Workgroup: TODO Working Group
Internet-Draft: draft-pfeairheller-ptel-01
Published: 28 July 2023
Intended Status: Informational
Expires: 29 January 2024
Authors: P. Feairheller
GLEIF

Public Transaction Event Logs (PTEL)

Abstract

TODO Abstract

Discussion Venues

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

Source for this draft and an issue tracker can be found at https://github.com/trustoverip/tswg-ptel-specification.

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 https://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 29 January 2024.

Copyright Notice

Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

The *Transaction Event Log* (TEL) is a hash linked data structure of transactions that can be used to track state. A *Public Verifiable Credential Registry* can be represented in several TELs to establish issuance or revocation state of a Verifiable Credential (VC). The KEL is used to establish control authority over the keys used to commit to the events of the TEL and sign the VC. The events of the TEL are used to establish the issuance or revocation state of the VCs issued by the controller of the identifier represented by the KEL. This document specifies a design for *public* VCs only. The use of a hash digest of the VC contents as the identifier of that VC or an attribute in a TEL event allows for correlation of uses of the VC.

1.1. Transaction Event Log

A KERI KEL can control a TEL by anchoring the TEL to key events of the KEL with the following:

1. Create the inception event for the TEL with the TEL specific unique identifier.

2. Generate a hash digest of the serialized content of the TEL inception event.

3. Attach anchoring seals from the KEL events to the TEL events they are authorizing.

4. Sign the KEL event as usual to commit to the digest of the serialized TEL event.

Any validator can cryptographically verify the authoritative state by validating the signatures of the referenced KEL. The TEL events do not have to be signed as the commitment to the event is in the form of the digest in the seal in the anchoring KEL event and the signatures on that event. Like KEL events, all TEL events have the fields i, s, and t. However, the s or sequence number field in TEL events represents the "clock" for that transaction set. Each transaction set can have its own "clock" (e.g. bitcoin block height, wall clock, etc) and is independent of the sequence number of the KEL events. In the case of the Verifiable Credential Registry, the s field is simply a monotonically increasing integer.

The events are anchored back to the KEL using Event Source Seals whose JSON representation is as follows.

```
{ "s": "3", "d": "ELvaU6Z-i0d8JJR2nmwyYAZAoTNZH3UfSVPzhzS6b5CM" }
```

For TEL events, this seal back to the KEL will be delivered as an attachment of event source seal triples in duple of (s, d).

```
-GAB 0AAAAAAAAAAAAAAAAAAAAAAw ELvaU6Z-
i0d8JJR2nmwyYAZAoTNZH3UfSVPzhzS6b5CM
```

Event source seal attachment example (line feeds added for readability)

1.2. Verifiable Credential Registry

A *Public Verifiable Credential Registry* (Registry) is a form of a *Verifiable Data Registry* that tracks the issuance/revocation state of credentials issued by the controller of the KEL. Two types of TELs will be used for this purpose. The first type of TEL is the management TEL and will signal the creation of the Registry and track the list of Registrars that will act as Backers for the individual TELs for each VC. The second type of TEL is the VC TEL which will track the issued or revoked state of each VC and will contain a reference to it's corresponding management TEL.

The following events will be used to create and maintain the TELs for the Registry.

Ilk	TEL	Name	Description
vcp	Management	Registry Inception Event	Inception statement for the Registry
vrt	Management	Registry Rotation Event	Rotation event for updating Backers
iss	VC	Simple Credential Issuance Event	Issue credential with no Backers
rev	VC	Simple Credential Revocation Event	Revoke previously issued credential with no Backers
bis	VC	Credential Issuance Event	Issue credential
brv	VC	Credential Revocation Event	Revoke previously issued credential
iis	VC	Simple Credential Issuance Event with VC Hash	Issue credential with no Backers, VC Hash as separate field
irv	VC	Simple Credential Revocation Event with VC Hash	Revoke previously issued credential with no Backers, VC Hash as separate field
ibs	VC	Credential Issuance Event with VC Hash	Issue credential, VC Hash as separate field
ibr	VC	Credential Revocation Event with VC Hash	Revoke previously issued credential, VC Hash as separate field

Table 1

1.3. Management TEL

The state tracked by the Management TEL will be the list of Registrar identifiers that serve as backers for each TEL under its provenance. This list of Registrars can be rotated with events specific to this type of TEL. In this way, Registrar lists are analogous to Backer lists in KERI KELs. Additional metadata can be tracked in this TEL, for example references to Schema. The Management TEL will have two events: vcp for Registry inception and vrt for rotation of the list or Registrars. The events will reference the controlling identifier in the ii field and be anchored to the KEL with an event seal triple attachment.

