Lappeenranta University of Technology

Faculty of Technology Management
Department of Information Technology

Authentication and authorization service for a community network

The topic of the Thesis has been confirmed by the Departmental Council of the Department of Information Technology on _____

Supervisors: Docent D.Sc. (Tech.) Jouni Ikonen
M. Sc. Janne Oksanen

Copenhagen, 13 October, 2008

Rustam Jemurzinov
Bøhmensgade 14, 4 TH, 2300 Copenhagen S, Denmark
Phone: +45 5018 3124

ABSTRACT

Lappeenranta University of Technology
Department of Information Technology
Rustam Jemurzinov

Authentication and authorization service for a community network

Thesis for the Degree of Master of Science in Information Technology
2008
71 pages, 21 figures and 2 appendices
Examiners: Docent D.Sc. (Tech.) Jouni Ikonen
M.Sc. Janne Oksanen
Keywords: wireless community network, authentication, authorization, software update, captive portal

Wireless community networks became popular in uniting people with common interests. This thesis presents authentication and authorization service for a wireless community network using captive portal approach including ability to authenticate clients from associated networks thereby combining multiple communities in a syndicate.

The system is designed and implemented to be reliable, scalable and flexible. Moreover, the result includes software management system, which automatically performs software updates at network's access points. Future development of the system can be concentrated on an improvement of the software management system.

Acknowledgements

I would like to thank all people who are involved in International Master's degree Programme in Information Technology and especially Ms Riitta Salminen. I am grateful to my supervisor Jouni Ikonen for all the effort put into this work. A special thanks to my friend Matylda Jabłońska for contribution and valuable comments.

I am also thankful to Tanya Petrova for making an excellent atmosphere and support. Last but not the least I thank my parents, who were worrying about my graduation for the last few months, for belief in me.

Table of contents

List of figures

Abbreviations

AAS – Authentication and Authorization Service
AP – Access Point
BIND – Berkeley Internet Name Domain
BSD – Berkeley Software Distribution
DHCP – Dynamic Host Configuration Protocol
DNS – Domain Name System
DSL – Digital Subscriber Line
SSID – Service Set Identification
GPL – General Public License
HTTP – HyperText Transfer Protocol
IANA – Internet Assigned Number Authority
IEEE – Institute of Electrical and Electronic Engineers
IP – Internet Protocol
ISP – Internet Service Provider
JDBC – Java Database Connectivity
LDAP – Lightweight Directory Access Protocol
MAC – Media Access Control
NAT – Network Address Translation
OAN – Open Access Network
PDA – Personal Digital Assistant
PGP – Pretty Good Privacy
PHP – Hypertext Preprocessor
RDBMS – Relational Database Management System
RFC – Request for Comments
SDL – Specification and Description Language
SMS – Software Management System
SSL – Secure Sockets Layer
URL – Uniform Resource Locator
WCN – Wireless Community Network
WEP – Wired Equivalent Privacy
WLAN – Wireless Local Area Network
XML – eXtensible Markup Language

1. Introduction

From ancient times people tend to live in communities. One of definitions of community is as follows: “community is a specific group of people, often living in a defined geographic area, who share a common culture, values, and norms and who are arranged in a social structure according to relationships the community has developed over a period of time [1].” In this thesis community is considered as a group of people and organizations, who share common interests and characteristics, regardless of their location.

Use of computer networks allows people to build online communities, where group of people primarily interact for social, educational or other purposes via computer networks rather than face to face [2]. In this thesis term wireless community network is interpreted as a computer network that uses radio technology to connect community members to one another. Being connected to a wireless community network, users have access to community-generated content as well as Internet connection. It is called a “community” network because it unites people with a common interest and provides community-wide coverage [3]. Further discussion on wireless community networks can be found in chapter 2.

In the recent years wireless community networks have become popular and currently many enthusiasts all over the world establish them. List of wireless community networks proofs that more than a hundred of communities all around the world are already deployed [4]. Wireless community networks allow to make regional communities and connect people together while giving them freedom of moving; they become mobile within community-wide coverage. A member of a community is able to connect to community services and to the Internet from any place within community network’s area. Since wireless community networks are based on use of computer networks, there are many issues to consider: security, privacy, trust to others, *etc*, which are mainly of a technical character. The wireless community network must keep secure private information of members, which implies only authorized access to community members’ information and to community services.

In order to allow only authorized access to community network's services, community networks need a service for authentication and authorization of members. This will ensure that users agree with policies of wireless community network which they are going to use. Moreover, authentication and authorization service provides different levels of access to community services, which makes sure that not authorized people do not have access to private data of community members. Different levels of access in a wireless community network mean, in particular, different roles of users – public users, access point owners and network administrators.

With having authentication and authorization service, it is desirable to get a good level of security without doubt about loss of private information. These issues were examined on the early stages of community networks and different software projects, like [5], [6], [7], [8] and [9], tried to solve them. A common idea of captive portals was developed and many implementations of captive portals were shown to the world (see [4] and [10]). Further discussion on captive portals can be found in chapter 3.

Currently, many solutions exist to build a wireless community network. Successful software projects such as Wifidog captive portal suite are running on many establishments of wireless community network. As all the software solutions in the world, they were made with assumptions and certain requirements. However, new prerequisites resulting from human nature appear every day. Therefore, due to new requirements to the wireless community networks this project was started.

The aim of this project – authentication and authorization service for a community network – is to provide flexible, reliable and scalable service with the following main features:

- The service must be able to authenticate and authorize users within the coverage of community network at any time
- The community members must be able to customize welcome page at their access points
- The access point owner must be able to do user management within his hotspot

- The access point owner has rights to stop being a member of the community at any time
- The content delivery system, that allows custom welcome pages at each access point, must be present in the community
- The network statistical information and reports must be provided at different levels – within an access point and the whole community network
- The community network must be able to authenticate and authorize users from associated communities
- Software management system for software updates must be present in the community

All these features were taken as requirements in development of new solution. A study of existing solutions was a part of this work and one of the approaches was taken as a basis for the new proposed system. Then it was extended by additional features. Section 3.3 discusses existing solutions of captive portals.

The research presented in this paper serves a way of extending characteristics of wireless community networks, for example by connecting the networks to each other and, therefore, giving more mobility freedom to the community users. Flexibility and scalability of the system are provided by means of using appropriate software and their extensions. Much effort was put on availability aspect, which was achieved by adding redundancy to the system and using distributed approach in design and implementation of core services of the system. New application was developed to provide functionality of software management system that is used to perform automatic updates of client software.

As a result of this work the authentication and authorization service for a community network with all new features was introduced. The new system was designed and implemented with use of Wifidog captive portal suite. Software management system was developed and implemented from the scratch. The analysis of the system was done and appropriate software testing was performed.

2. Wireless community networks

When we talk about computer networks, people think about wired infrastructure with some sort of physical transmission medium in the ground. Often such infrastructure is operated by large entities such as telecom operators. Wireless networking using the IEEE 802.11 standard is a new communication technology, which opens possibilities of building a network without cost of putting wired infrastructure for that [11]. The IEEE 802.11 standard is an open standard, which uses radio link as a transport medium [10].

With growing popularity of wireless networking using 802.11 standard, manufacturers made a marketing name for this type of wireless connection – wireless fidelity, for short Wi-Fi. Wireless Fidelity (Wi-Fi) refers to wireless networking based on the IEEE 802.11 standard [12]. The 802.11 standard is implemented in several versions, which differ in used frequencies, range and possible data transmission rate. From this point on in the thesis when we say "wireless", we mean the 802.11 b/g standard and Wi-Fi is considered as a synonym to it. Also, access point (AP) and hotspot stand as names for a device, which provides access to community services with use of the 802.11 b/g wireless networking standard.

With use of off-the-shelf wireless hardware components and open source (free) software we can build a low-cost network infrastructure for a local community [11]. The use of open source software is an advantage, which makes wireless community network (WCN) "open" for developers who wish to contribute. WCNs could be used as a "last mile" to provide the Internet connection to people, for example, they may be used to connect rural areas to the Internet; Min-You Wu describes how the WCNs make it possible to connect people into a connected community [13].

In this work, when we say "wireless community network", we mean a community with interest of sharing own Internet bandwidth with others in order to be able to connect to the Internet when they are away from home. To be a member of a community one should share his Internet bandwidth with others through a wireless AP. Outsiders also can use the Internet bandwidth of WCN, but they must be registered in the network and as they are not members, they will not receive a high speed connection.

2.1. Examples of wireless community networks

A study of existing projects in the world was done before development of authentication and authorization service (AAS) for WCN. Examples of WCNs were examined from the point of view of services they provide and features available in used software. Some of the examples are not exactly WCNs; they are open access networks (OAN), where users have credentials at different Internet service providers (ISP) and OAN acts as a medium to authenticate users of different service providers. The OANs were studied since some of the ideas were taken from their nature, for example idea how to authenticate people from different communities.

At the earliest stage of wireless technologies the Wi-Fi activists found each other through web connection and e-mails. They deployed cooperative public wireless networks and gave free access to use the network's services. The main incentive to establish a WCN was formulated as follows. "I will set up my free node in the hopes that others will too, and I will have access when I'm away from home [14]." Moreover, Elias C. Efstathiou *et al.* [15] show some ways how to stimulate the participation in wireless community networks and how to secure the participants from having their access points abused. With the idea to benefit from community network when mobile in mind many wireless networks were established all over the world and they became popular. According to wireless communities list (see [4]), there are more than 150 WCNs in the world and their number still rises.

Some of the WCN projects provide Internet access through hotspots for free or with a small fee. Hotspots can be located in parks, hotels and in any other place, where people would like to get access to the Internet. Such projects use broadband connection as a backbone and then share the connection through hotspots in the areas where no cable connection to the Internet is available. The goal of each project might be different, but all the projects are driven with a wish to provide an easy way of information exchange.

Many authors work on the topic of phenomena of WCNs. All of them see the community networks from different points of view. Battiti *et al*. [16] discuss the

evolution of wireless networks from wireless local area networks (WLAN) to OAN architecture. They primarily focus on service operator – service user relations; therefore, they pay attention to pricing and billing issues. They present a pilot of OAN – StockholmOpen.net. They discuss social and economic aspects of the OAN architecture and win-win scenario for the users and the operators. More information can be found on StockholmOpen.net project's web page [5].

Another example of OAN is WilmaGate project – a WLAN hotspot management system. It supports number of authentication protocols for a number of service operators. Users must be registered at a service provider in order to get authenticated by the WilmaGate system. This system focuses a lot on authentication and authorization, and on security issues. The system has two parts – GateKeeper and Gateway. The captive portal paradigm is used in the authentication and authorization procedure [6].

In Milano, Italy a community network was established and it is called Rete Civica di Milano (RCM). RCM is a community network and its members have services such as e-mail (local and the Internet), public discussion areas, chats, file server, *etc* [7]. RCM is a community network, but not completely wireless, because anyone can be a member of the network even without sharing access through an AP. Therefore, it does not fit to our definition of WCN.

A good example of WCN is WirelessLeiden project deployed in Leiden, the Netherlands. The network was established with rules of using free open source software and low-cost hardware. The requirements to the system were to be open to users and developers, reliable and low-cost. The project started as an alternative to proprietary solutions, which used a certain vendor and not low-cost hardware. The target customers were schools and other educational institutions. The WirelessLeiden provides a low-cost way of crossing the "last mile" to the users. With possession of network, educational institutions may use the new technologies in the teaching; a good example of using new technologies in teaching is presented in [17]. The WirelessLeiden project is not a community of people who just share their digital subscriber line (DSL) connection with other, but also partnership with major groups of potential users, hardware vendors, the educational institutions and content providers [11].

The examples given above show how WCNs can be used, how they may contribute to members and what kind of services they may propose. The studied projects are good examples of how to provide reliable services to people. Moreover, they help to answer some questions concerning development of AAS for a community network in order not to reinvent the wheel. These examples are a study on what was done in the field of WCNs before and they were taken as an input to this project.

2.2. Access points

Access point (AP) is a hardware device that provides a point of entry to wired network infrastructure for wireless clients. Through wired network wireless clients get connection to the Internet. AP acts as a bridge between wireless and wired infrastructures and as any bridging hardware, it has at least two network connections and shuffles traffic between them [10]. AP can support several wireless users simultaneously. Normally, APs are configured using a client program and they are secured by a password.

Due to mass demand for wireless devices their prices drastically dropped. Nowadays a client wireless adapter costs only 40 Euros, which may connect to the nearest wireless AP. If a person wants to share the bandwidth, s/he may do it by buying a wireless router for 90 Euros only, which may interact as an AP. The wireless router used in this project – Linksys WRT54GL – is a low-cost AP hardware and it is possible to buy it for less than 90 Euros. A detailed description of Linksys WRT54GL router can be found on the producer's web-site [18].

2.2.1. Security concerns

APs were designed to let a small group of trusted people connect to a wired network and lock out everyone else. Access control methods were implemented by most of vendors to control the access on the wireless side of APs. If someone wants to lock down his network at AP, he may use encryption: Wired Equivalent Privacy (WEP) or

Wi-Fi Protected Access (WPA); filtering on Media Access Control (MAC) address – the radio card's serial number; and a closed network option [10]. Combination of these methods may not make your network completely safe from an incursion into network, but may protect it from most of would-be network raiders.

Encryption can be used when you want to share wireless network access with a group of people by sharing a secret WEP or WPA key. Since the secret key is mutual among a group, you have to trust the group of people with whom you are sharing wireless connection. If one knows the secret key of a wireless network secured by encryption, then s/he may intercept any traffic in the network and read it as it is sent in the clear [10]. In addition to encryption, a static table of hardware MAC addresses may be useful in a small network. If MAC filtering is enabled, a client attempting to connect the AP must be listed in internal list before obtaining the access. In case of client's MAC was not found, the access to network is not permitted. In closed networks AP does not broadcast its service set identification (SSID) thereby it makes difficult for people to detect the network. In order to connect to the closed network each client must know the SSID in advance [10].

2.2.2. Vital services

Maintenance of AP network connections is needed to provide access to the Internet through the AP. This includes control of connection to wired infrastructure and management of wireless service supplied to clients. Dynamic host configuration protocol (DHCP) is used to automatically configure wireless client devices for current wireless network. Most of the APs can provide connection to wired network for multiple wireless clients with use of only one Internet protocol (IP) address from the wire. In order to do that in wireless environment, network address translation (NAT) services should be available for wireless clients. Wireless clients also need service called Domain Name System, which is used to resolve Internet domain names, *e.g. slashdot.org*.

DHCP provides a framework for passing configuration information to hosts on a computer network [19]. The client configuration is not only IP address. Along with

assigned IP address client gets information about Internet gateway and the local DNS servers. DHCP simplifies the maintenance of the network. Every time a new client computer connects, it gets a unique IP address automatically without manual interference.

DHCP is a requirement in wireless environment due to nomadic nature of wireless clients and it would be a nightmare to configure the network parameters manually for each network. It is more convenient to use DHCP service, which lets a client computer discover information about the network which it is connected to. Thereby, automatic configuration of client's node gives an easy way to get "online" without any prior knowledge about the network layout.

Domain Name System (DNS) is an Internet service that translates IP addresses into domain names and vice versa. The Internet is based on IP addresses; each IP address uniquely identifies a hardware connected to the global net. Since it is hard for a human to remember many numbers, DNS provides alphabetic human friendly domain names, which are easy to remember. For example, domain *www.example.org* might be assigned to IP address 198.68.1.5. Every time anyone uses domain name *www.example.org*, the DNS service translates the name into corresponding IP address.

DNS service is a hierarchical network and DNS servers are constructed in a tree structure. If a DNS server cannot translate a particular domain name, it asks a parent in the tree and so on until the IP address is returned. The root servers are coordinated by Internet Assigned Number Authority (IANA). IANA is responsible for maintenance of a number of key aspects of the DNS, including the root zone – the top level of the DNS hierarchy [20].

Like DHCP, the DNS is a must in wireless infrastructure; it provides name resolution services to wireless clients connected to the Internet. The network administrator may use existing DNS server, which already serves the wired infrastructure or he may want to provide additional DNS services at APs. It is a good idea to make a caching DNS server and thereby reduce the load on primary DNS server [10].

