

Framework for One Account Log-in From Multi Device On Mobile Application

Rafinno Aulya*, Ary Setijadi Prihatmanto** and Kyung Hyune Rhee*

*Dept. of IT Convergence and Application Engineering, Pukyong National University, Republic of Korea

**School of Electrical Engineering and Informatics, Bandung Institute of Technology, Indonesia

rafinno.aulya@gmail.com, asetijadi@lskk.ee.itb.ac.id, khree@pknu.ac.kr

Abstract

Many applications require users to log-in, some applications can allow users to use the same account is active on multiple devices at the same time. On location-based application that serves to record the movement of the position, account is active on multiple devices could not be permitted because it causes a different location data from multiple devices, so the location of the users become ambiguous. This paper describes a simple protocol to prevent users from using their accounts on multiple devices at the same time. This Protocol will turn off one account on a device when the account log-in on new devices.

1. Introduction

Nowadays, most of the applications requires users to have an account. Some applications permit the users to log-in on multiple devices with a single account and at the same time. Example : Facebook, KakoTalk, etc.

There are also some applications that prohibit it. i.e. Location-based applications, the application will track the user's location and record it periodically. If status of the account is active on some devices, data collision will occur, and will make ambiguous location of the user. Therefore a protocol is needed to prevent the data collision.

With the rapid development of network technology, user authentication scheme in ecommerce and m-commerce has been becoming one of important security issues [1]. Nowadays, location-based services are widely utilized, including identifying user locations, offering traffic status, providing point of interest (POI) information, and guiding routes [2]. The authentication procedures on more security-sensitive sites, such as those for banking, stock trading, email, and online social networks [3].

This paper describes a protocol which prevent the users to use their account to log-in on multiple devices.

2. Design of the application system

In this section, we describe the design of protocol. This protocol is divided into three parts : Mobile application (MA), Location Based Service (LBS) and Log-in Management Service (LMS). Overall, the design of protocol is illustrated in the Figure 1.

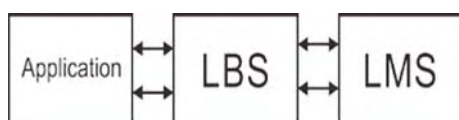


Figure 1. Overall System

When users use the application, the user is prompted to log-in with his account. Users also can register if the users do not already have an account. On LMS, log-in process and registration process are considered equal by the system, because they necessary to start a new session. After log-in,

application will communication with LBS. And every communicate, LBS will check about the session.

In the LMS, we use a database, and the structure of database can be seen in the Figure 2.


Name	Type
ID 	int(11)
UserID	int(11)
StartTime	datetime
LastTime	datetime
EndTime	datetime

Figure 2. Session Table In LMS

ID represents session_id, in this experiment we set the ID become auto increment. User_id is ID of user from LBS. StartTime is time starts of the session. LastTime is last time of the session is used, and EndTime is the expiration of the session.

• Log-in Process

This process is when users begin using the application.

1. Application send data log-in to LBS, e.g. username and password
2. If the log-in fails (e.g. username and password wrong), the LBS provides a response to the application failed log-in.
3. If there are success log-in, LBS sends user ID to LMS.
4. LMS takes the last session based on User ID. This will result in some cases :
 - a. No data. This case occurs if user registration, because there is no session is ever made by the user.
 - b. Data exist and EndTime equal to NULL. This case happens when the previous

session was not over yet or the user has not done a logout on the device before, and logged back on new devices. Therefore, LMS will given value of EndTime with current time. It means that previous session has expired.

- c. Data exist and EndTime not equal to NULL. This case is previous session has expired.
 - 5. LMS will create a new session. ID equal to Auto increment, User ID equal to ID of user, StartTime equal to current time, LastTime equal to current time, and EndTime equal to NULL.
 - 6. LMS will send the session ID to LBS
 - 7. LBS will send the response log-in success including session ID. And the session ID will be stored by the application for the next communication
- Communication Process
This process is data requests by applications to the server after log-in.
 - 1. Application send request data and session ID to LBS (e.g. update data).
 - 2. Before LBS respond to request data from application, LBS will send the Session ID to LMS. LMS will take row data session based on Session ID. In case :
 - a. No data. It means the session ID is fake, and LMS will respond to LBS that the data is invalid.
 - b. Data exist and EndTime not equal to NULL, LMS will respond to LBS that the session ID has expired. And LBS will refuse application request and send respond about session has expired to application
 - c. Data exist and EndTime equal to NULL. Then, LMS will update the LastTime with current time. And LMS will respond to LBS that the session ID is active and continue application request

- Log-out Process

This process is when the user is logged out of the application.

1. The application send logout request including Session ID to LBS
2. LBS will send the session ID to LMS
3. LMS will take row data session based on Session ID and update the EndTime with current time

3. Security Analysis

In this section, we will describe security analysis in this protocol.

Ambiguous location. This Protocol makes each account can only be active on one device, it makes data of every movement of the user is received by a server is only recorded and transmitted from one device only.

Secure communications. The user will get a different session id every log-in session Id used to be sent to the server for each request for data or communication with the server

after doing a log in as user authentication. So when the attacker try to access API (Application Programming Interface), the attacker must know the active session ID.

4. Implementation

In this section, protocol implementation is presented. We using PHP as API, MySQL as database, Postman installed on Google Chrome to try the APIs and XAMPP for local server. Scenario of this implementation we as follows:

Alice log-in on device A is shown in Figure 3.

