EAP Based Certificateless Authentication Technique to Access Cloud Services in Openstack

K. Raghavendra[1(✉)], B. Ramesh[2], and J. Chandrika[2]

[1] Computer Science and Engineering, AI-ML Department, Dayananda Sagar University, Kudlu Gate, Hosur Main Road, Bangalore, Karnataka 560068, India
raghavendrak-cse@dsu.edu.in

[2] Department of Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India

Abstract. Nowadays cloud services have gained more interest. The main advantage of the cloud is, it reduces management costs and efficient usage of resources. The major role of efficient authentication means not only providing authentication but it should also reduce the authentication traffic and provide security. There are many authentication mechanisms most of them were only for a web application. Here in this paper, we focus on both web and non-web applications authentication in cloud environment to access SaaS service. For non-web application services of cloud, ABFAB has created an architecture. ABFAB also defines how to use existing EAP/AAA and GSS-EAP for both web and non-web-based applications. In this paper, we mainly focus on efficient AAA to access cloud services. To demonstrate the proposed system, we use the moonshot from Github, freeradius an open-source, and Openstack for cloud service with our proposed authentication method. We have done the performance analysis of our authentication method compared with the other authentication mechanism to access cloud services. This analysis shows a significant reduction in the computation time required for authentication and reduction in the authentication traffic.

Keywords: EAP · EAP-TLS · AAA · SAML

1 Introduction

OAuth, OpenID, and SAML are the few web-based authentication technologies to access cloud services and also in federated identity management technologies used by Google, Amazon, and Microsoft. RADIUS and EAP methods support AAA for both web and non-web (generic) applications in accessing cloud services and also in federated identity management environments, in AAA federated environments each organization should deploy AAA server and deployed organizations should interconnect with all AAA servers, in this way federation of identity and AAA can be provided. For authentication, an extensible authentication protocol framework is used with an extensible set of "EAP methods (e.g. EAP-TLS, EAPTTLS, EAP-MD5)".

D. S. Guru et al. (Eds.): ICCR 2021, CCIS 1697, pp. 139–151, 2022.
https://doi.org/10.1007/978-3-031-22405-8_12

AAA is widely used for network access in federated environments. Eduroam is an example which is using AAA infrastructure for providing internet access through WiFi for the members of federated organizations students and research scholars. By the federated identity management, if the user of the same organization (home organization) once authenticates, he or she can access services provided by other federated organizations. Today more interests gaining in providing AAA for services, provided by cloud and internet access. An example of providing internet access using AAA infrastructure is eduroam, here RADIUS provides federated infrastructure and the Extensible authentication protocol provides authentication. The success of eduroam has gained more interest to use RADIUS infrastructure for AAA for any type of application services for example SSH, HHTP, and cloud services, including network access too. For providing AAA using RADIUS, EAP and to access any kind of services in the federated environment is defined in ABFAB. "Generic Security Service Application Program Interface (GSS-API)" is a new mechanism specified by ABFAB. "GSS-API is based on an Extensible authentication protocol". "GSS-API" is already included and supported by many of the application services.

In GSS-EAP authentication, several authentication message exchanges take place with the end-users home organization. In typical EAP authentication, first, there will be the establishment of tunnel TLS/SSL (using the EAP-TLS method) between the AAA server(home) and the end-user.

1.1 Extensible Authentication Protocol/AAA

EAP is a prime example of a generic authentication framework. EAP has gained popularity in the recent years. The popularity arises from the pressure imposed on the network designers and administrator to not only support the legacy authentication mechanisms required to run their existing platforms but also to support the newer and stronger authentication mechanisms. However, one of the important goals of the exercise of classification of authentication mechanisms was to help the designers and implementers to realize when the authentication mechanisms they provide belong to the same or different classes. The IAB recommends designers to resist the pressure of supporting multiple authentication mechanisms that essentially belong to the same class to the EAP framework. Also the IAB recommends designers to resist the pressure of supporting legacy authentication mechanisms due to increased risk of complexity and interoperability problems. Extensible authentication protocol architecture is very flexible because we can use any authentication techniques known as extensible authentication protocol methods (Fig. 1).