Normally, one IP address from public IP address space is provided from ISP to an ordinary user. This means that only one machine is able to use that IP address at a time to be online. What to do in case if we want to provide open access to the Internet, but we have only one public IP address and we do not want to pay for an extra range of public IP addresses for each extra user, since it is going to cost enormously too much. We need a tool which would be able to forward packets from internal network to the Internet and distribute incoming packets from the Internet among internal user in accordance with the destination of packets. Such tool exists – it is NAT.

NAT is the translation of an IP address used within one network to a different IP address known within another network [21]. A typical scenario is mapping local internal network addresses to one public IP address and unmapping the public IP address on incoming packets back into internal IP addresses. This gives a possibility to provide access to global network, such as the Internet, for all internal users by use of only one public IP address. NAT is described in Request for Comments (RFC) 1631 (see [22]) and a description of how the NAT works is available in [23].

3. Captive portals

While some people are happy to share their bandwidth, most of us are afraid to give open access to the networks and consequently be paid for neighbors. Users may use the connection for Internet browsing, and some of them can abuse other networks and their operations can be traced back to our network, to access point to which malefactor was connected. The owner of access point, who shares bandwidth with others is responsible for all traffic originated from his IP address. Therefore, not many of us are ready to share our bandwidth with unknown people. To make the bandwidth sharing responsible and secure we need a way to securely identify users when they connect and perhaps, give some restrictions to users on use of resources.

In order to restrict access to AP, its bandwidth, to make use of the AP's Internet connection responsible captive portals might be useful. The idea behind a captive portal is rather simple. Instead of using the built-in security features of 802.11 to control who can associate with an AP, the AP is configured as an open network with no encryption. When a user goes to an AP and brings up his browser, instead of getting on home page, he receives a custom page from service provider asking the user to identify himself, *e.g.* by entering login and password, and with information about the node they are connected to. If the AP can contact authentication server to determine the identity of the connected wireless user, then it dynamically changes firewall rules accordingly to that user's rights. This is how captive portals make sure that users identify themselves [24].

Once a user is signed in, a captive portal goes to pass-through mode. Next time the user sends a request, the captive portal looks up for his identity (usually MAC address of user's machine) and silently let him proceed with it, if the user is still in time-window of service; otherwise the client needs to authenticate again.

To illustrate the idea of a captive portal a drawing was made, which is shown in Figure 1. First of all, a traffic blocking mechanism is needed to protect the AP from anonymous usage. This is done by running dynamic firewalling inside the AP box; by default the whole access is denied. The solid lines in the pictures show intention of client to connect to *slashdot.org* web server in the Internet; for this purpose the client

must be authenticated and in order to pass authentication procedure he is redirected to the authentication server. In the picture redirection is shown by dashed lines. AP allows a client to connect to the authentication server through a secure connection. The connection is secured in order to protect client's information, login and password. If the client is authenticated and access to the Internet should be granted, the authentication server notifies the AP of the client's status, and firewall rules on the AP dynamically change to grant the Internet access to the client. The AP notification from the authentication server is shown by dash-dot line in the picture. The authentication server acts as a central repository system for user credentials. In reality, authentication server is the heart of captive portals, while gateway software, usually running on AP boxes, is used to play with firewall rules to grant or deny access and throttle bandwidth.

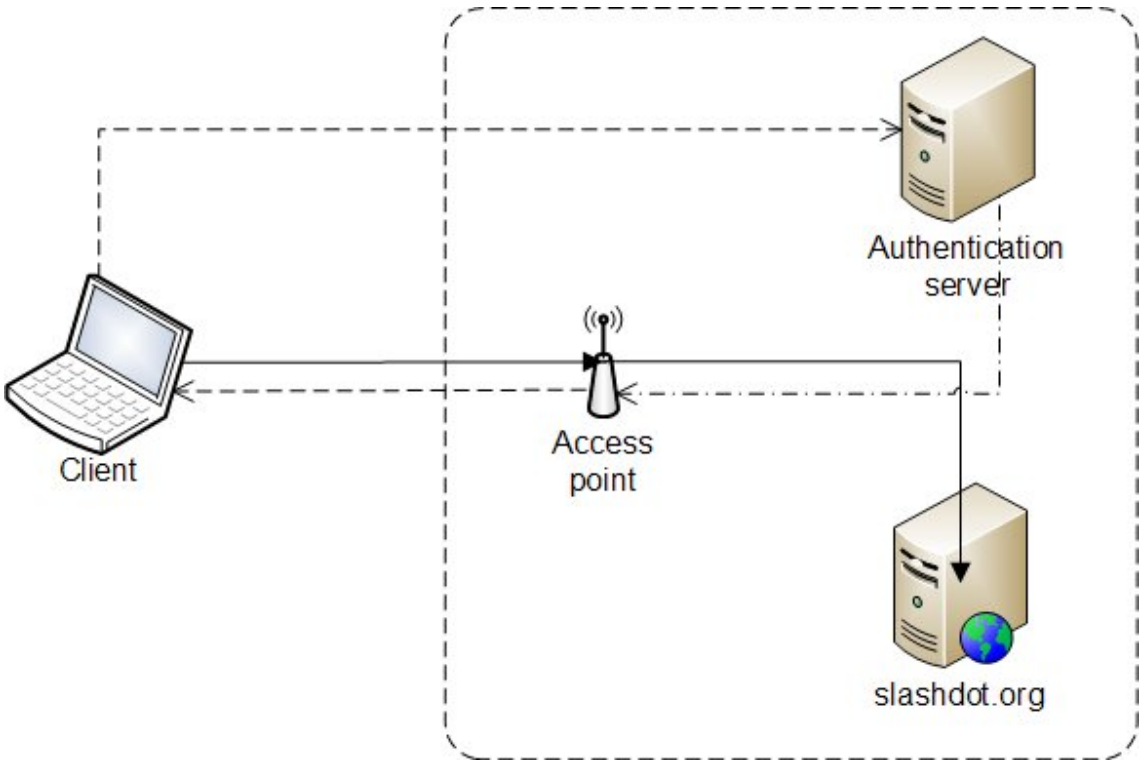

Figure 1 Captive portal definition

To summarize, if a service provider wants to establish a captive portal he needs:

- A traffic blocking mechanism, *e.g.* firewall,
- A redirection mechanism for Web-traffic,
- A secure sign-in mechanism, *e.g.* encrypted traffic over secured socket layer,
- A repository to safe users' credentials, *e.g.* a database.

If someone wants to implement his own captive portal solution, it is not necessary to build everything from the scratch. There are many existing solutions of components needed for a captive portal. Moreover, there are many existing solutions of captive portals; it might be useless to develop a new one, when we can use something already existing. Examples of captive portals will be given in section 3.3.

3.1. Advantages of captive portals

Some people might wonder why they need to use a captive portal while sharing Internet bandwidth. While providing open access services the way of doing it must be convenient and the owner should be able to maintain control of the network. By using public resources an intruder may abuse other networks and his activity can be traced back to service provider's network and the owner of public access network will be responsible for the consequences. A few examples of potential risk of giving open access are:

- Someone starts sending spam through an open AP and ISP threatens to shut down open access provider's account,
- Someone sends threatening or harassing emails/instant messages to someone and open access provider receives a warrant from police,
- Someone hacks neighbor's box (which is accessed the network via open AP) and starts hacking other networks. Problem with police, see above.

All the mentioned above proves the need in an authentication procedure to control users rights when we provide an open wireless access. The use of captive portals gives a convenient way of sharing open Internet bandwidth and responsible usage of network facilities by others.

When we use a captive portal, we keep AP as an open wireless network and the whole traffic between clients' devices and the AP is sent in the clear form to anyone in range. It is a must to protect clients' information, such as logins and passwords, in wireless community networks. How to protect clients' data from alien eyes while using wireless?

A good shield is possible by end-to-end encryption, which is provided by secure socket layer (SSL) tool for example. Browsing an SSL-enabled web page will keep the traffic between client and web server private. By use of end-to-end encryption, such as SSL, captive portal ensures that nobody else can obtain your credentials and personal information. Moreover, captive portals are designed in such a way that gateways cannot see the information passed from users to the authentication server; this ensures that the private information cannot be obtained by the owner of AP [10].

3.2. Disadvantages of captive portals

As usually, along with advantages of use of captive portals we have disadvantages as well. The main ones come from open access usage side. Since all ports, except 80, are blocked by default, a web browser must be loaded every time before we can use the Internet, because we have to be authenticated to use services. Since the authentication is web based, we need to have a web browser on our devices, if we wish to use the Internet in a WCN. After a client is authenticated, according to user's rights different firewall rules are performed to open appropriate ports and to give the access level associated with a certain user. Thus, a user can use a client software to check for new messages only after he has been authenticated, which can only be done from a web browser [25]. This is true, if we do not use MAC address authentication – when devices are authenticated and authorized by their MAC addresses, hence, devices with no web browser can be authenticated. By nature, all captive portal authentication solutions are vulnerable to MAC address spoofing and the case of providing MAC address authentication by a captive portal is insecure.

While requiring a web browser on clients' devices is disadvantage of all captive portals, for some of them it is necessary for web browser to be run in the background all the time to keep the connection alive, since they rely on scripts to keep track of the connection. This issue is solved in some of the proposals, for example in Wifidog project, but other solutions still depend on scripts running in the background, *e.g.* NoCat project. This brings difficulties to keep connection alive on some platforms, *e.g.* Portable Digital Assistant (PDA). The usage of captive portals from different platforms is restricted by the requirements made to end-users software. For example, usage of an

open AP is impossible from some models of mobile phones due to their inability to have a web browser with SSL encryption.

3.3. A study of existing solutions

In the recent years many solutions of captive portal came into the world. Proprietary solutions, such as FirstSpot, UseMyNet, HotSpot Studio, *etc*, were not examined in this project due to requirement of using only free open source software, even if they have very attractive characteristics [26], [27], [28]. More names of commercial solutions of captive portals you may find in the list made by PersonalTelco project's participants [29]. As regards open source solutions, there is a large selection of them. Many groups of enthusiasts implemented their own solutions of captive portals; all of them have their own pros and cons as well as a set of features. The free software was examined with the object to be customizable, providing network monitoring and logging, having user management and content delivery methods, allowing limiting shared bandwidth, *etc*. The open source captive portals – CoovaChilli, NoCat, Wifidog, M0n0wall, WilmaGate – were scrutinized [6], [8], [9], [29]. Two of them seem to fit best to our requirements; they are NoCat project and Wifidog captive portal suite. Each of them is presented in subsections below.

3.3.1. NoCat captive portal implementation

In this subsection, NoCat, a third party authentication system, is described. The NoCat package includes two main components: a centralized authentication system and numerous wireless gateways that communicate with the authentication server. The NoCat system fits to the idea of a captive portal shown in Figure 1. The NoCat captive portal solution is written in Perl and C, "it takes care of presenting the user with logon prompt, contacts a user database, such as MySQL database, to lookup for user credentials, securely notifies the wireless gateway of the user's status and authorizes further access [10]." On the AP side, the owner can set bandwidth throttling, firewall rules and a time limit for timing out old logins. The software is released under the General Public License (GPL) [30]. The NoCat system can work in two modes:

- Close Mode – when a nomadic user is prompted for login and password, and the user has to be authenticated to get a certain access rights.
- Open Mode – when a nomadic user is required to accept an acceptable use policy before he is granted a certain access rights.

The NoCat authentication system provides three possible classes of wireless user:

- Public class
- Co-op class
- Owner class

A public class user might be anyone, who wants to connect to the Internet. They might know nothing about the WCN; their aim is just to browse in the Internet. Typical public class users get a small bandwidth and they are restricted in what services they can access by the use of firewall rules. Public class users are granted access even without having login and password, but still being authenticated as in open mode. They are shown a splash page where they get information about the network, which they are connected to and use policy that they have to accept.

A co-op class user is a user of WCN, who has prearranged credentials and receives a better connection speed than a public class user. The rules for membership are defined by a local community groups and configured in the authentication system. This class of users is granted for more bandwidth and access to services, as they are considered responsible for their own actions.

The owner class is reserved for the owner of a given AP and anyone else to whom they want to grant access. An owner class user gets full access rights within given access point: uses all available bandwidth and has free use of all network resources.

When a roaming user associates with an AP, he is immediately issued by a DHCP lease. The whole access out of contacting the authentication server is denied by default. When the user types in the web browser an address, *e.g. http://slashdot.org*, his request is

intercepted by the gateway and the user is redirected to the authentication server's SSL login page as it is shown in Figure 2. A registered SSL certificate is required in production as the whole security of the system depends on it. A non-registered certificate would result in a security alert and may allow someone to spoof the authentication service.

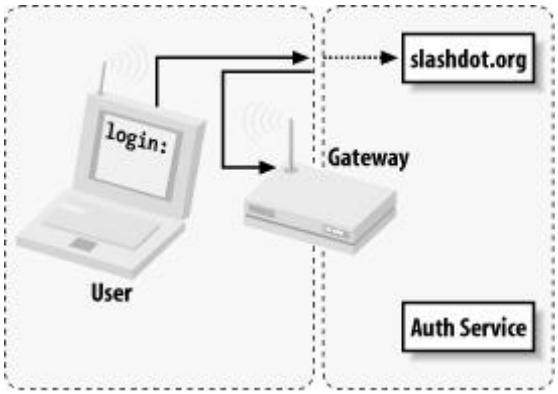

Figure 2 NoCatAuth protocol – phase 1 [10]

Then the user has three options: log in with prearranged credentials, click on the link to find out more about membership or click the *Skip login* button. "Once the user either logged in correctly or skipped the process, the Auth Service makes an outcome message, signs it with Pretty Good Privacy (PGP) encryption, and sends it back to the wireless gateway [10]" (see Figure 3).

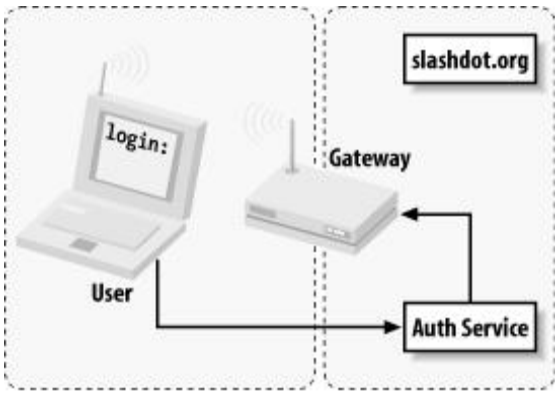

Figure 3 NoCatAuth protocol - phase 2 [10]

The veracity of the message is verified by the gateway that has a copy of the Auth service's public PGP key. "The digital signature prevents the possibility of other

machines posing as the Auth service and sending bogus messages to the wireless gateway [10].”

After everything has gone well for the user, the wireless gateway modifies its firewall rules to permit further access and redirect the user to the original site that he wanted to browse (see Figure 4).

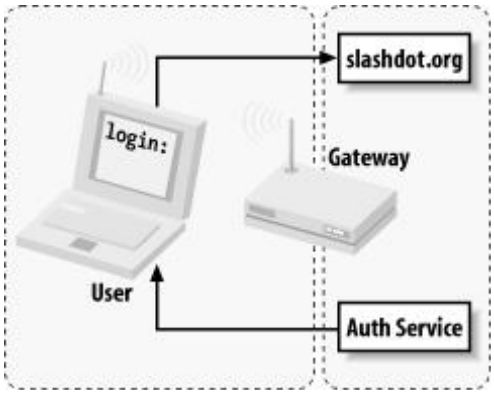

Figure 4 NoCatAuth protocol - phase 3 [10]

In order to avoid MAC address spoofing by unauthorized users, the client’s authentication must be periodically renewed. NoCat system opens a small window on the client side (via JavaScript) that periodically refreshes the login (see Figure 5). If the authorization is not refreshed within a given period of time, the client will be considered as gone and authorization will be revoked.

