ABFAB
Internet-Draft
Updates: RFC7055 (if approved)
Intended status: Experimental
Expires: April 30, 2015

A. Perez-Mendez
R. Marin-Lopez
G. Lopez-Millan
University of Murcia
F. Pereniguez-Garcia
Catholic University of Murcia

October 27, 2014

ERP extensions for the ABFAB architecture
draft-perez-abfab-wg-arch-erp-00

Abstract

The Application Bridging for Federated Access Beyond Web (ABFAB) architecture [I-D.ietf-abfab-arch] allows the use of EAP to perform access control to a wide range of applications. This document describes the extensions required to incorporate support for the EAP Extensions for the EAP Re-authentication Protocol (ERP) to this architecture, in order to provide an enhanced fast re-authentication and Single Sign-On (SSO) support and a better resource utilization. Authorization aspects are also described.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 30, 2015.

Table of Contents

1. Introduction

The Application Bridging for Federated Access Beyond Web (ABFAB) architecture [I-D.ietf-abfab-arch] allows the use of EAP to perform access control to a wide range of applications. This is mainly achieved by the definition of the GSS-API Mechanism for the Extensible Authentication Protocol [RFC7055], which can be used with any application that provides support for the GSS-API or the SASL framework (through the GS2 family of mechanisms). Since EAP is mostly used in conjunction with a backend Authentication, Authorization, and Accounting (AAA) infrastructure, the use of this mechanism automatically brings federated authentication capabilities to these applications. Besides, the ABFAB architecture also defines how attributes about the Client can be transported from the home organization to the application in order to perform an authorization process based on this federated identity information. Hence, altogether the ABFAB architecture provides a consistent federated access control framework usable in multiple scenarios, as described in [I-D.ietf-abfab-usecases].

However, neither EAP nor the GSS-EAP mechanism provide any particular means of performing fast re-authentication. This implies that, whenever a Client accesses to different applications, regardless they are deployed in the same organization or in different ones, a complete EAP authentication process must be performed for each one of them. Depending on the selected EAP method (e.g. EAP-TTLS, PEAP, etc.), this exchange may consist of several roundtrips executed between the Client and her Home organization, with the inherent costs in terms of network bandwidth consumption and computational resources used (e.g. to perform asymmetric cryptography operations).

Therefore, in order to provide a better resource utilization, Clients, organizations, and applications using EAP have to rely on the fast re-authentication capabilities provided by the selected EAP method, when possible. There are a number of EAP methods that already provide some fast re-authentication capabilities (e.g. EAP-

AKA, EAP-TTLS...). However, although they can achieve a reduction of the required round-trips to complete the authentication process, they all require the execution of at least two round-trips in the best case. Besides, these exchanges are always performed between the Client and the Home AAA server.

The EAP Extensions for the EAP Re-authentication Protocol (ERP) [RFC6696] modifies certain aspects of the EAP protocol to 1) allow the execution of a re-authentication process in a single round trip; and 2) allow performing the re-authentication process in a local fashion whenever the Client repeatedly accesses to different applications within the same organization. In particular, ERP allows

the Client and an AAA server (either in the home or in the visited organization) to mutually verify proof of possession of key material from an earlier EAP method run. Although ERP, as EAP, was originally conceived to be used only in the network access context, the ABFAB WG extended the EAP applicability statement [RFC7057] to allow its use for generic application authentication. This fact also enables ERP to be used to provide fast re-authentication for generic applications.

The extensions defined by ERP consist of a pair of new EAP codes, and some updates to the EAP state machines required to generate and process these codes accordingly. Therefore, adding support for ERP implies updating the EAP lower layer specifications. This document describes the modifications to the ABFAB architecture that are required to incorporate support for the EAP Extensions for the EAP Re-authentication Protocol (ERP).

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119]. When these words appear in lower case, they have their natural language meaning.

2. Applicability considerations

Although EAP, and thus ERP, was originally conceived to be used only

in the network access context, the ABFAB WG extended the EAP applicability statement [RFC7057] to allow its use for generic application authentication. This fact also enables ERP to be used to provide fast re-authentication for generic applications when used in combination with the GSS-EAP mechanism.

