ISSN 2067 – 4074

Location-Based Security for Resource Management

Iulia GRUMAZ
IT&C Security Master
Department of Economic Informatics and Cybernetics
The Bucharest University of Economic Studies
ROMANIA
iuliagrumaz@outlook.com

Abstract: Considering that wireless technology has widespread and mobile devices gain every day even more popularity, security concerns are increasingly shifting to these areas and solutions that take into account the high mobility users have been empowered with, are in demand. Hence, the aim of this paper is to propose a model for securing and controlling access of mobile clients to resources by using reliable location information, which is available by the means of access points already existing in network infrastructure of most organizations. The key part of the architecture, obtaining a reliable location proof and checking its authenticity, is based on a scheme derived from a validated cryptographic protocol, the Diffie-Hellman key-exchange. The software implementation demonstrates how current technology stack (Java, Android and Bluetooth etc.) can be used to create a workable system and a series of conclusions is drawn at the end to highlight the model's advantages, limitations and possible improvements.

Key-Words: Location, Mobile, Access Control, Partial Keys, Distributed System, Bluetooth, Android

1. Introduction

Given the rapid growth in the population of mobile devices over recent years and the wide adoption of wireless LAN systems as the leading technologies for Internet access, the demand of security measures to protect resources and maintain the integrity and confidentiality of data has heightened. Furthermore, location-based applications have developed in terms of value-added services they provide and complexity, using location awareness to improve data searches and condition or customize their offerings in numerous fields, such as education, social networks, e-commerce, medicine etc. In this context, one important security concern is the control of access in dependence to the position of the users and taking into account their high mobility.

With regards to the access control problem, the goal of this research is to put forward an architecture that enables an organization to manage availability of resources and regulate users' access to them taking into account their position within pre-defined areas of interest.

In this pursue, a secure scheme will be designed based on partial information – the keys, being distributed among several components of the system – access points, in order to allow the user to construct an undeniable location claim and the location-based authentication server to validate its authenticity.

A security analysis is to be performed after the resulted model is demonstrated with a prototype implementation, as well as drawing an outline of the strengths, possible limitations and further improvements in the concluding section.

2. Resource Access Control for Mobile Clients

Throughout the course of history, controlling access to resources such as files, devices, databases etc. was one of the main motivations of security mechanisms.

Access control is defined in [1] as "a process by which use of system resources is regulated according to a security policy and is permitted only by authorized entities according to that policy".

2.1 Types of access control

The main access control models defined are the following:

- Discretionary Access Control (DAC): a

model which lets users as owners of objects to grant access rights to other individuals; these kind of policies have a higher degree of flexibility, but with weak points: granting access is transitive, no restrictions apply to usage of information and privileges given by the owner may not be in concordance with the security requirements of the organization.

- Mandatory Access Control (MAC): there is a central authority that regulates the access of subjects to resources. Military security is probably the main area to use this model, examples of multilevel security models that formalize a kind of MAC policy being Bell-La Padula Confidentiality and Biba Integrity models. Main disadvantage is the high cost of implementation, but also lack of dynamic separation of duty or addressing fine-grained least privilege.
- Role-based Access Control (RBAC): access is given by the role that a user has within the system and on the rules that define if access is permitted or denied to certain roles. In this way, the management of privileges is simplified, rights for the role modified without repeating this operation for each individual and user can easily receive (or drop) all rights for their job tasks by assigning them to a certain role.

2.2 Location-Based Access Control

Location-based access control is an improved model that enforces access policies using the position of the users. In reality, this notion is very easy to comprehend: a worker needs to be at his workplace in order to operate a machine or a patient has to come to the hospital for medical services. It is the lack of remote control that supports the security policy in these examples, whereas software applications and services accessible by Wi-Fi must involve a location check to offer the same degree of security.

So an important security aspect is to have a verification mechanism in place in order to obtain accurate and reliable location information. For mobile users, this validation process has to adapt to an often and sudden switch of context as users move freely within the network-covered area.

