An Extensional Client Authorization Scheme for IoT Scenarios by Using OAuth 2.0 and PoP Token

Xing Xiaonan*, Sunggyun Jang*, Inwhee Joe*
*Mobile and Network Convergence Laboratory
Dept. of Computer Science, Hanyang University
xingxiaonan@hanyang.ac.kr, mrjang28@hanyang.ac.kr, iwjoe@hanyang.ac.kr

ABSTRACT

To improve the security of OAuth 2.0 access token transportation and satisfy the challenge of resources constraint caused by the bearer token access mechanism of the OAuth 2.0, we proposed an extensional client authentication scheme that is based on the Proof-of-Possession (PoP) token mechanism. By improving the integrity of PoP token, we bind a PoP key of a public/private key pair to the PoP token. The authorization server and the resource server can authenticate the identity of the client by verifying whether the client has the possession of the PoP token. If the client can prove that it has a PoP key that matches the PoP token, then the identity of the client can be authenticated. This experimental evaluation can confirm that this scheme effectively dealing with the issue of client identity authentication and reduce resources consumption.

1. Introduction

With the increase of Internet of Things (IoT) deployments, the need for a better user experience for handling the authentication and authorization tasks in constrained environments [1]. Several technologies have already been developed. Open Authorization especially the OAuth 2.0 Protocol can reduce the complexity and avoid the burden upon the service provider that can satisfy the needs of requirements for a better user experience in the authorization procedure.

OAuth is an open protocol to allow secure authorization from third-party applications in a simple and standardized way [2]. The standard OAuth 2.0 protocol is based on the bearer token access mechanism that allows an entity holding the bearer token to access the protected resources without providing the credentials. Moreover, the current OAuth 2.0 protocol discarded all the encryption and dropped signatures. That needs another mechanism to make sure the security during the transmission process. The OAuth protocol provides an authorization layer for HTTP-based service APIs, typically on top of a secure transport layer, such as HTTP-over-TLS (i.e., HTTPS) [3]. The HTTPS does ensure that the client and the server securely communicate and exchange the bearer token and prevent eavesdropping and tampering. However, if the token is sent to the wrong destination, it cannot help.

For addressing the issues mentioned above, in this paper, an extensional scheme has been proposed that using the OAuth 2.0 protocol and PoP token to build an extensional client authentication scheme. In this scheme, we use the JSON Web Token(JWT) to represents the PoP token and bind the PoP key to the PoP token. The client should use the ownership of the PoP key to proving that it has the possession of the PoP token. Thus, as a legitimate client to obtain access to the resources.

The rest of the paper is arranged as follows. In Section 2, related works are presented. In Section 3, the detailed workflow of the extensional scheme is presented. Section 4 is the performance evaluation of the proposed scheme. In Section 5 we draw out conclusions.

2. Related Work

OAuth 2.0 protocol is an authentication protocol that lets client obtain access to protected server resources on behalf of a resource owner [4]. Also helps end users authorize third-party access to resources server without having to share their credentials, such as user names and passwords.

OAuth defines four roles; the participants are as follows:

Resource Owner (RO). The resource owner is an entity that can grant the third-party application access to a protected resource. If the resource owner is a person, it always referred to as an end-user.

Resource Server (RS). The server that hosts the protected resources of resource owner, and can use the access token to response to the protected resources requests.

Client. An application that makes the protected resource requests and accesses the protected resource with the resource owner's authorization.

Authorization Server (AS). The server that receives the client's protected resource requests and response it with the access token when the requests are successfully authenticated.

The standard OAuth 2.0 protocol works as the following figure 1 shows:

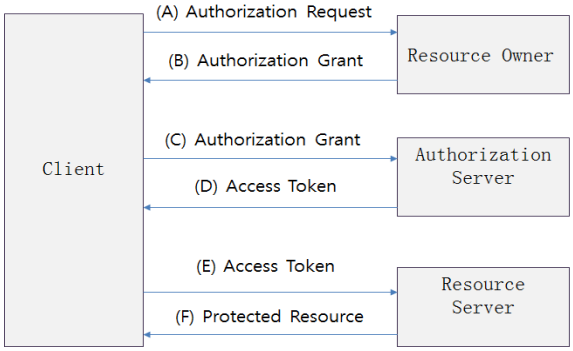

Figure 1. Authorization flow of standard OAuth 2.0 Protocol

(A) The client sends an authorization request to the resource owner for authorization grant.
(B) The resource owner response to the authorization request with authorization grant.
(C) The client uses the authorization grant received from resource owner to send requests to the AS for the

access token.

(D) The access token will be sent to the client if the AS authenticates the client and the authorization grant successfully.

(E) The client uses the access token received from the AS to request the access to protected resources to the RS.

(F) RS validates the access token and returns the protected resource to the client.

However, the standard OAuth 2.0 protocol cannot make sure the access token transport securely. Several works have been done about the security of OAuth 2.0. The scheme [5] proposed an extended authentication scheme to enhance security protection of OAuth 2.0 Protocol. It insisted a method that using the E-mail during the authentication between the third party and user to preventing the replay attack, phishing attack, and impersonation attack.

