authorization.

After the completion of the Authentication and Authorization procedures by the ReCRED components, the LDAP directory on the Authorization Server will be updated. Consequently, the CUT’s ISE will read the LDAP directory to retrieve the user’s group. After the retrieval of the user’s group, the ISE will find the appropriate access rules and it will notify the network controller. The network controller will make the appropriate actions so that the user will gain access to the selected network resources with the defined access rules.

3 Hardware Architecture

The Hardware architecture describes all the hardware components that are part of the pilot and their role, their configuration and the reason for choosing them. This is a hardware view of the pilot.

3.1 Access Point

The network access layer is based on the public radio frequencies at 2.4GHz and 5GHz, according to 802.11 a/b/g/n/ac Wi-Fi standards and compatible with all mobile devices for the last 8 years. Depending on the client density and expected network coverage, we use two specific solutions.

For a large campus network, as in Figure 2, we use a Cisco 5508 Wireless Controller (AIR-CT5508-K9) and Cisco Aironet 1100/1200/1600 Series Access Points. The controller manages the entire wireless infrastructure in terms of APs provision, radio channels allocation, clients’ authorization and traffic management. It can advertise multiple SSIDs (service set identifier), both open or secured, each one belonging to a given virtual LAN(VLAN). The APs might be connected anywhere in the campus network, even over a WAN link, once they can communicate with the controller over IP. The traffic between clients connected to different APs is considered as ‘local’, in the same Layer 2 domain and is also encrypted. The CAPWAP protocol helps to consider all wireless clients in our isolated network.

In this scenario, the controller requires a static IP address and APs could have dynamic IP, using DHCP. The only specific setting on APs is the controller’s IP address. On the controller side, the SSID ‘ReCRED’ is configured as open (no encryption nor authentication).

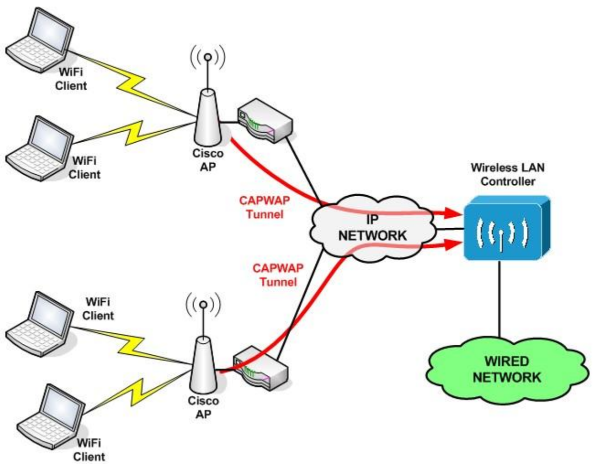


Figure 2: Access points deployment for a large campus

For a small network or laboratory, we use a Linksys WRT54GL wireless router. It provides the radio access point, Ethernet switching, IP routing and DHCP service in a single box. It is configured to broadcast the SSID ‘ReCRED’ as ‘open’ and assign IP addresses from the same subnet as ‘Authentication Server’. The server is attached to the wireless router on a LAN port.

In both scenarios, the Authentication Server must be the gateway for the mobile devices in order to access the outside the Wi-Fi resources.

3.2 Server

The server is a high end business solution from Lenovo, model M900 tower. It is well suited for virtualized environments due to it’s octa-core hyperthreaded processor, fast DDR4 dual-channel memory modules and large storage space.

The server hardware is composed of an Intel® Core™ i7-6700 processing unit, a total of 16 GB of DDR4 RAM, clocked at 2133 Mhz, one 256 GB Samsung EVO 750 SSD and a 1 TB Seagate 7200 rpm SATA HDD storage unit. The motherboard supports RAID levels 0,1, has a built-in Gigabit Ethernet chip and supports up to 8 USB 3.0 ports. As addons, the server is equipped with an optical unit, one Ethernet Gigabit addon card and one nVIDIA GT720 video card with 2 gigabytes of DDR5 memory.

CPU unit: Intel® Core™ i7-6700 Processor (8M Cache, up to 4.00 GHz), is based on the new Skylake architecture. “Skylake”is the codename used by Intel for a processor microarchitecture which was launched in August 2015 as the successor to the Broadwell microarchitecture. Skylake is a microarchitecture redesign using an already existing process technology, serving as a "tock" in Intel's "tick-tock" manufacturing and design model. According to Intel, the redesign brings greater CPU and GPU performance and reduced power consumption. Skylake uses the same 14 nm manufacturing process as Broadwell. The benefits that come with this new technology are as follows:

1 Improved front-end, deeper out-of-order buffers, improved execution units, more execution units(third vector integer ALU(VALU)), more load/store bandwidth, improved hyper-threading (wider retirement), speedup of AES-GCM and AES-CBC by 17% and 33% accordingly.

2 14 nm manufacturing process[50]

3 LGA 1151 socket for desktop processors

4 100 Series chipset (Sunrise Point)[51]

5 Thermal design power (TDP) up to 95 W (LGA 1151)

6 Support for both DDR3L SDRAM and DDR4 SDRAM in mainstream variants, using custom UniDIMM SO-DIMM form factor[53][54][55] with up to 64 GB of RAM on LGA 1151 variants. Usual DDR3 memory is also supported by certain motherboard vendors even though Intel doesn't officially support it.

7 Support for 16 PCI Express 3.0 lanes from CPU, 20 PCI Express 3.0 lanes from PCH (LGA 1151)

8 Support for Thunderbolt 3 (Alpine Ridge)

9 64 to 128 MB L4 eDRAM cache on certain SKUs

10 Up to four cores as the default mainstream configuration

11 AVX-512: F, CDI, VL, BW, and DQ for some future Xeon variants, but not Xeon E3

12 Intel MPX (Memory Protection Extensions)

13 Intel SGX (Software Guard Extensions)

14 Intel Speed Shift

15 Skylake's integrated Gen9 GPU supports Direct3D 12 at the feature level 12_1

16 Full fixed function HEVC Main/8bit encoding/decoding acceleration. Hybrid/Partial HEVC Main10/10bit decoding acceleration. JPEG encoding acceleration for resolutions up to 16,000×16,000 pixels. Partial VP9 encoding/decoding acceleration.

The processor has 8 MB L3 cache and is clocked at 3.4 Ghz with 4.0 Ghz achievable using Turbo boost overclocking technology. It has 4 independent central processing units, each supporting two separate threads of execution. Also the hardware configuration supports up to 64 GB of DDR4 RAM with a 34.1 GB/s maximum bandwidth, all with a TDP of just 65 Watts under base frequency and normal load.

RAM Modules: The server is equipped with 2 x 8 GB DDR4 HyperX DIMMs, working in dual-channel mode under 2133 Mhz frequency. HyperX HX421C14FB2K2/16 is a kit of two 1G x 64-bit (8GB) DDR4-2133 CL14 SDRAM (Synchronous DRAM) 1Rx8, memory module, based on eight 1G x 8-bit FBGA components

Storage Medium: The storage solution chosen to support the environment is composed of two storage units:

- **SSD Samsung 750 EVO 250GB SATA-III 2.5 inch** which hosts the operating system and all installed components for the virtualized systems. The solid-state drive solution was chosen as it would speed up boot times and access to various installed applications and backup operations with read and write speeds of up to 540MB/s. The 750 EVO features SED1 technology to help keep data safe. An AES-256-bit hardware-based full disk encryption engine secures data with significantly less performance degradation often experienced with software-based encryption. The 750 EVO is compliant with TCG™2 Opal v2.0 standards and IEEE®3 1667 protocol.

- **Seagate Barracuda ST1000DM003 1 TB** provides the needed storage space for the necessary virtualized environments and other operational needs. The device can sustain a maximum transfer rate of 210MB/s which should provide for the hosted virtualized guests.

Discrete video adapter nVIDIA GT720: The system comes with a video card module attached to a PCIe x16 slot. The card has 2 GB GDDR3 memory clocked at 1800 Mhz and 128-bit bus width able to transfer data at at rate of 28.8 GB/sec. The 40 nm Fermi GF108 processor is capable of rendering at 11200 MTexels/sec and supports a wide range of specific technologies like nVidia CUDA, PhysX, nVidia FXAA and Adaptive VSync. It is meant to provide 3D capabilities to the virtualized guest operating systems.

Network adapters: The system comes equipped with two Gigabit Ethernet adapters which will provide the integration within the project’s networked environment and also provide for either high availability or load balanced communications as needed.

3.3 User Device

3.3.1 Hardware platform

For testing the client mobile application, a Huawei Nexus 6P and an LG Nexus 5X smartphones were used. Nexus 6P is an Android based smartphone which runs Android 6 (Marshmallow) and which is equipped with a series of capabilities which enable the development of security applications. Regarding the hardware platform of Nexus 6P we enumerate the following:

- Chipset: Qualcomm MSM8994 Snapdragon 810
- CPU: Quad-core 1.55 GHz Cortex-A53 & Quad-core 2.0 GHz Cortex-A57
- GPU: Adreno 430
- WLAN: Wi-Fi 802.11 a/b/g/n/ac, dual-band, Wi-Fi Direct, DLNA, hotspot
- Sensors: Fingerprint, accelerometer, gyro, proximity, compass, barometer
- Memory: 3GB RAM + 32/64/128GB ROM
- Miscellanous: Bluetooth, GPS, NFC

Regarding the hardware platform of Nexus 5X we enumerate the following:

- Chipset: Qualcomm MSM8992 Snapdragon 808
- CPU: Quad-core 1.44 GHz Cortex-A53 & dual-core 1.82 GHz Cortex-A57
- GPU: Adreno 418
- WLAN: Wi-Fi 802.11 a/b/g/n/ac, dual-band, Wi-Fi Direct, DLNA, hotspot
- Sensors: Fingerprint, accelerometer, gyro, proximity, compass, barometer
- Memory: 2 GB RAM + 16/32 GB
- Miscellanous: Bluetooth, GPS, NFC

Taking into consideration the FIDO scenario, the fingerprint sensor plays an important role in authenticating the user. The FIDO Android client prompts a user authentication screen in order to unlock the FIDO cryptographic keys. On the Nexus 6P and Nexus 5X smartphones, the user can use the fingerprint sensor in order to authenticate to the FIDO client (the fingerprint security feature must be configured a priori from the OS security settings).

3.4 Network Access Controller

The network controller that is currently deployed at CUT’s premises is a Cisco Wireless controller 5508 (AIR-CT5508-K9). This controller will be configured by CUT’s IT department in order to provide the functionality that is described in Section 2.3. From a hardware perspective this component will be used as it is.

3.5 CUT’s Identity Service Engine

CUT’s Identity Service Engine consists of 4 nodes; 2 admin nodes and 2 policy nodes. The 2 admin nodes are 2 Cisco ISE3395 and they allow you to perform all the administrative operations that can be done on the CUT’s Identity Service Engine. They handle all system-related configuration that is related to authentication, authorization, etc. The 2 policy nodes are 2 Cisco NAC3355 and they provide a variety of functionalities such as client provision and profiling services. From a hardware perspective, these nodes are already deployed at CUT’s premises and will be used as they are.

4 Software Architecture

The Software Architecture describes all the software components of the pilot, including the OS and OS Tools. This is the software view of the pilot.

4.1 Operating System and Tools

4.1.1 Access Points (AP) Software and open Wi-Fi

The AP equipment involved in the project use Cisco IOS and OpenWrt as operating systems.

4.1.1.1 Cisco IOS

Cisco IOS is used for the APs in CUT, in a large campus network.

Cisco IOS (originally Internetwork Operating System) is software used on most Cisco Systems routers and current Cisco network switches. (Earlier switches ran CatOS.) IOS is a package of routing, switching, internetworking and telecommunications functions integrated into a multitasking operating system.

Cisco IOS is versioned using three numbers and some letters, in the general form a.b(c.d)e, where:

- a is the major version number.
- b is the minor version number.
- c is the release number, which begins at one and increments as new releases in the same a.b train are released. "Train" is Cisco-speak for, "...a vehicle for delivering Cisco software to a specific set of platforms and features.."
- d (omitted from general releases) is the interim build number.
- e (zero, one or two letters) is the software release train identifier, such as none (which designates the mainline, see below), T (for Technology), E (for Enterprise), S (for Service provider), XA as a special functionality train, XB as a different special functionality train, etc.

Rebuilds – Often a rebuild is compiled to fix a single specific problem or vulnerability for a given IOS version. For example, 12.1(8)E14 is a Rebuild, the 14 denoting the 14th rebuild of 12.1(8)E. Rebuilds are produced to either quickly repair a defect, or to satisfy customers who do not want to upgrade to a later major revision because they may be running critical infrastructure on their devices, and hence prefer to minimize change and risk.

Interim releases – Are usually produced on a weekly basis, and form a roll-up of current development effort. The Cisco advisory web site may list more than one possible interim to fix an associated issue (the reason for this is unknown to the general public).

Maintenance releases – Rigorously tested releases that are made available and include enhancements and bug fixes. Cisco recommend upgrading to Maintenance releases where possible, over Interim and Rebuild releases.

4.1.1.2 OpenWrt

OpenWrt is used for the APs in a small network at UPRC and in the test laboratory.