As [2] presents, the relative position of the user and his movement in the surrounding environment has become easy to compute once sensors were integrated in common devices and location technologies have evolved. In spite of advancements in this area, there still are limitations [3], some of which can be turned into advantages, depending on the context of the LBAC solution:

- Coverage is limited for Wi-Fi, but makes it a good candidate for indoor spaces or urban area with hotspots;
- GPS is the suitable choice for outdoor environments, but as shown in [4] it can be spoofed programmatically with no modifications on the device;
- Bluetooth is a choice when more accurate location data is needed (3-10 m) within a small range, from 10 up to 100 meters.

Further on, it is interesting to see how LBAC can contribute to certain goals of software solutions, motivating its necessity. Authors of [5] found and evaluated five such objectives:

- principle of separation of duty, as it can impose physical separation between users or the opposite, depending on the situation;
- principle of least privilege, because the model can be very fine-grained, conditioning user access based on proximity of other participants and restricting it only to the user's workplace;
- accountability, due to the fact unauthorized usage is traceable as users need to be in a location unusual for their work and, also, attackers need to expose themselves by visiting the location;
- maintainability: while leveraging management by defining permissions to locations rather than users, LBAC increases complexity with the prerequisite of defining physical spaces;
- usability, since LBAC resembles

human behavior in physical world (where a person needs to be in the room to turn on the TV), being natural and familiar to users.

Their study highlights the key use cases for LBAC solutions, presenting situations which could benefit from their characteristics:

a. Enterprises, where they can act as safety-nets to ensure protection of data, accountability and maintainability;
b. Health-care or educational institutions, where a mix between LBAC and RBAC gives the advantage of assuring least privilege, collaboration and visibility;
c. Pervasive systems, with few security concerns, such as museums or conferences, where key points are usability and proximity;
d. Data centers and military facilities, where adding location to MAC guarantees containment and separation of duties.

2.3 Security Requirements

Taking into account the aspects previously presented, a set of desirable security-related requirements for a location-based access control solution to be used in an organization is outlined:

- Determine the location of the user, meanwhile protecting user location privacy;
- Define areas under control, where access should be granted or denied to certain roles and users;
- Restrict availability of the resources to certain areas, likely by providing a way to map resources and users/roles to rooms;
- Give only temporary access to resources, so that leaving the area results in losing rights on resources;
- Combine with other types of policies or traditional access control models;
- Integrate user authentication in the scheme and also protect the user credentials.

2.4 Proposed Solution

In relation with the enumerated requirements, this paper recommends a solution that, compared with more common models of access, gives the next advantages.

First of all, the organization can reuse existing parts in its network infrastructure, since the components that prove the location can be Bluetooth-enabled or wireless access points. Consequently, these location evidence providers are here under strict control and it is harder to tamper with them on the scene, in comparison with external providers.

Confidential data can remain in the restricted zone, the possibility to encrypt the information with a key derived from the location, which is periodically changed, remaining open; in [6], an example of location-based cryptography is given. Also, the solution can be extended to allow encryption of data transferred through the location points in the infrastructure by deriving an individual key from the user's key and the partial key of the access points nearby [7].

Providing continuity of the authentication service even when one location point is affected is achievable, since the central authentication server can monitor the location points and take their status into account when requesting location proof.

Another essential advantage would be that it is very difficult for an outsider to authentication in the name of a real user, because first he would have to risk exposure by being present on the premises. Additionally, for an insider, the chances to fake his location in order to gain access to resource from another location than this own are lowered.

Finally, a similar model can be constructed for IoT (Internet of Things) applications, to turn on or operate smart devices and deliver services only in the presence of the owner(s) or another personal device.

3. Model for Location-Based Access of Mobile Clients to Resources

Taking into consideration the aspects presented in the previous exposition, the objective is to design a model for secure access control based on location, intended to be used by organizations that want to condition the access to certain resources with the presence of the requester in a controlled area. For instance, in a mobile learning scenario, a university may offer the possibility to take a test only if the student is in the classroom. Also, view of a patient's medical records can be restricted to the consulting room or consulting confidential financial reports can be limited to the people gathered in a board meeting.

Therefore, the first presumption of the location-based architecture is that the environment is under the control of the organization. In this way, access-granted or access-denied spaces can be defined in terms of location groups that contain several location points, for example: $G_{classroom} = \{LAP_1, LAP_2\}$.