The scheme [6] proposed an approach to provide a secure authentication mechanism for IoT network by using a security manager. The security manager is used to determine whether the user is authenticated by comparing the user ID obtained from the service provider with its local database. Only in successful authentication, the security manager allows the user to access the IoT network. This scheme prevents unauthenticated users to access the IoT network.

Darwish et al. [7] evaluated the implementation of Google's OAuth 2.0 for web server application. His work demonstrates that the implementation of Google's OAuth 2.0 protocol can generate a security flaw by exhausting the storage resources of the web server.

So, in this paper, we proposed a scheme based on OAuth 2.0 and PoP token for securing the service in the IoT open platform. The access token is a type of signed JSON Web Token (JWT) with the PoP key. The Proof-of-possession key is a means of proving that the party sending a message is in possession of a particular cryptographic key [8]. Using the PoP key as a proof to authenticate the message sender is the legal or not.

3. Proposed Scheme

The main aim of our proposed scheme is to improve the security of access token transportation and efficiently authenticate the identity of clients. Our proposed approach is based on the authorization code grant type of OAuth 2.0 protocol. It is the most common OAuth 2.0 flow, and it implements 3-legged OAuth.

The detailed step of the authentication and authorization process is as follow

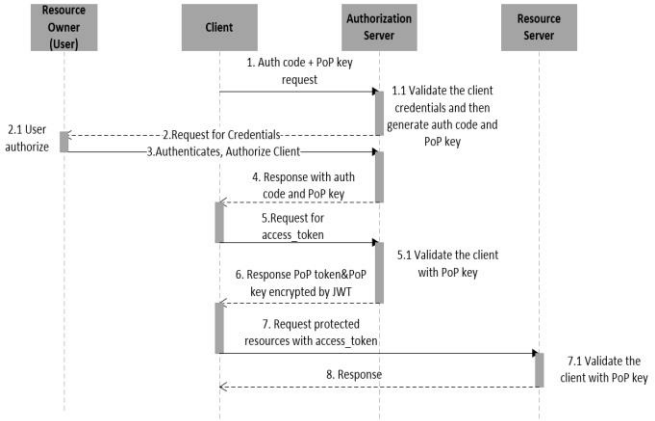

Figure 2. Based on OAuth 2.0 and PoP token to extend the client authentication

1) The client requests authorization code and PoP key in exchange for the access token to the server.
2) The AS authenticates the identity of the client that sends the request to the user.
3) If the user authorized the request, the client receives an authorization grant that is the authorization code. The authorization code is a JWT token. In this JWT token, the claim set of payload part contains the aud (audience) is meant to refer to the receiver that should accept the token. Besides, define a cnf (confirmation) claim in this JWT token. This claim is used to bind a public/private key pair to the token.
4) The client extracts the public/private key pair from the authorization code and sends for the access token to the AS. The client should prove to the AS that it has the private key corresponding to the public key. Sign the request with the private key and send it to the AS.
5) The AS receives the request and validates it with the public key. If the client identity is authenticated and the authorization code is valid, the AS issues an access token to the client. The access token is also a JWT token. The claim set contains the reserved claim names and the definition claim cnf, as follows:
   a) Audience (aud): It is meant to refer to the receiver that should accept the token.
   b) Confirmation (cnf): It is used to bind the pop key to the token. The cnf claim is used to prove that the token is in the possession of the token holder, then can prove that the client is the legitimate one.
   c) Expiration time (exp): It is the expiration time on or after which the token is invalid.
   d) Issue at time (iat): it means this token issued at the time.
   e) Issue (iss): It is the client that issued the token.
   f) JWT id (jti): It is used to the RS to identify the JWT token and used to prevent the replay attack by not allowing the issuer to reuse it.
   g) Subject (sub): It can be a resource owner or authorized delegate. In this case, it is the client.

The PoP token with the PoP key generated for the access token is as follows:


```
{
 "aud" : "fad38724-3662-4de0-9540-ba1bd0c0752d.oauthIoTplatform.hanyang",
 "cnf_pk" : "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCHu8yW4b98FyLivEM/JtfndbL
 "cnf_sk" : "MIICdwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAAoGBAIe7zJbhv3wXIuK
 "exp" : "1505723870972",
 "iat" : "1503131870972",
 "iss" : "AuthorizationServer",
 "jti" : "bc10681f7f30be5e9c49c83be1dab455",
 "sub" : "test"
}
```


Figure 3. PoP access token

6) The client request for protected resources to the RS by using the PoP token. The RS parse the access token with the public key which is authenticated from the AS. If verification is success, the client is legitimate and can gain the access to protected resources.

