IADIS International Journal on Computer Science and Information Systems
Vol. 5, No.2, pp. 37-55
ISSN: 1646-3692

TRUST-BASED HOTSPOT SELECTION

Xavier Titi. *University of Geneva.*
Xavier.Titi@unige.ch

Tewfiq El Maliki. *Ecole d'ingénieurs de Genève HES-SO*
tewfiq.elmaliki@hesge.ch

Jean-Marc Seigneur. *University of Geneva.*
Jean-Marc.Seigneur@unige.ch

ABSTRACT

Wi-Fi networks and its users are more and more numerous. Unfortunately, the risk of using one Wi-Fi or another varies because there is no means of selecting the most trustworthy hotspot. Mobile wireless connections are still risky and the performance varies greatly depending on the different nearby networks connections. For instance, new soft Access Point (AP) may easily spoof the identity of another AP under maintenance and thus compromise the privacy or the power consumption of the users' mobiles. Our solution proposes using trust management to mitigate malicious APs. We define a malicious AP in two ways: first the AP is defined such as an AP which cheats by capturing users' sensitive information while the user surfs the Web during their connection to this AP. Second, the AP is defined such as an AP which constrains user to use a lot of battery of the users' mobile by constraining to use encryption when the user does not need that. We have analyzed the network utilization and power consumption of mobile devices in the case of Geneva Hotspot AP locations. We have validated our proposed solution under resource constraints through simulation based on dynamic simulation tools named AnyLogic.
In second part of this paper we present a solution that solution allows the users to rate the networks they have used and to check that their rating corresponds to the true network quality they have experienced by measuring and certifying Quality of Service (QoS) evidence such as delay, jitter and packet loss. We use Quality of Service (QoS) evidence such as Delay, Jitter, Packet Loss to compute a trust value. This trust value will define the state of the AP (Good, Bad). We validate our solution by proving when this certification is irrefutable.

KEYWORDS

Security, Wi-Fi, Trust Management, QoS Evidence.

1. INTRODUCTION

The number of access point in the world increases significantly: they spread in many locations like airports, cafes, businesses and university campuses. This ongoing spreading, coupled with the inherent vulnerabilities of the deployed technologies, has provoked more breach of security. In addition to typical network threats, wireless networks present several challenges and attacks. This is due to the wide open air nature of the channel allowing more attacks, bandwidth limitations and constant topology changes because of node mobility. Furthermore, a lot of security breaches have been registered, such as Service Set Identifier (SSID) spoofing using soft AP. To steal credit card numbers and other personal information, thieves are using a soft AP to masquerade as legitimate Internet APs. For instance, it has been reported [1] that fake Wi-Fi networks have been set up in airports in order to capture users sensitive information as they surf the Web during their connection to these networks. In this case the basic security mechanisms may not be sufficient. To tackle the above-mentioned constraint problems, we need new and easily deployable security-adapted mechanisms respecting overall performances. It is important to know whether the Wi-Fi networks within range are trustworthy or not. For instance, we can take a young man who wants to send an email to his wife to prevent her that his plane will be late or a businessman who must make a banking transaction using Internet. In these two cases it is important to notice that the level of security required is not the same. According to the opinions of users who have used some APs, it is important to inform future users whether they can rely on these APs or not. In some locations, it is not rare to have more than five potential Wi-Fi networks to be connected.

We propose our solution to evaluate the level of security for the AP. Our solution can help user to detect which AP is more trustworthy by taking into account the security and energy consumption. The first part of this paper is organized as follows: Section II presents the problem statement and related work. In Section III, we present our solution and its assumptions. Section IV presents how we have implemented our solution to validate it. The results are presented in Section V. Section VI concludes the first part of the paper. The second part of this paper presents the solution that checks the trustworthiness of the rating of the user. We propose to use QoS evidence measurements, when the user is using the hotspot. Our solution will be used in order to control the ratings of the users meaning if the user tries to cheat by giving a bad rating to a hotspot whereas the hotspot is considered as a “Good” AP our solution will reject this bad rating. The second part of this paper is organized as follows: Section VII presents our system model. Section VIII presents our solution and Section IX concludes the second part of the paper.

2. PROBLEM STATEMENT AND RELATED WORK

2.1 Problem Statement

The number of Wi-Fi networks is growing very quickly because of the increasing number of shared personal and commercial Wi-Fi access. Most deployments of public Wi-Fi networks solutions are not based on security and optimized energy consumption. This is very important for network operators and users to find a way to improve the network utilization and energy consumption.

For instance a user who wants to connect to an AP and does not have a lot of battery life with his/her mobile phone will not be able to use the network during a long time. We have also another case which is when a user wants to do an important banking transfer and needs to use an AP which he/she can trust.

The goal of this work is to present an easy deployable solution that:

- Evaluates the level of trust on the security of the AP.
- Minimizes the energy consumption.
- Encourages the owner of the AP to act correctly.

2.2 Related Work

Salem et al. [5] propose a reputation system to enable the user to choose the best hotspot and discourage the wireless Internet service providers (WISPs) from providing a bad quality of service to the mobile nodes. In this paper, they consider a mobile node MN that is affiliated with a home network H and that wants to connect to the Internet via a hotspot managed by a wireless Internet service provider. The behavior of each WISP in their model is characterized by what they call a reputation record. This record represents an evaluation of the reputation of the WISP and is generated and signed by a trusted central authority. The reputation mechanism is maintained by the same trusted central authority. When a WISP first enters the network, the trusted central authority provides it with an initial reputation record that can afterwards increase (i.e., better reputation) or decrease, depending on the behavior of the WISPs. If MN has two neighboring WISPs that propose equivalents offer, i.e., same QoS and price MN will choose to connect to the access point managed by the WISP that has the best reputation record. They use a micropayment scheme to make sure that Mobile Nodes will pay for the service they received. Our solution in some point is similar but different as well because our solution can work on free or paid AP. Our solution takes also into account the energy consumption which is an important criterion when we talk about mobile device. We focus our work to detect the APs which want to cheat by stealing sensitive information or stress the user to consume a lot of battery life.