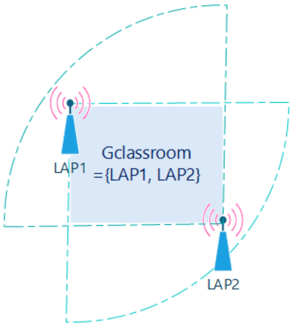

Figure 3 Defining location groups

These points are seen as part of the network of the organization, with the capability of communicating with a central server or the mobile clients through Wi-Fi, Bluetooth or RFID technologies, and the model proposed assumes they are physically secured.

Since location is the evidence necessary to get the desired resource, there are two main requirements to be met by the architecture:

- The mobile client has the ability to compute a location proof that the authentication server can validate.
- The location cannot be mocked, so the user will not be able to claim his presence in an area other than his current position.

For satisfying the first condition, the architecture includes an authentication server that keeps track of and distributes partials keys to the location points in such a way that, in the authentication phase, it is able to certify that a set of those keys were certainly used by the client in computing the shared key.

Because of the second specification, giving the mobile client the chance to determine its own location, through GPS, Internet positioning systems or other location data providers, independent of the controlled environment is not an option. The access points have the main role here, due to the fact that they can provide better accuracy for the positioning (especially if the chosen technology is Bluetooth), higher availability indoors and a greater level of assurance with regards to the location evidence.

The protocol proposed is inspired by the example given in [7] and the Diffie-Hellman key-exchange as it targets a similar result: arriving at a shared key between two entities, by exchanging messages over insecure channels. Another reason is that Diffie-Hellman can be easily extended to multiple parties and the intermediate keys can be composed in different manners.

To demonstrate the scheme, we define a group that contains two location points (the same steps being applied in case of n location points):

$$G = \{LAP_1, LAP_2\}.$$

The phases of the protocol are the following:

a. Generation of keys

The authentication server generates the global Diffie-Hellman parameters: g, p and the key-pairs for each of the location points:

$$LAP_{1pub} = g^{LAP_{1priv}} \bmod p \qquad (1)$$
$$LAP_{2pub} = g^{LAP_{2priv}} \bmod p \qquad (2)$$

b. Distribution of keys

The authentication server connects to each location point in a secure mode, based on mutual authentication, and serves its corresponding public key: LAP_{1pub}, LAP_{2pub}, while the private key remains a secret known only to the authentication server.

c. Gathering of keys

The location points broadcast the keys received in the nearby area (either by beacon messages, wireless data frames or Bluetooth service) in order for the mobile client within it to listen and gather all the keys for the location group G.

d. Mobile client computes the location claim key

Based on the D-H parameters obtained from the key server, the client will generate its own key-pair:

$$UK_{pub} = g^{UK_{priv}} \bmod p \qquad (3)$$

The location claim keys is calculated as the product of the location points keys raised to the power of the user private key using modulo arithmetic:

$$SK = (LAP_{1pub} \cdot LAP_{2pub})^{UK_{priv}} \bmod p \quad (4)$$

This is almost identical to the way the shared key from Diffie-Hellman agreement is obtained, with the difference that the base is a little more complicated.

e. Sending the location-based authentication request

With this request, two objectives have to be fulfilled: authentication of the user (by user-password tuple or other kind of credentials) and the authorization in relation to the location, by means of using the derived SK key to make the claim.

An example of how this can be achieved is to:

- use authentication by username and password, where only the hash of the password is stored;
- use the SK key to protect the password by encryption and to generate a HMAC signature of the request.

The necessary information: username, group G and client public key UK_{pub} will be included in the request, but other security controls may well be part of it, such as a nonce or a timestamp.

f. Examining the location-based authentication request:

The authentication server computes the same location claim key using the information in the request, by retrieving the private keys of the location points from the group specified and using the client public key:

$$SK = UK_{pub}^{(LAP1priv + LAP2priv)} \bmod p \qquad (5)$$

The equality between the values obtained by the client and by the server is confirmed by modulo arithmetic calculus:

$$\begin{aligned} SK_{client} &= (LAP_{1pub} \cdot LAP_{2pub})^{UK_{priv}} \bmod p \\ &= (g^{LAP_{1priv}} \bmod p \cdot g^{LAP_{2priv}} \bmod p)^{UK_{priv}} \bmod p \\ &= g^{(LAP_{1priv} + LAP_{2priv}) \cdot UK_{priv}} \bmod p \\ &= (g^{UK_{priv}} \bmod p)^{(LAP_{1priv} + LAP_{2priv})} \bmod p \\ &= UK_{pub}^{(LAP1priv + LAP2priv)} \bmod p = SK_{server.} \end{aligned} \quad (6)$$