1.2 Generic Security Services Application Program Interface

GSS-API [10] has defined a bunch of standard functions both for server and user applications. GSS context does two things i) mutual authentication between RP and client, ii) identity information will be given to RP of the user, and credentials for "AA authentication and authorization". In RP's viewpoint, GSS-API performs access control and EAP performs the authentication process. Using GSS-API EAP packets are transported between RP and Client for this purpose GSS-API mechanism is defined by ABFAB for

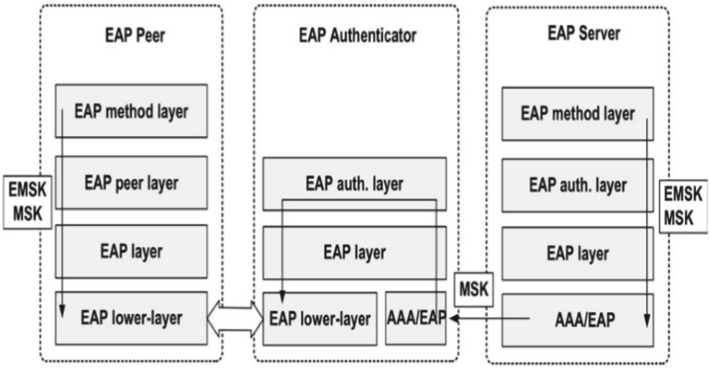

Fig. 1. Architecture of Extensible authentication protocol

EAP known as (GSS-EAP) it also defines how EAP packets sent over GSS tokens and how credentials (keying materials.

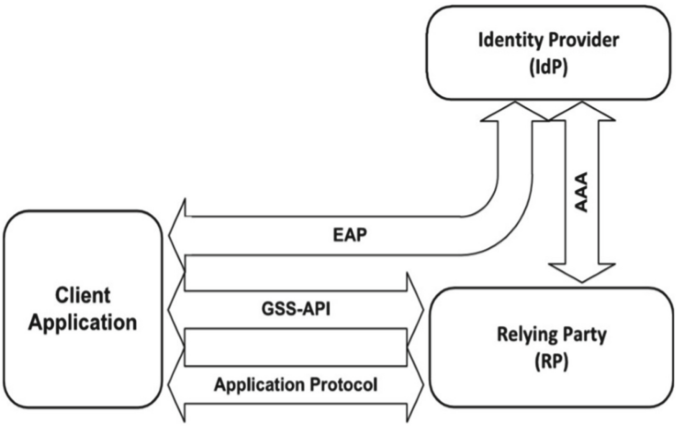

Fig. 2. ABFAB protocol and entities

1.3 Moonshot

Moonshot is based on AAA and it is separated into a few modules, the important and applicable is the one that actualizes the "GSS-EAP" system. The moonshot has 2 unique libraries.The EAP authenticator and EAP peer are implemented in the library named libeap. The libeap [14] has been modified to broaden the EAP, consolidating the advances and usefulness related to REAP. GSS-EAP methods functionality is implemented in the library mech_eap. For the EAP functionality, mech_eap uses the libeap library.

1.4 FreeRadius

FreeRadius is utilized to execute as a home AAA server (IdP), which does the EAP confirmation among clients and connects the cloud administration using AAA based on FreeRadius and it has a lot of modules. To help EAP strategies, In the EAP module of free-range (called rlm_eap) all the usefulness of EAP exists. Little changes are made to this module to help the EAP-REAP and use related to EAP methods.

1.5 OpenStack

Openstack is made up of different modules, each module provides different cloud services like virtual machines, object storage, file (swift services), and networking. Using HTTP RESTFUL APIs all the modules will communicate with each other for providing services. Authentication and authorization and identity management for the cloud are provided by the module called KEYSTONE. Keystone supports a variety of user credentials including X.509 and username/passwords. It also supports different authentication technologies (LDAP, Kerberos, etc.). Federated user credentials received are plotted to "OpenStack" roles and clusters for authorization purposes. Figure 2 shows how to access cloud services with ABFAB based validation provided by the OpenStack. In this process user first contacts the keystone and gets the unscoped token. This keystone is protected by the apache server and receives the authentication request and passes it to the authentication module(mod_auth_kerb). Then a legacy EAP authentication is performed between the user and the Identity provider, in this process RADIUS and GSS-EAP is utilized to pass the packets between them over the apache server. As soon as the user is authenticated, initially, the HTTP authentication request is allowed by the apache server to reach the keystone. Which thus produces and gives the unscoped token to the user.