In [6], the selection algorithms focus on AP signal strength as an important metric. They conducted an extensive field study of three neighborhoods in Chicago, which showed that choosing an AP based on signal strength misses significant opportunities for Internet connectivity. They presented the design and implementation of Virgil, an automatic AP discovery and selection system. Virgil quickly associates to each AP found during a scan, and runs a battery of tests designed to discover the AP's suitability for use by estimating the bandwidth and round-trip-time to a set of reference servers. Virgil also probes for blocked or redirected ports, to guide selection in favor of preserving application services currently in use. They evaluated Virgil in five different neighborhoods across three different cities. Their results show Virgil finds a usable connection from 22% to 100% more often than simply selecting based on signal strength alone. Virgil improves both performance and accuracy for neighborhoods the user commonly travels, by caching AP test results. Their work focuses on estimation of bandwidth and round-trip-time for assessing the AP. Our solution focuses on two other aspect security and energy consumption in order to assess the AP.

Ormond et al. [7] further examine network selection decision in wireless heterogeneous networks based on a user-centric approach, which they say allows a user to choose a network which meets their best requirements. Their network selection algorithm predicts the data rate

on each interface available to the mobile node and decides based on the predictions. Their approach is very interesting because they focus on the user requirements or preferences although they do not consider security and energy consumption as we do.

3. OUR SOLUTION AND ITS ASSUMPTIONS

We consider a mobile phone (MP) that integrates the Wi-Fi technology. The user wants to connect his MP to the Internet via a Wi-Fi AP. There are different APs which are available in proximity. In our solution, we use a trust mechanism to discourage the APs to cheat: the trust management is used during the selection process of the best AP for the users among the available APs. Thanks to our trust-based mechanism, the APs are encouraged to behave correctly: they do not steal sensitive data of connected users or force users to consume a lot of energy when connected for example by forcing strong encryption even if not needed.

3.1 Attacker Model Assumptions

The trust value evidence is maintained by a Trusted Authority (TA). We assume that:

- TA is trusted by the other parties;
- The user is not able to create many identities in order to cheat, for example, via Sybil attacks[2].We assume that the MP has a SIM card which unambiguously authenticates the identity of the user with a provider;
- There is a mobile Web site where we have a social network and the users can specify who their friends are;
- Communication between users' mobile clients can be done by using the Web site for mobile.
- The connection to the Web site for mobile will be done by GPRS or GSM.
- The AP has an unlimited amount of energy and a uniform transmission range.

3.2 Trust Model

The behavior of each AP in our model is characterized by what we call a trust value. This trust value represents the trustworthiness level of the AP about the security and energy consuming and this thrust value is signed by a trusted authority denoted TA. The trust value of an AP will be in [0…1].When an AP first enters the network; TA provides it with an initial trust value that can afterwards increase or decrease, depending on the behavior of the APs. In our simulation we have found the best value for this initialization (*see Section 5.1*). The users have the possibility to ask to his friend some recommendations about an AP**.** The user will do that by using a mobile Web site called Mobile Web site (MWs) where the user will store all his information about his friends and the AP used by him. The connection on this Web site will be done by using a UMTS, GPRS or 3G connection. The recommendations are useful when the users have no information about an AP. We will explain these recommendations in *Section 4***.** After using the AP, the user will rate the AP with three possible values: Positive, Negative and Neutral. We assume that in MP we implemented two functions: one for detecting a security breach during the session and the other one detecting big energy consumption.

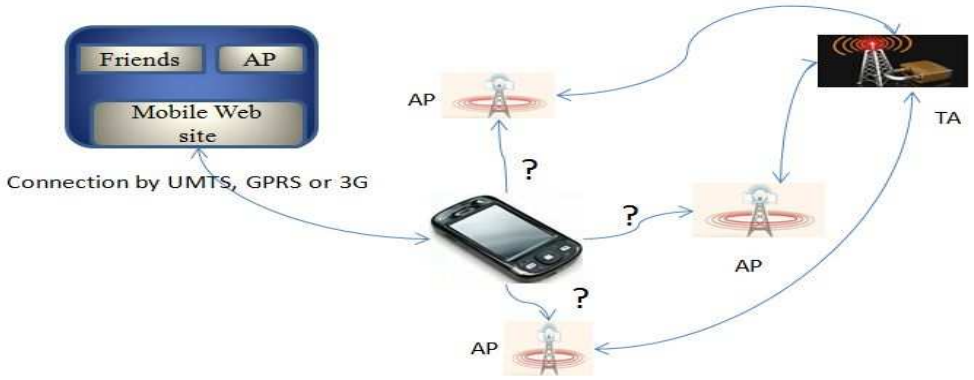

Figure 1. Trust model overview

Our solution proposes a trust management to help users to choose the most trustworthy AP. Two thresholds are used: trust_value_min and trust_value_max.

We have defined three cases:

- **First case**: the MP will connect to an AP by encrypting his connection if the trust value of this AP is between two values which are explained later in this paper
- **Second case**: the MP will not connect to an AP if the trust value is inferior to the trust_value_min.
- **Third case**: the MP will connect to an AP without encrypting his data when the trust value of the AP will be superior to trust_value_max

This is a part of our implementation code about these three cases.

```
{ if trust_value_min <trust_value_AP< trust_value_max
  then encrypted connection
if trust_value_min >trust_value_AP
        then no connection
if trust_value_max <trust_value_AP
  then normal connection
end.
}
```

To apply this algorithm, our solution needs trust functions in order to compute the trust value. We have implemented four functions. These functions are implemented on different part of our trust model such as the MP, AP and Mobile Web site (MWs).