OpenWrt is an embedded operating system based on the Linux kernel, primarily used on embedded devices to route network traffic. All components have been optimized for size, to be small enough for fitting into the limited storage and memory available in home routers.

OpenWrt is configured using a command-line interface (ash shell), or a web interface (LuCI). There are about 3500 optional software packages available for installation via the opkg package management system.

OpenWrt can run on various types of devices, including CPE routers, residential gateways, smartphones, pocket computers (e.g. Ben NanoNote), and laptops. It is also possible to run OpenWrt on ordinary computers, which are most commonly based on the x86 architecture.

4.1.1.3 Open Wi-Fi

For the WLAN access, in order to allow credential issuance to not yet authorized users, the Wi-Fi APs do not make use of link-level encryption protocols such as WPA2, but are instead open.

Open Wi-Fi networks are unprotected Wi-Fi networks and are present usually in public places. They may be a threat because the users are connecting to a network without knowing who else could be on the network.

The security risks raised by this approach are presented in the Security Considerations chapter while the mitigation is described in the chapter Future Work / Conclusions.

4.1.2 Server operating system

The operating system used on the hardware server is a Linux based OS, Suse Linux Enterprise Server (SLES) 12 Service Pack 1, the 64 bit version. SLES was first released on October 31, 2000 as a version for IBM S/390 mainframe machines and in April 2001, the first SLES for x86 was released. The installed version of SLES 12 SP1 was released on 18 December 2015 and is the latest stable version at the moment of this document writing. Major versions of this product are released at an interval of 3–4 years, while minor versions (called "Service Packs") are released about every 18 months. SUSE Linux Enterprise products, including SUSE Linux Enterprise Server, receive more intense testing than the openSUSE community product, with the intention that only mature, stable versions of the included components will make it through to the released enterprise product.

The kernel of the SLES 12 SP1 operating system is version 3.12 and the architecture is 64 bit. The installation was made using the GUI interface, YAST.

The two available disks were mounted directly into the OS as follows:

- 1 SSD disk of 250 GB mounted in / partition

- 1 non-SSD disk of 1TB mounted in /opt partition

Also, a 16GB swap partition was created on the SSD disk for I/O disk speed considerations.

The operating system was installed using these patterns groups of packages:

- Base System
- 32-Bit Runtime Environment
- KVM Virtualization Host and tools
- Minimal System (Appliances)
- GNOME Desktop Environment
- X Window System
- KVM Host Server
- DHCP

For security reasons, the remote authentication of users via SSH was changed to public key and the root access to SSH, restricted. Each user had generated a pair of keys (private and public). The public key had been copied into the .ssh/authorized_keys file located in the user’s home directory on the server.

```
/etc/ssh/sshd_config
…
PermitRootLogin no
RSAAuthentication no
PubkeyAuthentication yes
PasswordAuthentication no
ChallengeResponseAuthentication no
UsePAM Yes
…
```

4.1.3 DHCP

Dynamic Host Configuration Protocol (DHCP) is a client/server protocol that automatically provides an Internet Protocol (IP) host with its IP address and other related configuration information such as the subnet mask and default gateway. RFCs 2131 and 2132 define DHCP as an Internet Engineering Task Force (IETF) standard based on Bootstrap Protocol (BOOTP), a protocol with which DHCP shares many implementation details. DHCP allows hosts to obtain required TCP/IP configuration information from a DHCP server.

The DHCP protocol implementation in SLES is called dhcpd and in this latest version of the OS (SLES 12 SP1) the version of this is 4.3.3

```
#rpm -qi dhcp-server
```

```
Name         : dhcp-server
Version      : 4.3.3
Release      : 2.2
Architecture: x86_64
Group        : Productivity/Networking/Boot/Servers
Size         : 2328638
License      : BSD-3-Clause
Signature      :  RSA/SHA256,  Thu  Oct  29  17:16:07  2015,  Key  ID
70af9e8139db7c82
Source RPM   : dhcp-4.3.3-2.2.src.rpm
Build Date   : Thu Oct 29 17:13:26 2015
Build Host   : sheep20
Relocations : (not relocatable)
Packager     : https://www.suse.com/
Vendor       : SUSE LLC <https://www.suse.com/>
URL          : http://www.isc.org/software/dhcp
```

The installation was made with SLES YaST2 - universal configuration utility, using yast sw_single command and searching the dhcp-server and yast2-dhcp-server packages.

The configuration file is dhcpd.conf and is located in /etc folder. It has been generated with the yast dhcp-server utility, following the wizard.

The configuration file shows the parameters for the networking card that are going to be send to clients contacting this server. It also shows the pool of addresses that are going to be allocated to clients and also the default lease time and the maximum lease time.

To improve security, the SUSE Linux Enterprise Server version of the ISC's DHCP server comes with the non-root/chroot patch applied. This enables dhcpd to run with the user ID nobody and run in a chroot environment (/var/lib/dhcp). To make this possible, the configuration file dhcpd.conf must be located in /var/lib/dhcp/etc. The init script automatically copies the file to this directory when starting.

The chroot behavior of the DHCP server is configured by setting DHCPD_RUN_CHROOTED in /etc/sysconfig/dhcpd to "yes" (default).

4.1.4 Virtualization environment

Each server side software element running on the hardware server will be deployed in a virtual machine in order to ease the component's integration. The virtualization solution was chosen because the server side components could have different software dependencies requirements. By enforcing such a solution, the overall system architecture will be modular and each software element will implement a communication interface with the other components.

The server’s virtualization environment is composed of a mix of solutions intended to accommodate all possible configurations for guest machines.

The main virtualization solution is KVM (Kernel Virtual Machine) and it was preferred because of its strong integration with the host operating system, good support for different guest OSes, a long list of features and additions in the latest versions and the presence of many open-source management tools readily available. Oracle VM VirtualBox is also available for compatibility with the Open Virtualization appliance format, when it will be required. Also, the Xen hypervisor is available, along with lxc containers and Docker, if it will be necessary to run applications under a microkernel designed system.

4.1.4.1 KVM

KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). It consists of a loadable kernel module, kvm.ko, that provides the core virtualization infrastructure and a processor specific module, kvm-intel.ko or kvm-amd.ko.

Using KVM, one can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware: a network card, disk, graphics adapter, etc.

KVM is open source software.

4.1.4.2 VirtualBox

VirtualBox is a general-purpose full virtualizer for x86 hardware, targeted at server, desktop and embedded use. It is a hosted hypervisor (sometimes referred to as a "type 2" hypervisor). Whereas a "bare-metal" or "type 1" hypervisor would run directly on the hardware, VirtualBox requires an existing operating system to be installed. It can thus run alongside existing applications on the host environment.

The VirtualBox solution is portable to multiple operating systems: Windows, Linux and MacOS, thus eliminating host OS restrictions.

Oracle VM VirtualBox is an open-source virtualization solution which supports both software-based and hardware-based virtualization (Intel VT-x and AMD-V). Regarding the the hard-disk emulation, VirtualBox supports the following:

- VDI – a VirtualBox specific image
- VMDK- the same format used by VMWare
- VHD – the format used by Windows Virtual PC
- qcow – QEMU disks

4.1.4.3 XEN, LXC and Docker

Xen is a hypervisor using a microkernel design, providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.

LXC is a userspace interface for the Linux kernel containment features. Through a powerful API and simple tools, it lets Linux users easily create and manage system or application containers.

Docker is an open-source project that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating-system-level virtualization on Linux. Docker uses the resource isolation features of the Linux kernel such as cgroups and kernel namespaces, and a union-capable file system such as aufs and others to allow independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines.

4.1.5 Android OS

4.1.5.1 Android 6 features

Android is an open-source mobile operating system and software platform. Initially developed by Google, Android is now managed by the Open Handset Alliance. The Android OS is based on Linux kernel and user-space applications run in a Java virtual machine (Dalvik). Android applications are usually developed by using the Java programming language (along with the Android SDK), but native applications can also be developed by using C/C++ (along with the Android NDK).

Android 6.0 (Marshmallow), the OS used to test the pilot mobile client application, is an Android OS version which bring a series of performance and security improvements. Regarding the Android 6.0 security improvements we enumerate the following:

- Runtime permission: applications require permissions on runtime, instead of requesting them when installed.
- Verified boot: the bootloader to OS elements are verified against a set of cryptographic keys.
- Hardware isolated security: a new hardware abstraction layer used by fingerprint API, lockscreen, device encryption and client certificates has to role protects keys against kernel compromise.
- Fingerprints: the applications can use a fingerprint API as security element.
- Clear text traffic: the developers can use a StrictMode to ensure that an application does not use cleartext.
- System hardening: this feature is achieved by enforcing SELinux policies.

Along with the Android OS runs Trusty, a set of software components which support a TEE.

Trusty has the following elements:

- The Trusty OS operating system which runs in a TEE.
- Android drivers used to communicate with applications running under Trusty OS.
- A set of Android libraries used in communication with Trusty OS.

Applications which run under the Trusty OS kernel are isolated processes and each process has its own memory sandbox (by using the MMU capabilities of the TEE processor). Trusty based applications are written in C/C++ and they have access to a simple C library. These applications are initialized once during load and reside in memory until TEE processor is reset. Taking this into consideration Trusty applications are simple and have an event-driven architecture. Trusty based applications are developed by a single party and packed into the Trusty kernel. The Trusty kernel image is signed and verified during the boot process, thus third-parties cannot run applications in TEE mode.

4.1.5.2 Hardware backed keystore

Regarding the security features, Android 6.0 has the concept of user-authentication-gated cryptographic keys. This concept protects the cryptographic keys storage and usage from unauthorized users. In order to achieve this feature, the Android authentication components Gatekeeper (PIN/pattern/password) and Fingerprint communicate their state with a KeyStore service via a secured channel.

The presence of trusted execution environments permits the enhancement of security measures regarding the protection of cryptographic keys by using hardware backed keystore. Android 6.0 keystore has symmetric cryptographic primitives (AES and HMAC) and employs an access control system for hardware backed keys. The keystore uses a Keymaster Hardware Abstraction Layer (HAL). Android 6.0 keystore offers a broader range of capabilities compared to its predecessors. Besides expanding the cryptographic primitives operations Android 6.0 presents the following capabilities:

- A usage control scheme to mitigate keys misuse situations
- An access control scheme to ensure the key usage only by the authorized users

The Keymaster HAL is OEM-provided and provides hardware-backed cryptographic services. Sensitive operations are executed only in a secure environment, which is reached through a kernel interface.

4.1.5.3 KeyGuard manager (Fingerprint)

One of the main Android 6.0 features is the fingerprint system, used in our work for the FIDO implementation. Android 6.0 uses a Fingerprint HAL in order to connect to a vendor specific fingerprint sensor. Regarding the Fingerprint architecture, the HAL interacts with the following elements:

- The Fingerprint Manager API – it is an application instantiated element which communicates with the Fingerprint Service
- FingerprintService - a system service which communicates with fingerprint daemon
- fingerprintd (Fingerprint daemon) – is a process which wraps vendor specific Fingerprint HAL
- Fingerprint HAL vendor specific library

- KeyStore API and Keymaster – provide hardware-backed cryptography for secure key store in a TEE

Fingerprint data is processed and stored in a secure execution environment and storage and it is not compromised even though the device is rooted. The vendor specific fingerprint operations from the Fingerprint HAL operate in TEE.

Android 6.0 offers the following guidelines in order to protect the fingerprint data:

- Raw fingerprint data must not be accessible only from outside the sensor driver or TEE.
- Fingerprint acquisition, enrollment and recognition must be performed inside the TEE.
- The fingerprint data must be stored into the file system by using encryption.
- The fingerprint templates must be signed with a private, device-specific key in order not to make them operable on another device.
- The fingerprint data must be erased when the user is erased from the system.

Android 6.0 presents two features which help power consumption management capabilities: background CPU and network activity is deferred when the device is unused for long period of time (Doze) and background and network activity is deferred for applications with which the user has not recently interacted (App Standby).

While in Doze mode the Android system has the following restrictions: network access is suspended, wake locks are ignored, alarms are deferred, Wi-Fi scans are not performed, sync adapters and job schedulers do not run. Android recommends using the GCM (Google Cloud Messaging) for interacting with the application while in Doze mode. Also Doze permits application white listing.

4.2 ReCRED Components

4.2.1 Authorization Server

The Authorization Server is an integral part of the Wi-Fi pilot. It acts as the Relying Party and it is responsible for performing Authorization to the users. To this end, the Authorization Server consists of an OpenID connect Client and a REST API for revealing useful information to the user about the resources and the required attributes. Furthermore, it maintains an LDAP directory so that communication between CUT’s components and ReCRED components can be performed. Finally, it has an Access Control Policies module which consists of a Web front-end, which allows the administrator to manually add policies, and a respective Machine Learning back-end for suggesting new policies. The various components of the Authorization Server are described below.

4.2.1.1 OpenID Connect Client

In the Wi-Fi Pilot, the Authorization Server will be configured as an OpenID Connect Client (Relying Party). It will interact with the Authentication Server that runs the OpenAM solution and is configured as an OpenID connect provider. The Authorization Server is configured to follow the authorization code flow of OpenID Connect and it is extended in order to support the custom scopes and claims according to the ReCRED Wi-Fi pilot needs.