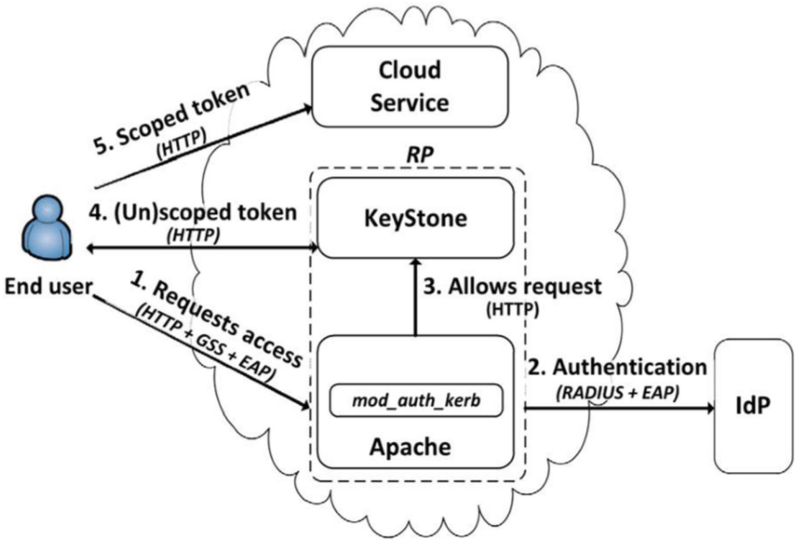

Fig. 3. Working of openstack

Lastly, the user can use the issued "token to request "supplementary" scoped" tokens from the keystone. This scoped tokens are used by the keystone for the authentication (Fig. 3).

2 Literature Survey

2.1 Related Works and Comparisons of Different AAA Mechanisms

In this paper, we are comparing EAP methods with the REAP method. We are comparing the requirements specified in RFC 4017, key properties, and maintenance of certificates. The disadvantage of EAP MD5 is, it is vulnerable to man-in-the-middle attack and dictionary attack [18]. EAP MD5 will not afford session key generation and mutual authentication.

Most of the EAP methods based on certificates. These methods mainly rely on certificates. The EAP method known as EAP-FAST provides both session key generation and mutual authentication. EAP-FAST is also not prone to MITM and dictionary attacks. Another EAP method is known as EAP-LEAP is vulnerable to dictionary attacks [23].

User identity protection, During the authentication process user's credentials, should be secured, for this purpose users' credentials are encrypted in the authentication process. To secure users' identity and credentials "EAP methods" like "EAP TLS, EAP TTLS, EAP PEAP, and EAP Fast" establishes a secure tunnel. Users' credentials are encrypted and transmitted via the secure tunnel therefore users' identity/credentials are hidden by these EAP methods.

User identities/credentials are also protected in EAP TLS and EAP SEM since they use TLS tunnels for hiding the user's identity. EAP-MD5, EAPLEAP, and EAP-SPEKE do not establish secure tunnels so these methods do not hide user identity/credentials.

[19] discussed how EAP TLS secure user credentials/identity by tunneling. Moreover, the first EAP method of Juang et al., the EAP method proposed by Park et al., Yoon et al. are prone to the dictionary attack and do not provide hiding of user identity. Hence, their methods do not provide and meet the specification of identity privacy. In REAP user identity is encrypted while exchanging authentication messages so it meets the specification of identity privacy.

The legacy EAP methods use asymmetric cryptographic algorithms such as RSA, DH, for key computation at both the server and the client sides which require more CPU cycles for computing keys and timeconsuming also decrease the performance. REAP method uses only symmetric encryption/decryption without asymmetric ones, Therefore, the computation cost of REAP is reduced by depending on different types of EAP methods as associated to the other EAP methods that have the same level of security as REAP.