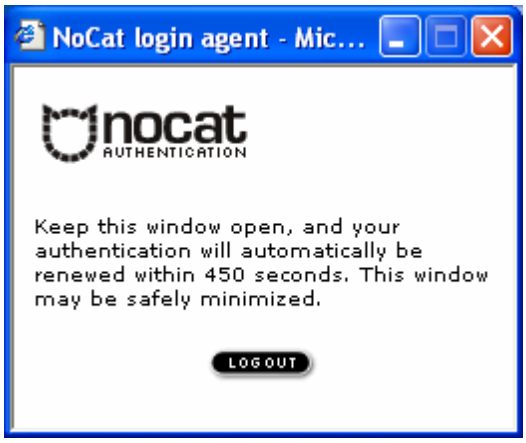

Figure 5 NoCat login agent

The NoCat authentication system seems to be popular (see [8]), and it is understandable as it was one of the first captive portal solutions. The NoCat deals fine with the technical aspects of catch and release but, on the other hand, has disadvantage of using JavaScript on the client side. Thereby nomadic users have to have a web browser with JavaScript enabled as NoCat uses a pop-up window to keep track of the connection. This was mentioned before as a disadvantage of many captive portal solutions.

3.3.2. Wifidog captive portal suite

Despite the fact that many solutions of captive portals exist, the group of developers from Montreal, Canada calling themselves Ile Sans Fil, decided to develop a captive portal solution for embedded systems, since the other solutions were not perfect for their needs. They wanted to make an embedded captive portal solution, but existing solutions were almost impossible to embed, for example the NoCat, which relies on Perl, GnuPG and OpenSSL. Other solutions, such as NoCatSplash, were designed to show only disclaimers with no access control at all [9].

The central server of the NoCat system did not have any mechanism to serve different content for each hotspot, the network statistics was poor and it was very difficult to get any useful statistics per hotspot from NoCat log. Finally, NoCat used JavaScript running in separate browser window to ping the gateway periodically, in order to keep a user's connection alive. This made it impossible to use the service from some mobile devices, which could not open more than one browser window, *e.g.* PDA. All these reasons made the group of developers implement a new captive portal, which will fit their needs. They named their project as Wifidog [31].

The authors of the Wifidog system defined it as follows: "The Wifidog project is a complete and embeddable captive portal solution for those, who wish to operate an open hotspot or network of hotspots while preventing abuse of their Internet connection [9]." The Wifidog captive portal solution has many features (see [9]) and main characteristics of the Wifidog, significant for this work, are:

- Centralized access control

- Full bandwidth accounting
- Local content specific to each hotspot
- It does not rely on JavaScript and works on any platform with a web browser
- Developed in C to make it easy to include in embedded systems. It has been designed for Linksys WRT54G.

The Wifidog project consists of two parts: authentication server coded in hypertext preprocessor (PHP) using a backend user repository, *e.g.* PostgreSQL database, and Wifidog gateway. All the duty of user authentication is placed on authentication server while the gateway is only playing with firewall rules to allow or deny certain access to the network. Every time a user submits information to the gateway, it connects to authentication server for future directives.

The initial connection process is similar to the NoCatAuth protocol's phase 1 (see Figure 2). A roaming user associated with an AP is immediately issued by a DHCP lease and all the access beyond contacting authentication server is denied by default. The Wifidog's client authentication flow is shown in Figure 6. A general description of the diagram is as follows. When the user starts to browse the web, his initial web request, *e.g. www.lut.fi*, is intercepted by Wifidog gateway, which redirects the user to the authentication server's login page. The user's device (Client) does its request to the Auth Server as specified by the Gateway (see Wifidog login protocol [32]). The Auth Server replies with a custom login page, where the user puts his identification information – username and password – and then submits them to the Auth Server. On successful authentication, the Client gets Redirect Request to the Gateway with its authentication proof – a one-time token. Then the Client contacts the Gateway and thus gives the token as a proof of authentication. The Gateway requests validation of the token from the Auth Server (see Wifidog client protocol [33]). After the Auth Server has confirmed the token, the Gateway redirects the Client to obtain the Success Page from the Auth Server. Finally, the Auth Server notifies the Client that his request was successful. The Success Page includes link to the original site, which the user intended to open at the beginning [34].

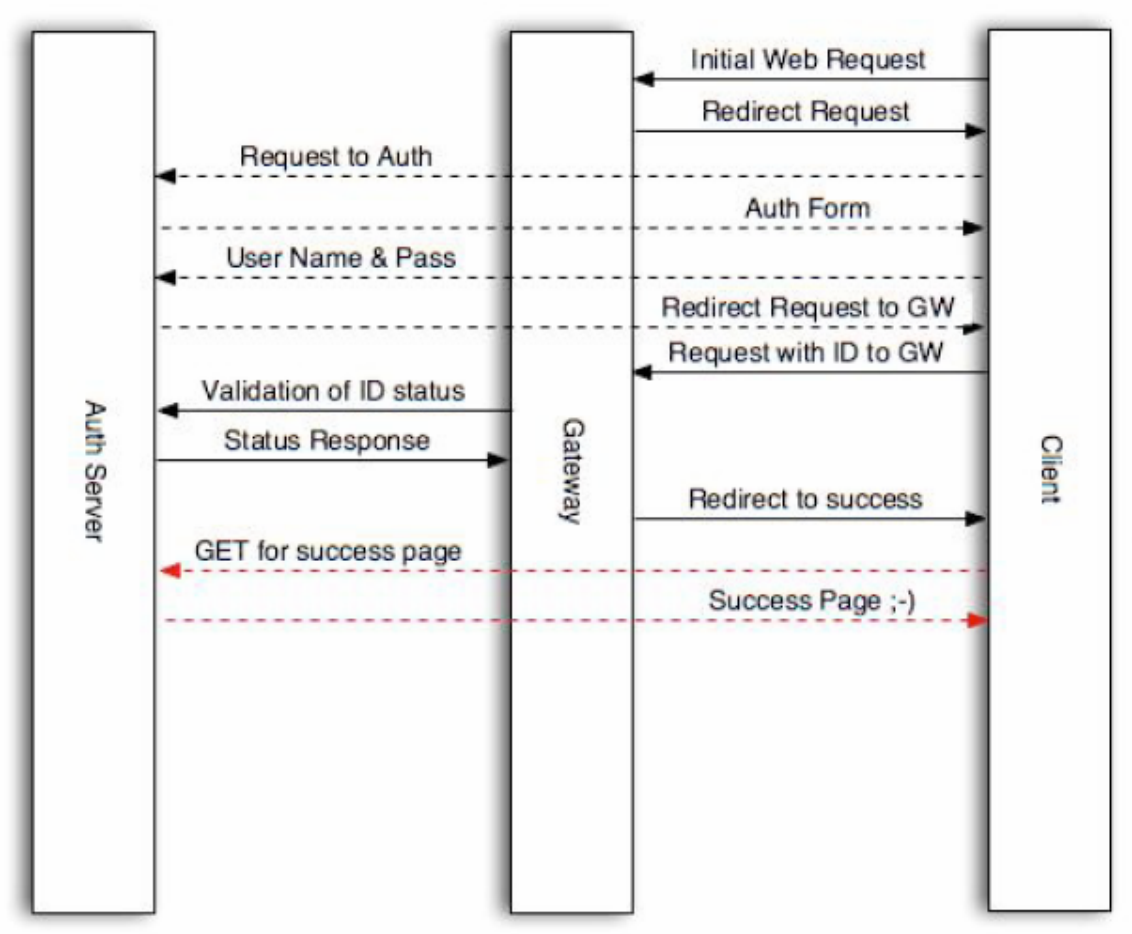

Figure 6 Wifidog Flow Diagram [34]

The Wifidog system seems to be a good implementation of a captive portal. It does not rely on JavaScript to keep the connection alive; instead, the Wifidog gateway provides heartbeats to the auth server. The gateway also talks periodically to auth server to update vital statistics including uptime, load, and traffic count per client. The Wifidog gateway runs on a device running Linux that has *netfilter* and *iptables* installations, which are requirements of the gateway. The auth server runs on any PHP-enabled web server and contacts a backend user repository, such as a PostgreSQL database, to verify user's identity.

4. Design of authentication and authorization service

4.1. Vision of the system

The aim of any software system is to make such a product, which will help users to do day-to-day job or to provide a service to them. The system developed for this project provides a centralized authentication and authorization service for a wireless community network. In other words, the system will help users of a community network to share their Internet bandwidth with other members of community and to be "online" within coverage area of community with use of a trustful authentication and presence of each user's responsibility. The system is intended to provide efficient and reliable AAS to the users.

A vision of the AAS for a community network is formulated as follows. Our WCN with a centralized authentication service will allow outside users to connect to the Internet through a Wi-Fi connection. The community users will be able to share their bandwidth using a wireless AP and will have rights to manage their APs. The system will be maintained by a network administrator, who will act as a support at the same time. The community network will share the Internet access; thereby it will improve lives of community's members by giving them ability to connect to the Internet from any point within community-wide coverage.

General conception of the AAS for a WCN is shown in Figure 7. APs and the authentication server are connected through the Internet. At the AP the whole access beyond contact the authentication server is denied. In order to get the access to the Internet, users must be authenticated.

In the figure community members – owners of *AP1* and *AP2* – are sharing their Internet bandwidth with other users in range. In order to provide responsible usage of their wireless networks, community members use AAS that is provided by authentication server – a captive portal is running on the server. A client has credentials at authorization server and is able to connect to the Internet from any AP registered in the system. A client may use any device with a web browser that supports SSL, for example

as it is shown in the picture – at *AP1* client connects from a laptop and at *AP2* he uses PDA.

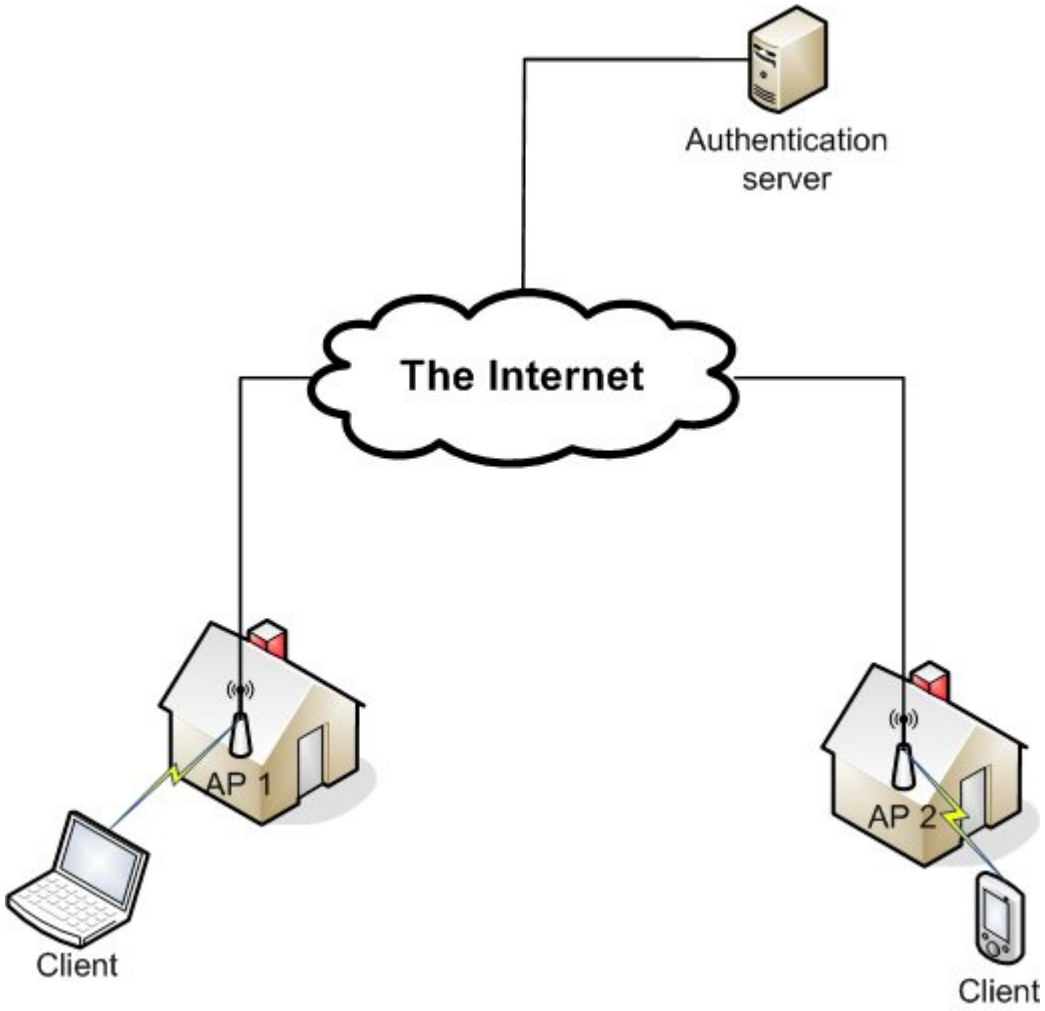

Figure 7 Wireless Community Network

Detailed statement gives general picture of the system with description of users who will use the product and of what they are able to do. The detailed statement of the solution is as follows. Our community network will allow users to connect to the Internet through a Wi-Fi connection. Our community network is a system which has centralized authentication and authorization service. The system has three different kinds of users: end-users, community users, and community administrator(s).

The community administrator(s) has rights to monitor the system (statistics, list of APs); to manage the accounting: delete users, make vouchers – according to location; to manage the network – to manage the software updates.

The community users have rights to register their AP(s) – make credentials; to manage the AP's access: guest, own; to customize the welcome message; and to look at statistics information of their AP(s). Moreover, community users will be able to set the advanced settings, such as wireless channels, IP address range for DHCP, the quality of service – set the amount of bandwidth to share.

An end-user is an outside user who does not have his own AP registered in the network and s/he wishes to get Internet access. S/he can do this by registering in the system and s/he will be provided by a limited bandwidth to use.

This section gave a general picture of the product, purpose and vision of the system. Now it is time to discuss more about the system and functional requirements to the designed application, use cases and make the actual design. The following sections cover all these topics.

4.2. Quality attributes

In order to design a system, an architect should know requirements to the designed system. Requirements are a reference point for architect; he makes such a solution that will fulfill all the requirements with appropriate assumptions. Requirements are defined by users of a computer system and they are interested in functional and behavioral requirements: what the system will be able to do. Moreover, the users are interested in how well the system will be working, *i.e.* usability, how fast the system reacts to users' requests, number of faults and handling exceptional situations. All these requirements are called quality attributes or non-functional requirements [35]. Quality attributes impose constrains on design of the system. This section discusses quality attributes of the AAS and the next section tells about functional requirements.

In this project the quality attributes were classified according to groups of users to whom one or another attribute seems more crucial. In the following paragraphs the quality attributes of each group of users are shown and described.

Quality attributes for network administrator are as follows:

- Maintainability – how easy it is to correct a defect or make a change in the software. How easily can the system be understood, changed and tested.
- Portability – the effort required to migrate from one operating environment to another.
- Reusability – the ability of the system to be a part of another project.
- Testability – the effort required to find and fix a problem.

A community user; who is sharing his own Internet bandwidth and uses advantage of community network to be online at any point of coverage area of network; he or she is interested in the following quality attributes:

- Availability – service should be available 24/7 at any time.
- Efficiency – how well the system utilizes the bandwidth, response time, latency.
- Flexibility – how easily new capabilities can be added.
- Integrity – only authorized access should be granted.
- Reliability – rely on service, ability to be online from own AP even if the service crashes.
- Robustness – stability of system, graceful response on users' errors.
- Usability – easy usage. The time spent to install and register an AP. Latency, easy to learn the system.

4.3. Functional requirements

Functional requirements are what the system should be able to do and how users can use the product. As the main feature of this project, it must be possible to authenticate and authorize users from other community networks. This implies assumption on what kinds of community networks can be combined to authenticate their clients regardless the network where clients are authenticated from. Also constraints on number of combined networks can be put. Further discussion about constraints and assumptions of

authentication and authorization of users from other networks see in dedicated section 5.4.

It is a requirement that software management system (SMS) must be present in the solution. SMS is a software product that supplies network's APs with updates of programs installed on them. Users may opt to get beta versions of software, but in this case they have to be aware of situations when beta software could crash hardware and entail other serious consequences. Client program of SMS must be publicly available on the community's web server; thereby community users can download and install it.