The Registry specific identifier will be self-addressing (see below (Section 1.4.1) for definition) using its inception data for its derivation. This requires a commitment to the anchor in the controlling KEL and necessitates the event location seal be included in the event. The derived identifier is then set in the i field of the events in the management TEL.

Though it is possible for a given identifier KEL to issue multiple types of credentials, it is anticipated that there will be relatively few (usually one) Management TELs anchored to a given

KEL. A more scalable approach to issuing multiple credential types from a single identifier would be to use delegated identifiers for the different types of credentials to be issued.

Label	Description	Notes
v	version string	
i	namespaced identifier of Registry	
s	sequence number of event	
t	message type of event	
p	prior event digest	
c	list of Configuration Traits/Modes	allows for config of no backer registry
a	digest seal of attachment meta-data for registry	
ii	issuer identifier	
vi	hash digest of VC contents	
b	list of backer identifiers for credentials associated with this registry	
bt	backer threshold	
ba	list of backers to add (ordered backer set)	
br	list of backers to remove (ordered backer set)	

Table 2

1.3.1. Configuration

The simplest (and most common) case for Registries relies on the witnesses of the controlling KEL and their receipts of the KEL events instead of Registry specific backers. To accommodate this case, the c element is added to the management TEL inception event with the configuration option NB to specify that the Registry will never have backers configured in the management TEL. In this case, there will only be one event in the management TEL for this Registry and the simple events iss and rev will be used for "simple issue" and "simple revoke" respectively in the VC specific TELs. For these events, the ri field will be the simple identifier referencing the management TEL.

Option	Description	Notes
NB	No Backers	No registry specific backers will be configured for this Registry

Table 3

1.3.2. Registry Inception Event

```
{ "v" : "KERI10JSON00011c_", "i" :
"ELh3eYC2W_Su1izlvm0xxw01n3XK8bdV2Zb09IqlXB7A", "ii":
"EJJR2nmwyYAfSVPzhzS6b5CMZAoTNZH3ULvaU6Z-i0d8", "s" : "0", "t" :
"vcp", "b" : ["BbIg_3-11d3PYxSInLN-Q9_T2axD6kkXd3XRgbGZTm6s"], "c" :
[] "a" : { "d": "EEBp64Aw2rsjdJpAR0e2qCq3jX7q7gLld3LjAwZgaLXU" } }-
GAB0AAAAAAAAAAAAAAAAAAAABwEOWdT7a7fZwRz0jiZ0DJxZEM3vsNbLDPEUk-
ODnif3O0
```

Registry inception event for establishing the list of Backers

```
{ "v" : "KERI10JSON00011c_", "i" :
"ELh3eYC2W_Su1izlvm0xxw01n3XK8bdV2Zb09IqlXB7A", "ii":
"EJJR2nmwyYAfSVPzhzS6b5CMZAoTNZH3ULvaU6Z-i0d8", "s" : "0", "t" :
"vcp", "b" : [], "c" : ["NB"] }-
GAB0AAAAAAAAAAAAAAAAAAAABwEOWdT7a7fZwRz0jiZ0DJxZEM3vsNbLDPEUk-
ODnif3O0
```

Registry inception event for "backer-less" configuration

1.3.3. Registry Rotation Event

```
{ "v" : "KERI10JSON00011c_", "i" :
"ELh3eYC2W_Su1izlvm0xxw01n3XK8bdV2Zb09IqlXB7A", "p" :
"EY2L3ycqK9645aEeQKP941xojSiuiHsw4Y6yTW-PmsBg", "s" : "1", "t" :
"vrt", "ba" : ["BXhpfP_H41hw8f-LluTidLfXxmC4EPwaENHI6CuruE6g"], "br"
: ["BbIg_3-11d3PYxSInLN-Q9_T2axD6kkXd3XRgbGZTm6s"] }-
GAB0AAAAAAAAAAAAAAAAAAAACQEOWdT7a7fZwRz0jiZ0DJxZEM3vsNbLDPEUk-
ODnif3O0
```

Registrar rotation event updates the list of Backers

1.4. Verifiable Credential TELs

The binary state (issued or revoked) of each verifiable credential (VC) will be tracked in individual TELs associated with each VC. The state changes will be represented by 4 sets of 2 events: iss for simple VC issuance and rev for simple revocation, bis for the issuance of the VCs with backers and brv for revocation of the VCs with backers and corresponding events iis, irv and ibs, ibr to be used when the identifier of the VC is not the self-addressing identifier of the VC and that identifier must be included is the separate vi field in the event. The events will be anchored to the KEL with an event seal triple attachment signified by the grouping counter -e##.