(REAP) Polynomial based keys are the sequence of keys that are computed using one-time symmetric key cryptography. In this technique, all the messages are encrypted or decrypted by using the sequence of keys generated. Since all the messages are encrypted or decrypted in this technique, if an attacker uses the compromised key, then it can be easily detected. Unlike sharing the keys to entities, in this technique, the keys are generated effortlessly using polynomial expression each time and used for encryption and

decryption. In session-based keys generation [21], for each session, separate keys will be generated and exchanged with the entities. In this technique, there is no concept of exchanging the key nor key trade in each session. Both timestamp and polynomial expressions are used to compute/generate the chain of keys used for encryption/decryption. The timestamp is passed as a parameter by the client to server and at both the end, polynomial expressions exist, which is used to compute the sequence of keys to encrypt/decrypt the messages.

That is how our technique utilizes the idea of Polynomial based keys When the succession of Polynomial based keys is spent, another succession of Polynomial based keys is produced by using Polynomial expression. Both the server-side and the user side performs the same procedure (Figs. 4 and 5).

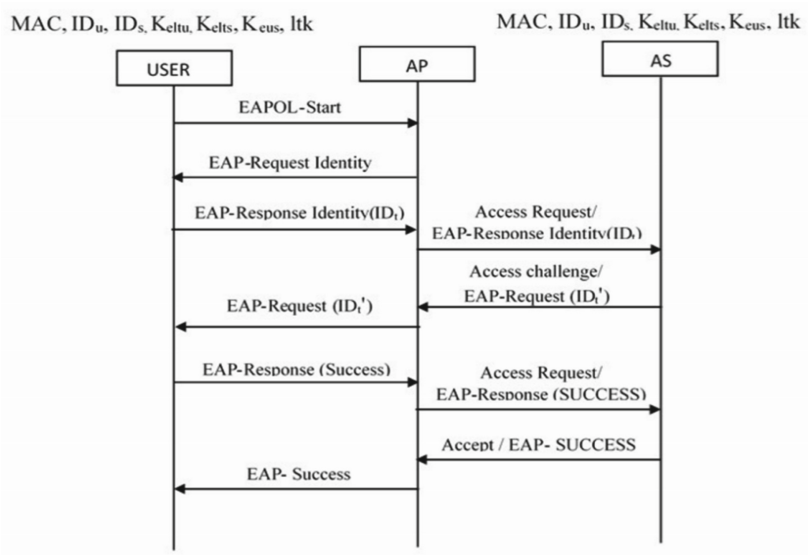

Fig. 4. REAP Authentication process

- Related works of EAP methods used in cloud scenario for AAA.

In the cloud scenario, the services provided by the cloud will be secured and access control is controlled by the access control mechanisms. Different cloud solutions will use different access control mechanisms and security mechanisms.

For example, OpenStack [15] supports numerous kinds of user credentials including X.509 and mechanisms for authentication like AAA. Cloud computing is now has the interest for access control as an approach to streamlining client connections to moderate the client the board exertion. OpenStack has incorporated the OS_FEDERATION augmentation [15], which allowed federated clients by performing the assignment of roles and groups all together with cloud access. It also provides a feature of federated access through which an efficient and secure authentication from the industry