Function 1

It is responsible to compute the trust value of an AP by taking into account the evaluation of the friends of the user. We have defined some different kind of level of friend. We consider the user U, n is the number of friends of user U, x is the trust value of AP store in the MP of user and y the value of the sum of recommendation coming from friends of user U and N is the AP. This function is implemented on the MP. Results of this function will be applied to the two thresholds trust_value_min and trust_value_max in order to know in which case we are.

$$TrustUpdate\ (U \rightarrow N) = \begin{cases} x\ if\ n = 0 \\ \dfrac{n(x+y)+x}{2*(n+1)} \end{cases}$$

Function 2

It is responsible to compute the recommendation coming from the friends of a user friend. We consider Ui all friends of user U and Friendshipfactor a factor of friendship explained in function 3, n it is the total number of friend of user U and N is the AP. This function is implemented on the mobile Web site. When the user will ask this recommendation the Web site will compute this value and send it to the user. We have chosen to put this function on the Web site in order to lighten the MP and the MP user will receive the value of recommendation in order to compute the trust value of AP.

$$Trust\,(U \rightarrow N) = \sum_{i}^{n} \frac{Friendshipfactor * Trust\,(Ui \rightarrow N)}{n}$$

Function 3

It is responsible to compute the Friendshipfactor of the user. We consider nblink as the level of friendship. The nblink must be inferior or equal to 6 by taking into account the theory of Small World [3]. A and B are users and there are friend. This function is used to help the computation of the recommendation by calculating the Friendshipfactor. This function is implemented on the mobile Web site.

$$Friendshipfactor\ A \rightarrow B = \begin{cases} 1\ if\ A\ is\ a\ friend\ of\ B \\ \dfrac{1+\frac{nblink}{2}}{nblink+\frac{1}{2}} \quad else \end{cases}$$

Function 4

It is responsible to compute the trust value of AP. We consider Pos like a good feedback, Neg is bad feedback and Neu is Neutral feedback. The trust value is certified by TA and stored in AP. Like we said we assume that this value cannot be changed by the user. This function is implemented on the AP. When the user finishes using the AP, he will choose between three different feedbacks (Pos, Neg, Neu). This function will be implemented also in the MP in order to have his own experience with the AP.

$$Trust_{Network\,(AP)} = \frac{\sum_{i=0}^{n} Pos + \frac{1}{2}\sum_{i=0}^{p} Neu}{\sum_{i=0}^{n} Pos + \sum_{i=0}^{m} Neg + \sum_{i=0}^{p} Neu}$$

4. IMPLEMENTATION AND VALIDATION METHODOLOGY

We have implemented our functions and validated our solution with AnyLogic, which is a simulation tool that supports all the most common simulation methodologies: System Dynamics, Process-centric (a.k.a. Discrete Event), and Agent Based modeling. It is based on Real-time UML and Java object-oriented language.

4.1 Model Set-up

The basic element of an Agent Based model is the agent itself. By using an Agent Based model we have created a new class than behaves as an AP. Each device is associated to a given agent matching with his location. As the device is not static, we have modelised his mobility using X and Y random variables.

The movement and the status of our agents are controlled by a state-chart which represents the exact behavior of the device [Fig.2].

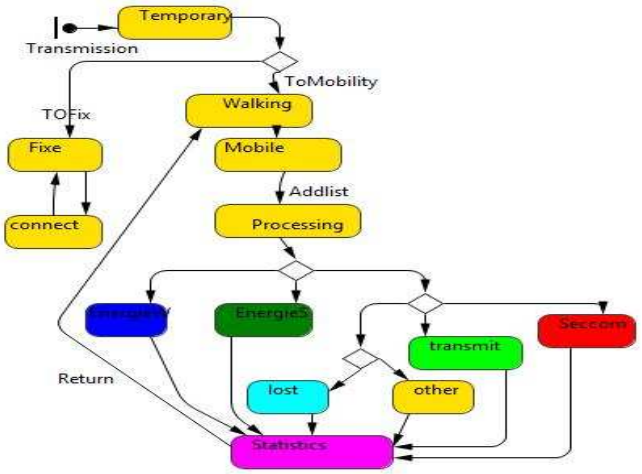

Figure 2. State-chart: Fix APs and mobile users

In Figure 2, each agent starts in a "Temporary" state in the state-chart. A part of our agents were APs, so they switched to the state "Fix". These agents were placed in the map according to their GPS positions taken from Swiss hotspots reference[1]. The rest of our agent were considered as mobile nodes and each of them followed a random mobility model and are switched to the state Walking. They are added to a list of an AP whenever they are within the coverage of this AP. After processing state which evaluates the trust function and choose the best value among all near-by APs.

We used agents that were in one of the 6 states:

- Transmit : a normal connection is done without encryption or problem (green)
- Energy Wasting : a encrypted connection is established although the encryption is not necessary (blue)
- Energy Saving : A non-encrypted connection is established even if the AP is asking for encryption (dark green)
- Security Compromising : The AP is compromising the security because the connection is not encrypted (red)
- Utilization Lost : A connection is not established because the Trust value is under min threshold although there is no danger (magenta)

[1] *http://www.swiss-hotspots.ch/*

- Other : there is no AP within a fixed range (yellow)

Each agent can be in those six states. After that the agent will go on the state Statistics to accomplish all necessary measures before to go back in his initial state.

