Workgroup: Network Working Group
Internet-Draft: draft-tigress-requirements-01
Published: 9 September 2022
Intended Status: Informational
Expires: 13 March 2023
Authors: D. Vinokurov   C. Astiz   A. Pelletier
         Apple Inc   Apple Inc   Apple Inc
         J. L. Giraud   A. Bulgakov   M. Byington
         Apple Inc   Apple Inc   Apple Inc
         N. Sha
         Alphabet Inc

Transfer Digital Credentials Securely - Requirements

Abstract

This document describes the use cases necessitating the secure transfer of digital credentials. The document also comprises a proposal, and defines requirements and scope.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://dimmyvi.github.io/tigress-requirements/draft-tigress-requirements.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-tigress-requirements/.

Source for this draft and an issue tracker can be found at https://github.com/dimmyvi/tigress-requirements.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
2. Conventions and Definitions
3. Use Cases
4. Assumptions
5. Requirements
6. Review of existing solutions
  6.1. Arbitrary Messaging Channel (Email / WhatsApp / SMS / Signal / etc.)
  6.2. GSS-API, Kerberos
  6.3. Signal Protocol
7. Out of Scope:
8. Security Considerations
9. IANA Considerations
10. Normative References
Acknowledgments
Authors' Addresses

1. Introduction

TODO Introduction

2. Conventions and Definitions

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

General terms:

*Credential information - data used to authenticate the user with an access point.

*Provisioning information - data transferred from Sender to Receiver device that is both necessary and sufficient for the Receiver to request a new credential from Provisioning Partner to provision it to the Receiver device.

*Provisioning - A process of adding a new credential to the device.

*Provisioning Partner - an entity which facilitates Credential Information lifecycle on a device. Lifecycle may include provisioning of credential, credential termination, credential update.

*Sender (device) - a device initiating a transfer of Provisioning Information to a Receiver that can provision this credential.

*Receiver (device) - a device that receives Provisioning Information and uses it to provision a new credential.

*Intermediary (server) - an intermediary server that facilitates transfer of provisioning information between Sender and Receiver.

3. Use Cases

*Let's say Ben owns a vehicle that supports digital keys which comply with the CCC [CCC-Digital-Key-30] open standard. Ben would like to let Ryan borrow the car for the weekend. Ryan and Ben are using two different mobile phones with different operating systems. In order for Ben to share his car key to Ryan for a weekend, he must transfer some data to the receiver device. The data structure shared between the two participants is defined in the CCC. In addition, the CCC requires the receiver to generate required key material and return it to the sender to sign and return back to the receiver. At this point, the receiver now has a token that will allow them to provision their new key with the car.

*Bob booked a room at a hotel for the weekend, but will be arriving late at night . Alice, his partner, comes to the hotel first, so Bob wants to share his key to the room with Alice. Bob and Alice are using two different mobile phones with different operating systems. In order for Bob to share his key to the hotel to Alice for a weekend, he must transfer some data to her device. The data structure shared between the two participants is proprietary to the given hotel chain (or Provisioning Partner). This data transfer is a one-time, unidirectional from Bob's device to Alice's. Once Alice receives this data, she can provision a new pass to her device, making a call to Provisioning Partner to receive a new credential.

4. Assumptions

- Original credential (with cryptographic key material) **MUST NOT** be sent or shared. Instead, sender **SHALL** be transferring its approval token for Receiver to acquire a new credential.

- Provisioning Partner **SHALL NOT** allow for two users to use the same credential / cryptographic keys.

- Security: Communication between Sender / Receiver and Provisioning Partner **SHOULD** be trusted.

- The choice of intermediary **SHALL** be defined by the application initiating the credential transfer.

- Sender and Receiver **SHALL** both be able to manage the shared credential at any point by communicating with the Provisioning Partner. Credential lifecycle management is out of scope for this proposal.

- Any device OEM with a digital credential implementation adherent to Tigress **SHALL** be able to receive shares, whether or not they can originate shares or host their own intermediary.

5. Requirements

- (Req-AnyPlatorm) Solution **SHOULD** be able to communicate with any mobile devices of any operating system and allow easy implementation of server-side components without requiring a specific Cloud stack.

- (Req-NontechnicalUX) Solution **SHALL** enable secure credential transfer for non technical users.

- (Req-SmoothUX) Solution **SHALL** allow for user experience where neither Sender nor Receiver is presented with raw data required only by the secure transfer protocol. The data **SHOULD** only be parsed programmatically and not required to be presented to the end user. This data **SHOULD** never be visible to said user in whichever messaging application the sender chose to initiate the transfer on. This eliminates the possibility of merely sending the requisite data inline, through an SMS or email for example, rather than leveraging an Intermediary server.

- (Req-Connectivity) Sender and Receiver **SHALL** be allowed to be online at different times. Sender and Receiver **SHALL** never need to be online at the same time.

*(Req-init) Solution **SHOULD** allow Sender to initiate credential transfer to Receiver over any messaging channel, with various degrees of security.

*(Req-P2P) A credential transfer **SHALL** be strictly from one device to another (group sharing is not a goal).