3. Terminology

Although this document is not intended to serve as an ERP or ABFAB guide, this section provides some terminology notes to help the reader to follow the rest of this document.

- Client. The entity which wants to access a particular service by means of ABFAB technologies. It plays the role of GSS-API initiator and EAP/ERP peer.

- RP (Relying party). The application server the Client is trying to access to. It plays the role of GSS-API acceptor and EAP/ERP

authenticator.

- Visited organization. The organization where the RP is deployed.

- Home organization. The organization where the Client has a registration.

- Visited AAA server. The AAA server that connects the Visited organization to the AAA federation. It plays the role of AAA proxy and of ERP server for the ERP local operation (see below).

- Home AAA server. The AAA server that connects the Home organization to the AAA federation. It plays the role of EAP server and of ERP server for the bootstrapping processes (see below).

- ERP keys. The keys used to perform authentication in ERP. They are named rIK/rRK when shared with the Home AAA server, and DS-rIK/DS-rRK when shared with the Visited AAA server. They are derived during the bootstrapping processes (see below). Besides, an rMSK is derived from them to play the role of a typical MSK, being installed in the RP.

- ERP Bootstrapping. When the Client tries to use ERP with a particular RP, but she does not share any ERP key with the Visited organization, a bootstrapping process is required to derive and install such keys.

  * Implicit bootstrapping. It is performed when the Client does not share any ERP key with the Home AAA server either. The Client authenticates with the Home AAA server by means of an EAP method. ERP keys for the Home and Visited organizations are derived from the cryptographic material generated by this method, and installed in the Home AAA server and Visited AAA server respectively.

  * Explicit bootstrapping. It is performed when the Client shares ERP keys with the Home AAA server from a previous implicit bootstrapping process. The authentication consists on a single round-trip ERP exchange where both entities prove their knowledge of these keys. New ERP keys for the Visited organization are derived and installed in the Visited AAA server.

- ERP local operation. When the Client tries to use ERP with a particular RP and she shares ERP keys with the Visited organization, the ERP authentication is performed locally by the Visited AAA server in a single round-trip exchange.

4. Motivation use case

As an example use case let us picture the following situation. Alice (Client) has a subscription with an organization (Home organization). This means that Alice has some long-term credentials that allow her to authenticate with this organization. Let us assume there are also other organizations, called Organization A and Organization B (Visited organizations), which have a set of services deployed (RPs). These three organizations are interconnected through an AAA federation, and the services they offer support the use of the ABFAB technologies for access control.

What Alice and the organizations would want is to allow her to access to any of the services deployed by the different organizations without requiring her to perform a complete EAP authentication for

each one of them. Instead, only one complete EAP authentication would be required, at the beginning of the session. The rest of authentication processes would be based on the cryptographic material derived from it, regardless whether the organization of the RP being accessed is different from the one where the first EAP authentication process was performed. Moreover, whenever possible, the authentication process should be performed without involving the home institution.

In this way, Alice avoids being prompted to introduce her long-term credentials once and again, and goes through the process as fast as possible to start enjoying the service. On the other hand, organizations are able to reduce the amount of network traffic exchanged between them, as well as the computational cost of performing this process. This should improve scalability and reduce infrastructure costs.

In this scenario, using the standard mechanism defined within the IETF to provide fast re-authentication in EAP (i.e. ERP) seems to be the most reasonable way to proceed. This solution would be valid not only for any application making use of the ABFAB architecture or the GSS-EAP mechanism.

This use case is directly applicable within the context of the CLASSe project [CLASSe], which focuses on allowing GEANT users to access to cloud services using their home institution credentials. Although this objective is mainly achieved by the use of ABFAB technologies, the project is also interested in ways to improve the SSO capabilities of the ABFAB architecture, in such a way that a Client moving among a number of different cloud services deployed within the same federation do not require performing more than one complete EAP authentication process.