4.4. Use cases

Description of a user interaction with the system is presented with utilization of use cases. Use cases – are description of sequences of operations, which are performed by a system in response to a user's action. Use cases are employed during the whole process of development to define functional requirements of a system. Use cases show functionality of a system from a user's point of view. A use case describes action, which user wants to get from system [36].

In the AAS for a community network two main groups of users exist. For each group a use case diagram is made. On the diagram each group is represented by an actor; and actors have system use cases, which are described below. The third group of community users is end-users; there is no use case diagram that is made for them, since their use of system is very simple. The presented use cases show main functionality of the system, which must appear in the final solution.

In Figure 8 the use cases of community network administrator are shown. As for any administrator, in the system there must be a way to manage the users – *User Management*. The network administrator must be able to do user management – add, remove, block users. Moreover, the network administrator can *make a voucher* – login and password for a user, who can use them for a limited period of time.

In case an AP is suspected to be an initiator of spam, viruses, *etc*, the owner of that AP will receive notice. If after the warning the owner of AP does nothing to prevent the threat to the network, the AP can be blocked or even deleted from the community. This functionality is *Block/Delete AP* action on the use case diagram.

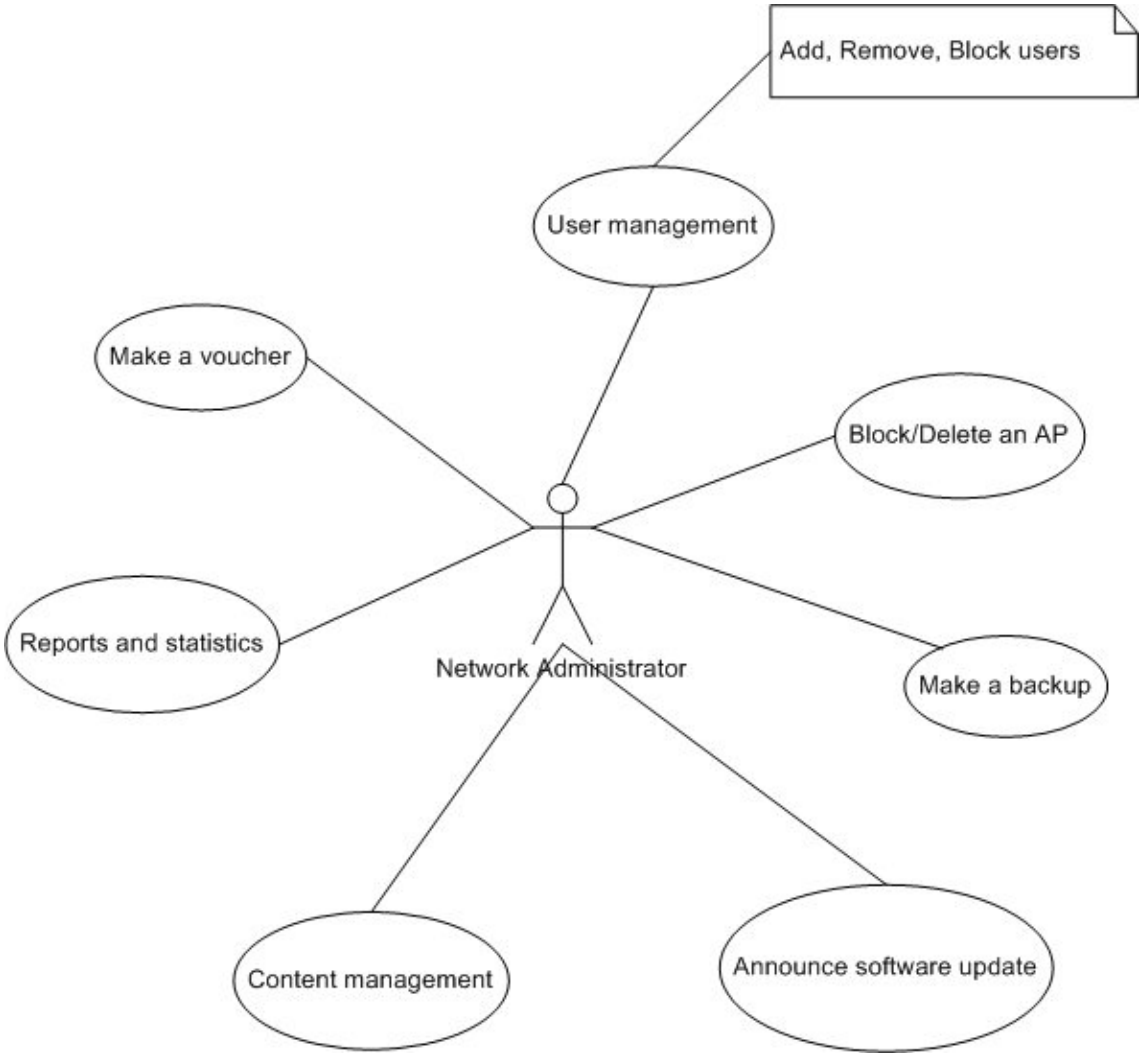

Figure 8 Use Case - Network Administrator

Since the system is intended to be reliable and available at any time, there should be action such as *making a backup* of the system. The network administrator must be able to make a reserve copy of the system in order to restore its capacity for work, if a crash of the system happens. The network administrator must ensure that all crucial information and services have a backup.

As the person responsible for all the functionality of the AAS for a WCN, the network administrator is also required to control software updates provided for the community's wireless APs. The administrator registers types of software, which are to be updated, and he announces new versions of that programs. This functionality can be triggered by *Announce software update* use case.

As community network has many APs in different places, a requirement for content delivery is made. Welcome pages at APs must be customizable and it must be possible to provide content to APs from the server side. A local content at an AP can be an announcement, advertisement, *etc*. The network administrator has *Content management* use case to provide such information to the APs.

To control the network and to identify threats in early stages the network administrator must be provided by a tool to see network's statistics and to make reports. The administrator may see a number of types of reports, *e.g.* traffic per AP, user activity per AP, *etc*. This feature is called *Reports and statistics* on the diagram.

Above the network administrator's use cases were described, now it is time to look at a community user's ability. A community user is person who shares the Internet bandwidth with others and benefits from it by being "online" within coverage area of a community network. This project also gives a possibility to authenticate and authorize users from other communities, *e.g.* users from other cities. Therefore, the community users (AP owners) could benefit in a larger extent by being able to use the services even at other community networks. The main activities of a community user as an owner of an AP are shown in Figure 9. The following paragraphs describe the use cases in details.

If anyone wants to be a part of community, he or she may register his or her own AP in the network; therefore, they agree to share the Internet bandwidth with others. The registration of AP is shown as *Register AP* use case. When a new AP is being registered, the owner provides information about the new AP, such as location, the owner of AP, *etc*. An existing community member can also add more APs to the network and, therefore, have more than one AP registered in the system. He uses the same use case to *register a new AP* in the network.

The user must be able to *make credentials*. Normally, user's credentials are login, password and e-mail address. The user might add more information into the system, *e.g.* address information. In order to keep a desired level of security, it is recommended to change the password on a regular basis. As a password is a part of user credentials, use case *Change credentials* can be used to change the password. Moreover, this use case provides functionality to change user's e-mail address.

The owner of access point must be provided by *Guest access management* – a possibility to share Internet bandwidth with users, who are not members of a community. Those users are clients of a community network, but they do not have their own APs registered in the system. For that group of users a limited Internet connection can be provided. The AP owner decides whether to provide the guest access or not. He also determines the bandwidth shared for each group of users – *Throttle bandwidth* use case.

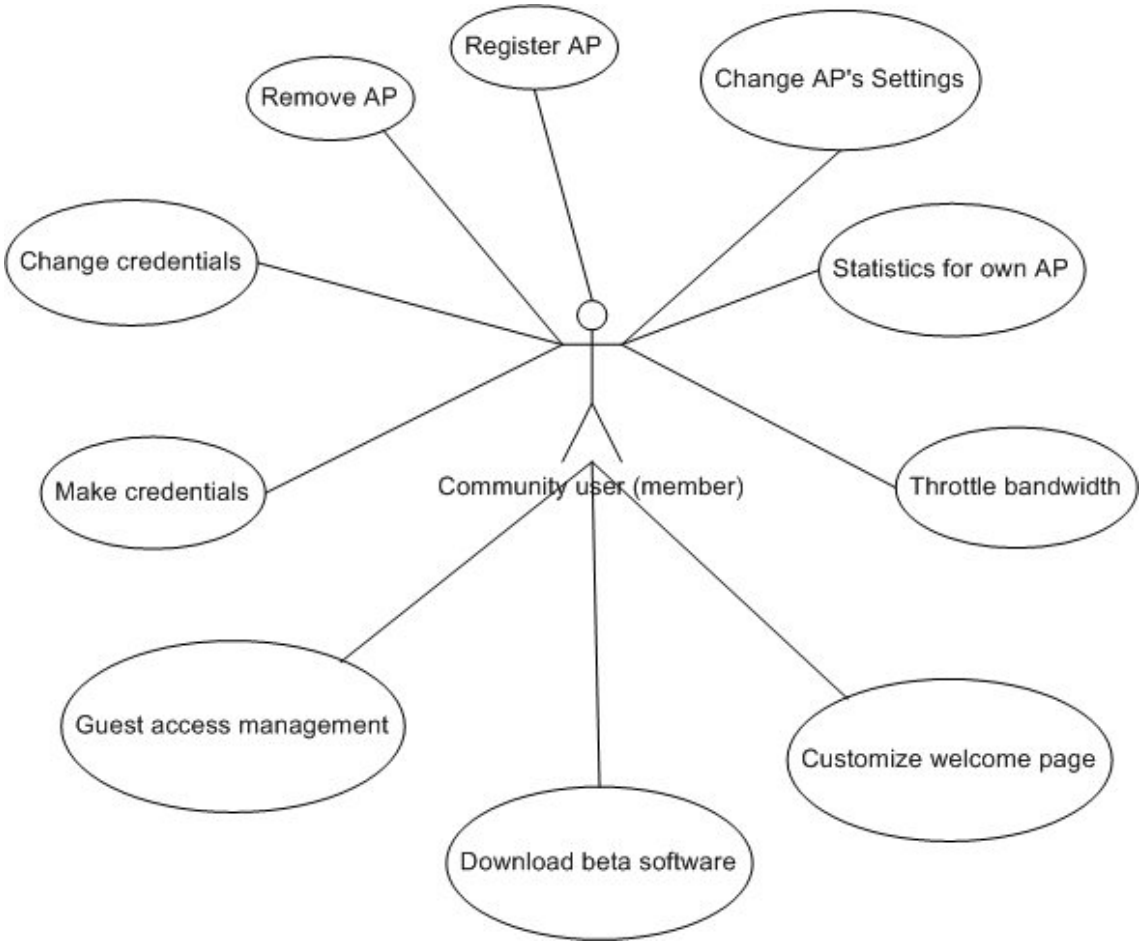

Figure 9 Use Case - Community User

Concerning the AP, it must have a customized welcome page. The AP owner customizes welcoming page utilizing *Customize welcome page* use case. The owner of AP can put any information on the page and change the layout of the page. Any content delivered from the network administrator must be present on the welcome page. Also, the owner must be provided by a possibility to manage his own AP – *Change AP's Settings*. AP management includes various setting of AP, such as a wireless channel, number of simultaneous users, *etc*. Finally, the AP owner must be able to see *statistics for own AP*, *e.g.* number of users for the last month, traffic passed through AP for last half a year, *etc*.

4.5. Design of the system

In this section we design the future system and describe how the system meets the requirements. The solution for the AAS for a community network is based on use of a captive portal. As a captive portal implementation, Wifidog was chosen, as it has many features implemented, such as user management, network statistics, content delivery, *etc* [9]. The Wifidog was also chosen due to author's background in languages used to develop the Wifidog project. The software management system is developed from the scratch. The development of system for software updates is done with intension to make its dependencies minimal and reduce them to the software packages used by Wifidog. The SMS is split into two programs: server and client. The server runs as a web application using the same web server as Wifidog and the client runs on AP's hardware.

As regards the authentication and authorization of users from associated networks, the authentication mechanisms supported by Wifidog system were examined. Thus, Lightweight Directory Access Protocol (LDAP) directories were chosen to store users' credentials. LDAP is an extensible Internet protocol used to access directory services [37]. The main reason for choosing the LDAP directories to store users' information is its ability to make search in an LDAP server hierarchy. This means that if an LDAP server could not find information in its directories, it can ask another referred LDAP server to search for requested information and return it, if it is found. Details about that are presented in the section dedicated to description of authorization and authentication

mechanism of a user from an associated community network (see section 5.4). More information about LDAP services can be found in [38]. Luckily, Wifidog supports LDAP authentication and LDAP directory services can be used as a register of user names and passwords.

4.5.1. Elements of the authentication server

In Figure 7 general picture of a WCN was shown. Now we describe each part of the system – authentication server and network's APs. The authentication server is a significant part of the solution and it includes many services, as it is shown in Figure 10. The authentication server consists of the following items:

- Web server – a program that accepts users' requests and provides responses back to the clients' web browsers. The communication protocol between clients' software (web browser) and the web server is hypertext transfer protocol (HTTP).
- Database server – a program that provides service to store the information into database (storage) and ways to retrieve the information from the storage. The database systems normally rely on client-server architecture.
- Mail server – a program that provides services to transfer e-mails from one computer to another regardless the network where machines are located. The mail server is also known as mail transfer agent or mail exchanger in context of DNS.
- DNS server or name server – a program that maps domain names to their IP addresses and vice versa.
- LDAP server – a program that implements LDAP directory services.

Figure 10 shows how parts of the authentication server are connected to each other. As it was said before, Wifidog runs on *web server* and takes user credentials from *LDAP server*. It also uses a *database server* to store information – many kinds of statistical information for network monitoring, report tools, *etc*. *Mail server* is used by Wifidog to send e-mails to community users; it could be a validation mail, a message from network administrator, *etc*. *DNS server* is not connected directly to any of other servers; it serves

the requests of community users. The used DNS server is a caching name server, which speeds up domain name resolving by using cache that allows DNS server to respond more quickly to multiple queries for the same domain or host. All this servers are components of authentication service presented in Figure 7 as *Authentication server*.

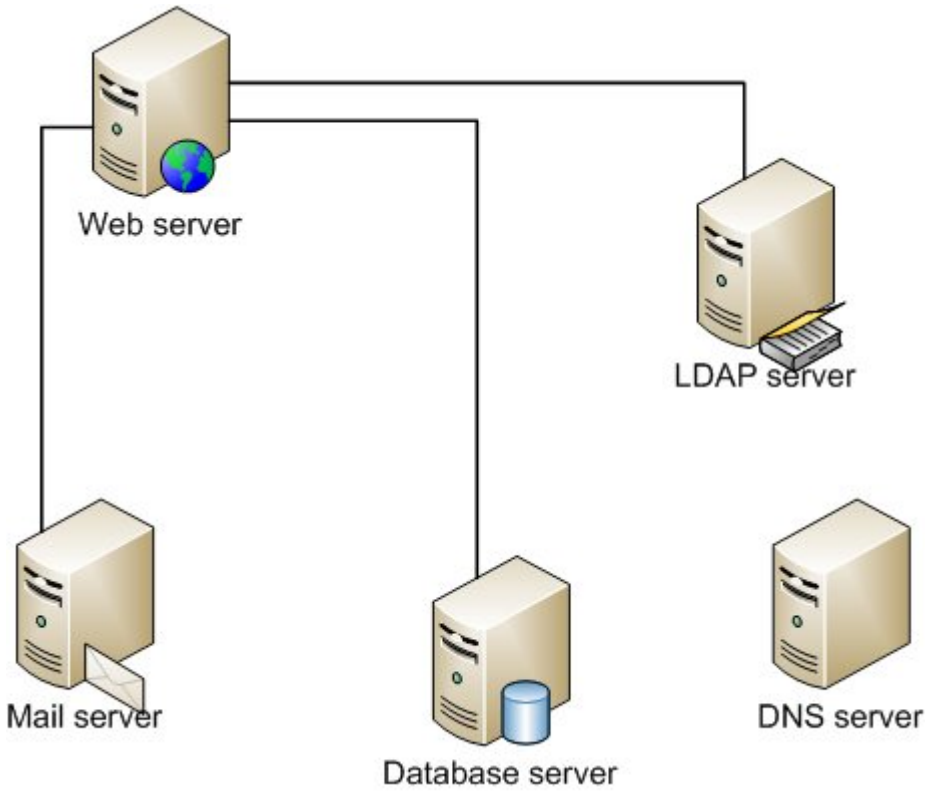

Figure 10 Authentication server's parts

4.5.2. Components of an access point

Community network's APs are essential as they provide wireless access to the Internet to community members. As it was shown in Figure 7, an AP consults with authentication server for granting the Internet access to clients. Analogically to the authentication server, an AP includes several components, which are shown in Figure 11. The AP's constituents are:

- Wifidog gateway – a program that is a part of Wifidog captive portal suite. It intercepts client's requests and redirects client to pass authentication procedure, if the client was not authenticated before.