1.4.1. Self Addressing Identifiers

The advantage of a content addressable identifier is that it is cryptographically bound to the contents. It provides a secure root-of-trust. Any cryptographic commitment to a content addressable identifier is functionally equivalent (given comparable cryptographic strength) to a cryptographic commitment to the content itself.

A self-addressing identifier is a special class content-addressable identifier that is also self-referential. The special class is distinguished by a special derivation method or process to generate the self-addressing identifier. This derivation method is determined by the combination of both a derivation code prefix included in the identifier and the context in which the identifier appears. The reason for a special derivation method is that a naive cryptographic content addressable identifier must not be self-referential, i.e. the identifier must not appear within the contents that it is identifying. This is because the naive cryptographic derivation process of a content addressable identifier is a cryptographic digest of the serialized content. Changing one bit of the serialization content will result in a different digest. A special derivation method or process is required.

1.4.2. Derivation Process

This process is as follows:

- replace the value of the id field in the content that will hold the self-addressing identifier with a dummy string of the same length as the eventually derived self-addressing identifier

- compute the digest of the content with the dummy value for the id field

- prepend the derivation code to the digest and encode appropriately to create the final derived self-addressing identifier replace the dummy value with the self-addressing identifier

As long as any verifier recognizes the derivation method, the 'self-addressing` identifier is a cryptographically secure commitment to the contents in which it is embedded. It is a cryptographically verifiable self-referential content addressable identifier.

Because a self-addressing identifier is both self-referential and cryptographically bound to the contents it identifies, anyone can validate this binding if they follow the binding protocol outlined above.

To elaborate, this approach of deriving self-referential identifiers from the contents they identify, we call self-addressing. It allows a verifier to verify or re-derive the self-referential identifier given the contents it identifies. To clarify, a self-addressing identifier is different from a standard content address or content addressable identifier in that a standard content addressable identifier may not be included inside the contents it addresses. The standard content addressable identifier is computed on the finished immutable contents and therefore is not self-referential.

1.5. Self-Addressing Identifiers in a TEL

ii issuer identifier is the controller prefix is self-certifying and may be also self-addressing (but may not be) wrt to its inception event (For GLEIF TELS the issuer identifier must be self-addressing)

ri, i registry identifier is self-addressing wrt the registry inception event i VC identifier is self-addressing wrt to the VC itself

There are two options for including a cryptographic commitment to the VC in the TEL VC events. The identifier of the VC can self-addressing using the same technique KERI uses for self-addressing identifiers. The VC identifier can be created by padding the VC id field and taking a hash digest of the serialized contents of the VC. This form of self-addressing identifier can be used as the i field in the TEL iss, rev, bis and brv events and no other reference to the VC is required. When the identifier of the VC is derived from some other method, the TEL events iis, irv, ibs and ibr are used, and a hash digest of the contents of the VC is placed in the vi field.

The VC identifier can be namespaced using DID syntax. In this case, the VC identifier in the TEL events would be the method specific identifier of the full DID. For informational purposes, the fully qualified DID can be included as an attachment to the TEL events.

The list of backers needed to sign each VC TEL event is maintained by the management TEL. Since that list can change over time with the rot management events listed above, the non-simple VC events (bis, brv) must be anchored to the event in the management TEL at the point when the VC event is published with the ra field. This way, the backer signatures can be indexed into the proper list of backers at the time of issuance or revocation.