Setting up our security model using Table 1, we can take advantage of state chart by monitoring the behavior of agents. The behavior of AP is summarized in the Table 1.

Table 1. Access Point Behaviours

Behaviour	**Definition**
Normal Transmit	Don't secure the connexion; AP has a normal Behaviour
Energy wasting	Establish a secure connection to misuse your energy
Security compromising	Don't secure access to compromise your security

4.2 Validation Methodology

In our experiments, we validated our proposed solution and analyzed the extended performance under a range of various mobility scenarios. All nodes are moving over rectangular 8.69 km x 6.08 km topography, and operating over one day of simulation time. In our simulations, we considered that the APs were those taken from Geneva hotspots and the nodes are the mobile users with their MP. The coverage of APs is limited to 100 meters. Each mobile device was configured to have a maximum communication range equal to 100 meters. We deployed the APs in an incremental mode, from AP_1 to AP_n, in the exact position taken from the true GPS position. Thus we estimated the impact of our solution for an existing network depending on the security and energy consumption aspect.

The movement pattern of mobile clients was totally randomized, in order to comply with a real hotspot application. To achieve this, we used the Random WayPoint (RWP) mobility model [4] with pause time equal to the time of network access and data transfer. We carried out our experiments considering thirty cases. In each studied case, we ran our simulations with different conditions. Our performance evaluation was the result of 1000x30 different simulations.

4.3 Scenario

In the first scenario, we fixed the percentage of mobile and APs and the percentage of each category of AP:

- 60% for mobile users
- 40% are APs :
  - 10% of APs have Energy wasting behaviour
  - 10% of APs have Security compromising behavior
  - 80% of APs have Transmit behavior

In the second scenario, we fixed the percentages of mobile and APs and the percentages of each category of AP:

- 60% for mobile users
- 40% are APs :
  - 25% of APs have Energy wasting behaviour
  - 25% of APs have security compromising behavior

- 50% of APs have Transmit behavior

The two scenarios were run using a wireless network composed of 600 users for 300 APs in case of high density and 400 users for 40 APs in case of low density.

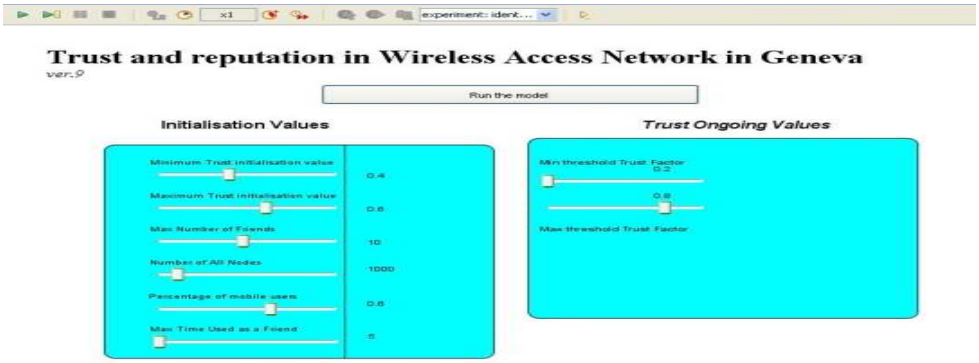

Figure 3. The launching interface.

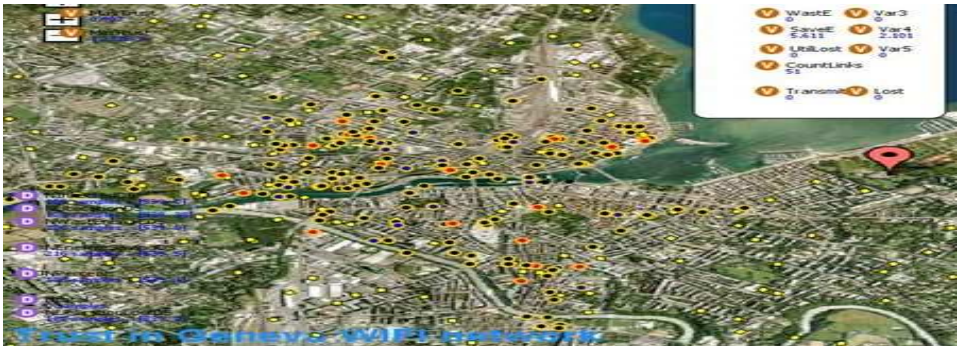

Figure 4. Animation interface.

Figure 3 shows the launching interface and Figure 4 shows the animation interface. The circles are APs and the square shapes are users whose their colors correspond to their behavior. In the background of the animation, we put the Hotspots APs according to their GPS location on the map of Geneva.

5. RESULTS ANALYSIS AND SYNTHESIS

5.1 Results Analysis

In this section we present our results of the validation of our solution by using the simulation. Firstly we carry out to find the best trust value to initialize the AP and to find the best value for the two thresholds trust_value_min and trust_value_max.

Table 2. Initialization and effect of our variables

Threshold min — max --> Initialisation of AP value l v	0.1–0.3	0.2 –0.4	0.3-0.5	0.4–0.6	0.5–0.7	0.6 -0.8
0.1	No	No	No	No	No	No
0.2	No	No	No	No	No	No
0.3	No	No	No	No	No	No
0.4	No	No	No	No	No	No
0.5	No	No	No	No	No	No
0.6	No	No	No	No	No	No
0.7	No	No	No	No	No	No
0.8	No	No	Yes	No	No	No
0.9	No	No	Yes	No	No	No
1	No	No	Yes	No	No	No

The results of Table 2 show that the best value for our thresholds is 0.3 for trust_value_min and 0.5 for trust_value_max because in this case we are able to detect the malicious AP after a number of connections to this AP. The best initial trust values for AP is between [0.8…1].