- Firewall – a program that filters traffic between clients' devices and the other side of AP and allows only authorized traffic to pass.
- DHCP server – a program that assigns network addresses to client devices. This avoids manual setting an address to each device in range.
- DNS server – a caching DNS server to speed up clients' requests.
- NAT – to share Internet bandwidth by having only one public IP address.

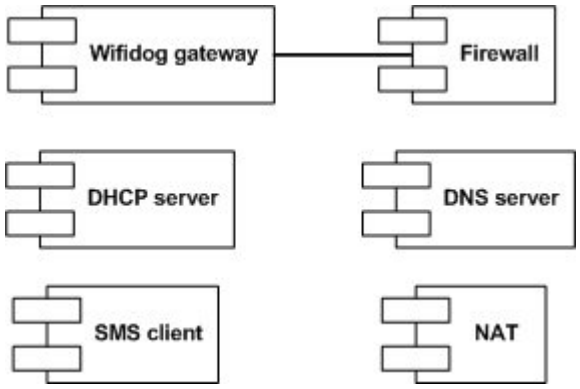

Figure 11 Wireless Access Point's components

Wifidog gateway uses firewall to control traffic going through the AP. When a new client tries to access a web page, the Wifidog gateway will redirect them to the authentication server, where they can log-in. The Wifidog gateway plays with firewall rules and according to authentication server's directives, either allows or denies certain network access.

As discussed in section 2.2.2, DHCP, DNS and NAT are vital services of an AP. DHCP is used to automatically configure wireless clients' devices; NAT is used to provide the Internet access to multiple users using only one public IP address from ISP. A separate caching DNS server is used on an AP in order to reduce load on central DNS server of the community network.

A client of software management system – *SMS client* – is a component of an AP in case a member of community network wishes to get software updates. *SMS client* runs in the AP box and periodically checks for software updates on central server; if an update is available; the client downloads and installs it automatically without user's

meddling. Launching *SMS client* should be scheduled by the owner of the AP by using an appropriate software package for it, *e.g. crontab* task scheduler on Linux platform.

4.5.3. Conformity to requirements

This subsection discussed conformity to requirements and makes sure that we did not omit any use case. Wifidog's functionality covers most of the functional requirements to the system. By use of the Wifidog auth service the network administrator is provided by the following use cases:

- User management
- Block/Delete AP
- Content management
- Reports and statistics
- Make a voucher

Other use cases, which are not covered by the Wifidog's functionality, are provided by use of other software products. The use case *Make a backup* is provided by means of software packages used in the system, therefore, each service has to be in possession of software with backup tools included in it, in order to be able store a copy of the whole system. The services that needed to have a backup copy are all the servers at the authentication service side. Moreover, the DNS server has to be redundant in order to meet a non-functional requirement – availability. The use case *Announce software update* is provided by the SMS, which runs on the same web server as Wifidog auth server. All the quality attributes are met in the design:

- Maintainability is provided by use of the Wifidog system, which requires knowledge of PHP and C programming languages to correct defect or to make changes in the software.
- Portability is present in the solution as the Wifidog auth server and software management system's server do require a PHP-enabled web server, regardless the platform, which it is running on. The backend services, database, name

server, mail server and LDAP directory services can be run on different platforms too.

- Reusability is present as a WCN that could be a segment of a large installation of many networks.
- Testability is provided by tools of network monitoring and logging, as well as reporting and statistics tools of the Wifidog system.

As regards a community user's requirements, all of them are present in the design and they are provided by different software. The use cases provided by the Wifidog gateway are:

- Register AP
- Remove AP
- Statistics for own AP
- Throttle bandwidth
- Customize welcome page
- Guest access management
- Make credentials
- Change credentials

The use case *Change AP's setting* is provided by firmware running on an AP. Normally, it provides to a user a web interface with a possibility to put a range of addresses for DHCP leases, to configure DNS server, to set a wireless channel and many other options. The use case *Download beta software* is performed by SMS client running in an AP's box. As for user's non-functional requirements, all of them were taken into account while designing the system. Mainly, the quality attributes are met with use of distributed services on the auth server side and by adding redundancy to them. This provides to the designed solution a high availability, efficiency, flexibility, reliability and robustness. The integrity is ensured by the Wifidog authentication service and we fully rely on it. The usability of the Wifidog system was considered as acceptable by the author of this work, as he did not spend much time to learn how to use web interfaces provided by the Wifidog.

5. Implementation of authorization and authentication service for a wireless community network

This section depicts implementation details of the authentication and authorization service for a community network. The construction of the system is done according to the solution design made in previous chapter. A special emphasis is put on the authentication server as it is the most crucial part of the system. The authentication and authorization of users from associated network is implemented as well as software management system. Moreover, this chapter presents implementation of quality attributes of the system.

5.1. The whole system

The system is implemented with use of available open source packages and is constructed on Linux platform. Wifidog captive portal suite was used as main part of authentication server. Originally Wifidog was designed to use a database as user credentials repository, but it also supports other back-ends for repository, one of them is LDAP directory. Supporting by Wifidog authentication against LDAP directory was used in the implementation mainly to provide capability to authenticate clients from other networks.

Operating system used for AP is OpenWrt [39]. OpenWrt is a Linux distribution for embedded devices and it provides a fully writable file system with package management. Building an application as a package for OpenWrt frees developers from having to build a complete firmware. Moreover, many packages already exist in OpenWrt and we can benefit from that. Thus, OpenWrt packages were used to provide vital services of an AP. Details of used software are in section 5.2.

Before digging into details of the implementation we will give a general overview of the system. A wireless community network was shown in Figure 7 and stuffing of authentication server and an AP were given in Figure 10 and Figure 11 respectively. Figure 12 gives detailed description of WCN's constituents in its complete

implementation. An AP side is fairly simple and it contains all the AP services mentioned in section 4.5.2. Figure 12 also shows that some of the services in AP are combined and provided by one software package. Thus, *iptables* package provides firewalling and NAT services, and *dnsmasq* package supplies DHCP and DNS services.

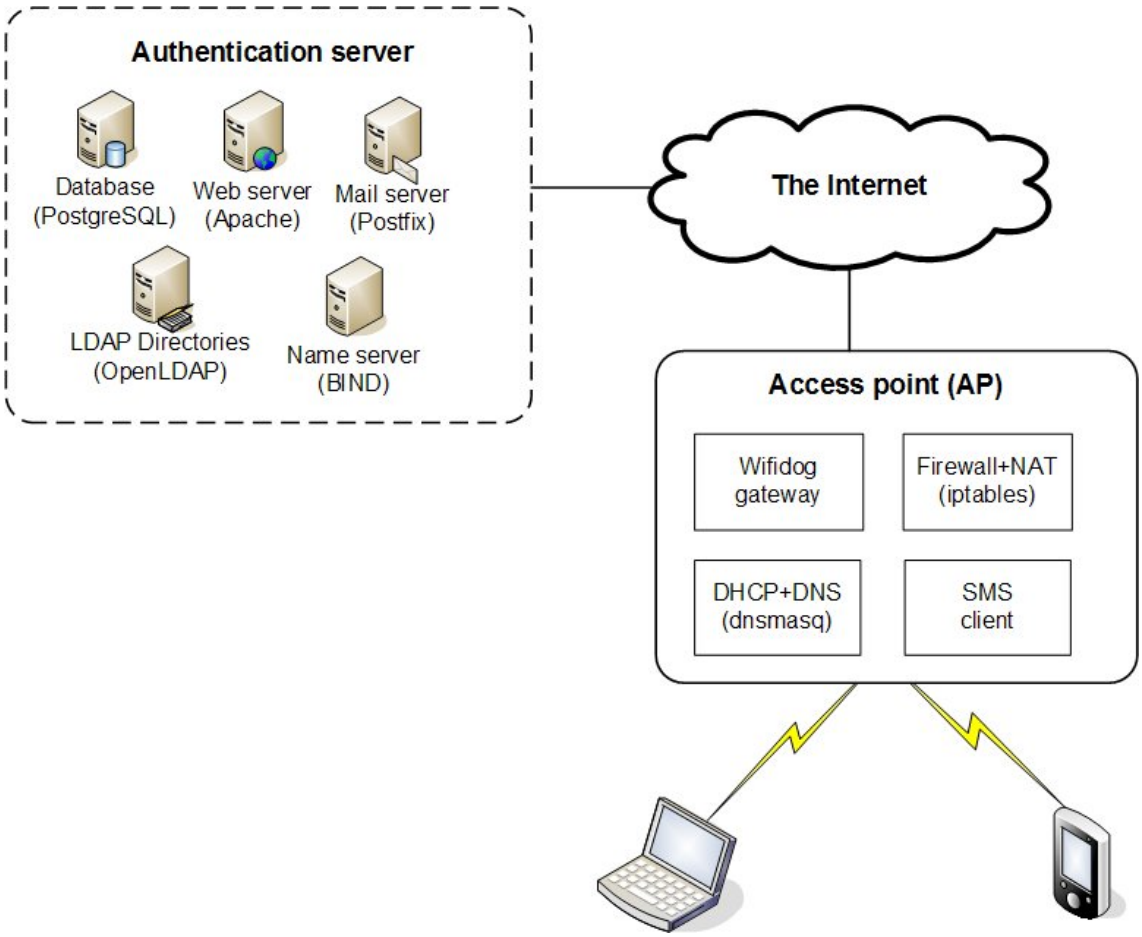

Figure 12 Wireless Community Network (Detailed)

The main component of the server side is a web server as it supplies many services. Two essential services run on web server: Wifidog authentication server and SMS server. Other servers, such as a mail server, a database server and LDAP directories, are used by Wifidog captive portal as helper services. DNS server is present to serve clients request in domain names and hosts resolving.

We will use sequence diagrams to describe the intercommunication between AP components and authentication server parts. We will start from the time point when a client connects to a network's AP. When a roaming user connects to an AP, his client

machine needs to obtain configuration information for current network. It is preferred to do the configuration automatically without user's interference and the DHCP protocol is the right choice to do that. Figure 13 briefly illustrates the message interchange between client and the DHCP server. In the figure we can see that the client sends a DHCP discover request to the network and the network's DHCP server offers an IP address and configuration information to the client. As we know the wireless infrastructure of an AP is served by the DHCP server running in the AP's box, therefore, the client will get a DHCP offer from the DHCP server of the AP. After receiving a configuration offer, the client requests the information from the server thereby confirming the acceptance of the configuration. The server acknowledges the request, thus the client is configured for use of current network. More information about DHCP can be found in [19].

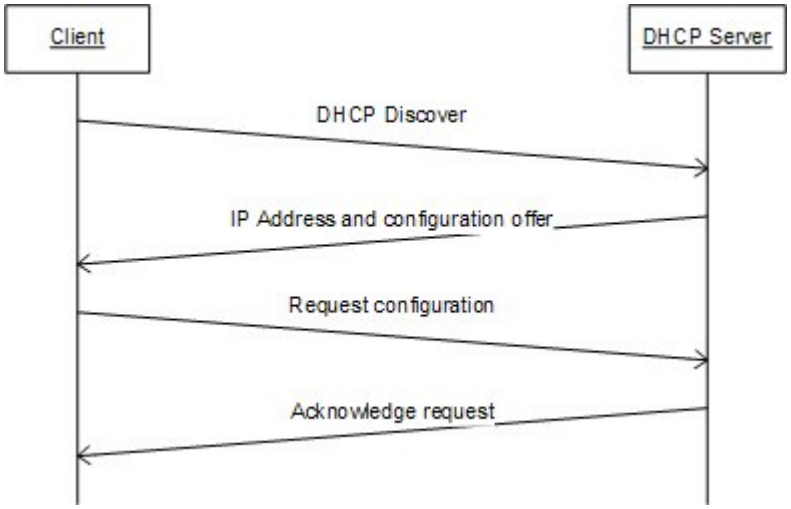

Figure 13 A DHCP lease to client

The client configuration performs automatically when a roaming user gets into range of WCN and connects to an AP. The client is immediately issued a DHCP lease and it is ready to work in wireless infrastructure of the AP. The client authentication flow of Wifidog was shown in Figure 6 and describes general message flow without details of interaction between constituents of the AP and the authentication server. Detailed client authentication message flow is given in Figure 14 with representation of all involved components.

Initial web request sent by client is intercepted by Wifidog gateway and the client is redirected to the authentication server to pass the authentication process. The user gets authentication form, where he is asked to put his user name and password. After the user name and password are sent to the authentication server, it sends a lookup request to the LDAP server and if credentials are found, LDAP server acknowledges the request. Then the client is redirected to gateway with a unique identification. The client makes a request to gateway with received identification whereupon gateway verifies the identification. If the authentication server has verified the identification, the gateway changes firewall rules for the authenticated client. Firewall changes rules to allow client packages pass through the AP. Allowing rules are based on client's IP address and MAC address. Wifidog gateway controls the time limit of the client being inactive, thus if the client was inactive within a certain period, access to the network is revoked.

Here is one more service involved, which was not mentioned, it is NAT. NAT is needed to serve the wireless infrastructure of an AP by using only one public IP address from the wire. In the implementation NAT is provided by firewall as the same package is used for firewalling and NAT. Firewall is configured to do message forwarding from private IP range of wireless infrastructure to single public IP address from wire and vice versa. As we can see, NAT and DHCP services are connected to each other. NAT service should be configured to the same address range, which DHCP uses to issue leases to clients.