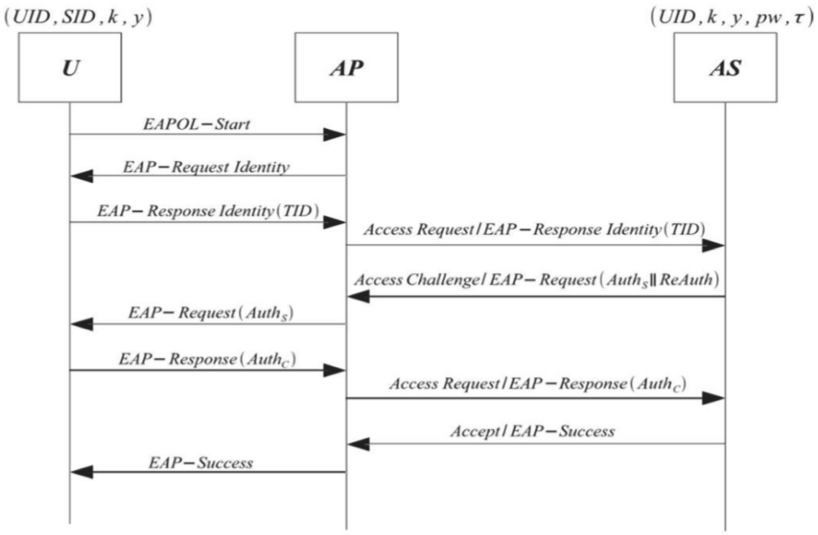

Fig. 5. EAP method proposed by [8]

and R&D for a long time. The advantages of secure and efficient authentication are, it gives regarding expanded convenience for clients (users need not have to present their certificates/credentials every time in the authentication process), too as the eminent decrease on the multifaceted nature of the verification the measure required when accessing to the cloud services have made a significant "Identity Management" (IDM) theme. From the individual perspective, a large portion of the secure, efficient, and SSO-empowered access control mechanisms are just centered around applications of the web. "SAML" [10] and "OAuth" [18] are outstanding models, permitting clients and services of applications to access various resources.

Some other important web services are "OpenID" [16], "OpenID Connect" [16] these have been only essentially framed for applications of web.

Kerberos [17] gives a conventional access control convention, in light of the circulation of confirmation tickets, that is broadly upheld by numerous applications & that gives an authentication and "SSO" feature. Even though the standard doesn't especially care about security, there are a few propositions, for example, PrivaKerb or KAMU, that give augmentations to supporting improved security. Moreover, Kerberos underpins an activity mode, "Kerberos cross-domain" organizations has not been generally conveyed because of some perceived issues, just as to the reality of establishing an autonomous foundation aside those effectively settled for the access to web applications (for example "SAML-based") & access to the network (for example "AAA-based").

Based on the literature survey we are comparing with the authentication methods (EAP-methods) that are secure, efficient, and recent technology proposed by different authors. We are comparing these methods with the REAP method integrated to access cloud services. Also, performance analysis and results are discussed below (Fig. 6).

Fig. 6. Proposed System

3 Proposed System

The cloud services are provided to the user, for example, SaaS (e.g. Openkm) using OpenStack. The "application service supporting ABFAB employing the extension of GSS-EAP". The components in the proposed system are listed below.