5. Modifications to the ABFAB architecture

The integration of ERP with the ABFAB architecture requires of some changes to the GSS-EAP mechanism, some modifications to the authorization handling, and some updates to the back-end (AAA) infrastructure. The following subsections describe them with further detail. Section 6 provides a detailed flow based description of the overall ABFAB/ERP process.

5.1. ERP extensions to GSS-EAP

ERP follows a different exchange scheme than EAP. Whereas a typical EAP exchange is a server-initiated flow, ERP may follow a client-initiated one. Therefore, supporting ERP would require changes on how these lower layers deal with re-transmissions. However, the GSS-API (and thus the GSS-EAP mechanism) operation also follows a client-initiated scheme. This aspect favors the integration of ERP into its message flow, minimizing the number of adaptations required to incorporate the ERP functionality. Specifically, this allows an easier negotiation of ERP capabilities and parameters between the GSS-EAP initiator (Client) and the GSS-EAP acceptor (RP), and a simpler handling of error conditions (e.g. re-transmissions).

Following subsections describe how the different states of the GSS-EAP mechanism must be adapted to enable ERP functionality.

5.1.1. GSS-EAP initial state

As defined in [RFC7055], the Initial State is used to start the context establishment process. In particular, it is used to exchange information such as vendor information or acceptor (RP) name. This document defines a new Subtoken, called ERP-Supported Subtoken, that is exchanged in this state, and it is used to negotiate ERP-related parameters. It MUST be included by any compliant Client willing to start an ERP exchange. The presence of this Subtoken indicates that the Client supports this specification, and the kind of ERP exchange to be performed. Section 5.1.1.1 provides further details on this Subtoken.

When the Client sends an ERP-Supported Subtoken indicating that an implicit bootstrapping is requested, a compliant RP MUST include an ERP-Supported Subtoken in the response, indicating that it supports ERP and the realm name to be used to derive the ERP keys. As the Subtoken is not marked as critical, non-compliant RPs will ignore such Subtokens and continue as indicated in [RFC7055].

On the contrary, when an explicit bootstrapping or an ERP local operation is requested, the RP MUST start the ERP exchange by

including an EAP-Initiate/Re-auth-Start packet within the first EAP

Request Subtoken. This packet is sent instead of the EAP request/ identity one. The reception of this ERP packet will notify the Client that the RP supports the ERP extensions, and provide the realm name to be used to derive the ERP keys.

Although ERP allows the Client to send an EAP-Initiate/Re-auth packet without having received any EAP-Initiate/Re-auth-Start packet first, this behavior is not allowed in this specification for two main reasons. First, the Client might not know the ERP realm of the application yet, as it could not be inferred from the DNS name or GSS-API Acceptor Name. Indeed, the RP (acceptor) name is being negotiated during this state. Hence, the Client must wait until an EAP-Initiate/Re-auth-start message, or an ERP Supported Subtoken, is received from the RP, as explained above. Besides, during the GSS-EAP initial state, neither the Client nor the RP are expected to perform any authentication processing. That belongs to the authentication state (see below). Allowing authentication in this state will require deeper changes to the GSS-EAP state machine, resulting on a more complex specification.

5.1.1.1. ERP-Supported Subtoken

The ERP-Supported Subtoken is sent from the Client to the RP, indicating that the former wishes to start an ERP exchange with the latter, and specifying whether it will be an implicit bootstrapping or not. Besides, it can be sent from the RP to the Client whenever an implicit bootstrapping has been requested by the Client, in order to 1) indicate that ERP is supported; and 2) to provide the Client with the ERP realm name required to derive they ERP keys at the end of the EAP authentication process.