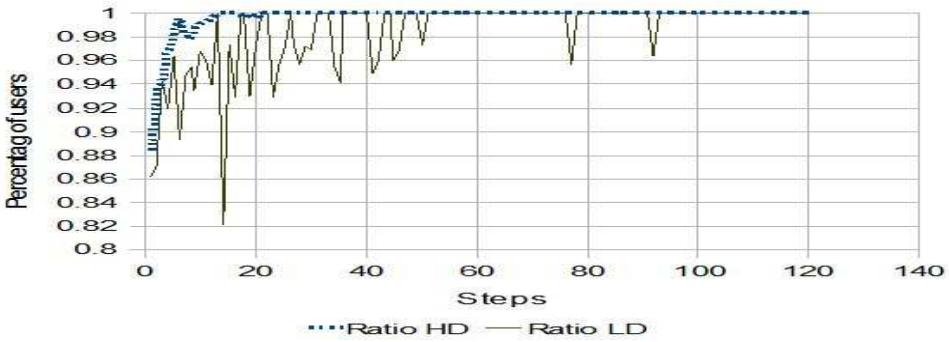

Figure 4. Comparison of low and high density APs in the 1st scenario

The Figure 4 shows that with our solution and in the case of high AP density we can detect the malicious AP in more or less 20 connections. However in case of low density we can detect malicious AP with our solution in more or less 90 connections. This is due to the fact that there are many isolated APs which are evaluated without any comparison with any other APs trust values. In other words, in the first case the users can easily evaluate many trust values of AP and choose the best one than in the second case. Therefore, our solution is well-adapted for the rapid evolution of the number of APs mainly in cities in the near future.

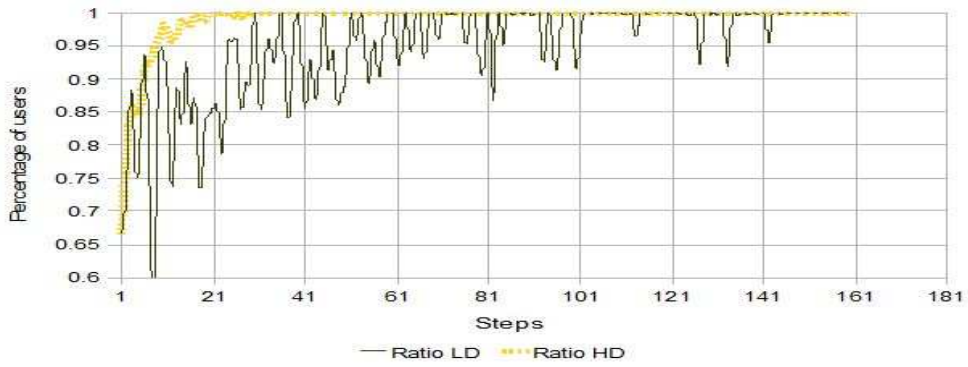

Figure 5. Comparison of low and high density in the 2nd scenario.

Figure 5 illustrates the evolution of the percentage of normal users when using the second scenario (25% Security compromising, 25% Energy wasting) for low and high density. In case of high density the number or connection for detecting malicious AP are more or less around 21 connections and in the case of low density it is more or less 142 connections. We can deduct that between the two scenarios there is not a big difference in the case of high density because the user can detect the malicious AP with approximately the same number of connections. However in the case of low density there is a big difference between the two scenarios because of the number of connections for scenario 2 for instance is widely more numerous than that of scenario 1.

5.2 Synthesis

Our solution provide a way to find the number of connections done by different user for detecting malicious AP by taking into account the fact that there is no attackers.

Table 3. Statistics of average number of connections in 2nd scenario

	High density	Low density	Ratio LD/HD
Average of connections to detect malicious AP	20.81	96.87	4.66
Standard deviation	4.13	23.68	5.74
Connections for 99% confidence level	30.41	151.95	5

In Table 3, we found the average values of connection to detect the malicious AP in the case of high density and low density for 30 samples. Then, we calculate the ratio between Low density and High density. We also calculate the standard deviation (SD) and the confidence interval that describe how reliable our results are. Indeed, for 99% of time the estimated number of connections to convergence will be less or equal to the average plus 2.326 of standard deviation (A + 2.326 * SD). For high density, the number of connection for detecting malicious AP is less than for low density; the ratio is around 5.

Table 4. Statistics of average number of connections in 2nd scenario

	High density	Low density	Ratio LD/HD
Average of connections to detect malicious AP	27.78	154.00	5.54
Standard deviation	5.31	35.34	6.66
Connections for 99% confidence level	40.12	240.86	6

In Table 4, we depicted the average values for high density and low density and their ratio in the case of 25% of Security compromising APs and 25% of Energy wasting APs. We also calculate the standard deviation and the confidence interval that describe how reliable results are. The ratio is between low and high density. The detection of malicious AP in the case of low density is from 5 to 6 times longer than for high density depending on the 1st scenario or the 2nd scenario.

6. CONCLUSION

The great increase of access points in the world offers the opportunity for users to connect to the Internet almost anywhere anytime. Unfortunately, some access points remain untrustworthy. Our proposed solution provides to the users the possibility to find the most trustworthy hotspot and to give their point of view regarding the quality they have experienced using it. This paper presents the simulation of our solution applied to Geneva Hotspot network APs based on random mobility and AnyLogic simulation tools. The results of our solution shows that with our given scenario two scenarios and by taking account the density aspect our solution is able to provide to the user the way to choose a good AP (AP not malicious). For instance for the two scenarios and with a high AP density, around 20 connections is needed to detect a malicious AP. But when we are in the low AP density we need a lot of connections to detect a malicious AP. For instance for the scenario 1 we need around 90 connections and for the scenario 2 we need around 142 connections. To conclude by that we see that our solution is appropriate in the case of high AP density.