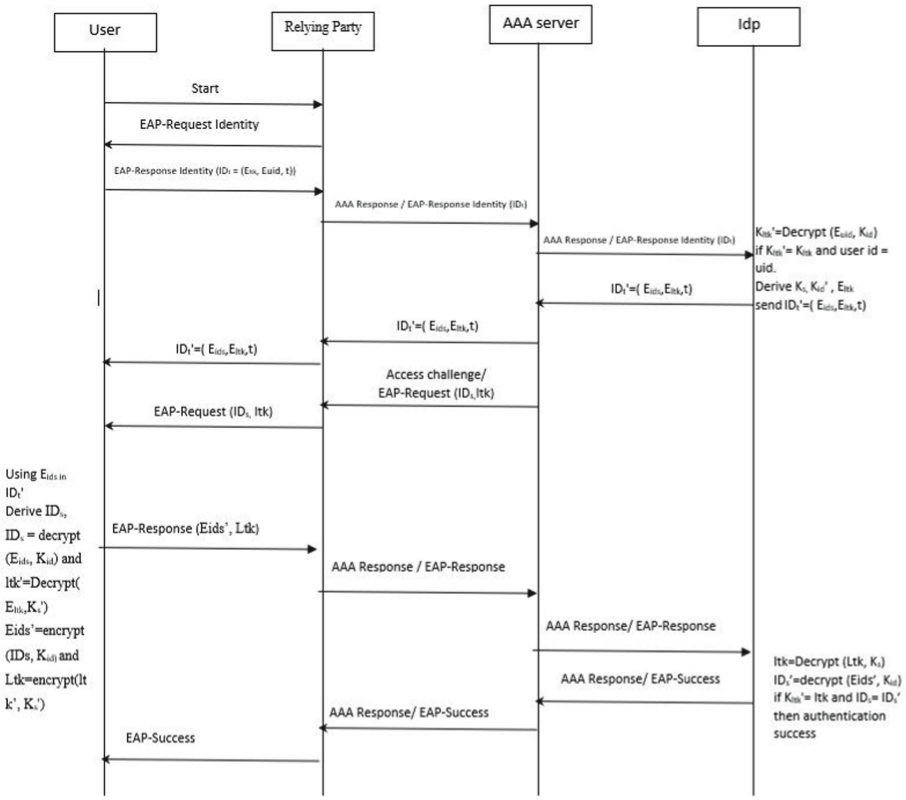

Fig. 7. Working of the proposed system

- "User" (U). The user is an entity who is intent on using the cloud services provided by the cloud service provider. The user will use his username/password for authentication by the AAA infrastructure and with EAP methods.
- "Identity Provider (IdP)". IdP uses a username/password to verifies the user, and it will act as an "AAA server", known as "IdP in ABFAB".
- "Cloud service/Relying Party (RP)". Also known as RP provides cloud services to the User. GSS-EAP manages access control (Fig. 7).

1) Start: in this first step both the user and the sever generates the secret key to encrypt all the messages transmitted between user and server. When user requires to access the cloud service EAP initiates the full EAP authentication process.
2) After the EAP initiation the relaying party requests for the identity of the user by encapsulating the eap request identity message in EAP-Request packet.
3) Relaying party acts as an authenticator between user and server sends the identity along with the encrypted keys (IDt) to the authentication server (AAA sever).
4) IdP identity provider extracts the encrypted keys in (IDt), computes the text by decrypting, K_{ltk}' = Decrypt(E_{uid}, K_{id}) and compute the ID of the user by decrypting, uid = Decrypt(E_{uid}, K_{id}). After computing K_{ltk}', $K_{ltk,}$ and the userid, if K_{ltk}' = K_{ltk} and user id = uid then it computes K_s = K_{elts} (t), computes K_{id}', K_{id}' = K_{eus}(t), compute E_{ids}, where E_{ids} = Encrypt(ID_s,K_{id}) and computes E_{ltk}, E_{ltk} = Encrypt(ltk,K_s). Then server sends ID_t', ID_t' = (E_{ids},E_{ltk},t) to AAA server and AAA server sends to RP.
5) Relaying party sends this encrypted key to user.
6) After receiving the keys, the user computes K_{id}', K_{id}' = K_{eus}(t), computes K_s' = K_{elts} (t), computes id of server ID_s, ID_s = decrypt (E_{ids}, K_{id}) and computesthe longtermkey ltk' = Decrypt(E_{ltk},K_s'). If longtermkey(ltk') and server-id are the same then both are mutually authenticated. The server sends EAP success to the RP.
7) Rp sends EAP-success to the user. User is now authenticated and allowed to access the service provided by the server in this case it is an application named (OpenKM).

4 Testbed Configuration

In the testbed, the entities are deployed using VM and the entities used in this testbed are listed below.

- "FreeRADIUS" is open-source software that is installed in the VM.
- IdP (RADIUS server). IdP handles the moonshot.test.in the realm and executes a "FreeRADIUS" instance.

The indicators A and B are measured for the performance analysis, elements A and B are the components: User, RP, and IDP.

- ATC: the time consumed by component A to accomplish the necessary process. Network delays are excluded in this indicator.

- ATW: Total time spent by entity A for requests sent and waiting for the responses. It is calculated by the value as the total time spent in element A amongst transmitting the request message till the response message is received for the request.

ATD(A, B):- The time taken(spent) to deliver the messages between components A and B. ATD(A, B) = ATW(A) − ATC(B) − ATW(B).