Sign In

http://localhost/tracker/api/user/signin.php

form-data x-www-form-urlencoded raw

Email: alice@email.com

Pass: 000000

OS: 1

Key: Value

Send Save Preview Add to collection

Body Headers (7) STATUS 200 OK TIME 210 ms

Pretty Raw Preview JSON XML

```

1 {
2   "MyProfile": {
3     "ID": "1",
4     "Nickname": "alice",
5     "AvatarID": "0",
6     "Poin": "1000",
7     "JoinDate": "2015-08-18 06:10:31",
8     "LastLocation": {
9       "Speed": "0",
10      "Altitude": "0",
11      "Latitude": "0",
12      "Longitude": "0",
13      "TimeLocation": "2015-08-18 06:10:31"
14    },
15     "Email": "alice@email.com",
16     "CountryCode": "82",
17     "PhoneNumber": "1012345678",
18     "Birthday": "2015-08-18",
19     "Gender": "1"
20   },
21   "SessionID": "2"
22 }

```

Figure 3. Sign In Process (Device A)

Because the username and password are entered by Alice is true, then the LBS sends Alice's profile and Session ID. After that, the device A will communicate with the LBS using the Session ID = 2. i.e. Alice can search for other users based on the keywords that entered, such as by name or email. For this search, device A requesting data through access API and send the session id to the server, it shown in Figure 4.

Search User

http://localhost/tracker/api/friend/search.php

Auth: ITBTMDG8BSTS

Sessionid: 2

Header: Value

form-data x-www-form-urlencoded raw

Keyword: fin

Key: Value

Send Save Preview Add to collection

Body Headers (7) STATUS 200 OK TIME 100 ms

Pretty Raw Preview JSON XML

```

1 [
2   {
3     "ID": "2",
4     "Nickname": "rafinno",
5     "AvatarID": "0",
6     "Poin": "1000",
7     "JoinDate": "2015-08-18 06:19:50",
8     "LastLocation": {
9       "Speed": "0",
10      "Altitude": "0",
11      "Latitude": "0",
12      "Longitude": "0",
13      "TimeLocation": "2015-08-18 06:19:50"
14    },
15    "Email": "rafinno@email.com",
16    "CountryCode": "82",
17    "PhoneNumber": "1012345678",
18    "BirthDay": "2015-08-18",
19    "Gender": "1",
20    "Relation": null
21  }
22 ]

```

Figure 4. Request Search User with active Session ID

LBS respond to requests from the device A, because its submitted a valid session id and still active. Therefore, each device must store the session id obtained from LBS. Because the session id can authenticate the user profiles and also can be used in every communication with LBS.

After that, Alice log-in on device B without logging out on device A, it shown in Figure 5. Because this application does not allow users to log-in with an account on two devices at the same time, in these case the role of this protocol is needed, this protocol will reject previous communication devices with LBS, thus LBS only receive data from the device that is the device B.

Sign In

http://localhost/tracker/api/user/signin.php

form-data x-www-form-urlencoded raw

Email: alice@email.com

Pass: 000000

OS: 1

Key: Value

Send Save Preview Add to collection

Body Headers (7) STATUS 200 OK TIME 180 ms

Pretty Raw Preview JSON XML

```

1 {
2   "MyProfile": {
3     "ID": "1",
4     "Nickname": "alice",
5     "AvatarID": "0",
6     "Poin": "1000",
7     "JoinDate": "2015-08-18 06:10:31",
8     "LastLocation": {
9       "Speed": "0",
10      "Altitude": "0",
11      "Latitude": "0",
12      "Longitude": "0",
13      "TimeLocation": "2015-08-18 06:10:31"
14    },
15    "Email": "alice@email.com",
16    "CountryCode": "82",
17    "PhoneNumber": "1012345678",
18    "BirthDay": "2015-08-18",
19    "Gender": "1"
20  },
21   "SessionID": "4"
22 }

```

Figure 5. Sign In Process (Device B)

On device B, id session to Alice's profile is changed to 4. When a device tries to communicate with the LBS, the LBS will reject the communication request, because the session id for the Session ID = 2 has expired, it shown in Figure 6.

Search User

http://localhost/tracker/api/friend/search.php

Auth: ITBTMDG8BSTS

Sessionid: 2

Header: Value

form-data x-www-form-urlencoded raw

Keyword: fin

Key: Value

Send Save Preview Add to collection

Body Headers (6) STATUS 408 Request Timeout TIME 15 ms

Pretty Raw Preview JSON XML

1 SESSION TIME-OUT

Figure 6. Request Search with expired Session ID

5. Conclusion

In this paper, simple protocol to handle multiple log-in on mobile application is presented. We do not explain about encryption, cryptography and other security analysis. In the future work, it can be added to this protocol.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2014R1A2A1A11052981).

References

- [1] An, Younghwa & Joo, Youngdo. 2013. Security Analysis and Improvements of a Password-Based Mutual Authentication Scheme with Session Key Agreement. International Journal of Security and Its Applications Vol 7 No. 1. Page 85-94
- [2] Ma, Shang-Pin & Yan, Duan-Yi. 2012. Location-Based Web Service Delivery: A Data-Mining-Based Approach. International Symposium on Computer, Consumer and Control. Page 666-669
- [3] Haque, S.M. Taiabul. Wright, Matthew & Shannon, Scielzo. 2014. Hierarchy of users' web passwords: Perceptions, practices and susceptibilities. International Journal of Human-Computer Studies 72. Page 860-874
- [4] Wang, Junhan. Qiao, Fei & Lu, Jianfeng. 2013. Research and application of the location information in the intelligent transportation. International Workshop on Cloud Computing and Information Security (CCIS). Page 522-525.
- [5] Sampatha, Sreedevi & Bryceb, Renée C. 2012. Improving the effectiveness of test suite reduction for user-session-based testing of web applications. Information and Software Technology Volume 54 Issue 7. Page 724-738
- [6] MySQL : <http://dev.mysql.com/doc/>
- [7] PHP : <http://php.net/docs.php>
- [8] Google Chrome : <https://www.google.com/chrome>