Our future work is to improve our solution in order to decrease the number of connections for mitigating the malicious AP.

In this second part of the paper we present a solution that will check the trustworthiness of the user rating.

There are an increasing number of websites offering an assessment of the Wi-Fi networks available in a given location [8, 9, 10, 11, 12]. This is due mainly to the great success of commercial and shared Wi-Fi access. For the user, it is important to know if the AP is trustworthy. There is a lot of proposed solution to solve this problem by giving the possibility to the user to evaluate himself the AP. This evaluation must be available to the others users in order to help them to choose the most trustworthiness AP. However, there is no means to detect when the user tries to cheat by giving an unsuitable evaluation. Our solution provides an application that will give the possibility to verify the trustworthiness of the evaluation of

the user. With our solution we will able to verify the trustworthiness of the evaluation of the hotspot that represents his state.

7. SYSTEM MODEL

We are considering using a mobile phone (MP) that integrates with the Wi-Fi technology and this MP will have our Android application running in background. If the user wants to connect his MP to the Internet via a hotspot and there are different hotspots available, then the MP will have a SIM card which links the identity of the user with a provider. In this case, we assume that the user is not able to create many identities in order to cheat, for example, Sybil attacks [2]. Our solution provides an Android application that runs in the background and collects the QoS evidence (packet loss, jitter, delay) and also the kind of application used by the user. This information is encrypted and signed in order to certify the information sent to our servers. We use more than one server because in time of high peak one server can be overloaded so we provide other servers to prevent this from happening. For instance when we have one server overloaded, we share the users between our different servers. In the case of overloaded servers, we will have bad values of QoS evidence (delay, jitter). Thus, we will have bad values of QoS Evidence not because of the QoS of the hotspot but due to a overloading of the server. This is why our solution provides the users several servers in order to prevent the case of overloaded servers. Our solution does not require changing anything from the existing infrastructure, no update of hotspot firmware, of MP OS. The user will have the possibility to rate the hotspot after using it. It will have two choices: “Good” or “Bad”. This rating will be sent to our servers. The server with the QoS evidence and the information of the kind of application used by the user will compute a value of trust to estimate the trustworthiness of the user rating. The Figure 6 is an overview of our system.

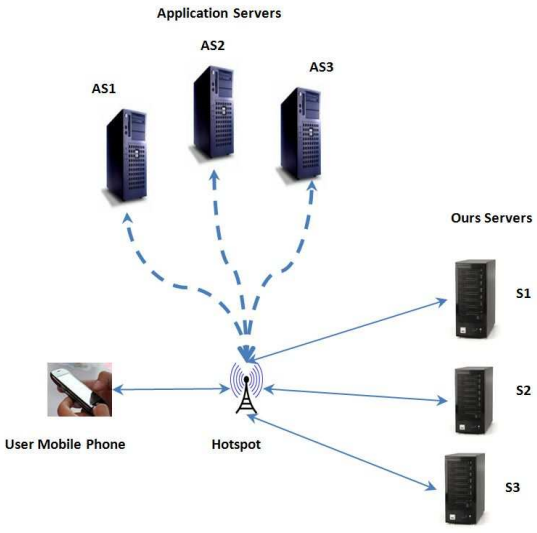

Figure 6. System Model

8. SOLUTION

We present our solution to verify whether the hotspot user rating is trustworthy. The hotspot user rating can be “Good” or “Bad”. Our solution provides an Android application in order to verify the trustworthiness of this hotspot user rating and this application needs a public key (PuK) / private key (PrK). This is why we propose to the user that after downloading our application they need to generate a public key (PuK) / private key (PrK) pair in order to sign the message. We have chosen the Android platform because this platform is an open source and it provides a lot of functionalities. On the Android platform we can identify the application that is running on the mobile device without any problem. The application will run in the background and will collect QoS evidence in order to assess the quality of the network. This assessment will be based on the requirements of the used applications as video, voice and messaging. For instance when the user uses a Voice over IP application, the recommendation G.114 of ITU [15] defines as an acceptable value of delay. In according of ITU [15] this acceptable value must be between 0 and 150 ms in order for the user to be satisfied with his conversation. G.114 is an ITU recommendation that addresses acceptable delays for voice applications. We distinguish different cases depending on the applications used; this is because different applications have different requirements. We will have a database which will contain the requirements of the different applications. This database will take place in our servers. We will store this information on the server in order to prevent overloading the device with a lot of information. Our solution calculates a trust value based on the QoS evidence values that are comparable to those required for the proper functioning of the application that has been used by the user. We have chosen these parameters as evidence of QoS:

- Packet loss [13] occurs when one or more packets of data traveling across a computer network fail to reach their destination.
- Jitter [14]: Packets from the source will reach the destination with different delays. A packets’ delay varies with its position in the queues of the routers along the path between source and destination and this position can vary unpredictably. This variation in delay is known as jitter and can seriously affect the quality of streaming audio and/or video.
- Delay [14]: It might take a long time for a packet to reach its destination, because it gets held up in long queues, or takes a less direct route to avoid congestion. In some cases, excessive delay can render an application such as VoIP or online gaming unusable.

We have chosen these parameters because with them we can know whether or not the use of an application such as video, voice or messaging went well.