TT:- The total time needed by the user (U) to access the services. TT = ATC(U) + ATD(U, RP) + ATC(RP) + ATD(RP) + ATD(IDP) + ATC(IDP). It is similar to ATC(U) + ATW(U).

- ADB(A, B). Amount of data in bytes that is communicated between elements A and B. the calculation of this indicator is calculated by taking the size of the responses or requests i.e. communicated between elements A and B.
- TAT = ADB (U, RP) + ADB(RP) + ADB (IDP).

To measure these indicators, we have used the open-source tool Wireshark to analyse and capture packet movements network (network traffic) of all the entities in the process of accessing cloud services. The data in this file are the messages captured during the accessing services. It also provides the size, type of packet (RADIUS), and timestamp for each message that has been sent and received. These files are analysed by using python scripts that extract and read all the lines and also calculates values for all the indicators with respect to time spent between messages.

5 Results

The results are obtained by executing the following testbed configurations.In particular, we have executed testbed configuration with legacy EAP method used in moonshot and with configuration using REAP method. Since virtual machines are used to configure our testbed, the challenges and performance considerations that have been mentioned in [22] apply. Before dissecting the outcomes, it should be noted that EAP authentication is only the process of accessing the cloud services (Openstack) other processes are the creation of OpenStack tokens and GSS-API packet encapsulation in keystone messages.

In Table 1, both Moonshot with legacy EAP method and the Moonshot with REAP configurations shows the overall time (TT) authentication time of the 3 methods discussed above. The time required for all three methods is shown in Table 1. In the initial authentication process, it is required to execute the complete authentication process "(EAP authentication) with the identity provider". Execution of the authentication method (REAP authentication process) reduces the authentication time to 3790.7ms approximately 30% compared with the other proposed authentication methods. The reduction of overall time includes processing time and accessing the OpenStack cloud services. In Table 1 and in Table 2 we can observe that by the use of the REAP authentication method the total amount of authentication time3790.7ms and data transmitted (TAT) is 89860.9ms21% to 30% reduction is achieved when compared with other methods.

Table 1. Performance based on Authentication time

Testbed Configuration	Taufik Nur, Hidayat Imam Riadi et al. [15]	Mohamed A. Abo-Soliman et al. [5]	Yin-Hui et al. [1]	Our method (REAP)
ATC(U)	1292	1346.7	979.2	758.5
ATD (U, RP)	2466	2667.26	2446	2007.2
ATC(RP)	1154.4	1248.1	1231	1013.9
ATD (RP, IDP)	13.6	19.8	12.3	11.1
Total authentication time taken (TaT)	4926	5281.86	4668.5	3790.7

Table 2. Results based on authentication traffic

Testbed Configuration	Taufik Nur, Hidayat Imam Riadi et al. [15]	Mohamed A. Abo-Soliman et al. [5]	Yin-Hui et al. [1]	Our method for (REAP)
ADB (U, IDP)/ TAT	91323.5	92313.2	90290.6	89860.9

6 Conclusions

The proposed work expects to give security and an optimized / efficient authentication process to access (SaaS) cloud service and also in the situations of federated access control environments where ABFAB is used, reducing the authentication data traffic. Also, by the results of the proposed model we can observe the decrease in authentication time needed to access the cloud service (SaaS) implemented using OpenStack. After obtaining the result by the performance analysis, the work proposed will be able to reduce the authentication time by the use of the REAP method to access the cloud services around 3790.7 ms. In future work, we try to implement the proposed method in Dockers and containers environment.

References

1. Pimple, N., Salunke, T., Pawar, U., Sangoi, J.: Wireless security — An approach towards secured Wi-Fi connectivity. In: 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)
2. Dictionary Attack on Cisco LEAP, http://www.cisco.com/warp/public/707/cisco-sn-20030802leap.shtml (2018)
3. Raghavendra, K., Ramesh, B.: Managing the digital identity in the cloud: the current scenario. In: 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) (2015)