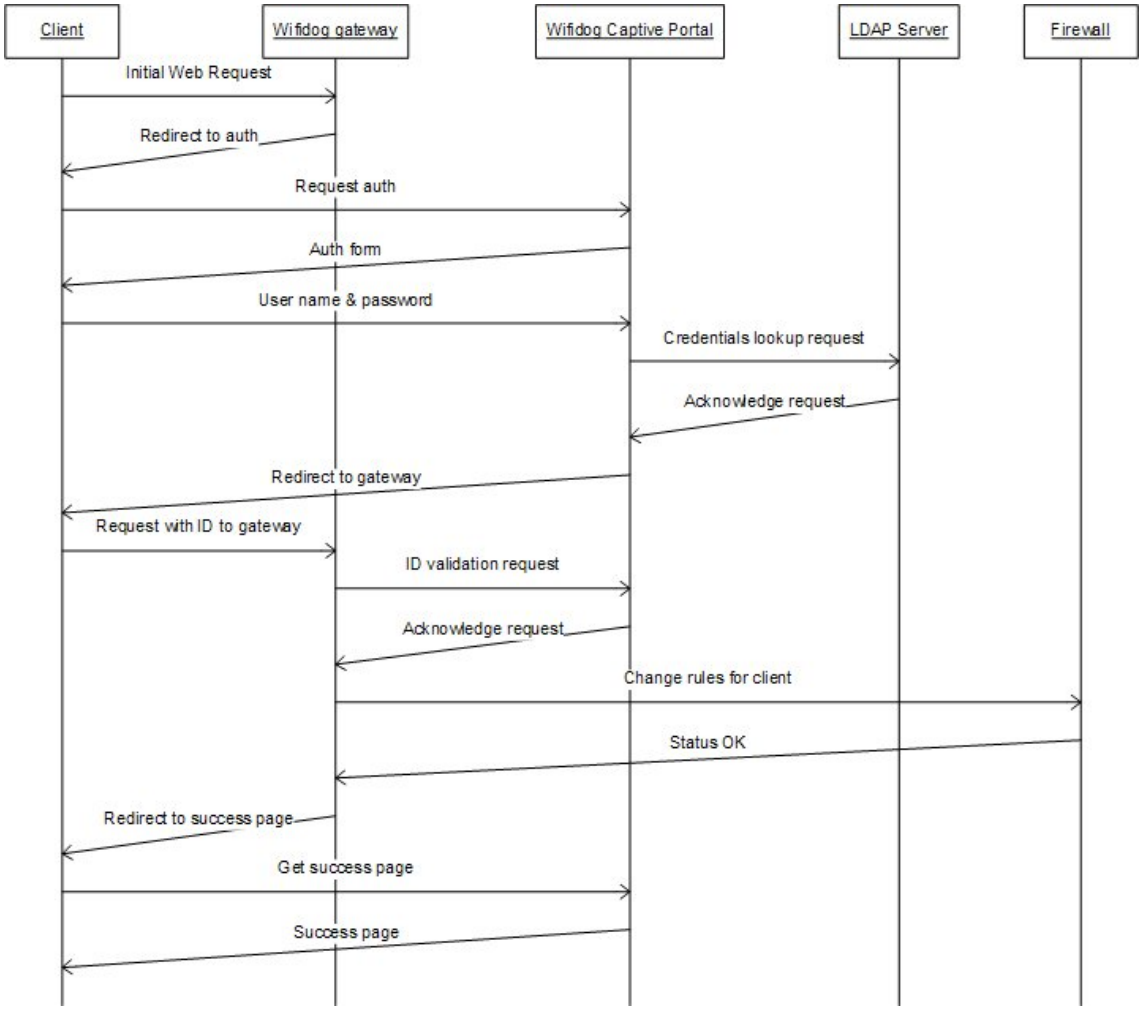

Figure 14 Detailed client authentication flow

A caching DNS server is used in an AP box to reduce the load on central DNS server and to speed up the clients' Internet connection. Apparently DNS cache will contain the most often used domain name and clients will get faster responses to their requests. One more constituent of the AP is SMS client, but details about it are given in section 5.5.

The authentication server parts, such as mail server and database server, are not described in details due to the fact that they are used by Wifidog captive portal and they are not the main focus of this thesis. Simply, Wifidog requires a mail server to send validation emails to newly sign up users. Database is used to store network statistics, which are used to provide various types of reports in Wifidog system.

5.2. The software used in the system

In previous section we were talking about software components used in the system, but we did not say what exactly was utilized. First of all, we would like to mention that the software used in this project is open source and published under GPL license or under licenses that are compatible with GPL. Therefore, the AAS for a WCN is published under GPL license. This means that the solution is free and everybody can use it in terms of GPL and any enthusiasts are welcome to contribute to the project.

The server software consists of several servers, which provide the core service to the system. The web server is Apache HTTP server from Apache Software Foundation. It is open source, provides reliable service and it has been the most popular web server in the Internet since April 1996 [40]. Apache web server has many extensions and many of them are required by Wifidog captive portal. One of the main required extensions is PHP extension. It is a must in the web server as Wifidog captive portal is written in PHP. Moreover, Wifidog uses a PostgreSQL database; therefore, the PHP pgsql extension must be installed. If the reader is interested in widening his knowledge of working with PostgreSQL databases from PHP, we refer him to Richard Stones *et al.* book (see [41]).

PostgreSQL is a relational database management system (RDBMS); it is open source software and published under liberal open source license – the Berkeley Software Distribution (BSD) license. "It is a very powerful database system and has more than 15 years of active development and a proven architecture that has earned it a strong reputation for reliability, data integrity, and correctness [42]." Since we are working with PostgreSQL from PHP, a library interface for PHP must be installed with core system of PostgreSQL.

Wifidog captive portal also sends emails to community users, *e.g.* validation email. Wifidog uses PHPMailer to be able to send emails automatically from PHP scripts. The used mail server is Postfix – a fast mail server, which appears to be easy to administer and secure.

LDAP Directory service is provided by OpenLDAP – an open source implementation of the Lightweight Directory Access Protocol. “The OpenLDAP project is a collaborative effort to develop a robust, commercial-grade, fully featured and open source LDAP suite of applications development tools [43].”

DNS service is supplied by Berkeley Internet Name Domain (BIND), which is an implementation of the DNS protocols and provides an openly redistributable reference implementation of the major components of the DNS [44]. The BIND distribution includes the following components:

- Domain Name System server
- Domain Name System resolver library
- Tools for managing and verifying the proper operation of the DNS server

The DNS server is used to provide a stable architecture on top of which a naming architecture can be built. The translation between domain names and IP addresses is done by DNS resolver library and the library intended to be connected with applications requiring naming service [44].

Concerning the AP’s box software packages, services, such as DHCP, DNS, NAT and firewalling are provided by OpenWrt packages. As it is visible from Figure 12 DHCP and DNS services are supplied by one package – *dnsmasq*. *Dnsmasq* is a lightweight DNS forwarder and DHCP server, it is targeted to use in embedded devices and to serve a small network. *Dnsmasq* supports static and dynamic leases, thereby the AP owner has an option to configure static addresses for own computers.

The firewalling and NAT services are provided by *iptables*, a software package that allows configuring the tables containing chains of rules for the treatment of packets. Every network packet arriving at or leaving from the AP is processed by traversing the chains. *Iptables* is used by Wifidog gateway to grant or deny the access to connected wireless clients. We mentioned Wifidog gateway, it is a client side program of Wifidog captive portal suite.

One more component of the AP is the SMS client. It automatically supplies software updates and currently it is in beta version. Startup of software management system's client must be scheduled by a scheduler service, such as *crontab*.

5.3. Reliability of services

When we talk about reliability of AAS for a WCN, we talk about reliability and availability of the authentication server. In Figure 10 and Figure 12 the authentication server parts were shown. The most demanding service is the web server as Wifidog server and the SMS server run on it. Other services with a high demand are PostgreSQL database, which has frequent data inputs: network statistics; and LDAP directories, which store user credentials. The reliability of the system depends on durability of services with high demand; therefore, they must be implemented as robust services.

Mail server and DNS server are not considered as crucial since the WCN can be working without them, but with limited functionality. Each DNS server's client is configured to use multiple DNS servers: if community network's DNS server is down, client will use another DNS server higher in DNS hierarchy. This means that clients still will be able to resolve domain names in the Internet. In case of mail server's failure, the AAS will not be able to send validation emails to new registered users, thereby the user will not able to sign up for community network during the break down of mail server. This is considered as not a big loose of functionality, as normal user authentication and authorization is not dependent on mail server.

To make services reliable a number of ways can be used: to make a service redundant, make a distributed solution, *etc*. In this project the reliability and availability are ensured by means of distributed applications approach. Starting from the most important part, the web server, where the Wifidog and SMS servers run, is a distributed Apache2 web server. The way of having the web server distributed makes certain the availability of the service. In the distributed environment replication and load balancing become a major topic for discussions. Quanzhong Li and Bongki Moon implemented a practical solution of distributed web server and described their Distributed Cooperative Apache (DC-Apache) web server approach in their work [45]. DC-Apache is a very

scalable, flexible and cost effective web server for WCN. The data replication and load balancing schemes make certain the avoiding the problem with overload. In case of WCN, the growth of the network will cause a bigger load on the web server, since it provides the core services for AAS – the captive portal. Adding a new server is very easy and not costly, any available machine could be added as a cooperating web server [45].

Behind the web server, the database server runs to support the main services. All the information concerning the community network: APs, users' activity, statistical information, *etc*, is stored in the database; therefore, the database service is critical to the AAS. Due to high demand the database service must be available at any time and as the same with web server, with growth of the network the demand to database will increase. In the solution for AAS for a WCN PostgreSQL RDBMS is used.

Many tools exist to make a cluster based on PostgreSQL database servers. The master-slave replications are popular in the database replication world. One of the proposed solutions for PostgreSQL database master-slave replication is Pronto – "a protocol that orchestrates the execution of replicated database," introduced by Fernando Pedone and Svend Frølund [46]. Pronto protocol provides highly efficient database replication and reduces the database failover. The Pronto protocol acts as a middleware between business application and the database. The Pronto can not be used in the AAS for a WCN project, since it was designed for Java-based application and it is actually a substitute for Java database connectivity (JDBC) with replica functionality inside.

Another master-slave based database replication system is Slony [47]. The Slony replication system supports cascading and aims primarily for large enterprise database replications. The Slony uses synchronous replication and assumes that all the nodes are available during the replication process. This system is not applicable for the system we implement due to the risk of losing frequently updated data, *e.g.* the AP statistics, in case of failure.

Multi-master replication systems have a good approach of doing the data replication. The systems as Postgres-R provide an efficient replication for a cluster of databases. "The primary use of Postgres-R is to build load-balancing and high-availability database

systems on commodity hardware [48].” The Postgres-R is an extension of PostgreSQL that provides scalable and reliable replication in a cluster of computers. The Postgres-R system is an implementation of eager replication approach, which emphasizes on data consistency [49], [50].

The last replication system examined in this project is pgreplicator. Pgreplicator was developed to extend the original potentials of PostgreSQL RDBMS. The pgreplicator system provides asynchronous replication and, therefore, the possibility for PostgreSQL to be able to manage distributed databases [51]. Since pgreplicator uses asynchronous nature of data replication, it is very appreciated in the systems with frequent data inputs. As the community network is one of systems with frequent data inputs, pgreplicator was used in this project.

The users’ credentials are stored in the LDAP directories due to its ability to provide hierarchical search. It is expected that there will be no often input of data, whereas read operation will be performed often in the LDAP directory services and because of this there is no urgent need to use runtime synchronization between master and slave of LDAP directories. The software used to provide LDAP directories services is OpenLDAP open source software. In this work, LDAP directories data replication is done by means provided with the OpenLDAP suite [52].

5.4. The authorization and authentication of clients from associated community network

The use of LDAP Directory services was not a spontaneous decision. The LDAP was chosen for its extensible hierarchical search functionality. With use of LDAP hierarchical search the authentication of a client from an associated community network becomes a trivial task. To authenticate clients from an associated community a number of requirements must be met. First of all, all associated communities have to store user credentials in LDAP directories. Secondly, their LDAP directories have to know about each other, details about connection of LDAP directories will be given later in this section. General picture of connection between two communities’ LDAP servers with ability to authenticate the clients from both of networks is shown in Figure 15. The

Wifidog authentication server runs on the web server (DC-Apache) and it consults with LDAP directories for user credentials.

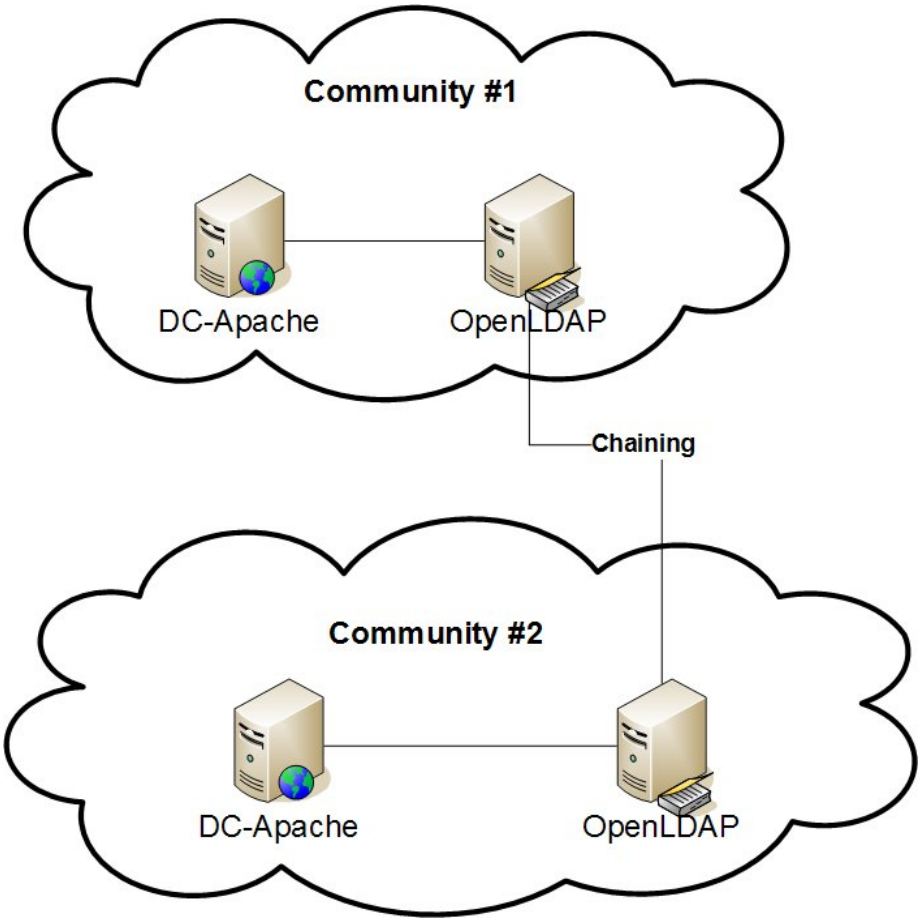

Figure 15 LDAP Directories connectivity between communities

While a user from an associated community (community #2 in Figure 15) passes the authentication procedure at community #1, the Wifidog server consults with local LDAP directory service for user credentials. Since user is from an associated community, local LDAP server can not find user's credentials in local repository and it sends request to LDAP server of community #2 for user credentials. LDAP server of community #2 serves the information to LDAP server of community #1, which acknowledges the Wifidog server's request for user's credentials and Wifidog authenticates and authorizes the user from the associated community, as his credentials were retrieved.

All the “magic” with LDAP servers asking each other for the user information is done by means of the LDAP serves themselves. They have a configuration option called *referral*, which enables a master-slave relations. Referral is a link to other LDAP server, which contains requested information. The LDAP servers can be mutual master-slave; it means that they can ask from each other. In this work LDAP servers were configured to follow the referrals and return to client a complete result, this is called *chaining*. An example of how chaining works is shown in Figure 16 [53].

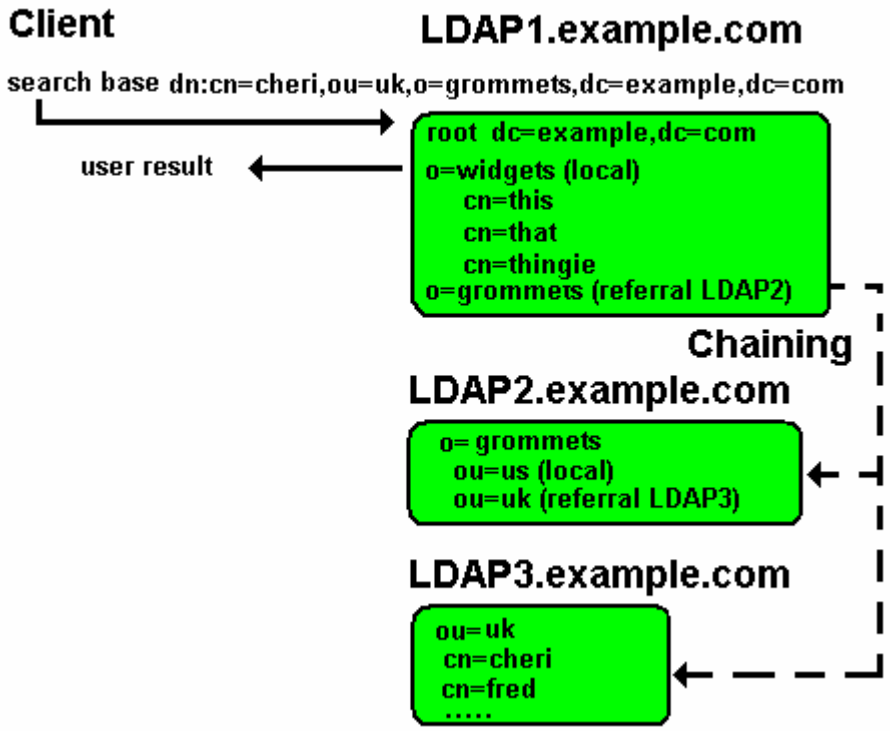

Figure 16 LDAP chaining [53]

In Figure 16 a client of LDAP service requested information for element *cheri* from an organizational unit *uk* and with distinguished name *grommets*. In LDAP1 server *grommets* is marked as referral to LDAP2 and LDAP1 server follows the referral to LDAP2 with request for information. In LDAP2 server the *uk* unit is marked as referral to LDAP3 server and LDAP2 server follows the referral to LDAP3. The name *cheri* was found in the server LDAP3 that returns the result to LDAP2, which in its turn returns the result to LDAP1 server. Finally, LDAP1 server serves the client with a complete result. The feature of LDAP server to do chaining and hierarchical search

made possible to authenticate clients from associated community networks without much of development effort.

