ABFAB
Internet-Draft
Intended status: Informational
Expires: September 7, 2014

L. Nordberg
NORDUnet
J. Howlett
JANET(UK)

March 06, 2014

Ephemeral keying for ABFAB
draft-linus-abfab-ephemeral-keying-01

Abstract

This document describes how EAP-GSS provides forward secrecy by encrypting each session in an ephemeral key generated in the initial state of the context establishment. This Diffie-Hellman key is shared by the initiator (EAP peer) and acceptor (EAP authenticator).

The goal is to stop a passive attacker with access to the traffic between an ABFAB user and the service she uses (Relying Party), from getting access to key material and information linkable to the user or from being able to fingerprint the user.

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 September 7, 2014.

Table of Contents

1. Introduction

The ABFAB architecture [I-D.ietf-abfab-arch] defines a GSS-API mechanism for the Extensible Authentication Protocol [RFC7055]. This mechanism provides support for the security services offered by the GSS-API, including the confidentiality of context tokens. This confidentiality service is available once a GSS context has been negotiated successfully between the initiator and acceptor.

However there is a possibility that a passive observer could extract information from this negotiation that could potentially compromise aspects of the confidentiality of the context tokens and/or the privacy of the initiator and/or acceptor.

This document defines an extension to [RFC7055] to deny a passive observer access to this information by encrypting the tokens used to establish the GSS context.

2. Information potentially accessible to a passive observer

This section describes the information available to a passive observer of an [I-D.ietf-abfab-arch] authentication, working from the lowest layers of the protocol stack upwards.

2.1. RADIUS

The realm component of the NAI [RFC4282] is generally exposed. While the user name component of the NAI is easily anonymised, the realm (which effectively names the user's identity provider (IdP)) will provide a strong indication of the organisational affiliation of a user.

In the event that RADIUS/UDP is being used instead of RADIUS/TLS, not only do the intermediate proxies between the acceptor and the IdP have access to the EAP MSK but a passive observer does too. Knowledge of the MSK could facilitate the compromise of the GSS context, which is derived from this key, potentially allowing decryption of the GSS session.

2.2. EAP method

The EAP methods most commonly used with [RFC7055] use X.509 server certificates to authenticate the IdP. This certificate will include information identifying the IdP's server.

A passive observer may also be able to fingerprint the EAP implementation [FIXME].

In cases where a TLS-based EAP method is used, a passive observer may be able to fingerprint the client based on TLS session resumption, for example as described in [RFC5077] section 5.8.

2.3. GSS-API data

A variety of information is available at the GSS-API layer.

- The acceptor name is carried in name requests and responses during the initial phase. This can be used for fingerprinting users since it indicates what service is requested and supplied. In settings where the endpoint's IP addresses and other identifying information don't link the user to the service, exposing the acceptor name is detrimental to privacy.

- GSS channel bindings are also available in the extensions state; these bindings typically identify the acceptor to the initiator.

- The currently defined flags leak information about which application protocol is being used and pose a threat to user privacy. Future flags might increase this threat.

- Finally the mechanism MIC is also exposed and error subtokens are also exposed [FIXME].

3. Solution

Generate a Diffie-Hellman key in the initial state of the context establishment and use it to encrypt other context tokens. Note that the DH key, shared by initiator and acceptor, is unique per GSS-API session, not per context token. [Elaborate on why?]

[describe where in initial the DH key exchange happens and how; point at general description? copy from existing standard?]

[describe how we signal algorithm and key size]

[describe the use of a nonce/sequence number for temporality, either in the key or in the payload, covered by the MIC and verified by the other end - mitigates replay, reflection and reordering attacks]

[describe how we derive a symmetric key from the DH key and encrypt the context token (perhaps in a GSS "wrap token"?)]

[describe how to mix in the DH key with the MSK to form the CRK (7055 sect 6) - this will make a MITM kexing with both ends unable to create a MIC which validates properly (and a MITM relaying DH kex will not know the key and thus not the CRK)]

3.1. Why do this at the GSS-EAP layer

Using a short lived key for providing confidentiality between an ABFAB client and the IdP could arguably be done at the EAP layer rather than at the GSS-API layer. A general solution for EAP would give better protocol reuse.

EAP methods run between the EAP peer and server. A Diffie-Hellman key exchange between these endpoints can not start with the first message sent from the client since the client doesn't talk to the EAP server (the IdP) directly and can not be helped with doing that until the EAP authenticator knows where the IdP is to be found. Most of the mentioned leaks at the GSS-API layer would thus still be present in this solution.

[maybe expand on how TEAP [draft-ietf-emu-eap-tunnel-method] could solve the problem of AAA proxies learning the MSK, impersonating the RP]

An alternative place to protect ABFAB authentication with a short lived key would be in the application level protocol. While some applications are using protocols already able to protect the GSS-API traffic using a TLS session with an ephemeral key (XMPP, IMAP, SMTP) it's not mandatory to use such a tunnel. Other applications use protocols which might be hard to protect in a tunnel (NFS, SSH).

4. Keying algorithm

This section defines an algorithm, based on the Diffie-Hellman protocol, enabling the initiator and acceptor to negotiate a shared key during the initial phase of the GSS context establishment. This key is used to encrypt all subsequent context tokens. The key is unique per GSS-API session, and is not rotated for each successive context token. [Elaborate on why not?]

5. Costs

- This will cost FIXME extra round trips.
- [No new GSS mech. Thus no complexity cost of picking the right one.]

6. Open questions

- Should we make the ephemeral keying and encryption optional?

  Might have to - asking the list about breaking backward compatibility.

- Bid down attacks - detect, prevent

  Fascinating idea from Sam: 6067 CB implementing 5056 CB could detect MITM before end of extension state (MIC).

- Include the nonce/sequence number in tokens or fold it into the key?

7. Security considerations

TBD

8. IANA considerations

TBD.

9. Contributors

The whole idea of adding ephemeral keys to ABFAB was suggested by Sam Hartman who also contributed substantial ideas and discussions on this subject.

Jim Schaad has made several valuable comments with corrections and suggestions.

10. 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-12 (work in progress), February 2014.

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, December 2005.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.

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

Authors' Addresses

Linus Nordberg
NORDUnet

Email: linus@nordu.net

Josh Howlett
JANET(UK)

Email: Josh.Howlett@ja.net