The user will first access the ReCRED-developed smartphone application and then using that application, the user will access the Authorization Server. The Authorization Server is responsible to interact with the Authentication Server so that the OpenID connect authorization code flow can be performed. To achieve this, the Authorization Server provides an interface that makes the appropriate calls to the OpenAM instance that is running at the Authentication Server and acts as the OpenID connect provider. Subsequently, the Authorization Server receives the replies from the Authentication Server and once it receives the verified user info then it continues with the Authorization of the user according to the Access Control Policies module.

4.2.1.2 REST API

The Authorization Server provides a REST interface to the ReCRED mobile application. The mobile application is able to acquire important information regarding the identity attributes and the available resources. Specifically, the Authorization Server provides the following:

- The mobile application can provide a set of identity attributes and acquire the set of resources that can be granted.
- The mobile application can request and get all the available resources
- The mobile application can request which set of identity attributes are required in order for the user to gain access to a specific resource.

The aforementioned operations are performed using the REST API calls that are described below.

4.2.1.2.1 GET /v1/public/resources

Description: Provides a list of the available network resources

REQUEST

```
GET /v1/public/resources
Accept: application/json
```

RESPONSE EXAMPLE

```
200
Content-Type: application/json

{
"resources":   ["Internet", "Research Network", "Webmail"]
}
```

Response Body: The response body contains a list with all the available resources.

4.2.1.2.2 POST /v1/public/policies

Description: Provides a list of the resources that can be gained by revealing the attributes that are provided in the request body. Note that we only provide the attributes without the attribute values.

REQUEST

```
POST /v1/public/policies
Accept: application/json

{
  "first_name" : "",
  "last_name" : "",
  "age" : "",
  "nationality": "",
  "religion": "",
  "country": "",
  "city": "",
  "phone": "",
  "email": "",
  "title": "",
  "department": "",
  "year_study": "",
   "semester": "",
   "teaching_years": "",
   "scholarship": "",
   "courses": ""

}
```

RESPONSE EXAMPLE

```
200
Content-Type: application/json

[{"idaccess_policies": 2, "resource": "Internet"}, {"idaccess_policies": 3, "resource": "Internet"}, {"idaccess_policies": 7, "resource": "Research Network"}]
```

Request Body: In the request body we can have any set of attributes that are used for the Wi-Fi pilot

Response Body: The response is a list with the resources that can be revealed when we provide the attributes that we listed in the request body. Note that the idaccess_policies is returned for debugging reasons.

4.2.1.2.3 GET /v1/public/policies/<resource>

Description: Returns a list of all the possible combinations of identity attributes that must be revealed in order to gain access to the requested resource.

REQUEST
`GET /v1/public/policies/<resource>` `Accept: application/json`
RESPONSE EXAMPLE

`200`
`Content-Type: application/json`
 	
[{"department": "", "title": ""}, {"courses": "", "title": ""}]

Request URI: In the request URI we provide the name of the resource that we want.

Response Body: Provides a list with all the possible combinations of identity attributes that can be used in order to gain access to the specified resource. For instance, in the above response example the requested resource can be granted by either revealing the department and title attributes or the courses and title attributes.

4.2.1.3 LDAP Directory

As mentioned before, the Authorization Server maintains an LDAP directory so that the CUT’s Identity Service Engine can grant resources to a user. Specifically, the Authorization Server maintains a mapping between groups and resources. A group may consist of multiple resources. After the completion of an Authorization request, the Authorization Server decides in which group the user should be placed according to the resources that he should gain access to. The Authorization Server runs the appropriate LDAP back-end operation in order to dynamically change a user’s group. After the update of the group, the CUT ISE reads the LDAP directory and retrieves a user’s group. Finally, the ISE is responsible for applying and granting the resources according to the selected group.

4.2.1.4 Access Control Policies

The Authorization Server provides a Web Front-end to the network administrator to enable the addition of access control policies. Also, it has an ABAC policy reasoning and creation module (Machine Learning back-end) that uses advanced machine learning techniques for complex attribute-based access control policy reasoning and creation. In other words, this module is responsible to provide suggestions for new access control policies to the system administrator.

The Inputs and the Output of the module are presented below.

Inputs:

The inputs of the module are four matrices:

1. The users(U)/attributes(A) matrix,
2. The resources(R)/users(U) matrix,
3. The resources/level of criticality matrix, and

4. The attributes(A)/level of assurance (LOA) matrix.

An example of the four matrices is presented below.

Table 1: The resources (R)/users (U) matrix.

R/U		U1	U2	U3	U4	U5
Electrical Engineer Lab network (print)	**[R0]**			x		
Multimedia and Graphics Arts Lab network (use BitTorrent)	**[R11]**	x				
Moodle (upload content - Virtual Reality)	**[R42]**	x				
Moodle (upload content – Machine Learning)	**[R243]**					
Moodle (view the content - Operating Systems)	**[R367]**					x
Moodle (view the content – Electronic Commerce)	**[R502]**				x	x
Moodle (post messages to the Control Systems forum)	**[R601]**			x		
Moodle (read messages on the Control Systems forum)	**[R895]**			x		x
Webmail	**[R996]**	x	x	x	x	
Internet	**[R997]**	x	x	x	x	x
Research network (print)	**[R998]**	x	x	x		x
Research network (use BitTorrent)	**[R999]**			x		x

This matrix shows which resource (e.g. Moodle) will be granted to users and with which permissions (e.g. only U1 should be able to upload content for the Virtual Reality course). A subset of users and a subset of resources (permissions) are presented in the above matrix.

For example:

- U2 can get access to the Internet, the Webmail and the Research network with the permission to print.
- U5 can get access to the Internet, the Moodle with the permission to view the content of the Operating Systems and Electronic Commerce courses, he can also get access to the forum of the Control Systems course with the permission to read the messages and finally can get access to the Research network with the permission to print and use BitTorrent.

This matrix should be transformed to three columns corresponding to the users, the resources and the decisions in order to be able to be used from the machine learning techniques of the module. The result of the transformation is presented below with a small part of the columns.

Table 2: the The users, the resources and the decisions columns

Users	Resources	Decision
U0	R0	x
U0	R11	
U1	R11	x
U1	R367	
U2	R42	
U2	R998	x
U3	R997	x
U3	R998	x
U4	R999	
U5	R367	x
U5	R895	x
U5	R999	

Table 3: The resources (R)/level of criticality (LOC) matrix

RESOURCE	LEVEL OF CRITICALITY
Electrical Engineer Lab network (print) [R0]	2
Multimedia and Graphics Arts Lab network (use BitTorrent) [R11]	3
Moodle (upload content - Virtual Reality) [R42]	4
Moodle (upload content – Machine Learning) [R243]	4
Moodle (view the content - Operating Systems) [R367]	1
Moodle (view the content – Electronic Commerce) [R502]	1
Moodle (post messages to the Control Systems forum) [R601]	3
Moodle (read messages on the Control Systems forum) [R895]	1

RESOURCE	LEVEL OF CRITICALITY
Webmail [R996]	3
Internet [R997]	1
Research network (print) [R998]	2
Research network (use BitTorrent) [R999]	3

This matrix shows the level of criticality of each resource. The LOC 4 is the highest level of criticality. A subset of resources (permissions) with the corresponding level of criticality is presented in the above matrix.

For example:

- The Moodle resource with the permission of content uploading for a course has the highest level of criticality (LOC 4). It is extremely important only the appropriate professors get access to this resource because they should also upload the exam grades for the course.
- The Internet resource has the lowest level of criticality (LOC 1) because it is not critical for a user to get access to this resource.

Table 4: The users (U)/ attributes (A) matrix

U/A	Title [A14]	First Name [A0]	Department [A15]	Year of Study [A18]	Chosen Courses [A21]	Teaching Courses [A22]
U0	Professor	Kostas	Electrical Engineer, Computer Engineer and Informatics	-	-	-Pattern Recognition -Machine Learning -Operating Systems
U1	Professor	Christos	Multimedia and Graphic Arts	-	-	-Virtual Reality -Informatics -Design and Computing
U2	Student	George	Communication and Internet Studies	2	-Communication Theory -Internet Technologies -Web Design -Alternative Media	-

U/A	Title [A14]	First Name [A0]	Department [A15]	Year of Study [A18]	Chosen Courses [A21]	Teaching Courses [A22]
U3	Student	John	Electrical Engineer, Computer Engineer and Informatics	4	-Control Systems -Electronics II -Electrical Lab III -VLSI Systems Design	-
U4	Student	Maria	Commerce, Finance and Shipping	1	-Financial Accounting -Electronic Commerce -Marketing Principles -Business Ethics	-
U5	Guest	-	-	-	-	-

This matrix shows the attribute values of each user. A subset of users and a subset of attributes with the respective attribute values are presented in the above matrix.

For example:

- U2 is a student with the first name George, belongs to the department of Communication and Internet Studies. He is in the second year of his study. He chose the following courses: Communication Theory, Internet Technologies, Web Design and Alternative Media.
- U5 is a guest, he is a person without credentials so he must be vouched by a registered user in order to get access to some resources.

This matrix should be transformed to users(U)/attribute values(AV) matrix in order to be able to be used from the machine learning techniques of the module. A part of the transformation is presented below.

Table 5: The users (U)/ attributes values (AV) matrix.

U/AV	Student	Professor	Guest	Electrical Engineer Department	Multimedia and Graphic Arts Department	Second Year of Study	Fourth Year of Study	Virtual Reality Teaching Course	Machine Learning Teaching Course	Electronics II Chosen Course
U0		x		x					x	

U/AV	Student	Professor	Guest	Electrical Engineer Department	Multimedia and Graphic Arts Department	Second Year of Study	Fourth Year of Study	Virtual Reality Teaching Course	Machine Learning Teaching Course	Electronics II Chosen Course
U1		x			x			x		
U2	x					x				
U3	x			x			x			x
U4	x									
U5			x							

Table 6: The attribute (A)/level of assurance (LOA) matrix.

ATTRIBUTE		LEVEL OF ASSURANCE
Title	**[A14]**	4
First Name	**[A0]**	4
Department	**[A15]**	4
Year of Study	**[A18]**	4
Chosen Courses	**[A21]**	3
Teaching Courses	**[A22]**	2

This matrix shows the level of assurance of each attribute. The LOA 4 is the highest level of assurance. A subset of attributes with the corresponding level of assurance is presented in the above matrix.

For example:

- The **Title** attribute has the highest level of assurance (LOA 4) because the certificate which is used to prove this attribute is very trusted.
- The **Teaching Courses** attribute has a LOA 2 because the certificate which is used to proof this, is not very trusted. A teaching course may be changed between the professors in a semester either for health reasons of a professor or for academic reasons.

This matrix should be transformed to attribute values(AV)/level of assurance(LOA) matrix in order to be able to be used from the machine learning techniques of the module. A part of the transformation is presented below.

Table 7: The attribute values (AV)/level of assurance (LOA) matrix

ATTRIBUTE VALUES	LEVEL OF ASSURANCE
Student	4
Professor	4
Guest	4
Electrical Engineer Department	4
Multimedia and Graphic Arts Department	4
Second Year of Study	4
Fourth Year of Study	4
Virtual Reality Teaching Course	2
Machine Learning Teaching Course	2
Electronics II Chosen Course	3

<u>Output:</u>

The output of the module is the resource(R)/attribute values(AV) matrix. This matrix shows the attribute values that needed to access a resource. Basically, this matrix provides suggestions for attribute-based access control policies to the system administrator. A part of this matrix is presented below.

Table 8: The resources (R)/attribute values (AV) matrix

R/AV	Student	Professor	Guest	Electrical Engineer Department	Multimedia and Graphic Arts Department	Second Year of Study	Fourth Year of Study	Virtual Reality Teaching Course	Electronics II Chosen Course
Electrical Engineer Lab network (print)		x		x					
Electrical Engineer Lab network	x			x			x		

R/AV	Student	Professor	Guest	Electrical Engineer Department	Multimedia and Graphic Arts Department	Second Year of Study	Fourth Year of Study	Virtual Reality Teaching Course	Electronics II Chosen Course
(print)									
Multimedia and Graphic Arts Lab network (use BitTorrent)		x			x				
Moodle (upload content - Virtual Reality)		x			x			x	
Moodle (view content – Electronics II)	x								x
Moodle (post messages - Virtual reality forum)		x			x			x	
Moodle (read messages - Virtual reality forum)			x						
Webmail	x								
Webmail		x							
Internet			x						
Research network (print)		x							
Research network (print)	x						x		

Examples of suggestions:

- “If a user is a **professor**, belongs to the **Multimedia and Graphic Arts Department** and he is responsible for the **teaching of the Virtual Reality course**”, he can get access to the Moodle with the permission to upload content for the Virtual Reality course.
- “If a user is a **student** and he is in the **fourth year of his study**”, he can get access to the Research network with the permission to print.
- “If a user is a **guest**”, he can get access to the Moodle with the permission to read the messages of the Virtual Reality forum.

This matrix should be transformed to resources(R)/attributes (A) matrix in order to be able to be stored in the Authorization’s Server access control policies database. A part of the transformation is presented below.