If, using this key SK, the validation of the signature upon the request passes and the decrypted password is the same as the one stored, then the next statements are confirmed to the authentication server:

- User is located in the access area defined by the group of location points in the claim, hence he can receive authorization to access certain resources;
- User introduced the correct password, so the authentication phase is successful;
- The information from the request has not been altered or tampered with in transit, the HMAC signature guaranteeing its integrity;
- Confidentiality of the password was assured: only the server and the client can arrive at the shared key and decrypt its value.

Further on, the nonce and the timestamp, suggested as part of the request, act as a security measure against message-replay attacks in addition to the periodical refresh of the location point keys.

4. Implementation Using an Android Mobile Application and Bluetooth-Enabled Access Points

The model for access control to resources depending on the location of the user was implemented by an architecture which consists of four major components: an authentication server, the access points with Bluetooth capabilities, the Android mobile clients and a database to store the persistent information.

The Location-Based Authentication Server, one of the central pieces of the architecture, is a Java application which executes three essential jobs: re-generating the Diffie-Hellman parameters once a day, refreshing the location keys every 30 minutes and distributing them to the location access points. For scheduling these tasks, the Quartz component of Apache Camel framework was employed, due to easy configuration and out of the box integration with Akka [8], the toolkit chosen for ensuring scalability, high concurrency and real-time processing for the system. In addition, all the jobs can be triggered manually if needed, via a secure API, which can be consumed in the future by an UI module where the administrators can start and monitor the execution of these tasks. The present values for the parameters and the key pairs (private and public key with expiry time) are stored in a memory cache (handled by a Location Keys Service), which is cleared and filled every time new values are generated.

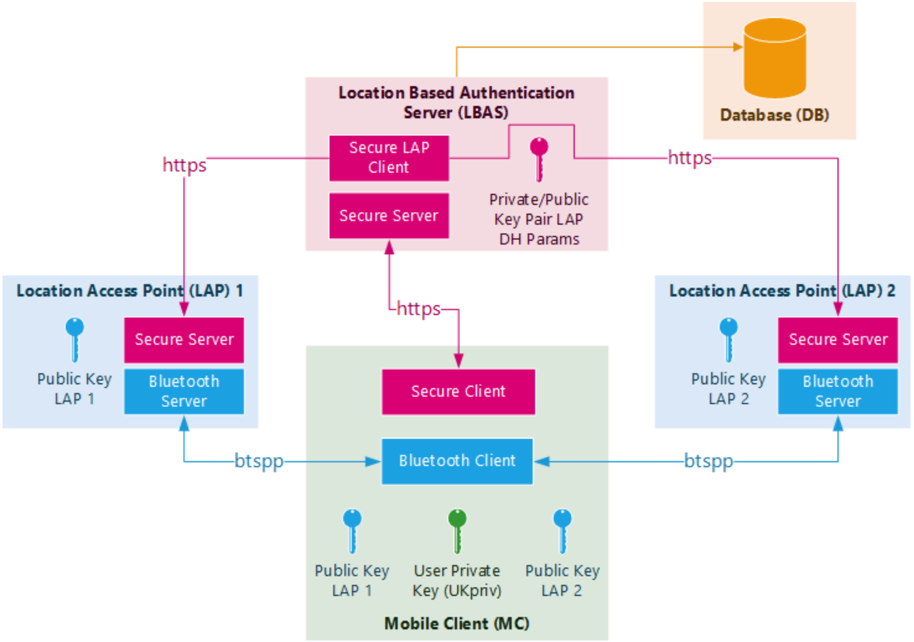

Figure 4. The architecture of the Location-Based Resource Management solution

For the purpose of distributing the partial keys, a REST client was designed to connect to the location access points, using Basic Authentication, and send over HTTPS its public key with the expiry time attached and the groups with which the point is currently associated. The connection is mutually authenticated, the certificate of the Authentication Server being installed for each new Location Access Point, while the Authentication Server has in its trust-store the certificates of all access points in the organization.