4. Performance Evaluation

In the standard OAuth 2.0 protocol, because of the access token has an expiry time, there need the refresh token to obtain the new pair of access and refresh token. Hence, the refresh tokens need to be stored securely in the client. In our proposed scheme, the claims of JWT are encrypted, it doesn't require the client to ensure the security of the refresh token. At the meanwhile, if the client needs a new access token, it will create a new JWT token. Therefore, the client also doesn't need to store the refresh token. Figure 4 illustrates the storage space of storing refresh token in different approaches. The x-axis represents the total number of requests and the Y-axis represents the storage required in KBs. Since refresh token and client secret require secure storage, they are encrypted and stored in the database, the storage space of standard scheme is increased more rapidly than the proposed scheme from 200 requests to 1000 requests.

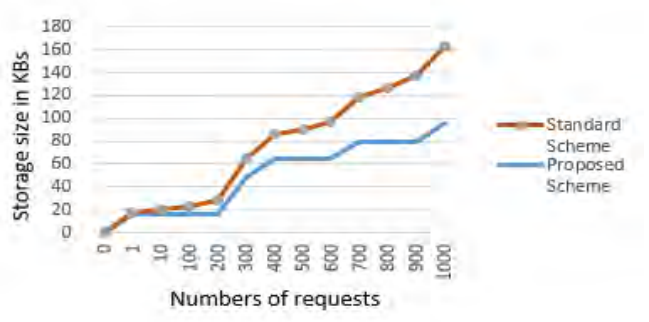

Figure 4. Storage space required in different schemes

In addition, in the standard OAuth 2.0 protocol, if the access token is expired, the client needs to obtain the valid access token through refresh token. In our proposed scheme, we don't use the refresh token to re-obtain the access token. Figure 5 illustrates the response time for re-obtaining an access token for each approach. The x-axis represents the schemes and the y-axis stands for the computation time in milliseconds

Figure 5. Time cost for re-obtain an access token in different approaches.

5. Security Analysis

The implementation of the proposed client authentication scheme depends on the PoP key and the PoP token, which are faced with various threats and attacks during the delivery and use. For example, the access token may be threatened by eavesdropping, tampering, replay and the PoP key may be eavesdropped. Especially in the transport layer does not provide adequate security mechanisms, end-to-end security of the application layer must be provided, to protect their integrity, confidentiality and prevent replay.

In our proposed approach, we encrypting the symmetric session PoP key to protect its confidentiality, signing the PoP token to ensure its integrity, using a one-time random number to prevent the replay attack, using the aud parameter to specify the token's receiver to prevent token redirection, etc. In this way, phishing attack and replay attack can be avoided in our scheme.

6. Conclusion

The standard OAuth 2.0 protocol relies completely on HTTPS to ensure the access token transportation. The HTTPS does make sure that the security of the access token transportation. But if the access token redirects to the wrong destination, the HTTPS doesn't work anymore. Besides, the client identity authentication of the standard OAuth 2.0 protocol is not consummation. To solve the mentioned issues, in this paper, we have developed an extensional client authentication scheme that based on the OAuth 2.0 protocol and the PoP token. The PoP token can enhance the security of access token transportation by binding a PoP key and enable the communication parties to authenticate each other. This scheme can decrease storage space requirement of refresh token and increase the response time of re-obtain access token. But there are also a lot of work to do in the future.

Several vendors have determined the size of the access token and refresh token. In our scheme, we don't consider the issue of PoP token's size. Although the OAuth 2.0 doesn't specify the restriction size of the access token, the next we would like to focus on how to reduce the size of PoP token for resource-constrained IoT scenario.

References

[1] H. Tschofenig, "The OAuth 2.0 Internet of Things (IoT) Client Credentials Grant", ACE, Internet-Draft, Jan. 2015

[2] E. Hammer-Lahav, "The OAuth 1.0 Protocol", RFC 5849, Internet Engineering Task Force, Apr. 2010.

[3] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, Internet Engineering Task Force, Aug. 2008.

[4] D. Hardt, "The OAuth 2.0 authorization framework," Internet Engineering Task Force(IETF) RFC 6749, 2012.

[5] C. Cheol-Joo, C. Kwang-Nam, C. Kiseok, Y. Yong-Hee and S. YounJu, "The Extended Authentication Protocol using E-mail Authentication in OAuth 2.0 Protocol for Secure Granting of User Access" Journal of Internet Computing and Services (JICS), pp. 21-28 Feb. 2015.

[6] Shamini Emerson, Young-Kyu Choi, Dong-Yeop Hwang, Kang-Seok Kim and Ki-Hyung Kim, "An OAuth based Authentication Mechanism for IoT Networks", Information and Communication Technology Convergence (ICTC), IEEE 2015.

[7] Darwish, Marwan, and Abdelkader Ouda. "Evaluation of an OAuth 2.0 protocol implementation for web server applications" Computing and Communication (IEMCON), 2015 International Conference and Workshop on IEEE, 2015.

[8] Michael B. Jones. "The Increasing Importance of Proof-of-Possession to the Web" W3C Workshop on Authentication, Hardware Tokens and Beyond. August 14, 2014.