Table 9: The resources (R)/attributes (A) matrix

R/A	Title [A14]	Department [A15]	Year of Study [A18]	Chosen Courses [A21]	Teaching Courses [A22]
Electrical Engineer Lab network (print)	Professor	Electrical Engineer			
Electrical Engineer Lab network (print)	Student	Electrical Engineer	4		
Multimedia and Graphic Arts Lab network (use BitTorrent)	Professor	Multimedia and Graphic Arts			
Moodle (upload content - Virtual Reality)	Professor	Multimedia and Graphic Arts			Virtual Reality
Moodle (view content – Electronics II)	Student			Electronics II	
Moodle (post messages - Virtual reality forum)	Professor	Multimedia and Graphic Arts			Virtual Reality
Moodle (read messages - Virtual reality forum)	Guest				
Webmail	Student				
Webmail	Professor				
Internet	Guest				

R/A	Title [A14]	Department [A15]	Year of Study [A18]	Chosen Courses [A21]	Teaching Courses [A22]
Research network (print)	Professor				
Research network (print)	Student		4		

4.2.1.4.1 Machine Learning Techniques

The machine learning techniques which are used in our case are presented below.

Matrix Factorization (MF):

Matrix factorization is a popular collaborative filtering technique, assuming that the matrix of ratings can be written as the inner product of two low-rank matrices, comprising latent (unobserved) features assigned to each user and item pair.

In our case, the ratings are the decisions and the items are the resources. There are latent features for the resources but the features of the users are known (observed). Specifically, the features of the users are the attribute values of them.

Bayesian Matrix Factorization:

In our case, we use Bayesian matrix factorization instead of the normal matrix factorization. Bayesian Matrix Factorization is an inference approach which infers posterior distribution over the resource latent features and replaces the inner product with an arbitrary function that we learn from the data at the same time as we learn the latent feature posterior for the resources. We use multi-layer feed-forward neural network (MLFFNN) to achieve this.

The advantages of using Bayesian inference approaches are that can better model sparse data and can take uncertainty into account.

Variational Bayesian (VB) inference:

The latent_feature posterior distribution for the resources is intractable so we use the Variational Bayesian (VB) inference which is a popular approximate inference technique for probabilistic graphical models with the aim of obtaining a variational approximation to the intractable posterior by optimizing a lower bound to the intractable log-marginal likelihood of the model.

Stochastic Gradient Variational Bayes (SGVB) estimator:

Unfortunately, the Variational Bayesian (VB) approach requires analytical expressions of the expectations of the model latent variables w.r.t the approximate posterior so we use the Stochastic Gradient Variational Bayes estimator (SGVB) which is a recently proposed technique with the aim of reparameterizing the variational lower bound and optimize it using standard stochastic gradient ascent techniques.

4.2.2 Authentication Server

The Authentication Server is the one of the key components in this pilot. It acts as the Identity Provider with the responsibility to authenticate end-users and to provide identity attributes to the Authorization Server (RP). To achieve this, a custom authentication module has been configured on the Authentication Server to dictate which FIDO Endpoint the client should contact to in order to perform the FIDO UAF Authentication. In addition to that, the Authentication Server is also responsible to interface with the FIDO server to know the outcome of this transaction in order to issue the authorization code to the Authorization Server (RP). The Authorization Server (RP) can then use this authorization code to get an access and ID token from the token endpoint of the Authentication Server.

4.2.2.1 OpenAM

ForgeRock OpenAM is a web-based open-source solution that provides Authentication, Authorization, Entitlement, and Federation services. OpenAM out-of-the-box supports 25 authentication methods and offers the flexibility to create custom authentication modules based on the JAAS (Java Authentication and Authorization Service) open standard. In addition to that, OpenAM’s federation services allow federated members to securely share identity information with each other.

For the Wi-Fi pilot, the Authentication Server uses ForgeRock OpenAM 13 which has been deployed on Tomcat 7. The custom authentication module has also been configured on OpenAM to interface with the FIDO server. In addition to that, OpenAM has also been configured as an OpenID Connect Provider for this pilot.

4.2.2.1.1 Configure Authentication Server as an OpenID Connect Provider

OpenAM has been configured as an OpenID Connect provider for the Wi-Fi pilot. Steps have been documented below to explain as how to configure OpenAM as an OpenID Connect Provider:

1. Login to OpenAM as amadmin.
2. Click on Configure OAuth2/OpenID Connect.

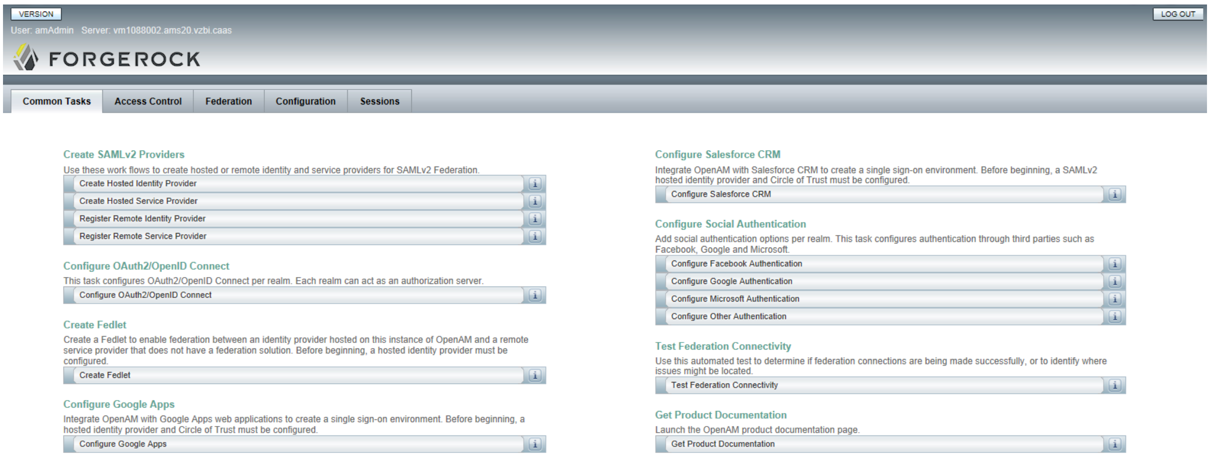


Figure 3: Configure OpenAM as an OpenID Connect provider

3. Following configuration creates OAuth2Provider Service in Top level Realm. Adjust the lifetime of the tokens.

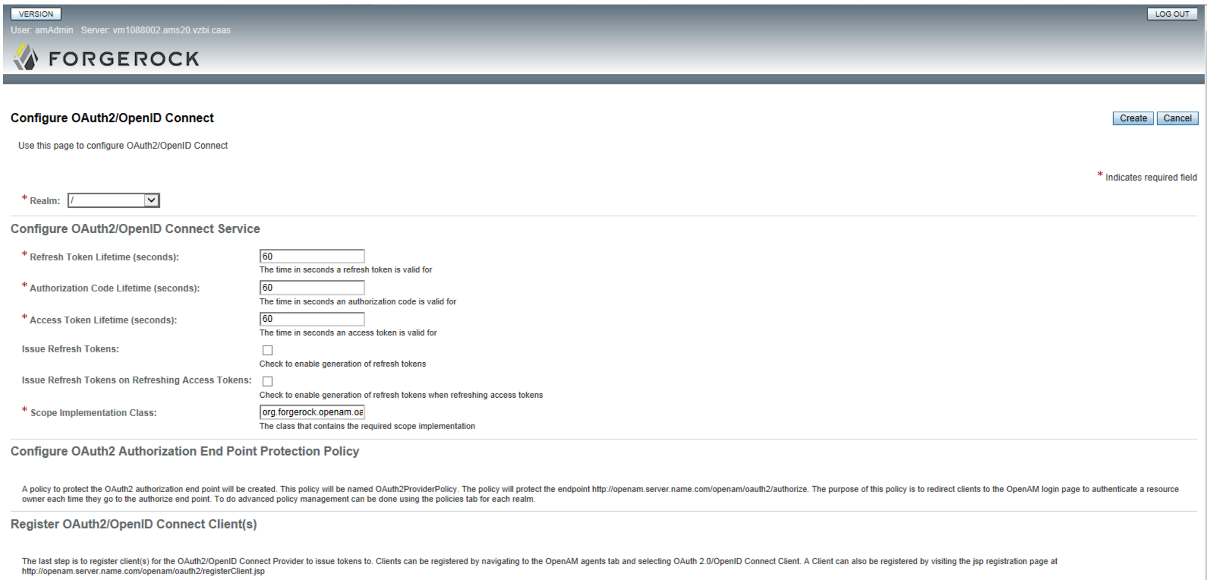


Figure 4: Configure lifetime of the tokens

4. Click on Create.
5. OAuth2Provider Service and Policy is created for Top Level Realm.

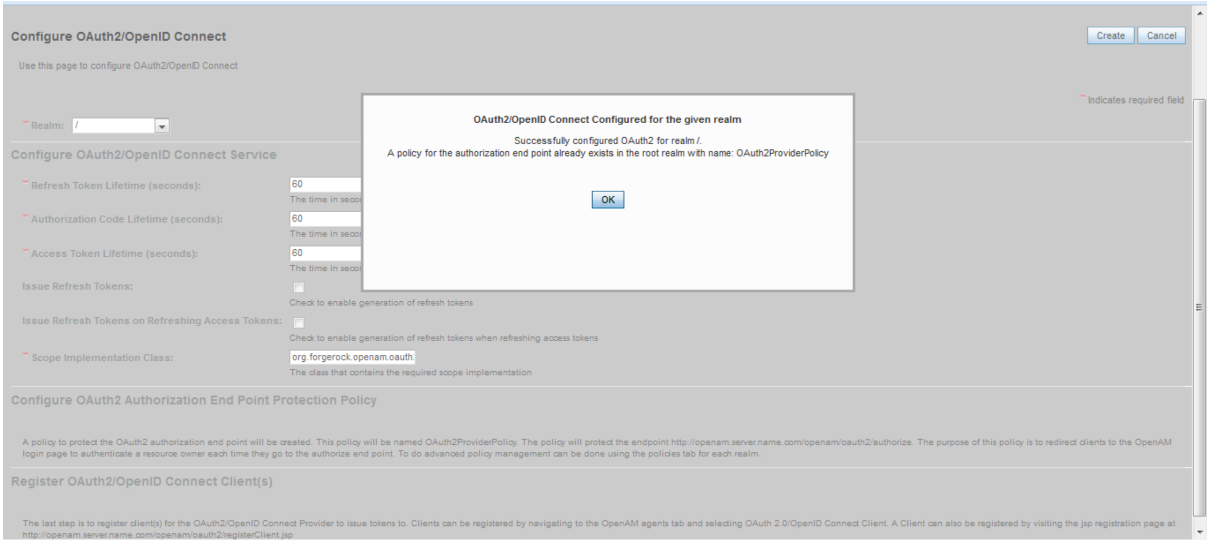


Figure 5: Creation of OAuth2Provider Service and Policy for Top Level Realm

6. Further modification in the configuration of OIDC Provider is possible after creating it in the OpenAM console under Access Control > Realm Name > Services > OAuth2 Provider.
7. Click on OAuth2 Provider to adjust the settings.

Figure 6: Adjust the settings of OAuth2 Provider

4.2.2.1.2 Create Custom Profile Attributes

For the Wi-Fi pilot, OpenAM has been configured to support custom profile attributes. The custom profile attributes can then be used to create custom OIDC scopes and claims.