Another role of this component is to manage and keep up-to-date the information about users and their credentials, location points and the groups they form in order to define areas with different resources available for the user.

In that respect, the CRUD (create, read, update, delete) operations on the data are performed either as part of the execution of different tasks (for example, when the location access point is unreachable, its status is updated in the database or when the user signs up, his password is stored) or some of them by the administrator via the REST API available (when a new location or group should be added, for instance).

The Location Access Points are the intermediary actors in this architecture and give the mobile clients the means for proving their location to the Authentication Server. The Java application running on these access points has three sub-components: a secure REST entry point "/key" to receive their partial key from the Authentication Server, a cache to store it and a Bluetooth server that awaits client connections and makes use of the BlueCove library as an interface to the Bluetooth stack of the operating system [9]. Apart from their other functionality, such as offering wireless Internet access, they act as fixed location points that broadcast in a closed area the Bluetooth service to be accessed by mobile clients. In this way, the mobile application can obtain the necessary partial keys to compute the proof of their location in certain zones of the building delimited by a number of these location points.

Another key component of the architecture is the **mobile client**, represented in this implementation by an Android application with the Login activity as the main functionality. Apart from handling the input from the user: username and password, validating it and pre-processing in the scope of authenticating to the server (hashing the password), the application runs in background a number of tasks for determining its location proof. It sets up a broadcast receiver for the Bluetooth operations and starts the inquiry for devices in its proximity. The detected devices are filtered using the know UUID of the service the Location Access Points offer and are interrogated about their partial key, each request on a different thread. At the end of this process, a map of location keys by group is obtained and will be used for the authentication request. The application communicates with the other components of the architecture on two sides: with the Location Access Points via the Bluetooth client, utilizing most of the features of the Android Bluetooth API [10] (scanning, querying for services, pairing, establishing RFCOMM channels, transferring data) and with the Authentication Server through a REST client built using the Retrofit and OkHttp libraries. The mobile application implements its own TrustManager [11], which enhances the default one with the certificate of the Authentication Server and is needed for the configuration of the secured connections using TLS protocol, assuring a high level of security for that part of the communication. The confidentiality and authentication for the Bluetooth connection is based on a PIN that has to be entered on both devices.

A close component to the Authentication Server is **the database**, which stores the information regarding the location access points (URL, name, status: up/down etc.), the way they are grouped in order to define areas with different resources available for the users (location groups), the users of the application and their credentials and, also, historical data on successful and failed logins for auditing and accountability purposes. In this particular implementation, we have a MySQL database, on top of which there is the Hibernate ORM as a persistence layer. However, in view of the possibility to use other types of storage, the business logic is decoupled from the database and persistence layer, by using an intermediary layer for data access.

Authentication and authorization of user

For the login functionality, the Authentication Server provides for the mobile clients, first of all, an entry point "/login/dh-params" to retrieve the global Diffie-Hellman parameters, which need to be used for generating the key-pair for the user and for computing the shared key to prove the location of the requester. The response from the Authentication Server also contains a timestamp and a nonce, which is set to have a life-time of 5 minutes.

The mobile client will extract from this response the parameters to generate his own Diffie-Hellman (DH) key-pair. From his private key and the collection of public keys from a group of location points the DH derivate key will be computed, then hashed and the first 128 bits will represent both the shared key (SK) between the client and the server and the proof of the user's location. The hashed password that the user introduced is protected by encrypting it with this key (AES-128 algorithm). SK is also the key material for generating a HMAC signature of the authentication request (algorithm chosen was HMAC-SHA-256). Finally, the authentication request is issued with the proper data: username, group id, secret (encrypted hashed password), timestamp, nonce (from the server's response when getting the DH parameters), the client public key and the signature.

On the server side, the authentication request follows validation steps ranging from the most basic (assert no required information is missing, fields are correctly formatted etc.) to checks of integrity and freshness. The timestamp and the nonce are verified to see if they are in the accepted time interval, so that the same request cannot be replayed later on. Afterwards, a similar step to the one in the mobile application is performed to derive the shared key (SK), but having as input the public key of the client and the private keys of the location points. Once the signature is checked using this key and the password stored for the user matches the one decrypted from the request, the authentication server can confirm the identity of the user and the fact that he is in the area defined by the location points for that group. The last part is to invalidate the nonce used, log the success/failure of the login operation, generate a session token and return it in the response to the client, so that the user can access for a limited interval of time (30 minutes) the resources available for that zone of the building.