The ERP-Supported Subtoken has the structure depicted in Figure 1

Pos	Description
0..3	TBD (not critical)
4..7	length of token
8	implicit bootstrapping? (0x00=no, 0x01=yes)
9..8+length	Visited organization's realm (sent by RP only)

Figure 1: ERP-Supported Subtoken

5.1.2. GSS-EAP authentication state

As defined in [RFC7055], this state is used to perform the exchange of EAP packets between the Client and the RP. When performing an ERP implicit bootstrapping, a compliant Client MUST derive and store the ERP keys associated with its Home organization for later use (i.e rRK and rIK). Besides, it MUST derive and store the ERP keys associated to the Visited organization whenever the latter indicates support of ERP (i.e. DS-rRK and DS-rIK). Both derivation processes are performed after having completed a successful EAP authentication.

Besides, during an implicit bootstrapping, a compliant RP SHOULD notify the Visited AAA server whenever an ERP implicit bootstrapping process has been requested. This notification can be performed by including an AAA attribute, such as the ERP-Realm one defined in [RFC6942], to the first request sent to the Home AAA server. This would trigger the Visited AAA server to request a DSRK from the Home AAA server. Without this notification, an ERP-capable Visited AAA server would need to send a DSRK request for every EAP authentication, regardless the Client supports ERP or not. This would result into the generation of an amount of useless state in the Visited and Home AAA servers when the Client does not support ERP.

When performing other types of ERP exchanges (i.e. explicit bootstrapping or local operation), this specification adds an additional way to end the EAP conversation to the ones listed in [RFC7055]. In particular, when an ERP exchange is successfully executed, the conversation will end with the generation (and reception) of an EAP-Finish/Re-auth message. The behavior at GSS-EAP layer associated to this message is identical to the one associated with the EAP Success message. It also includes the derivation of the ERP keys associated with the Visited organization, and the rMSK derived from them by the Client. Note that the rMSK is used as the MSK for all purposes, including the derivation of the GSS-API Master Session Key (GMSK) defined in [RFC7055].

5.1.3. GSS-EAP extensions state

This state requires no modifications, and it can be performed as described in [RFC7055].

5.2. Authorization Considerations

Another important aspect has to do with how authorization should be handled in ABFAB when ERP is in use. The ABFAB architecture defines that the Home AAA server generates a SAML assertion with information

about the Client, and delivers it to the RP, after the EAP authentication has been successfully completed

[I-D.ietf-abfab-aaa-saml]. Whenever an ERP bootstrapping process (either implicit or explicit) is performed, no changes are required to this behavior. That is, when the Home AAA server generates the EAP-Finish/Re-auth message, it also generates and delivers the SAML in the same way it would do with an EAP-Success message.

However, there is an important consideration when it comes to the ERP local operation. Since the Home AAA server is not involved in the process, in principle, an SAML assertion is not generated as specified in GSS-EAP. To overcome this limitation, and to take advantage of the ERP local operation, the Visited AAA server SHOULD store the SAML assertion received during any of the bootstrapping processes (either implicit or explicit) executed previously, associated to the Client's identifiers (e.g. User-Name, CUI...). In this way, the stored SAML assertion can be delivered to the RP during the ERP local operation.

5.3. Updates to the backend (AAA) infrastructure

Another architecture element that needs to be updated, in order to add support for ERP to the ABFAB architecture, is the backend AAA infrastructure. Basically, the Home AAA server and the Visited AAA server need to implement ERP as described in [RFC6696]. In particular, the Home AAA server needs to be updated to support the different ERP bootstrapping exchanges. This includes the generation and parsing of ERP messages, as well as the derivation, storage, distribution and usage of the ERP-related keys (i.e. rIK, rRK, DS-rIK, rMSK...).

In the same way, the Visited AAA server needs to be updated in order to be able to request the required ERP key material (i.e. DSRK) from the Home AAA server during any of the ERP bootstrapping exchanges. This key material needs to be stored for its use during the ERP local operation (see below). Besides, the Visited AAA server MAY need to support the storage of the SAML assertion as described in Section 5.2.

Finally, the Visited AAA server also needs to be updated to support the execution of the ERP local operation with the Client, based on

the key material obtained during any of the bootstrapping exchanges performed before.