Steps have been documented below to explain as how to create custom profile attributes in OpenAM:

1. SSH to OpenAM server and then using any text editor open the following file: /var/lib/tomcat/webapps/openam/WEB-INF/classes/amUser.xml

2. Add the following lines under the <user> section of amUser.xml:

```
<AttributeSchema name="university"
              type="single"
              syntax="string"
              any="display"
              i18nKey="University">
```

```
                </AttributeSchema>
      <AttributeSchema name="department"
                    type="single"
                    syntax="string"
                    any="display"
                    i18nKey="department">
                </AttributeSchema>
```

3. Locate ssoadm script and then run the following commands:

```
./ssoadm delete-svc --adminid amadmin --password-file /tmp/pwd.txt -
-servicename iPlanetAMUserService
./ssoadm create-svc --adminid amadmin --password-file /tmp/pwd.txt -
-xmlfile /var/lib/tomcat/webapps/openam/WEB-INF/classes/amUser.xml
```

4. Create custom LDIF file by typing the following command:

```
touch /root/extendschema.ldif
```

5. Add the following lines to extendschema.ldif file:

```
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( temp-custom-attr2-oid NAME 'university' EQUALITY
case
 IgnoreMatch ORDERING caseIgnoreOrderingMatch SUBSTR
caseIgnoreSubstrings
 Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
attributeTypes: ( temp-custom-attr3-oid NAME 'department' EQUALITY
case
 IgnoreMatch ORDERING caseIgnoreOrderingMatch SUBSTR
caseIgnoreSubstrings
 Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
-
add: objectClasses
objectClasses: ( temp-custom-oc2-oid NAME 'testuniversity' SUP top
AUXILIARY
  MAY university )
  add: objectClasses
objectClasses: ( temp-custom-oc3-oid NAME 'department' SUP top
AUXILIARY
  MAY department )
-
```

6. Locate ldapmodify script and then run the following command:

```
./ldapmodify --port 50389 --hostname openam.example.net --bindDN
"cn=Directory Manager" --bindPassword <Enter Password> --filename
/root/extendschema.ldif
```

7. Restart tomcat service.
8. Login to OpenAM web console using amadmin.
9. Click on Access Control -> / (Top Level Realm) -> Data Stores -> OpenDJ

10. Add user object class (e.g. testuniversity) in the LDAP User Object Class list.

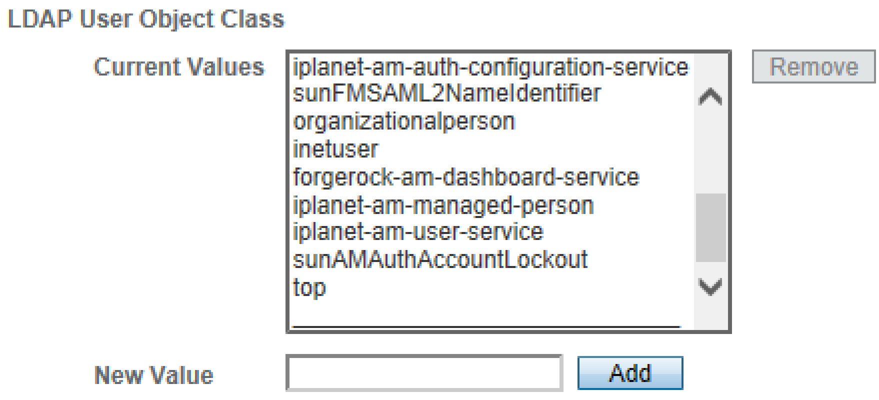


Figure 7: Update LDAP User Object Class list

11. Add LDAP user attributes (e.g. university) in the LDAP User Attributes list:

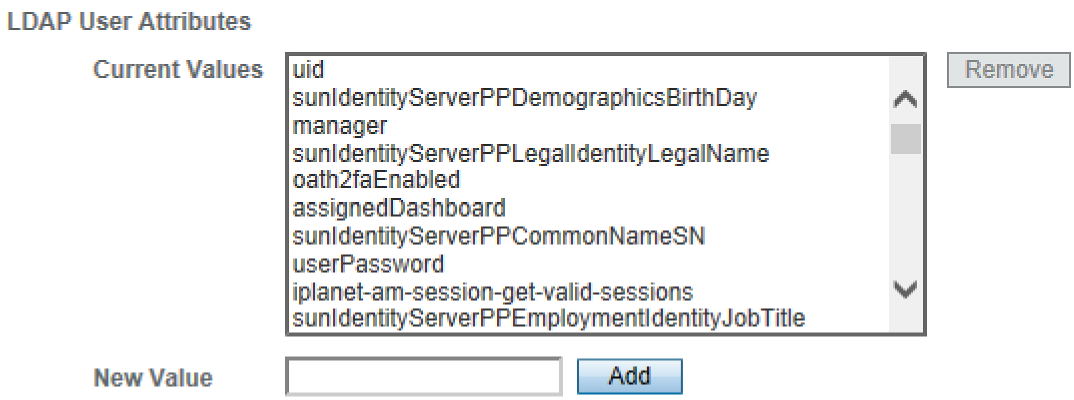


Figure 8: Update LDAP User Attributes list

12. Click on Save.

4.2.2.1.3 Create Custom OIDC Scopes and Claims

For the Wi-Fi pilot, OpenAM has been configured to support custom OIDC scopes and claims. Once the relying party (Authorization Server) gets an access token (access_token), it can then use this access token to get identity attributes of the end-user from the UserInfo Endpoint of the OIDC Authorization Server.

Steps have been documented below to explain as how to create custom OIDC scopes and claims in OpenAM:

1. Click on Access Control -> / (Top Level Realm) -> Services -> OAuth2 Provider
2. Edit OIDC Claims Script and add the following lines under claimAttributes section:

```
"department": attributeRetriever.curry("department"),
"university": attributeRetriever.curry("university")
```

   Also add the following lines under scopeClaimsMap section:

```
"department": [ "department" ],
"university": [ "university" ]
```

3. Click on Upload to upload the edited script.

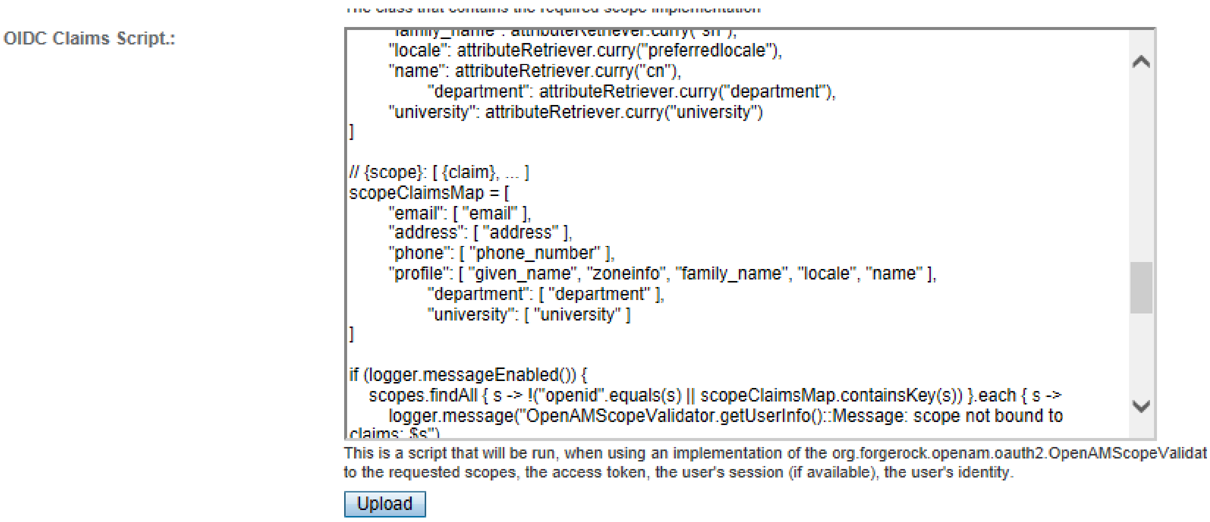


Figure 9: Upload the OIDC Claims Script

4. Add custom scopes (e.g. department) in the Supported Scopes list.

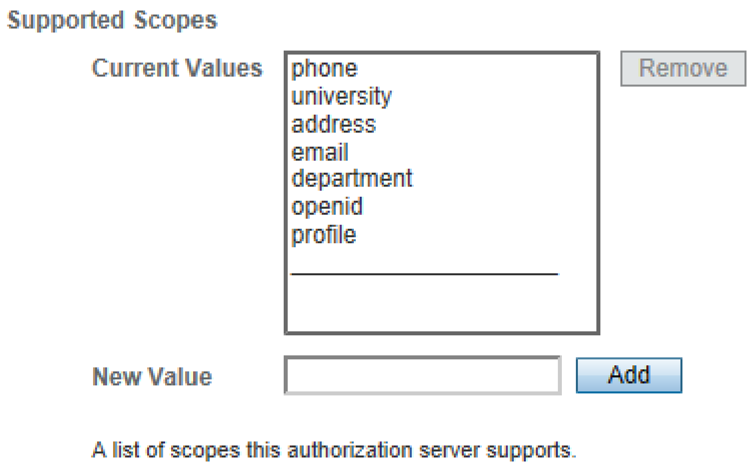


Figure 10: Update Supported Scopes list

5. Add custom claims (e.g. department) in the Supported Claims list:

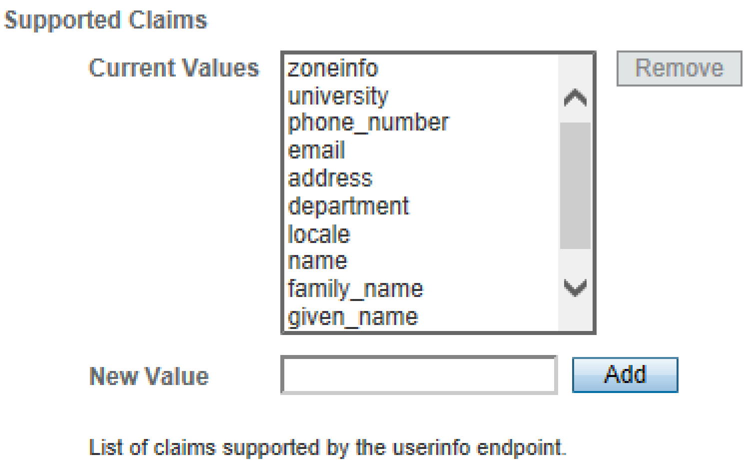


Figure 11: Update Supported Claims list

6. Add custom scopes (e.g. department) in the Default Client Scopes list.

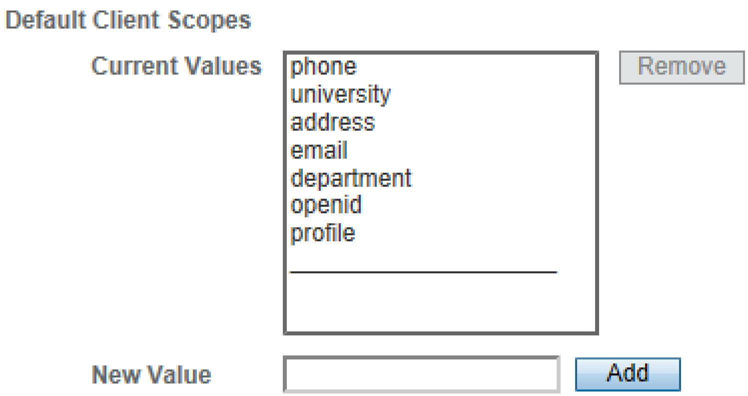


Figure 12: Update Default Client Scopes

4.2.2.1.4 Create Custom Authentication Module

The custom authentication module provides the mobile application with the FIDO-server after receiving the username. The mobile application lets the user authenticate using the FIDO-server. When the user is successfully authenticated the custom authentication module verifies this with the FIDO-server and this result in a session.

This is done in three stages which are as follow:

- The first stage asks for the username. The response contains a callback that should be used to post the username to the /authenticate endpoint.
- The second stage gets the username and reply with a fixed FIDO-server.
- The third stage checks the authentication at the FIDO-server using a REST call to obtain the username and timestamp.

The authentication module is a standard OpenAM extension; using out of the box extension points. More detail on technicalities of OpenAM authentication module development and configuration can be found on:
https://wikis.forgerock.org/confluence/display/openam/Write+a+custom+authentication+module

4.2.2.1.5 Authorization Server (RP) Registration

To register Authorization Server (Relying Party or OIDC Client) into OpenAM, OpenID Connect Client needs to be created with Relying Party details.

1. Login to openam as amadmin
2. Go to Top Level Realm → Agents →OAuth2.0/OpenIDConnect Client
3. Click on new.
4. Enter Client ID in the name field and Client Secret in the password field.

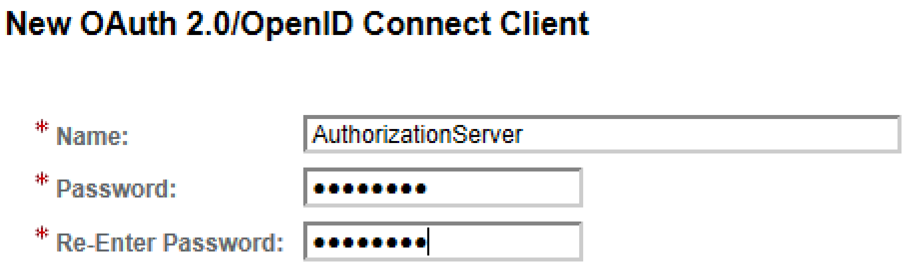


Figure 13: Create OpenID Connect Client

5. Enter Redirection URI and Scope as mentioned in the below screen shot.

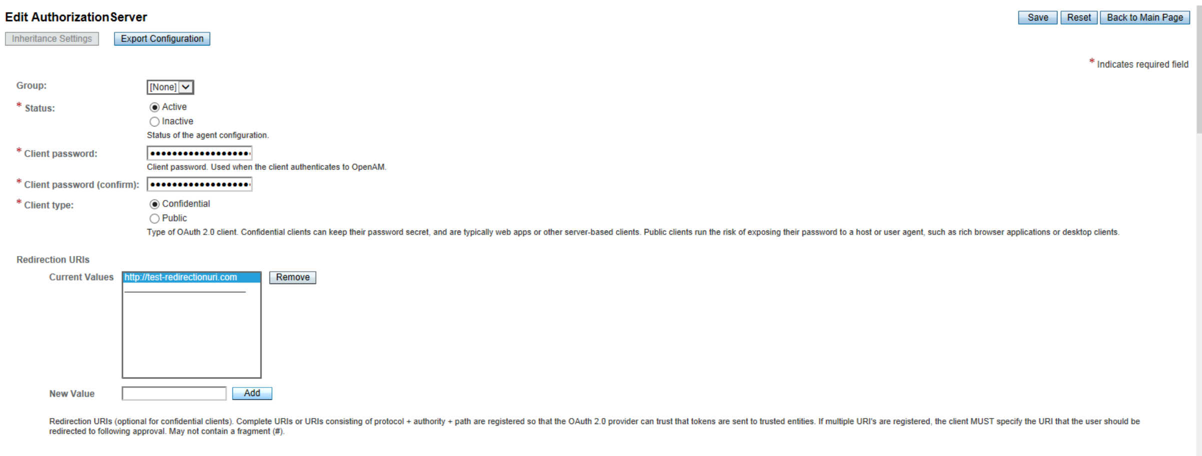


Figure 14: Define Redirection URI and Scope for OpenID Connect Client

4.2.2.1.6 Authentication Server APIs

Step 1: Start Authentication

```
curl --request POST --header "Content-Type: application/json"
https://isic.vzgcis.com:443/openam/json/authenticate -k
```