Thus the application collects the values of these parameters and we use digital signature to certify who is sending this information to the server. Our application will also indicate the type of application used and the telephone number of the user to server. We do not send the phone number but the hash of this number by using the SHA-256 protocol. The server will use the QoS evidence coming from our application to compute a trust value that shows whether the hotspot user rating is really trustworthy. In accordance with the trust value the server will know whether the user tries to cheat or not. Our solution provides a way to confirm the hotspot user rating such as when the hotspot user rating is “Good”, our solution verifies whether or not it is true.

We assume that there is no congestion problem on the hotspot due to the presence of a lot of co-located users.

8.1 Certifying QoS Evidence

To clarify our solution, we list the different situations in which the user may be placed. We present and explain what our solution proposes according to different cases. To summarize the different cases, we define table 5.

Table 5. Summarize of the Possible Situations

Hotspot User Rating	Server Evaluation	Status
Good	Good	User has not cheated
Good	Bad	User has cheated
Bad	Good	User has cheated
Bad	Bad	Unsure(Future Work)

In the first line, the table shows that the hotspot user rating is “Good” and the server computes a trust value that corresponds to the quality of the network as “Good” also, this is provided by the hotspot, so we can now validate the hotspot user rating. There is no way for the user to cheat here. In this case, the server finds that the hotspot is “Good” and the server knows, because it computes a trust value in accordance of the QoS evidence that is certified so the user cannot cheat because the QoS evidence is certified when the server finds that the hotspot is “Good”.

In the second line, this case represents the cheating of the user. For instance if the user tries to modify the source code of our application so that the server computes a bad value, it would make no sense for this user to try to evaluate this hotspot by giving a good evaluation, because his evaluation will be rejected. So the only reasonable reason that can provide this case will be when the user tries to cheat by giving a “Good” rating where as the server finds that the hotspot is “Bad” in accordance with QoS evidence.

In the third line, we assume that there is no problem with the application server because we compute the QoS evidence by exchanging message between MP of the user and our server through the hotspot. Thus we can have the QoS evidence of the hotspot; we assume that application servers work well and do not have any problem with overloaded work. By taking our assumptions into account, the case of the third line is considered as the user cheating and the evaluation of the user will be rejected.

The last line is more complicated because the user can cheat by changing the source code to make the hotspot “Bad” because our solution cannot certify the QoS evidence that corresponds to a “Bad” quality of network provided by the hotspot. For the last line, it will be part of our future work. We do not yet have any solution to be sure that the user has not cheated in order to make the hotspot “Bad” for other users or for his personal purpose. For instance the user can change the source code of our application which is running on his device in order to send false value of QoS evidence. The user can only send false QoS evidence by sending bad value. It is not possible for him to send Good value in order to make a “Bad” hotspot as a “Good” hotspot. We will explain how we get the QoS evidence with our Android application and we will also explain how this information is certified.

Our solution takes into account the synchronization problem. Clock synchronization is a problem from computer science and engineering which deals with the idea that internal clocks of several electronic devices may differ. For instance the MP of a user will not have the same internal clocks as our servers.

With our solution, we can detect when the user tries to cheat with his rating so our solution cannot replace the user rating because we need these ratings in order to detect which user tries to cheat. This information will be very useful in the case where *Hotspot User Rating* will be set to "Bad" and *Server Evaluation* will be set as "Bad". Additionally, we will not launch our application at all times because if all users launch their applications at the same time then our servers will become overloaded. The user is at the center of our system so their feedback is very important, even if the user tries to cheat. This information about cheating will be very useful to our future work which uses the reputation concept.

8.2 Delay and Jitter

The jitter and delay are intrinsically linked. By verifying the value of the delay, we verify also the value of jitter. This is because the jitter is the variation of the value of delay. Our application that runs in background will collect every second of the measurement of the delay; therefore we will also have the jitter. To calculate these parameters we use the exchange of signed and encrypted messages between the user and the server. To sign these messages we use a key pair (public / private). The user will have the key pair PuK / PrK that they generate themselves. At the launch of our application in the background, the user and the server exchange their public key. The details of this procedure will follow. Firstly, the user starts by sending a signed request (Req1) at the time (T0) in accordance with his clock. The server receives this query Req1 at time T1 according to its clock. The two clocks are not synchronized to departure. The server returns a query Req2 to the user in order to receive the response of the user. The server receives the response of the user at time T2 according to its clocks. The Figure 1 shows how it works. When the user sends the response req3 and the server receives at time T2, the server will know that the delay will be equal to (T2-T1) / 2. We will calculate the delay several times in order to compute the jitter, for instance we will have the jitter as (((T4-T3) / 2) – ((T2-T1) / 2)) but at the beginning the user and the server have to exchange their public key (PuK).

To summarize that we will have:

The message sent is defined: ID of the sender and the type of the message. We have four types of messages:

- Start: This kind of message comes from the user in order to inform the server that it represents the beginning of the procedure
- Ask Confirmation: The server asks to the user to reply when he or she receives this request
- Confirmation: The user replies to the request coming from the server.
- Key: the user uses this message in order to send their public key.

User sends his public key (PuK) to the server.
User->Server sends PuKUser [UserID, Key]

Server sends his public key (PuK) to the user.

Server->User sends PuKServer [ServerID, Key]

User encrypts the message M in order to sign M with his PrK and after the user encrypts this signed message with the PuK of the server.

User->Server sends Req1[UserID,Start]

(PrK of the user /PuK of the server)

The server receives the message and uses its PrK to decrypt the message and then after it will use the PuK of the user to certify the sender.

Server->User Replies Req2[ServerID,Ask]

(PrK of the server /PuK of the user)

User encrypts the message M in order to sign M with his PrK and after the user encrypts this message with the PuK of the server.