4. Dierks, T., Rescorla, E.: The TLS Protocol Version 1.2. RFC 5246 (August 2015)
5. Cam-Winget, N., McGrew, D., Salowey, J., Zhou, H.: The flexible authentication via secure tunneling extensible authentication protocol method (EAP-FAST). RFC 4851 (May 2014)
6. Alexandra, C., Laura, G., Daniel, R.: A practical analysis of EAP authentication methods. In: Proc. 9th Roedunet International Conference (RoEduNet), pp. 31–35 (2017)
7. Fan, C.I., Member, IEEE, Lin, Y.-H., Hsu, R.-H.: Complete EAP method: user efficient and forward secure authentication protocol for IEEE 802.11 wireless LANs. IEEE Trans. Parallel Distrib. Syst. **24**(4) (APRIL 2016)
8. Hutzelman, J., Salowey, J., Galbraith, J., Welch, V.: Generic security service application program interface (GSS-API) authentication and key exchange for the secure shell (SSH) protocol. IETF RFC 4462 (May 2006)
9. Smith, R. (ed.): Application bridging for federated access beyond web (ABFAB) use cases. IETF RFC 7832 (May 2016)
10. Mamidisetti, G., Makala, R., Anilkumar, C.: A novel access control mechanism for secure cloud communication using SAML based token creation. J. Ambient. Intell. Hum. Comput. (2020)
11. Configuring keystone for federation. http://docs.openstack.org/developer/keystone/configure_federation.html
12. Li, W., Mitchell, C.J., et al.: User access privacy in OAuth 2.0 and OpenID Connect. In: 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). https://doi.org/10.1109/EuroSPW51379.2020
13. Raghavendra, K., Nireshwalya, S.: Application layer security issues and its solutions. **2**(6) (2012)
14. Zhang, Y., Zhou, M., Stol, K.-J., Wu, J., Jin, Z.: How do companies collaborate in open source ecosystems? An empirical study of openstack. In: 2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)
15. Bilal, M., Wang, C., Yu, Z., Bashir, A.: Evaluation of secure OpenID-Based RAAA user authentication protocol for preventing specific web attacks in web apps. In: 2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)
16. K Balakrishna F Mohammed CR Ullas CM Hema SK Sonakshi 2021 Application of IOT and machine learning in crop protection against animal intrusion Global Transitions Proceedings 2 2 169 174
17. Chen, C.M., Chang, T.-H.: The cryptanalysis of WPA & WPA2 in the rule-based brute force attack, an advanced and efficient method. In: 2015 10th Asia Joint Conference on Information Security. IEEE (2015)
18. Adnan, A.H., Abdirazak, M., Shamsuzzaman Sadi, A.B.M., Anam, T., Khan, S.Z., Rahman, M.M.: A comparative study of WLAN security protocols: WPA, WPA2. In: 2015 International Conference on Advances in Electrical Engineering (ICAEE) (2015)
19. Sood, P., Taveggia, D.: From General ESL to EAP A Fall Leap", Vol. 8 (2019): Proceedings of the conference "Building Bridges for English Language Centers", (2019)
20. Khadem, B., Abedi, S., Sa-adatyar, I.: An idea to increase the security of EAPMD5 protocol against dictionary attack. In: 3rd International Conference on Combinatorics, Cryptography and Computation, December 15, 2018 in IUST, Iran (2018)
21. Younes AsimiEmail authorAhmed AsimiAzidine Guezzaz, "Robust Cryptographical Applications for a Secure Wireless Network Protocol", 2020
22. WS Juang JL Wu 2009 Two efficient two-factor authenticated key exchange protocols in public wireless LANs Comput. Electr. Eng. 35 1 33 40
23. K Raghavendra B Ramesh J Chandrika 2021 Secure, efficient and certificateless authentication scheme for wired and wireless networks Int. J. Adv. Res. Eng. Technol. (IJARET) 12 2 167 175

24. Raghavendra, K., Ramesh, B., Chandrika, J.: A secure and efficient authentication technique to access cloud services in Openstack. Int. J. Innov. Res. Technol. **8**(5), 167–175
25. Raghavendra, K., Nireshwalya, S.: Application layer security issues and its solutions. Int. J. Comput. Sci. Eng. & Technol. **2**(6) (2012)