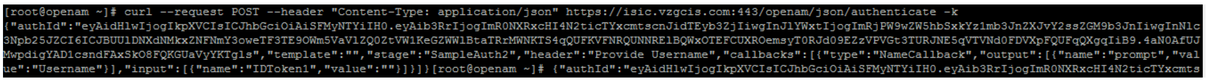


Figure 15: Start Authentication

Step 2: Provide Username

```
curl --request POST --header "Content-Type: application/json" --data
'{"authId":"eyAidHlwIjogIkpXVCIsICJhbGciOiAiSFMyNTYiIH0.eyAib3RrIjog
ImR0NXRxcHI4N2ticTYxcmtscnJidTEyb3ZjIiwgInJlYWxtIjogImRjPW9wZW5hbSxk
Yz1mb3JnZXJvY2ssZGM9b3JnIiwgInNlc3Npb25JZCI6ICJBUU1DNXdNMkxZNFNmY3ow
eTF3TE9OWm5VaVlZQ0ZtVW1KeGZWWlBtaTRrMWNKTS4qQUFKVFNRQUNNRElBQWxOTEFC
UXROemsyT0RJd09EZzVPVGt3TURJNE5qVTVNd0FDVXpFQUFqQXgqIiB9.4aN0AfUJMwp
digYAD1csndFAxSkO8FQKGUaVyYKTgls","template":"","stage":"SampleAuth2
","header":"Provide
Username","callbacks":[{"type":"NameCallback","output":[{"name":"pro
mpt","value":"Username"}],"input":[{"name":"IDToken1","value":"91991
0431224"}]}]}'  https://isic.vzgcis.com:443/openam/json/authenticate
-k
```

Figure 16: Provide Username

Step 3: Again provide username

```
curl --request POST --header "Content-Type: application/json" --data
'{"authId":"eyAidHlwIjogIkpXVCIsICJhbGciOiAiSFMyNTYiIH0.eyAib3RrIjog
ImR0NXRxcHI4N2ticTYxcmtscnJidTEyb3ZjIiwgInJlYWxtIjogImRjPW9wZW5hbSxk
Yz1mb3JnZXJvY2ssZGM9b3JnIiwgInNlc3Npb25JZCI6ICJBUU1DNXdNMkxZNFNmY3ow
eTF3TE9OWm5VaVlZQ0ZtVW1KeGZWWlBtaTRrMWNKTS4qQUFKVFNRQUNNRElBQWxOTEFC
UXROemsyT0RJd09EZzVPVGt3TURJNE5qVTVNd0FDVXpFQUFqQXgqIiB9.4aN0AfUJMwp
digYAD1csndFAxSkO8FQKGUaVyYKTgls","template":"","stage":"SampleAuth3
","header":"FIDO                                        Server
Details","callbacks":[{"type":"NameCallback","output":[{"name":"prom
pt","value":"https://recred-dev.certsign.ro:8443/fidouaf-
ci"}],"input":[{"name":"IDToken1","value":"919910431224"}]}]}'
https://isic.vzgcis.com:443/openam/json/authenticate -k
```

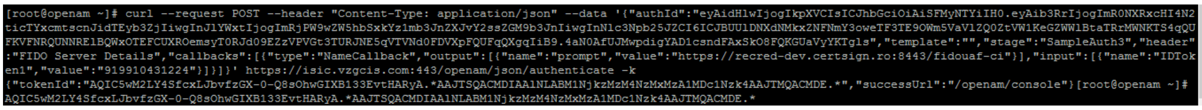
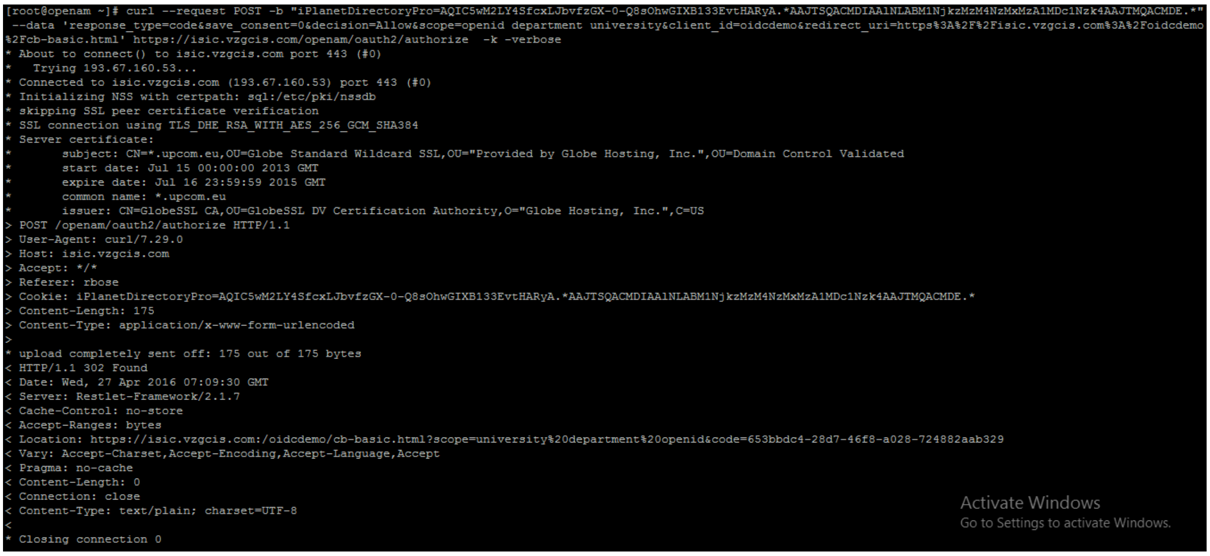


Figure 17: Again provide username

Step 4: Get Authorization Code

```
curl --request POST -b "iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-
0-
```

```
Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4A
AJTMQACMDE.*"                                                    --data
'response_type=code&save_consent=0&decision=Allow&scope=openid
department
university&client_id=oidcdemo&redirect_uri=https%3A%2F%2Fisic.vzgcis
.com%3A%2Foidcdemo%2Fcb-basic.html'
https://isic.vzgcis.com/openam/oauth2/authorize  -k –verbose
```



```
[root@openam ~]# curl --request POST -b "iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*"
 --data 'response_type=code&save_consent=0&decision=Allow&scope=openid department university&client_id=oidcdemo&redirect_uri=https%3A%2F%2Fisic.vzgcis.com%3A%2Foidcdemo
%2Fcb-basic.html' https://isic.vzgcis.com/openam/oauth2/authorize  -k -verbose
* About to connect() to isic.vzgcis.com port 443 (#0)
*   Trying 193.67.160.53...
* Connected to isic.vzgcis.com (193.67.160.53) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
* skipping SSL peer certificate verification
* SSL connection using TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*       subject: CN=*.upcom.eu,OU=Globe Standard Wildcard SSL,OU="Provided by Globe Hosting, Inc.",OU=Domain Control Validated
*       start date: Jul 15 00:00:00 2013 GMT
*       expire date: Jul 16 23:59:59 2015 GMT
*       common name: *.upcom.eu
*       issuer: CN=GlobeSSL CA,OU=GlobeSSL DV Certification Authority,O="Globe Hosting, Inc.",C=US
> POST /openam/oauth2/authorize HTTP/1.1
> User-Agent: curl/7.29.0
> Host: isic.vzgcis.com
> Accept: */*
> Referer: rbose
> Cookie: iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*
> Content-Length: 175
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 175 out of 175 bytes
< HTTP/1.1 302 Found
< Date: Wed, 27 Apr 2016 07:09:30 GMT
< Server: Restlet-Framework/2.1.7
< Cache-Control: no-store
< Accept-Ranges: bytes
< Location: https://isic.vzgcis.com:/oidcdemo/cb-basic.html?scope=university%20department%20openid&code=653bbdc4-28d7-46f8-a028-724882aab329
< Vary: Accept-Charset,Accept-Encoding,Accept-Language,Accept
< Pragma: no-cache
< Content-Length: 0
< Connection: close
< Content-Type: text/plain; charset=UTF-8
<
* Closing connection 0
```


Figure 18: Get Authorization Code

Step 5: Get access token

```
curl --request POST -b "iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-
0-
Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4A
AJTMQACMDE.*"            --header            "Authorization:Basic
b21kY2RlbW86cGFzc3dvcmQ="                                      --data
'grant_type=authorization_code&redirect_uri=https%3A%2F%2Fisic.vzgci
s.com%3A%2Foidcdemo%2Fcb-basic.html&code=653bbdc4-28d7-46f8-a028-
724882aab329'
https://isic.vzgcis.com/openam/oauth2/access_token?service=apiServic
e -k –verbose
```

```
[root@openam ~]# curl --request POST -b "iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*"
 --header "Authorization:Basic b21kY2RlbW86cGFzc3dvcmQ=" --data 'grant_type=authorization_code&redirect_uri=https%3A%2F%2Fisic.vzgcis.com%3A%2Foidcdemo%2Fcb-basic.html&
code=653bbdc4-28d7-46f8-a028-724882aab329' https://isic.vzgcis.com/openam/oauth2/access_token?service=apiService -k --verbose
* About to connect() to isic.vzgcis.com port 443 (#0)
*   Trying 193.67.160.53...
* Connected to isic.vzgcis.com (193.67.160.53) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
* skipping SSL peer certificate verification
* SSL connection using TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*       subject: CN=*.upcom.eu,OU=Globe Standard Wildcard SSL,OU="Provided by Globe Hosting, Inc.",OU=Domain Control Validated
*       start date: Jul 15 00:00:00 2013 GMT
*       expire date: Jul 16 23:59:59 2015 GMT
*       common name: *.upcom.eu
*       issuer: CN=GlobeSSL CA,OU=GlobeSSL DV Certification Authority,O="Globe Hosting, Inc.",C=US
> POST /openam/oauth2/access_token?service=apiService HTTP/1.1
> User-Agent: curl/7.29.0
> Host: isic.vzgcis.com
> Accept: */*
> Cookie: iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*
> Authorization:Basic b21kY2RlbW86cGFzc3dvcmQ=
> Content-Length: 144
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 144 out of 144 bytes
< HTTP/1.1 200 OK
< Date: Wed, 27 Apr 2016 07:11:27 GMT
< Server: Restlet-Framework/2.1.7
< Cache-Control: no-store
< Accept-Ranges: bytes
< Vary: Accept-Charset,Accept-Encoding,Accept-Language,Accept
< Pragma: no-cache
< Content-Type: application/json;charset=UTF-8
< Connection: close
< Transfer-Encoding: chunked
<
* Closing connection 0
{"scope":"university department openid","expires_in":599,"token_type":"Bearer","refresh_token":"5ee4abcd-116b-4505-86c8-922f12169355","id_token":"eyAidHlwIjogIkpXVCIsIC
JhbGciOiAiSFMyNTYiIH0.eyAidG9rZW5OYW1lIjogImlkX3Rva2VuIiwgImF6cCI6ICJvaWRjZGVtbyIsICJzdWIiOiAiOTE5OTEwNDMxMjI0IiwgImF0X2hhc2giOiAiOVdSb0xJSmF5bURBaGVvRHJ2anhuZyIsICJpc3
MiOiAiaHR0cHM6Ly9pc2ljLnZ6Z2Npcy5jb206NDQzL29wZW5hbS9vYXV0aDIiLCAib3JnLmZvcmdlcm9jay5vcGVuaWRjb25uZWN0Lm9wcyI6ICI4OGI5NTAzNi00OWQxLTQ1YTgzOTA5MC04ZjMwMjUzMzMyMmYiLCAiaW
F0IjogMTQ2MTc0MTA4NywgImF1dGhfdGltZSI6IDE0NjE3NDA5MDMsICJleHAiOiAxNDYxNzQxNjg3LCAidG9rZW5UeXBlIjogIkpXVFRva2VuIiwgInJlYWxtIjogIi8iLCAiYXVkIjogWyAib2lkY2RlbW8iIF0sICJjX2
hhc2giOiAiOGlPLWxTaS00TDFHMmlWUDl4U0tBdyIgfQ.mcpD5AWk2y2G8Xv09sinp32ZMKB8BHhSX3O9XaiFzBA","access_token":"2d6910d5-741f-4b3e-93db-1a3bd388053c"}[root@openam ~]# 2d6910d
5-741f-4b3e-93db-1a3bd388053c
```