1.6. Credential Issuance/Revocation TEL

Label	Description	Notes
v	version string	

Label	Description	Notes
i	namespaced identifier of VC	
s	sequence number of event	
t	message type of event	
dt	issuer system data/time in iso format	
p	prior event digest	
ri	registry identifier from management TEL	
ra	registry anchor to management TEL	

Table 4

1.6.1. Simple Credential Issuance Event

```
{ "v" : "KERI10JSON00011c_", "i" : "Ezpq06UecHwzy-
K9FpNoRxCJp2wIGM9u2Edk-PLMZ1H4", "s" : "0", "t" : "iss", "dt":
"2021-05-27T19:16:50.750302+00:00", "ri":
"ELh3eYC2W_Su1izlvm0xxw01n3XK8bdV2Zb09IqlXB7A" }-
GAB0AAAAAAAAAAAAAAAAAAAAAwELvaU6Z-
i0d8JJR2nmwyYAZAoTNZH3UfSVPzhzS6b5CM
```

1.6.2. Simple Credential Revocation Event

```
{ "v" : "KERI10JSON00011c_", "i" : "Ezpq06UecHwzy-
K9FpNoRxCJp2wIGM9u2Edk-PLMZ1H4", "s" : "1", "t" : "rev", "dt":
"2021-05-27T19:16:50.750302+00:00", "p" :
"EY2L3ycqK9645aEeQKP941xojSiuiHsw4Y6yTW-PmsBg" }-
GAB0AAAAAAAAAAAAAAAAAAAAABAELvaU6Z-
i0d8JJR2nmwyYAZAoTNZH3UfSVPzhzS6b5CM
```

1.6.3. Credential Issuance Event

```
{ "v" : "KERI10JSON00011c_", "i" : "Ezpq06UecHwzy-
K9FpNoRxCJp2wIGM9u2Edk-PLMZ1H4", "s" : "0", "t" : "bis", "dt":
"2021-05-27T19:16:50.750302+00:00", "ra": { "i":
"ELh3eYC2W_Su1izlvm0xxw01n3XK8bdV2Zb09IqlXB7A", "s": "2", "d":
"Ezpq06UecHwzy-K9FpNoRxCJp2wIGM9u2Edk-PLMZ1H4" } }-
GAB0AAAAAAAAAAAAAAAAAAAAAwELvaU6Z-
i0d8JJR2nmwyYAZAoTNZH3UfSVPzhzS6b5CM
```

1.6.4. Credential Revocation Event

```
{ "v" : "KERI10JSON00011c_", "i" : "Ezpq06UecHwzy-
K9FpNoRxCJp2wIGM9u2Edk-PLMZ1H4", "s" : "1", "t" : "brv", "dt":
"2021-05-27T19:16:50.750302+00:00", "p" :
"EY2L3ycqK9645aEeQKP941xojSiuiHsw4Y6yTW-PmsBg", "ra": { "i":
"ELh3eYC2W_Su1izlvm0xxw01n3XK8bdV2Zb09IqlXB7A", "s": "4", "d":
"Ezpq06UecHwzy-K9FpNoRxCJp2wIGM9u2Edk-PLMZ1H4" } }-
GAB0AAAAAAAAAAAAAAAAAAAAABAELvaU6Z-
i0d8JJR2nmwyYAZAoTNZH3UfSVPzhzS6b5CM
```

1.7. Use Case

The *Verifiable Legal Entity Identifier* (vLEI) provides a lightweight, easy to understand use case for a *Transaction Event Log* as a *Verifiable Credential Registry*. Issuing a VC has been described above. Verification of a VC will start with the presentation of a vLEI VC as proof (all vLEI VCs are public and therefore proof presentation will include the entire vLEI VC). The verifier will extract the DID of the issuer from the VC, and calculate the hash digest of the serialized contents of the VC. By parsing the namespaced identifier of the VC, the verifier will perform the following steps:

1. Retrieve the key state from the KERI did method (or appropriate DID method tunnel) using the controller identifier embedded in the VC identifier

2. Retrieve and verify the KEL against the key state of the issuer

3. Retrieve the management TEL using the Registry identifier embedded in the VC identifier and determine the Registrars to use to retrieve the VC TEL.

4. Retrieve the VC TEL and calculate the issuance/revocation state of the VC from the events in the TEL.

5. Using the keys from the KERI event to which the iss event is anchored, verify the signature on the VC.

1.8. Security Considerations

1. To avoid DDoS attack by flooding an Endorser with TEL events that are not associated with any identifiers they are associated with, TEL events need to be placed in escrow until an anchoring KEL event is seen for the TEL identifier.

2. IANA Considerations

This document has no IANA actions.

Acknowledgments

TODO acknowledge.

Author's Address

Phil Feairheller
GLEIF

Email: Philip.Feairheller@gleif.org