When the Visited AAA server does not provide support for ERP, the Client will only be able to perform bootstrapping processes. Although this is not optimal, a typical explicit bootstrapping process will require a single round-trip performed between the Client and its Home AAA server. The main drawback of not having ERP-capable AAA proxies is the impossibility of executing the ERP local

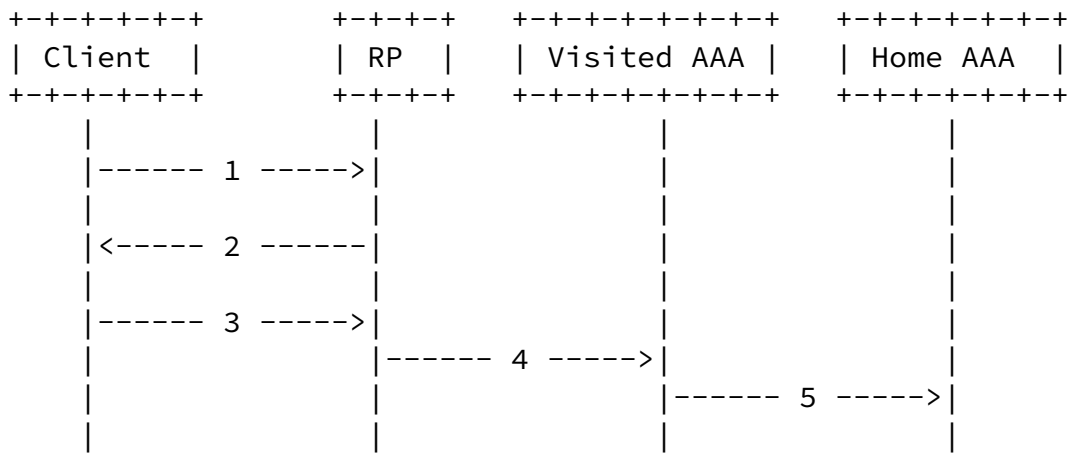
operation, which implies communicating with the home AAA server to perform the authentication.

6. Operation

Previous section has explained the extensions that are required to GSS-EAP for supporting ERP. This section provides a detailed description of how the different ERP exchanges would be executed. In these descriptions it is assumed that all the participants support the ERP extensions described in this document.

6.1. ERP implicit bootstrapping

TBD This ERP exchange is performed when the Client wants to use ERP with a particular RP, but she does not have neither a DS-rRK/DS-rIK shared with the Visited AAA server nor a rRK/rIK shared with her Home AAA server. The process is depicted in Figure 2 and described below.



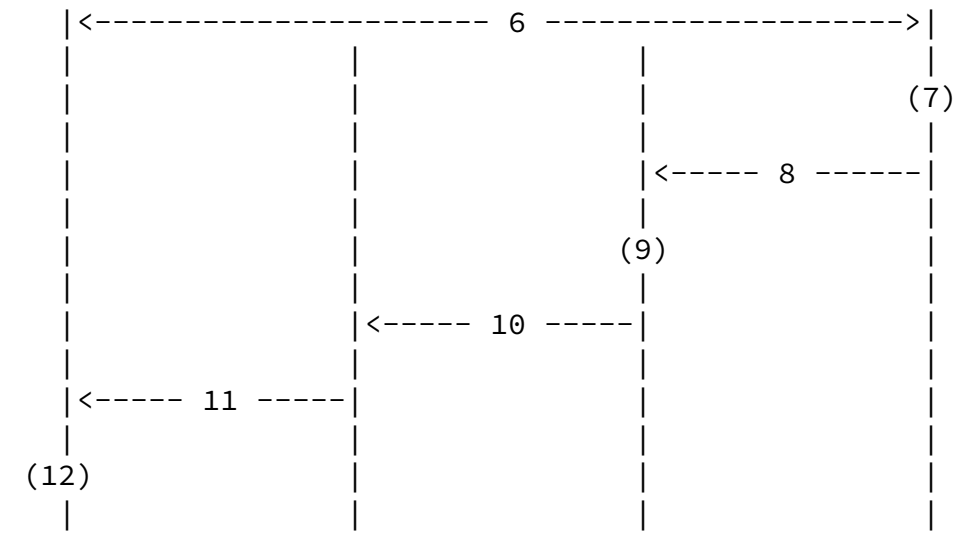


Figure 2: ERP implicit bootstrapping

1. The Client starts the GSS-EAP mechanism, also including an ERP Supported Subtoken indicating an implicit bootstrapping is required.