Figure 19: Get access token

Step 6: Get UserInfo

```
curl --request GET -b "iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*" --header "Authorization:Bearer 2d6910d5-741f-4b3e-93db-1a3bd388053c" https://isic.vzgcis.com/openam/oauth2/userinfo -k -verbose
```

```
[root@openam ~]# curl --request GET -b "iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*"
--header "Authorization:Bearer 2d6910d5-741f-4b3e-93db-1a3bd388053c" https://isic.vzgcis.com/openam/oauth2/userinfo -k -verbose
* About to connect() to isic.vzgcis.com port 443 (#0)
*   Trying 193.67.160.53...
* Connected to isic.vzgcis.com (193.67.160.53) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
* skipping SSL peer certificate verification
* SSL connection using TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*       subject: CN=*.upcom.eu,OU=Globe Standard Wildcard SSL,OU="Provided by Globe Hosting, Inc.",OU=Domain Control Validated
*       start date: Jul 15 00:00:00 2013 GMT
*       expire date: Jul 16 23:59:59 2015 GMT
*       common name: *.upcom.eu
*       issuer: CN=GlobeSSL CA,OU=GlobeSSL DV Certification Authority,O="Globe Hosting, Inc.",C=US
> GET /openam/oauth2/userinfo HTTP/1.1
> User-Agent: curl/7.29.0
> Host: isic.vzgcis.com
> Accept: */*
> Referer: rbose
> Cookie: iPlanetDirectoryPro=AQIC5wM2LY4SfcxLJbvfzGX-0-Q8sOhwGIXB133EvtHARyA.*AAJTSQACMDIAAlNLABM1NjkzMzM4NzMxMzA1MDc1Nzk4AAJTMQACMDE.*
> Authorization:Bearer 2d6910d5-741f-4b3e-93db-1a3bd388053c
>
< HTTP/1.1 200 OK
< Date: Wed, 27 Apr 2016 07:13:23 GMT
< Server: Restlet-Framework/2.1.7
< Accept-Ranges: bytes
< Vary: Accept-Charset,Accept-Encoding,Accept-Language,Accept
< Content-Type: application/json;charset=UTF-8
< Connection: close
< Transfer-Encoding: chunked
<
* Closing connection 0
{"university":"University of Rome","sub":"919910431224","department":"Computer Science"}[root@openam ~]#
```

Figure 20: Get UserInfo

4.2.2.2 FIDO UAF Server

The FIDO server component comprises two main elements: the FIDO core protocol and the FIDO servlet. The FIDO core protocol is a software module which is instantiated on both the server and the client and is in charge with implementing the details of the FIDO protocol.

The FIDO servlet component is in charge with handling the HTTP interface of the implemented FIDO solution. The FIDO server component is deployed in an Apache Tomcat environment and it uses the Jersey Servlet as JAX-RS implementation. The FIDO server component is an open-sourced eBay solution, modified in order to meet the ReCRED application requirements.

One of the solution modification was adding an extension in order to specify the last authentication timestamp of a certain user. This extension will be used by the other server-side components. The added extension has the following form:

```
{
"username_id": "recred_user",
"last_auth_timestamp": 1460555380698
}
```

The *last_auth_timestamp* field specifies the number of milliseconds elapsed since 1970.

The communication between the FIDO server and the other ReCRED components is realized through REST calls. The FIDO server implements the following FIDO functionalities:

- Registration
- Deregistration
- Authentication

The FIDO functionalities are accesible by using the following REST calls:

- Get authentication request - `/fidouaf/v1/public/authRequest`
- Post authentication response - `/fidouaf/v1/public/authResponse`
- Post Deregistration response - `/fidouaf/v1/public/deregRequest`
- Get registration request - `/fidouaf/v1/public/regRequest/`
- Post registration response - `/fidouaf/v1/public/regResponse`
- Get last authentication of a username - `/fidouaf/v1/public/lastAuth/{username}`
- Get a history of operatoins – `/fidouaf/v1/history or /fidouaf/v1/history`

In Figure 21 are depicted the FIDO software and hardware components.

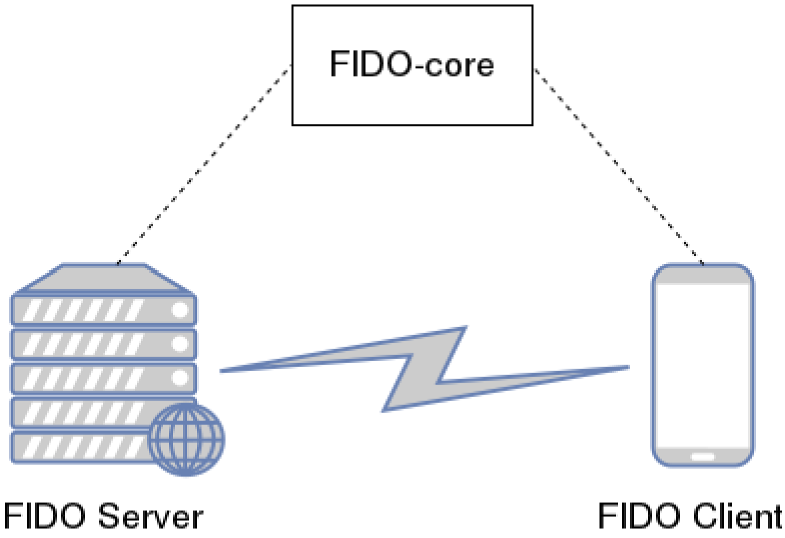


Figure 21: FIDO software and hardware components

The FIDO messages exchanged between the server and the Android client are serialized using JSON (JavaScript Object Notation). The FIDO servlet hardcodes a series of information like the FIDO application ID or the allowed FIDO authenticator AAID (Authenticator Attestation ID). In future versions of this application this information will be separated in another modules (database). Another drawback this solution is the in-memory storage of FIDO statistics.

4.2.3 Mobile Application

The mobile application for the ReCRED project, is a native Java application. It handles the communication with the web services through REST HTTPS calls and uses the JSON format for data transferring.

The app uses the Java HttpURLConnection class in order to create the client to handle all the REST calls.

The initial trigger is the action from a UI Button that is pressed by the user, in which case an Asynchronous task makes the initial Authentication REST call, of POST type, to the OpenAM web service and creates a JSON object that includes the authentication ID from the service’s answer. Subsequently, the app adds the username value to this Object and makes another POST call to the OpenAM web service and gets a response that contains the authentication token and the FIDO server URL.

Afterwards the Authorization procedure starts, where the app creates a Cookie with the previous mentioned token and makes another set of REST calls, but this time both POST and GET. All the required parameters for OAuth2 are also added to the call URL’s header.

For all the other calls that just display data, such as the user’s identity attributes and the available resources, the app uses normal HTTP GET calls and JSON objects.

The OpenID Connect and the FIDO UAF Client are analytically described in the following subchapters.

4.2.3.1 OpenID Connect Client

The OpenID Connect Client uses the APIs provided by the Authentication server. The APIs and communication between the server and the client are described in chapter 4.2.2 Authentication Server.

4.2.3.2 FIDO UAF Client

The FIDO UAF Client used in the Wi-Fi pilot is an Android application open-sourced by eBay. The FIDO Client has a modular structure having two main components: the FIDO protocol core and the FIDO UI client. The FIDO UI component consists of Android activities and of an HTTP client which facilitates the interaction with the FIDO server component. The FIDO protocol core implements all the FIDO protocol elements, an instance of this component being also employed by the FIDO server.

The client implements the following FIDO actions:

- Registration
- Authentication
- Deregistration

The access to the FIDO generated keys are protected by an authentication screen, where the user has to input the password or present a correct fingerprint. Taking into the consideration the hardware platform used in testing the FIDO client (Nexus 6P and Nexus 5X), the fingerprint scenario was successfully tested.

The fingerprint user authentication screen is implemented using the KeyguardManager class from the Android SDK. This class can be used to lock and unlock the keyboard, being authentication mechanism agnostic (for instance the method *createConfirmDeviceCredentialIntent* supports pin, pattern, password or fingerprint).

The FIDO cryptographic material: the generated key pair (RSA or ECC) and the attestation certificate are store in the application preferences. This aspect will be improved in our future work by delegating the storage of this data to Android OS Keystore.

Regarding the FIDO client UI structure we enumerate the following Android activities:

- Settings activity: permits the settings management. The user can configure the FIDO server address and the FIDO operations URL path.
- Main activity: presents the FIDO registration status and permits FIDO registration, deregistration and authentication.

- Confirmation activity: the user is shown the FIDO message content and the actual FIDO operation has to be approved by authenticating with the fingerprint.

For the Wi-Fi pilot scenario the FIDO username was replaced by the device unique ID. The Android device unique ID (Secure.ANDROID_ID) is a 64 bit string which is changed after each device factory reset. This action was taken in order to create a link between the FIDO registration and the actual device which is authorized to use the Wi-Fi services.

Regarding the FIDO core component, the authenticator is a software emulated element which implements the ASM interface. The attestation process is also emulated by using a hardcoded certificate and a signature by employing Java Security classes.

4.2.3.3 Mobile Application UI

In this section, we will provide some screenshots from the mobile application in order to demonstrate the user’s interface.

Figure 22: Main Screen

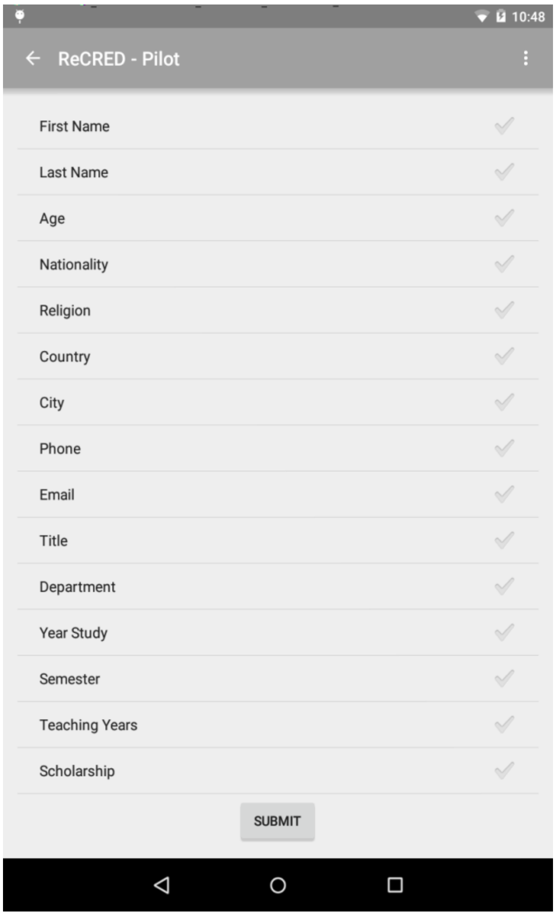


Figure 23: Screen where the user selects identity attributes

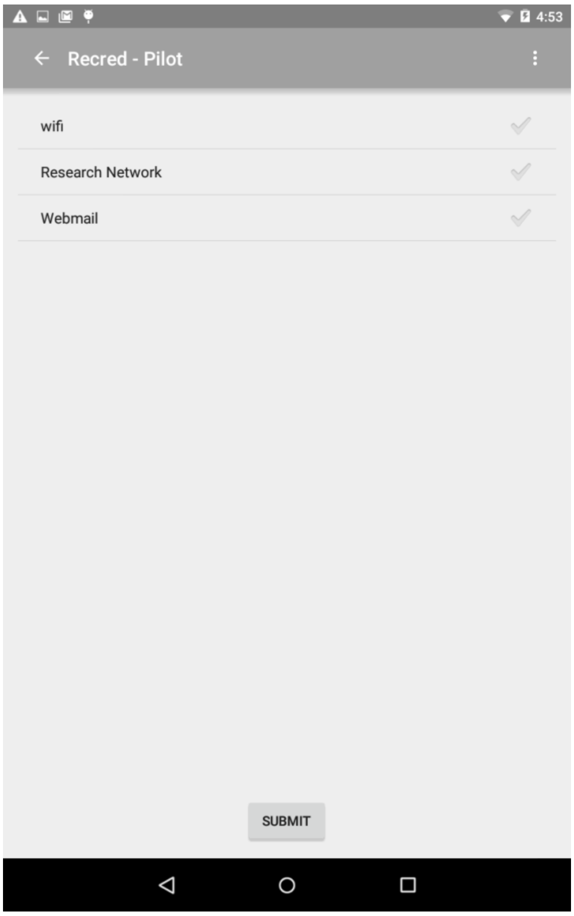


Figure 24: Screen where the user selects network resources

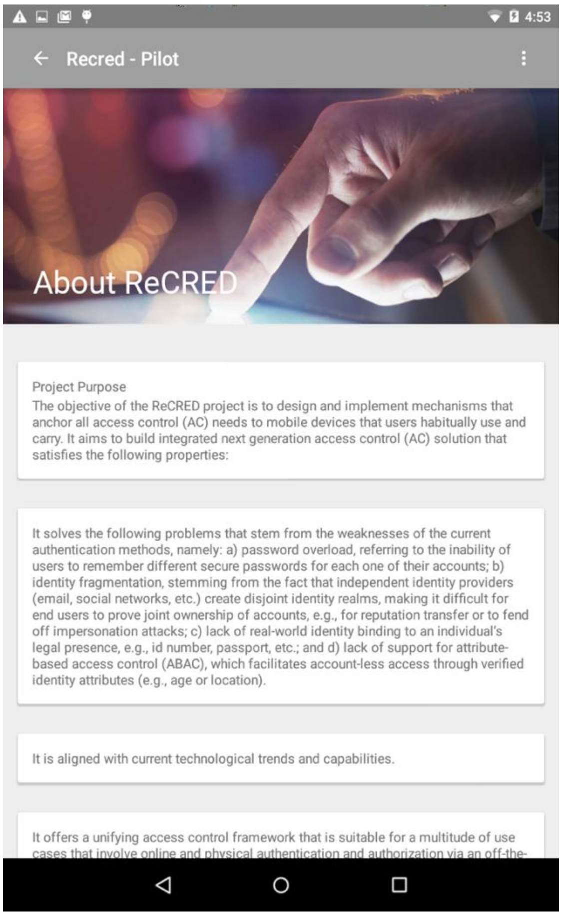


Figure 25: About us screen

4.3 Environment and Deployment

The FIDO server component is HTTP based and needs to be deployed in a Java servlet engine. To accomplish these requirements an Apache Tomcat servlet engine was chosen for the FIDO deployment. The Apache Tomcat version installed for the Wi-Fi Pilot is 8.0.32. Tomcat is an open-source web server which implements multiple Java EE specifications like: Java Servlet, JavaServer Pages (JSP), WebSocket and many others.