User->Server Replies Req3[UserID, Confirmation]

(PrK of the user /PuK of the server)

When the server receives the Req1, it sends a request directly to the user and the user has to reply directly when he or she receives this request. The user can only cheat if he wants to delay the response but, in no case can the user deny receiving the request of the server earlier than the real time of the reception of this request. That means when the server receives the response at the time T2, He knows that (T2-T1) / 2 is equal to the delay. Thus when the hotspot user rating gives "Good" and the server gives "Good" as we know it cannot be cheated because the user can only cheat by pretending that the network is bad. The biggest advantage of this method is that we do not need to synchronize the mobile devices with our servers. We do not need to synchronize the mobile devices and our servers before hand in order to compute the delay. The Figure 7 shows how we compute the Delay.

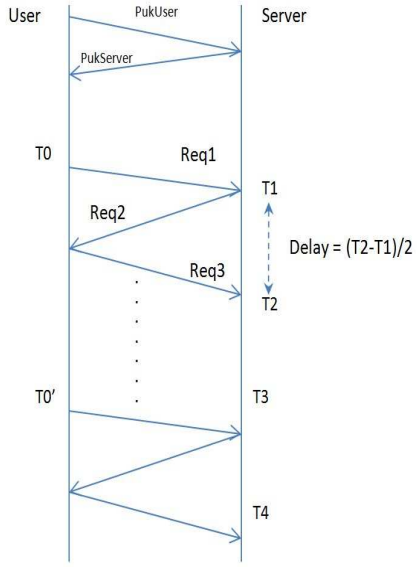

Figure 7. Computing of Delay

8.3 Packet Loss

To measure the rate of Packet Loss we use the fragmentation protocol. IP Fragmentation protocol is a process that is divided into datagram smaller (fragments) that can be transmitted so that we can propose to send our servers a packet larger than the MTU (Maximum Transfer Unit) of the hotspot. The hotspot will be forced to fragment this Packet in numerous fragments. Therefore, the server will be able to know which fragments are missing because the fragmentation protocol indicates the position of each fragment in order to help the receiver of this Larger Packet to reassemble all fragments. Thus the value of the number of missing fragments allows us to know the rate of Packet Loss. This technique has a great advantage because it does not depend on the user but on the hotspot. Therefore, even if the user wants to cheat by modifying the source code he will not be able to change this value.

Rules

We propose to implement a rule which will help the server to interpret the results obtained of QoS evidence:

- If 0<delay<Delay_maximum then X=1 else X=-1
- If 0<jitter< Jitter_maximum then Y=1 else Y=-1
- If 0<packet loss< Packet_Loss_maximum then Z=1 else Z=-1

With X=1,Y=1,Z=1 corresponding to acceptable values for the delay, jitter, packet loss and when X=-1,Y=-1,Z=-1 corresponds to unacceptable values for the delay, jitter, packet loss, Delay_maximum, Jitter_maximum and Packet_Loss_maximum will not have the same values depending on the application used by the user.

In accordance with the recommendations of the ITU we will choose the best value for Delay_maximum, Jitter_maximum and Packet_Loss_maximum, for instance in the ITU recommendation G.114 delay must be between 0 and 150 ms for the user to be satisfied with his conversation. In Judging a user rating based on ITU specifications, this may appear easy but with these specifications we can predict that the users experience went well because if the value of our QoS evidence exceeds the bounds we know that the users experience was not good, for example, in using a Voice over IP application, we can determine whether the speaker was functioning correctly or not. Technically the user cannot have a good experience if the parameters of QoS evidence exceed the bounds.

9. CONCLUSION

In this work, we have presented our solution that provides an Android application that allows a set of servers to verify whether or not the hotspot rating of the user is trustworthy. We verify by measuring QoS evidence: delay, jitter and packet loss. The measuring process follows an innovative protocol that certifies the measurement in different cases.

There are even cases when the user can still cheat. In fact, there is one remaining case that we cannot certify. This is when the user rates a hotspot badly and when the delay measure is indeed bad.

We intend to mitigate this case with trust management in future work; also, we will take into account the overload of the hotspot when the users are co-located.

ACKNOWLEDGMENT

The research leading to these results has received funding from the EU IST Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n° 224024, project PERIMETER (User-centric Paradigm for Seamless Mobility in Future Internet) and from the EU IST Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n° 257418, project ULOOP (User-centric Wireless Local Loop).

REFERENCES

1. Gralla Preston, 2007. Don't fall victim to the 'Free Wi-Fi' scam. In: *ComputerWorld Networking & Internet*.
2. Douceur John R., 2002. The Sybil Attack. In: *Proceedings of 1st International Workshop on Peer-to-Peer Systems*.
3. E. Gray et al, 2003. Trust Propagation in Small Worlds. In: Proceedings of *the First International Conference on Trust Management*, LNCS 2692, Springer-Verlag.
4. Bettstetter C. and al., 2004. Stochastic Properties of the Random Waypoint Mobility Model. In: *ACM Wireless Networks*, vol. 10, pp. 555–567, Sept.
5. Ben Salem Naouel et al, 2004. Fuelling Wi-Fi deployment: A reputation-based solution. In: Proceedings of *WiOpt*.
6. Nicholson Anthony J. et al, 2006. Improved Access Point Selection. In: *MobiSys'06*.
7. Ormond O. Perry, P. Murphy, J., 2005. Network Selection Decision in Wireless Heterogeneous Networks. In: Proceedings of *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications*.
8. www.wefi.com
9. www.journaldunet.com/wifi
10. www.freespot.ch
11. www.cafes-wifi.com
12. www.jiwire.com
13. http://en.wikipedia.org/wiki/Packet_loss
14. http://en.wikipedia.org/wiki/Quality_of_service
15. http://www.itu.int/rec/T-REC-G/e.