2. The RP continues with the standard GSS-EAP mechanism, also including an ERP Supported Subtoken indicating implicit bootstrapping and the Visited organization's realm.

3. The Client sends the EAP-Response/Identity message.

4. The RP forwards this EAP message to the Visited AAA server, also including an indication (e.g. ERP-Realm attribute) that an implicit bootstrapping was requested.

5. The Visited AAA server forwards the EAP message to the Home AAA server, also including a request for a DSRK key (used to derive the ERP keys).

6. The Client and the Home AAA run the EAP protocol.

7. The Home AAA server derives the MSK and SAML assertion(as usual). It also generates the rRK and rIK, and the requested DSRK.

8. The Home AAA server sends the EAP-Success, MSK, SAML assertion

and DSRK to the Visited AAA server.

9. The Visited AAA server derives the DS-rRK and DS-rIK from the DSRK, and stores the SAML assertion.

10. The Visited AAA server sends the EAP-Success, the SAML assertion and the MSK to the RP.

11. The RP sends the EAP-Success to the Client and finalizes the GSS-EAP mechanism.

12. The Client derives the MSK, rRK, rIK, DS-rRK, and DS-rIK.

6.2. ERP explicit bootstrapping

TBD

6.3. ERP local operation

This ERP exchange is performed when the Client wants to use ERP with a particular RP, having DS-rRK/DS-rIK shared with the Visited AAA server. The process is depicted in Figure 3 and described below.

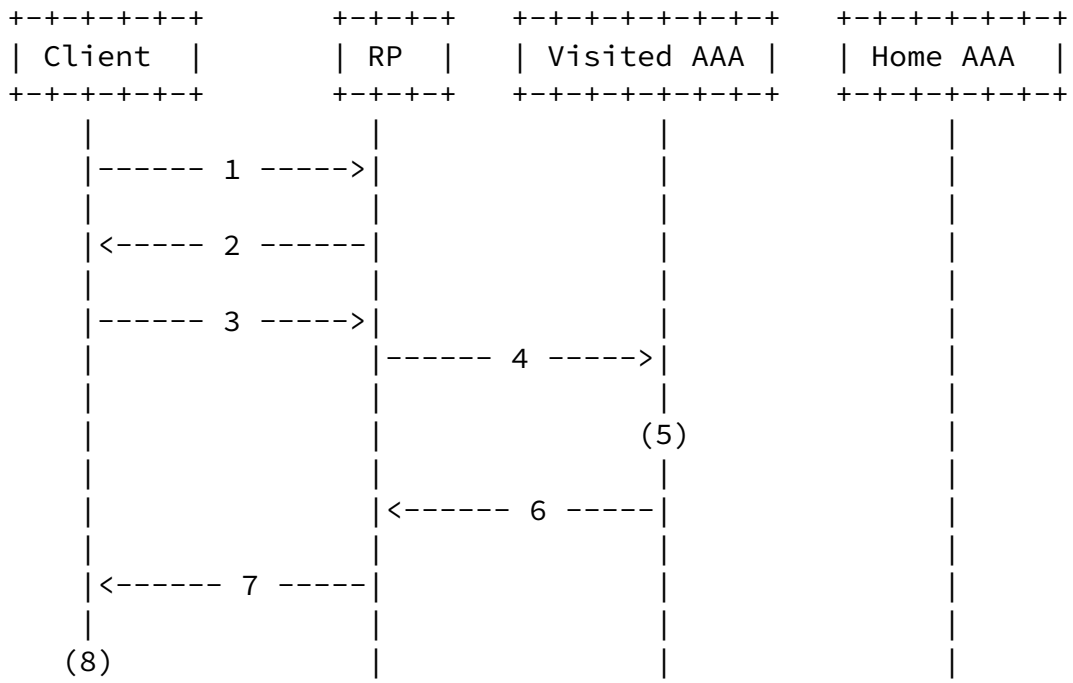


Figure 3: ERP local operation

1. The Client starts the GSS-EAP mechanism, also including an ERP Supported Subtoken indicating an implicit bootstrapping is not required. At this point the Client does not know whether an explicit bootstrapping or an ERP local operation will be performed, as the ERP realm is unknown.