The following are the main Tomcat components:

- Catalina – is a Tomcat servlet container.
- Coyote – is a Connector component for Tomcat that supports the HTTP 1.1 protocol as a web server.

- Jasper – is a Tomcat JSP Engine (it parses JSP files to compile them into Java code as servlets).
- Cluster – this components is used when managing large applications (e.g. load balancing).
- High availability – this component eases the system upgrades without affecting the live traffic.
- Web application – it consists of user and system based application which permits deployment in a variety of environment. The web application permits the deployment and management from both a graphical web interface and from REST calls.

The JRE (Java Runtime Environment) version installed for the FIDO server deployment is 1.8.0_05.

The communication between the FIDO client and the FIDO server is secured by using TLS. The component which handles the TLS session is gateSAFE. The gateSAFE server terminates the TLS and forwards the HTTP request to the FIDO servlet, via the Tomcat server.

Regarding the Tomcat configuration, all the roles are disabled, except the *manager-gui* role which is password protected. The *manager-gui* role has access to the Tomcat HTML interface and can manage some of Tomcat features. Other Tomcat management roles which can be activated are: *manager-status, manager-script* and *manager-jmx.*

5 Security Considerations

The current version of the pilot does not yet make use of Privacy-Preserving Attribute-Based Access Control mechanisms such as Idemix or U-Prove. This implies that the privacy of the users can be threatened by an attacker capable of linking together multiple credential shows (to the same or different relying parties) or by an attacker which is able to trace the identity of the user starting from the credential issuance process up to the credential show to relying parties.

For what concerns WLAN access, in order to allow credential issuance to not yet authorized users, the WLAN APs do not make use of link-level encryption protocols such as WPA2, but are instead open to any STA. Unauthorized STAs are allowed to access only the first segment of the network, in order to be able to request access to resources. The rest of the network is closed by using a firewall. When a STA is authorized, its MAC address is whitelisted in the firewall, giving it access to the requested resources.

An attacker capable of eavesdropping on the open WLAN and generating frames with arbitrary source MAC addresses could thus wait for a STA to be authorized, eavesdrop its MAC address and then implement an impersonation attack by spoofing frames having the same source MAC address of the authorized STA. Such attacker would be able to access the same resources as the user of the authorized STA.

An attacker capable of generating arbitrary ARP packets could implement a man in the middle attacks at the IP layer by generating spoofed ARP replies and thus poisoning the ARP caches of the STAs and the gateway. Such setup will allow the malicious user (i.e. the user that spoofs the address of a legitimate user) to be authorized to access the network despite not having the right credentials. Moreover, the malicious user capable of manipulating packets in the WLAN could also perform Denial of Service attacks towards the legitimate user or endanger the privacy of the connected users by collecting packets belonging to clear-text user sessions. Even in case of encrypted sessions, the malicious user could still attempt to attack the user’s privacy, by collecting metadata from the packet headers, website traffic fingerprinting (through statistical analysis) or similar attacks.

6 Future Work / Conclusions

As stated in the previous chapter, the presented scenario has a series of security vulnerabilities which must be overcome in later versions of the pilots.

We will replace OpenID connect with Privacy-Preserving Attribute-Based Access Control mechanisms (Idemix and U-Prove) with unlinkability and untraceability features. To this aim, as described in Deliverable D5.1 “Specification and initial design of the ABAC infrastructure”, credential issuance, verification and credential management at the user device components are being developed.

The main security vulnerability is represented by the spoofing attacks, because an unauthorized user can gain access to Wi-Fi resources by sniffing the network packets (by connecting to the open Wi-Fi).

To overcome the spoofing vulnerability, we would like to approach the following directions:

- Use pairs of APs with two different SSIDs: one AP will be open, while the second AP will be configured to employ IEEE 802.1X. The open AP will allow only STAs to verify their attributes and obtain a credential which can be used at the second, IEEE 802.1X enabled AP, to access the requested resources.
- Establish an encrypted tunnel between the user's smartphone and the network controller, after the user is authenticated.

Multiple methods will be taken into consideration because one of the solution component will be an unrooted Android device, thus having permissions and security restrictions.

When deployed in IEEE 802.11 WLANs, IEEE 802.1X employs mutually-authenticated sessions with per-client encryption keys. In this way the impersonation and source address spoofing attacks by STAs connected to the AP, described in Chapter 5, can be avoided.

Moreover, the AP will be configured in order to not allow STAs on the same WLAN to communicate at the MAC layer between each other, thus avoiding also ARP spoofing attacks.

From the Android point of view, the Wi-Fi 802.1X network authentication use the Android WifiConfiguration class from the Android SDK. Manipulating the Wi-Fi state from an application may require the following Android permissions:

- ACCESS_WIFI_STATE
- CHANGE_WIFI_STATE
- CHANGE_WIFI_MULTICAST_STATE

The Android Wi-Fi API provides a means by which third-party applications can communicate with the low level Wi-Fi network access. An application can access a series of Wi-Fi supplicant data like: network link speed, IP address and many others. Also through the Android Wi-Fi API, a third-party application can scan, add, save, terminate and initiate Wi-Fi connections.

The user first authenticates to the Identity Service by connecting to the open Wi-Fi, using the mechanisms described in this paper (OpenAM and FIDO). After the user is authenticated it switches to the closed Wi-Fi by using some credentials/cryptographic material obtained during the authentication phase. In this scenario the open Wi-Fi operations are a bootstrap for the secured Wi-Fi connection. The protocol 802.1X is an IEEE standard which provides authentication mechanism to devices which want to access LAN or WLAN. This protocol defines the encapsulation of the Extensible Authentication Protocol (EAP) over multiple IEEE 802 LAN technologies.

The Android SDK classes which permit the management of Wi-Fi connection are the following:

- WiFiConfiguration.AuthAlgorithm: IEEE 802.11 authentication algorithms (LEAP, OPEN and SHARED)
- WiFiConfiguration.GroupCipher: recognized group ciphers (CCMP, TKIP, WEP104, WEP40)
- WiFiConfiguration.KeyMgmt: recognized key management schemes (IEEE8021X, NONE, WPA_EAP, WPA_PSK)
- WifiConfiguration.PairwiseCipher: recognized ciphers for WPA (CCMP, NONE, TKIP)
- WiFiConfiguration.Protocol: recognized security protocols (WPA2/IEEE 802.11i, WPA/IEEE 802.11i/D3.0)

In addition to the classes presented above there is a WiFiEnterpriseConfig class which permits the configuration of EAP methods and associated credentials for connecting to a Wi-Fi. This class permits the setting a password or X.509 certificates and private key for server and client authorization.

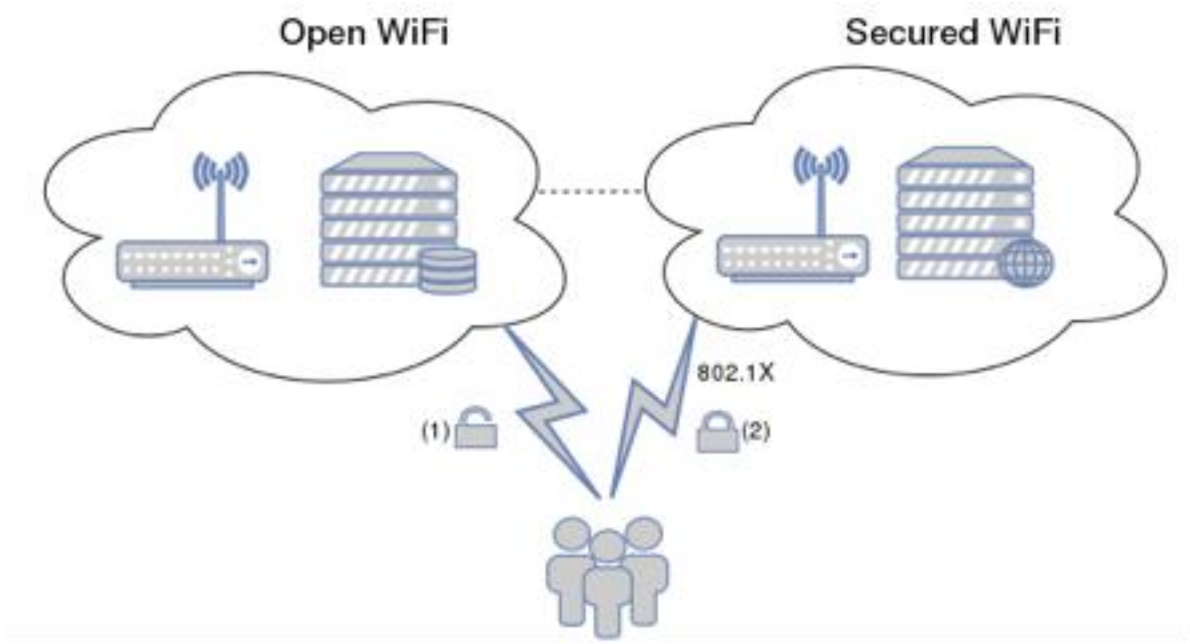


Figure 26: Open Wi-Fi/Secure Wi-Fi architecture

An alternative to the AP pairs approach proposed above would be to set up an open AP that allows connections only to a local VPN server.

The user could provide the credentials in order to be authenticated and authorized to access the VPN, which would in turn provide access to the network. The VPN server would need to be configured in order to not allow address spoofing and direct communication between clients.

Moreover, this solution would potentially allow for Traffic Flow Confidentiality techniques such as the one proposed in [1] for IPsec. By using such a method, the spoofing attacks will be mitigated, because the user device will be authenticated during the IKE phase of the IPSEC tunnel establishment and all subsequent data packets will be transmitted over a secure and authenticated channel. In this scenario, like in the first one, the cryptographic material used during IKE phase (pre-shared key or a temporary X.509 certificate) will be obtained in an authentication phase similar to the one presented in this paper.

The Android SDK permits the creation and management of VPN connection by third-party applications, by using the VpnService and VpnService.Builder classes. The usage of the latter requires the BIND_VPN_SERVICE Android permission. The VpnService is a base class for extending and building own VPN solutions: it creates a virtual network interface (tun/tap), configures addresses and routing rules. The interface is running on IP, so the applications can complete a VPN connection by processing and exchanging packets with a remote server (Network Controller).

Regarding the FIDO client, the existing solution could be improved by exposing all the FIDO functionalities into an Android Service. All the FIDO UAF Client methods will be called formalized by using an AIDL interface. By using such an architecture, the the FIDO solution will gain modularity and the integration with the other ReCRED components which will employ the FIDO functionalities will be eased.

From the security point of view, the existing FIDO solution could be improved by adding a TLS channel binding. By adding TLS channel binding the FIDO security will be improved because MITM (Man-in-the-Middle) attacks will be mitigated, the application layer (FIDO) being ensured that the two end-points are the one which established a secure transport channel. FIDO specifications recommend the usage of TLS channel binding, but this feature is not mandatory. The TLS channel binding types are formalized in RFC 5929 and the FIDO specifications recommend the usage of the following TLS channel binding types: tls-unique and tls-server-end-point.

Another security improvement for the FIDO client would be storing the FIDO generated keys into the Android hardware-backed Keystore. Currently the generated FIDO keys are stored as base64 strings into the application preferences. Storing the keys into the Android Keystore will be a security improvement because the key access and operations is controlled by the Android OS and only authenticated users (by means of fingerprint or gatekeeper) can execute operations with the keys. The same design could be applied to the FIDO attestation X.509 certificate and private key. The FIDO client authenticator is software emulated and by delegating the cryptographic operations and storage to the Android keystore a series of security vulnerabilities will be mitigated.

7 References (ALL)

[1] Kiraly, C., Teofili, S., Bianchi, G., Cigno, R. L., Nardelli, M., & Delzeri, E. (2007). Traffic flow confidentiality in IPsec: Protocol and implementation. In The Future of Identity in the Information Society (pp. 311-324). Springer US.