The example shown in Figure 16 presented multiple levels of chaining. This is called *cascading chaining* and it is recommended to limit number of server in each chain. As more servers we have in a chain, the longer is the response time to LDAP client. The network administrator must ensure an acceptable level of LDAP requests latency. Also, number of referrals within one LDAP server should be limited, which gives a restriction on number of community networks that can be put together in a syndicate.

LDAP servers of associated community networks can be configured in a hierarchical structure with one root server or in a distributed manner, where each LDAP server can directly talk to another. The approach with central LDAP server gives advantages as well as disadvantages. The advantage of having a root LDAP server is easy management of LDAP structure; changes to LDAP structure can be made easily, adding a new associated community to an existing syndicate of community networks will entail only adding a new branch in the root server. Deleting a community network from a syndicate is also a change at one server. Disadvantage of this approach is that authentication servers from all community networks of a syndicate will request user credentials from one root server, which entails a high load to central LDAP server and makes it a bottleneck.

On the other hand, having no central LDAP server and configuring all local community's LDAP servers to speak to each other entails a lot of management and administrative work. Adding a new community to a syndicate or deleting one network from it becomes not a trivial task. In this case all network administrators of all community networks in a syndicate must do appropriate changes in their LDAP servers. If LDAP server of a new associated community network was not added in one of the networks in syndicate, then users from that new associated network will not be authenticated in the community network, where appropriate changes were not made. Thus, approach with no central LDAP server gives a lot of administrative work, while removing a bottleneck as a root LDAP server. It is up to a deployment manager to decide, which approach to use.

5.5. The software management system

Historically, in software industry a product does not live a long time without changes. While using a product, users sometimes want to get more from the software, and thereby change existing functionality, *etc*. With new wishes from users, new requirements come to existing solutions, which have to be changed. Programmers use versioning to keep track of software changes, therefore, with time there might appear a number of versions of a particular product. Normally, the latest version is the best, where most of bugs and errors are fixed.

Another reason to have different versions of a product could be different functionality included into a particular version. For example, a new version of a product appeared on market with a new functionality. For some users the functionality of the product they have might be enough, and they do not have any need to buy the new version, while others were waiting for the new functionality. Thus, the new version does not have to be always a substitute for the old versions. It is completely up to users whether to take the new version or not.

To provide software updates to community users, who wish to get latest versions of software installed on their APs, the software management system was implemented in this project. In Figure 12 the components of server and AP side are shown. The SMS server is written in PHP and runs on the same web server as Wifidog authentication server. In AP box a lightweight SMS client is installed. This client triggers the software updates for an AP.

The SMS is intended to automatically provide updates of installed software, *e.g.* firmware. If a user opted to get automatic updates, the system will work without user's interference. The system will download a new version of software and install it. While software update some services might be down during the installation, therefore, the user configures the SMS on preferred time of running, when the AP does not have high loading, for example in the nights.

The SMS is split into two parts: the server part and the client part. The server part is written in PHP and, therefore, needs a web server to run its services. In this project the software management system's server and Wifidog server run on the same web server. Moreover, the SMS server uses a database to store and retrieve the information. The database is placed in PostgreSQL RDBMS, the same one, which is used by Wifidog server. The SMS client is designed to be able to run in embedded devices with OpenWrt Linux distribution version White Russian 0.9. It is written in C and has minimal dependencies in installation.

5.5.1. Design of the database

The aim of the SMS is to automatically provide software updates to the clients. Hereby, the SMS must be able to keep track of software types and their versions. Moreover, it is necessary to ensure that software provided to a client will run on its operating system. Thus, we have to store following information:

- Software types – what kind of software is available for updates, *i.e.* software name,
- Software versions – each version of a software is represented by a package,
- Supported operating systems – the operating systems supported by one software or another

All this information is stored in a database, which is designed as shown in Figure 17. The figure shows entity-relationship diagram (see [54]) of the database, where the entity "swlist" represents the software types, the entity "swpack" signifies the software packages and the entity "oslist" denotes the supported operating systems.

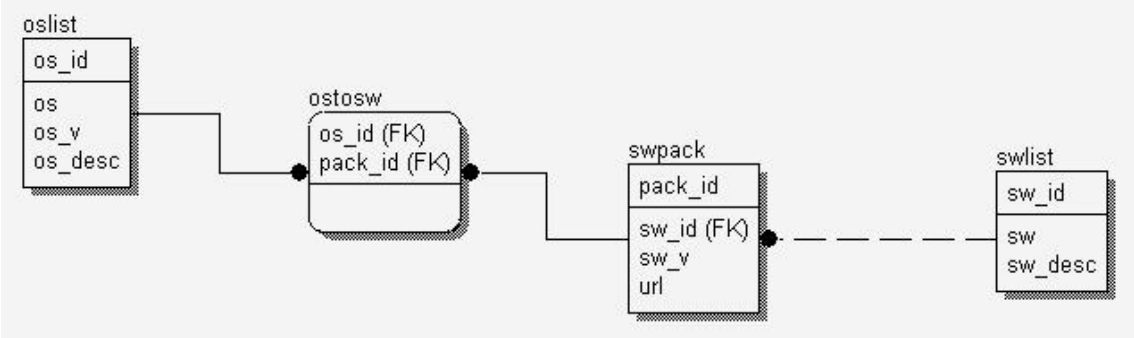

Figure 17 The database of software management system

The relationships between entities are different. The entities "swlist" and "swpack" are related by one-to-many relationship, which means one software type can have many packages and each package refers to only one software type. The entities "oslist" and "swpack" are connected by many-to-many relationship, which means one software package can be run on many operating systems and one operating system can be supported by many software packages. In order to provide many-to-many relationship between "oslist" and "swpack" the cross-reference entity "ostosw" was made. The detailed description of each entity is given in order to depict the whole database structure. The physical model of the database is defined by the structured query language. The script that creates this model can be found in Appendix I.

The entity "oslist" consists of:

- os_id – serial primary key,
- os – operating system name,
- os_v – operating system version,
- os_desc – a short description of operating system.

The entity "swlist" consists of:

- sw_id – serial primary key,
- sw – software type, *i.e.* software name,
- sw_desc – a short description of software.

The entity "swpack" consists of:

- pack_id – serial primary key,

- sw_id – reference to software type from “swlist” entity,
- sw_v – version of software,
- url – location of software package.

The entity “ostosw” consists of:

- os_id – reference to operating system from “oslist” entity, primary key,
- pack_id – reference to package from “swpack” entity, primary key.

5.5.2. The server of software management system

The server of SMS consists of two logical units: first, administration tool for the network administrator; second, the public site for all the clients. They must be separated by putting them in separate directories of a web site or by defining two different web sites for each. It is important that the network administrator controls the access to the administration unit. In this project the administration unit was put in a separate directory of the SMS web site and the Apache web server was configured to require visitors of this directory to login with a user identification and password.

The main screen of the administration tool of the SMS is shown in Figure 18. The SMS administration tool provides following functionality:

- Addition of new software type – button “New software” in the figure,
- Addition of new supported operating system – button “New OS” in the figure,
- Addition of new software package – button “New” in the figure,
- Editing information of a software package – button “Edit” in the figure,
- Removal of a software package – button “Remove” in the figure.

The main usage of the administration tool is to announce software updates. This can be done in different ways. If we announce support of a new software type, first, the new software type must be created in the system. Second, a new software package should be made for that software type. While addition of a new package we specify the version of software and what operating systems are supported by the new software. The

announcement of new version of existing software type can be done in two ways: by creation a new software package or by editing one of the existing packages. The first way keeps the old version in the system, while second replaces the old package by a new package. The new software updates are automatically processed by the SMS clients without user interference. The user interface of administration tool is intuitive and all its components can be found in Appendix II.

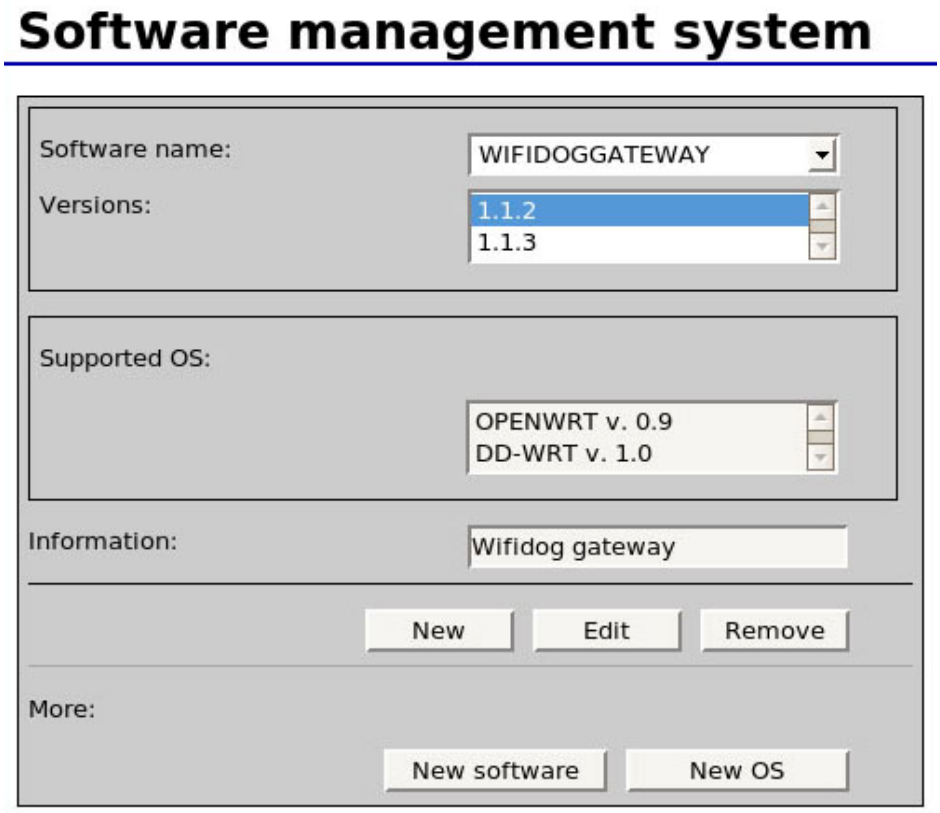

Figure 18 The main screen of SMS administration tool

On the public site of SMS one PHP script is present. The script serves the clients' requests and uses the HTTP POST array as an input to receive the information from the clients [55]. The information sent to the server by a client is as follows:

- Operating system on which the client runs,
- Version of operating system,
- Software name which is to be updated,
- The software version.

In Figure 19 specification and description language (SDL) diagram of the server is shown. The server gets one request and serves one response each time of interaction with the client.

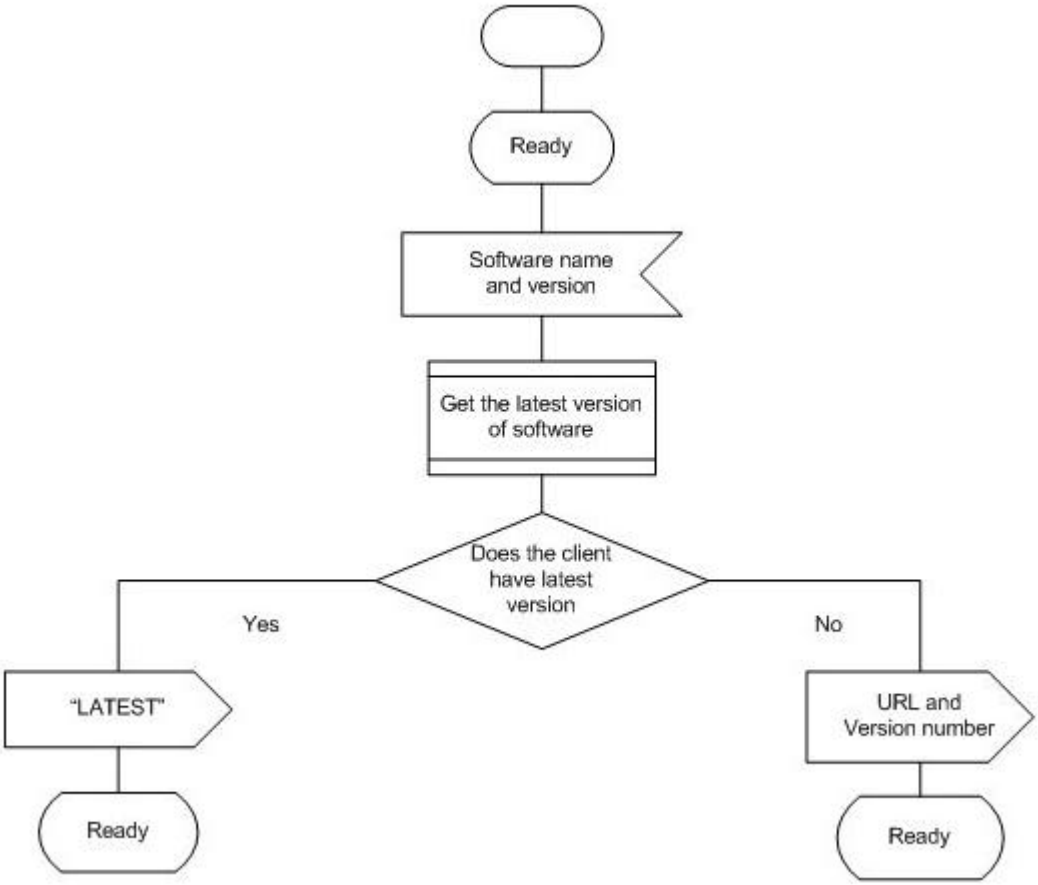

Figure 19 SDL diagram - the SMS server

First, the PHP script intercepts the client's request and receives the input parameters. Then the PHP script connects to the database and requests the latest available version of client software. If the version number received from the client and the version number obtained from the database are identical, then the script sends message "LATEST" to the client, this means there is no new updates for requested software. In case the newest version is available for update, the server sends to client a uniform resource locator (URL) containing the HTTP address, where the updates are available for the client. The server sends messages to the clients in the extensible markup language (XML) format as shown in Figure 20. The tags "path" and "version" are used for the URL and the version

number of an available software update respectively. In case of no updates for a client, the “LATEST” message is put in both “path” and “version” tags of the XML structure.

```
<XML>
   <PATH>
   …
   </PATH>
   <VERSION>
   …
   </VERSION>
</XML>
```

Figure 20 The XML structure sent by the server to the clients

5.5.3. The client of software management system

The SMS client performs checking for new versions of software as well as downloads new packages and installs them. Configuration file is used to adjust the settings of SMS client. The configuration file consists of the following information:

- Server’s address
- Server’s port number
- Path on the server
- Client’s operating system
- Operating system’s version
- Software package, which is intended to be updated
- Software package’s current installed version

All this information can be changed by user; therefore, the user has to be careful while editing the configuration file. The check-up for new software updates should be performed on a timely basis, *e.g.* weekly; therefore, the SMS client needs to be triggered by a scheduler service, *e.g. crontab*. The SDL diagram of the client is presented in Figure 21. On start up the SMS client reads the configuration information and based on that it sends HTTP POST request to the server. The request contains

information about client's operating system and software, which client wants to update. After sending the request to the server, the client waits for server's response. As soon as the client receives the response, it determines whether it is a URL or not. If no URL was received, the client stops its work, as no URL means that no updates for the requested software are available.