2. The RP continues with the standard GSS-EAP mechanism, sending an EAP-Initiate/Re-Auth-Start, instead of an EAP-Identity/Request, indicating the Visited organization's realm.

3. The Client sends an EAP-Initiate/Re-Auth generated using the available DS-rRK/DS-rIK keys.

4. The RP forwards this EAP message to the Visited AAA server.

5. The Visited AAA server verifies the EAP message using the stored DS-rRK/DS-rIK keys, and derives the rMSK. It also retrieves the stored SAML assertion.

6. The Visited AAA server sends the EAP-Finish/Re-Auth, the SAML assertion and the rMSK to the RP.

7. The RP sends the EAP-Finish/Re-auth to the Client and finalizes the GSS-EAP mechanism.

8. The Client derives the rMSK.

7. Security Considerations

TBD

8. IANA Considerations

The authors request that GSS-EAP Subtoken types defined in this

document be registered by the Internet Assigned Numbers Authority (IANA) from the "Extensible Authentication Protocol Mechanism for the Generic Security Service Application Programming Interface (GSS-EAP) Parameters" registry defined in section 7.3 of [RFC7055], in accordance with BCP 26 [RFC5226].

9. Acknowledgements

This work has been partly funded by the GN3plus OpenCall CLASSe project [CLASSe] .

10. References

10.1. Normative References

[I-D.ietf-abfab-arch]
Howlett, J., Hartman, S., Tschofenig, H., Lear, E., and J. Schaad, "Application Bridging for Federated Access Beyond Web (ABFAB) Architecture", draft-ietf-abfab-arch-13 (work in progress), July 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC6696] Cao, Z., He, B., Shi, Y., Wu, Q., and G. Zorn, "EAP Extensions for the EAP Re-authentication Protocol (ERP)", RFC 6696, July 2012.

[RFC7055] Hartman, S. and J. Howlett, "A GSS-API Mechanism for the Extensible Authentication Protocol", RFC 7055, December 2013.

10.2. Informative References

[CLASSe] "CLASSe project home page", <http://www.um.es/classe/>.

[I-D.ietf-abfab-aaa-saml]
Howlett, J. and S. Hartman, "A RADIUS Attribute, Binding, Profiles, Name Identifier Format, and Confirmation Methods for SAML", draft-ietf-abfab-aaa-saml-09 (work in progress), February 2014.

[I-D.ietf-abfab-usecases]
Smith, R., "Application Bridging for Federated Access Beyond web (ABFAB) Use Cases", draft-ietf-abfab-usecases-05 (work in progress), September 2012.

[RFC6942] Bournelle, J., Morand, L., Decugis, S., Wu, Q., and G. Zorn, "Diameter Support for the EAP Re-authentication Protocol (ERP)", RFC 6942, May 2013.

[RFC7057] Winter, S. and J. Salowey, "Update to the Extensible Authentication Protocol (EAP) Applicability Statement for Application Bridging for Federated Access Beyond Web (ABFAB)", RFC 7057, December 2013.

Authors' Addresses

Alejandro Perez-Mendez (Ed.)
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 46 44
Email: alex@um.es

Rafa Marin-Lopez
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 85 01
Email: rafa@um.es

Gabriel Lopez-Millan
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 85 04
Email: gabilm@um.es

Fernando Pereniguez-Garcia
Catholic University of Murcia
Av Jeronimos, 135
Murcia, 30107
Spain

Email: pereniguez@um.es