*(Req-Privacy) If Intermediary server is required - it **SHALL** not be able to correlate users between exchanges, or create a social graph. Intermediary server shall not be an arbiter of Identity.

*(Req-Security) Solution **SHOULD** provide security of the provisioning data transferred (MITM, brute-force attacks on the content, DDOS attacks etc).

*(Req-Notify) Solution **SHOULD** support a notification mechanism to inform devices on the content update on Intermediary server.

*(Req-Revoke) Solution **SHALL** maintain access control, allowing Sender to revoke before the share has been accepted, and for Receiver to end transfer at any time.

*(Req-IntermediaryProvision) If Intermediary server is required - it **MUST** not be able to provision credential on their own.

*(Req-Opaque) If Intermediary server is required - Message content between Sender and Receiver **MUST** be opaque to the Intermediary.

*(Req-ArbitraryFormat) The solution **SHALL** support arbitrary message formats to support both keys that implement public standards like CCC as well as proprietary implementations of digital keys.

*(Req-UnderstoodFormat) Both Sender application and Receiver application **MUST** be able to recognize the format.

*(Req-Simplicity) Where possible, the system **SHOULD** rely on simple building blocks to facilitate adoption and implementation.

*(Req-IntermediaryAttestation) If any Intermediary is required - it **SHALL** implement mechanisms to prevent abuse by share initiating device, verifying that the device is in good standing and content generated by the sender device can be trusted by the Intermediary. The trust mechanism could be proprietary or publicly verifiable (e.g. WebAuthN).

*(Req-RoundTrips) Solution **SHALL** allow for multiple round trips or multiple reads/writes between one set of Sender and Receiver devices.

- *(Req-ReceiverTrust) If any Intermediary is required - the Receiver device **SHOULD** evaluate the trustworthiness of the Intermediary using a list of trusted/approved intermediaries.

- *(Req-Preview) Solution **SHOULD** allow for extensibility and discoverable extensions (preview of share invitation).

- *(Req-RedemptionHandling) ShareURL **SHOULD** route Receiver to redeem Provisioning Information using the designated Credential Management Application (e.g. Wallet).

6. Review of existing solutions

A number of existing solutions / protocols have been reviewed in order to be used for secure credential transfer based on the requirements: GSS-API, Kerberos, AWS S3, email, Signal. None of the existing protocols comply with the requirements; the effort of modifying the existing protocols has been accessed to be significantly higher than introducing a new solution to solve this problem.

6.1. Arbitrary Messaging Channel (Email / WhatsApp / SMS / Signal / etc.)

The Provisioning Information **MAY** be sent from Sender to Receiver over an arbitrary messaging channel, but that would not provide a good user experience. Users **MAY** need to copy and paste the Provisioning Information, or need a special application to handle some new file type. This violates (Req-SmoothUX). If multiple round trips were required the user would need to manually managing multiple payloads of Provisioning Information. This would be very hard for anyone non technical and greatly limit adoption. This violates (Req-NontechnicalUX).

6.2. GSS-API, Kerberos

GSS-API [RFC2078] and Kerberos [RFC4120] are authentication technologies which could be used to authenticate Sender, Receiver and intermediary. However, as they provide strong authentication, they would allow the Intermediary server to build a social graph in violation of (Req-Privacy). Their setup also require strong coordination between the actors of the system which seems overly costly for the intended system. AWS S3 could be used as an Intermediary server but it would force all participants to use a specific cloud service which is in violation of (Req-AnyPlatorm).

6.3. Signal Protocol

As a messaging protocol, Signal could be used between Sender, Receiver and Intermediary but this protocol is fairly complex and its use would most like violate (Req-Simplicity). The system will however support the Signal service for share initiation, in line with (Req-init).

7. Out of Scope:

*Identification and Authorization - solution shall not require strong identification and authentication from user (e.g. using PKI certificates).

*Fully stopping people from sharing malicious content ("cat pictures").

*Solving problem of sharing to groups.

*Detailing how credentials are provisioned either on a device or with a provisioning partner.

8. Security Considerations

TODO Security

9. IANA Considerations

This document has no IANA actions.

10. Normative References

[CCC-Digital-Key-30] Car Connectivity Consortium, "Digital Key – The Future of Vehicle Access", November 2021, <https://global-carconnectivity.org/wp-content/uploads/2021/11/CCC_Digital_Key_Whitepaper_Approved.pdf>.

[RFC2078] Linn, J., "Generic Security Service Application Program Interface, Version 2", RFC 2078, DOI 10.17487/RFC2078, January 1997, <https://www.rfc-editor.org/rfc/rfc2078>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, DOI 10.17487/RFC4120, July 2005, <https://www.rfc-editor.org/rfc/rfc4120>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Dmitry Vinokurov
Apple Inc

Email: dvinokurov@dezcom.org

Casey Astiz
Apple Inc

Email: castiz@apple.com

Alex Pelletier
Apple Inc

Email: a_pelletier@apple.com

Jean-Luc Giraud
Apple Inc

Email: jgiraud@apple.com

Alexey Bulgakov
Apple Inc

Email: abulgakov@apple.com

Matt Byington
Apple Inc

Email: mbyington@apple.com

Nick Sha
Alphabet Inc

Email: nicksha@google.com