The SMS client initiates software update, if a URL to the new version of software was received. The *ipkg* package manager was used to download and install the software packages. The *ipkg* software manager is able to take a URL as a parameter and interpret it as a source of the package to be installed. The call of *ipkg* is done in a thread safe manner; it means that the client gracefully handles the errors of downloading new packages. Errors while downloading and installation the new software package result in not updating the target software. No log system is implemented in this version of SMS; therefore, it is difficult to identify errors during run of the system. The current version of SMS client depends on *ipkg* package management system.

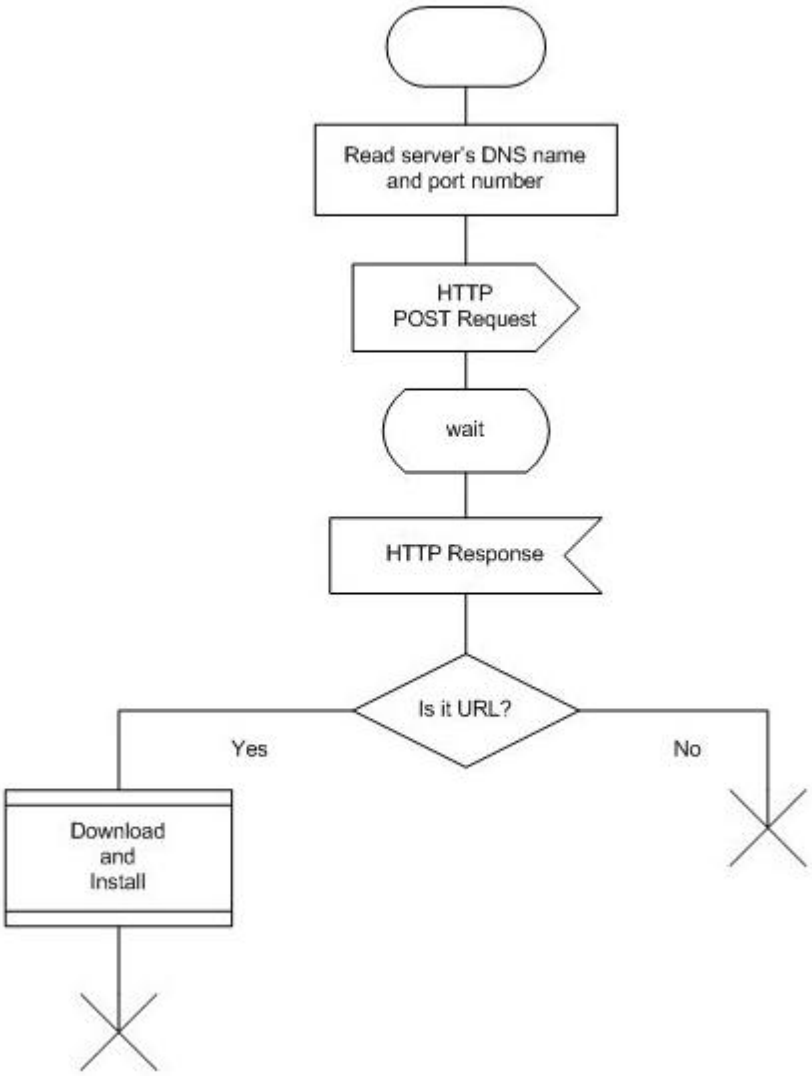

Figure 21 SDL diagram - the SMS client

If an error occurs during the software update process, it terminates and the software is left not updated. The next run of the SMS client will try to update the software again. After successful software update, the SMS client updates currently installed version of the software package in the configuration file.

6. Conclusions and recommendations

The authentication and authorization service implemented in this project shows an alternative way of how communities can be connected to each other and, therefore, the user may get a wider coverage area for mobility. The connection of two or more communities is done by use of well known LDAP protocol that allows hierarchical search among the whole network of LDAP servers.

The non-functional requirements such as flexibility, reliability and availability are achieved by use of, perhaps, the best open source software and then combining them in such a way that they fit well. The software extensions were religiously examined with an objective to achieve the desirable level of conformity to non-functional requirements.

All the functional requirements were implemented. The user management system, content delivery system, AP control and management, blocked user and APs, *etc*, all the required functionality is included in the solution. Honestly, most of them were already implemented in the Wifidog project; acknowledgements to its authors.

One of the main attributes of this project is the software management system, which was designed, implemented and integrated into the authentication and authorization service. The system is designed in such a way that allows using the software, which is already used by Wifidog system, at high extent. The current version of the software management system is beta-version with a lot of limitations and agreements on the usability. However, the system is able to provide the required service and, of course, can be developed further by anyone, who wishes to contribute to the project; it is published under GPL license as open source software.

Concerning the analysis of the whole system, it seems that the work is well done. The non-functional requirements to the project were pushing the author to search for methods to meet them and it was decided to use well known approaches, such as distributed services. The distributed installation of web servers and databases provides reliable, highly available and robustness services to the users. Making a copy of running services is highly recommended in order to bring services up in case of a breakdown.

This is network administrator's responsibility to keep an eye on backups of the system. As regards future improvements, in the first place, there is further development of the software management system, which is at its beta-version at the moment.

References

[1] Department of Health and Human Services, Terminology, http://www.cdc.gov/healthyplaces/terminology.htm , Last accessed date: 20.08.2008

[2] Wikipedia – the free encyclopedia, Virtual community, http://en.wikipedia.org/wiki/Virtual_community, Last accessed date: 20.08.2008

[3] Wireless community network blog, frequently asked questions, http://wcn.cnt.org/news/faq, Last accessed date: 20.08.2008

[4] Personal Telco project, Wireless communities, http://wiki.personaltelco.net/WirelessCommunities, Last accessed date: 07.08.2008

[5] StockholmOpen.net project, http://www.stockholmopen.net/, Last access date: 12.07.2008

[6] Mauro Brunato, Danilo Severina: "WilmaGate: a new open access gateway for hotspot management", Wireless Mobile Applications and Services on WLAN Hotspots. Proceedings of the 3rd ACM international workshop on Wireless mobile applications and services on WLAN hotspots, 2005, pp. 56-64.

[7] Casapula, G. Chizzoni, A. De Cindio, F.: "RCM: an open architecture to build a community", Community Networking, 1996. Proceedings of 3rd Workshop on Community Networking, May 1996, pp. 123-126.

[8] NoCat.net project – an implementation of captive portal, http://nocat.net, Last accessed date: 12.07.2008

[9] Wifidog – a captive portal suite project, http://dev.wifidog.org/, Last accessed date: 12.07.2008

[10] Rob Flickenger: "Building Wireless Community Networks. 2nd Edition", O'Reilly, June 2003, 182 pages. ISBN: 0596005024

[11] Rudi van Drunen, Jasper Koolhaas, Huub Schuurmans, Marten Vijn: "Building a Wireless Community Network in the Netherlands", Proceedings of USENIX 2003 Annual Technical Conference, 2003, pp. 219-230.

[12] Scrutiny of Acts and Regulations Committee, Victorian Electronic Democracy, Final Report, Glossary, http://www.parliament.vic.gov.au/SARC/E-Democracy/Final_Report/Glossary.htm, Last accessed day: 01.09.2008

[13] Min-You Wu: "Wi-Fi community area networks enable a connected community", Emerging Technologies: Frontiers of Mobile and Wireless Communication 2004. Proceedings of the IEEE 6th Circuits and Systems Symposium, Vol. 2, June 2004, pp. 671-674.

[14] Terry Schmidt, Anthony Townsend: "Why Wi-Fi wants to be free", Communications of ACM. Wireless networking security, Vol. 46, Issue 5, May 2003, pp. 47-52.

[15] Efstathiou, E. C.; Frangoudis, P. A.; Polyzos, G. C.: "Stimulating participation in wireless community networks", INFOCOM 2006, 25th IEEE International Conference on Computer Communications, April 2006, pp. 1-13.

[16] Roberto Battiti, Renato Lo Cigno, Mikalai Sabel, Fredrik Orava, Björn Pehrson: "Wireless LANs: from WarChalking to open access networks", Mobile Networks and Applications, 2005, pp. 275-287.

[17] Harri Hämäläinen, Jouni Ikonen, Jari Porras: "Applying wireless technology to teaching environment", 12th International Summer School on Telecommunications. Workshop on Applications of Wireless Communications 2003, August 2003, pp. 29-36.

[18] Web site of Linksys company, www.linksys.com, Last accessed date: 01.09.2008

[19] R. Droms, "RFC2131: Dynamic Host Configuration Protocol", IETF, 1997

[20] Internet Assigned Numbers Authority, http://www.iana.org/, Last accessed date: 08.07.2008

[21] Definition of NAT from whatis.com, http://searchnetworking.techtarget.com/sDefinition/0,,sid7_gci214107,00.html, Last accessed date: 12.07.2008

[22] K. Egevang, P. Francis, "RFC1631: The IP Network Address Translator (NAT)", IETF, 1994

[23] Hot Stuff Works, http://computer.howstuffworks.com/nat.htm, Last accessed date: 12.07.2008

[24] Personal Telco project, Captive portal definition, http://wiki.personaltelco.net/CaptivePortalDefinition, Last accessed date: 05.09.2008

[25] Kjell Jørgen Hole, "Indoor WLAN Design", http://www.kjhole.com/Standards/WiFi/WiFi-PDF/WLAN5alt.pdf, Last accessed date: 05.08.2008

[26] FirstSpot Wi-Fi hotspot management software home page, http://www.patronsoft.com/firstspot/, Last accessed date: 17.09.2008

[27] UseMyNet Wi-Fi hotspot system home page, http://usemynet.biz/ , Last accessed date: 17.09.2008

[28] HotSpot Studios captive portal home page, http://www.hotspotstudio.co.za/, Last accessed date: 17.09.2008

[29] Personal Telco project, portal software, http://wiki.personaltelco.net/PortalSoftware, Last accessed date: 17.09.2008

[30] The GNU General Public License, http://www.gnu.org/copyleft/gpl.html, Last accessed date: 17.09.2008

[31] Michael Lenczner, "Wireless portals with Wifidog", Linux Journal, 2005 Online version: http://www.linuxjournal.com/article/8352, Last accessed date: 12.07.2008

[32] Wifidog project, Wifidog login protocol, http://dev.wifidog.org/wiki/doc/developer/LoginProtocol, Last accessed date: 17.09.2008

[33] Wifidog project, Wifidog client protocol, http://dev.wifidog.org/wiki/doc/developer/ClientProtocol, Last accessed date: 17.09.2008

[34] Wifidog project, Wifidog Flow Diagram, http://dev.wifidog.org/wiki/doc/developer/FlowDiagram, Last accessed date: 12.07.2008

[35] Karl E. Wiegers: "Software Requirements", Microsoft Press, 1999, 341 pages. ISBN: 0735606315

[36] Sergey Trofimov, "Use Case", Computer-Aided Software Engineering Club, 2003, http://www.caseclub.ru/articles/use_case.html, Last accessed date: 12.07.2008

[37] M. Wahl, T. Howes, S. Kille, “Lightweight Directory Access Protocol (LDAP), version 3”, IETF, 1997

[38] Timothy A. Howes, Tim Howes, Mark C. Smith, Gordon S. Good: “Understanding and Deploying LDAP Direcoty Services”, Addison-Wesley, 2003, 936 pages. ISBN: 0672323168

[39] OpenWrt project, http://openwrt.org, Last accessed date: 12.07.2008

[40] Apache HTTP server project, http://httpd.apache.org/, Last accessed date: 12.07.2008

[41] Richard Stones, Neil Matthew: “Beginning Databases with PostgreSQL”, Wrox Press, 2001, 547 pages. ISBN: 1861005156

[42] PostgreSQL RDBMS project, http://www.postgresql.org/, Last accessed date: 12.07.2008

[43] OpenLDAP – community developed LDAP software, http://www.openldap.org, Last accessed date: 12.07.2008

[44] BIND Domain Name System service, http://www.bind9.net/, Last accessed date: 12.07.2008

[45] Quanshong Li, Bongki Moon: “Distributed cooperative Apache web server”, Proceedings of the 10th international conference on World Wide Web, Hong Kong, 2001, pp. 555-564.

[46] Fernando Pedone, Svend Frølund: “Pronto: High availability for standard off-the-shelf databases”, Journal of Parallel and Distributed Computing, Vol. 68, Issue 2, February 2008, pp. 150-164.

[47] Slony – enterprise-level replication system, http://www.slony.info/, Last accessed date: 12.07.2008

[48] Postgres-R – eager multi-master replication for Postgres, http://www.postgres-r.org/, Last accessed date: 12.07.2008

[49] Bettina Kemme, Gustavo Alonso: “Don’t be lazy, be consistent: Postgres R, A new way to implement Database Replication”, Proceedings of the 26th International Conference on Very Large Data Bases Conference, 2000, pp. 134-143

[50] Shuqing Wu, Bettina Kemme: “Postgres-R(SI): Combining Replica Control with Concurrency Control Based on Snapshot Isolation”, Proceedings of the 21st International Conference on Data Engineering, 2005, pp. 422-433.

[51] M. Cavallery, R. Prudentino, U. Pozzoli, G. Reni: "A set of tools for building PostgreSQL distributed databases in biomedical environment", Engineering in Medicine and Biology Society, 2000. Proceedings of the 22nd Annual International Conference of the IEEE, Vol. 1, July 2000, pp. 540-544.

[52] LDAP Sync Replication, www.openldap.org/doc/admin22/syncrepl.html, Last accessed date: 12.07.2008

[53] LDAP for Rocket Scientists, LDAP Referrals, http://www.zytrax.com/books/ldap/ch7/referrals.html, Last accessed date: 12.07.2008

[54] The home page of Dr. Peter Chen – the originator of the Entity-Relationship Model, http://bit.csc.lsu.edu/~chen/chen.html, Last accessed date: 13.10.2008

[55] PHP Tutorial, http://www.w3schools.com/PHP/DEfaULT.asP, Last accessed date: 12.07.2008

Appendices

Appendix I

The structured query language script that creates the database for the software management system

```
CREATE SEQUENCE os_id_seq;
CREATE TABLE oslist
(
    os_id       integer NOT NULL PRIMARY KEY DEFAULT
                nextval('os_id_seq'),
    os          varchar(50) NOT NULL,
    os_v        varchar(20) NOT NULL,
    os_desc     varchar(255),
    CONSTRAINT  oslist_uniq UNIQUE (os, os_v)
);

CREATE SEQUENCE sw_id_seq;
CREATE TABLE swlist
(
    sw_id       integer NOT NULL PRIMARY KEY DEFAULT
                nextval('sw_id_seq'),
    sw          varchar(50) NOT NULL,
    sw_desc     varchar(100),
    CONSTRAINT  swlist_uniq UNIQUE (sw)
);

CREATE SEQUENCE pack_id_seq;
CREATE TABLE swpack
(
    pack_id     integer NOT NULL PRIMARY KEY DEFAULT
                nextval('pack_id_seq'),
    sw_id       integer NOT NULL,
    sw_v        varchar(10) NOT NULL,
    url         varchar(255) NOT NULL,
    CONSTRAINT  swpack_sw_fk FOREIGN KEY (sw_id) REFERENCES "swlist"
                (sw_id) ON UPDATE CASCADE ON DELETE RESTRICT,
    CONSTRAINT  swpack_uniq UNIQUE (sw_id, sw_v)
```

```
);

CREATE TABLE ostosw
(
    os_id       integer NOT NULL,
    pack_id     integer NOT NULL,
    CONSTRAINT  ostosw_pk PRIMARY KEY(os_id, pack_id),
    CONSTRAINT  ostosw_pack_fk FOREIGN KEY (pack_id) REFERENCES
                "swpack" (pack_id) ON UPDATE CASCADE ON DELETE
                RESTRICT,
    CONSTRAINT  ostosw_os_fk FOREIGN KEY (os_id) REFERENCES "oslist"
                (os_id) ON UPDATE CASCADE ON DELETE RESTRICT,
    CONSTRAINT  ostosw_uniq UNIQUE (os_id, pack_id)
);
```

Components of SMS administration tool user interface.