5. Conclusions

In this paper, a model of resource management for mobile clients was presented using a location-based approach. The role of each component in the architecture was described alongside with the protocol at the end of which the location evidence is obtained (based on the well-known Diffie-Hellman key agreement). Additionally, a practical implementation, with the associated technology stack, was given for testing its feasibility and discovering the possible weaknesses in a real scenario.

While the model fully satisfies the security requirements defined, it is still vulnerable to attacks such as a wormhole attack, where two attackers collaborate on gathering the location keys for a group, even though they would not have been individually authorized for that set of resources. However, an implementation using wireless access points instead of Bluetooth-enabled ones would reduce the chances for this kind of attack by forcing the mobile client to switch periodically between the access-points of the group and, therefore, guarantee it is in the range of each of them.

Also, an attacker from the inside could impersonate an access point, injecting fake location keys and compromising the authentication of mobile clients, in a so called Sybil attack [12]. However, the level of impact and extension of such a threat is low, a thorough inspection of the premises being sufficient to discover the author inside the organization.

If the mobile client is standing between areas represented by 2 different groups, there might be the cases when: a) all the location keys for both groups are obtained, b) all location keys are retrieved at least for one group, the other only partially, c) for both groups, only a part of the location keys are acquired. The current implementation tries to authenticate applying the previous algorithm on each group until it succeeds, which means scenarios a) and b) are covered. The application can be extended in such a way that, for the first scenario, the user is able to choose for which group he wants to be authorized depending on the resources he wants to access. The last situation is a matter of positioning the access points in order to delineate as well as possible the areas of interest in the building.

Acknowledgement

Parts of this paper were presented at The 8th International Conference on Security for Information Technology and Communications (SECITC 2015), Bucharest, Romania, 11-12 June 2015.

References

[1] "Internet Security Glossary, Version 2,", available online, http://tools.ietf.org/html/rfc4949, accessed on March 20, 2015.

[2] C. A. Ardagna, M. Cremonini, E. Damiani, S. d. C. Vimercati and P. Samarati, "Supporting Location-Based Conditions in Access Control Policies," *Proceedings of the 2006 ACM Symposium on Information, computer,* p. 222, 2006.

[3] J. Koppers, "Location Based services: Positioning techniques,", available online, http://www.positioningtechniques.eu/lbs_technique_checker.asp, accessed on April 2, 2015.

[4] The University of Texas at Austin, "Todd Humphreys' Research Team Demonstrates First Successful GPS Spoofing of UAV,", available online, http://www.ae.utexas.edu/news/features/todd-humphreys-research-team-demonstrates-first-successful-gps-spoofing-of-uav, accessed on May 5, 2015.

[5] A. v. Cleeff, W. Pieters and R. Wieringa, "Benefits of Location-Based Access Control: A Literature Study," *GREENCOM-CPSCOM '10 Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing,* 2010, pp. 739-746.

[6] C. Boja, "Location Based Encryption with Distributed Shared Keys in Mobile Services," *Proceedings of Information Technology and Communications Security Conference,* 2011, pp. 67-74.

[7] Y. Sun Cho and L. Bao, "Secure Access Control for Location-Based Applications in WLAN Systems," *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference,* 2006, pp. 852-857.

[8] C. Ibsen and J. Anstey, "Appendix E: Akka and Camel," in *Camel in Action*, Manning Publications, 2011, pp. 487-492.

[9] E. C. Ortiz, "Using the Java APIs for Bluetooth Wireless Technology, Part 1 - API Overview,", available online, http://www.oracle.com/technetwork/articles/javame/index-156193.html, accessed on April 10, 2015.

[10] "Android Bluetooth,", available online, http://developer.android.com/guide/topics/connectivity/bluetooth.html, accessed on March 10, 2015.

[11] "SSL on Android: The Basics,", available online, https://commonsware.com/blog/2013/03/04/ssl-android-basics.html, accessed on February 20, 2015.

[12] J. R. Douceur, "The Sybil Attack," 2002., available online, http://research.microsoft.com/pubs/74220/IPTPS2002.pdf, accessed on March 14, 2015.