hackmd.io

KERI OOBI - HackMD

20–25 minutes

Vacuous discovery of IP resources such as service endpoints associated with a KERI AID requires an Out-Of-Band Introduction (OOBI) to associate a given URL with a given AID. The principal reason for this requirement is that KERI AIDs are pseudonymous and completely independent of internet and DNS addressing infrastructure. Thus an IP address or URL could be considered a type of Out-Of-Band Infrastructure (OOBI) for KERI. In this context an introduction is an association between a KERI AID and a URL which may include either an explicit IP address for its netloc or a DNS name. We call this a KERI OOBI (Out-Of-Band-Introduction) and is a special case of Out-Of-Band-Infrastructure (OOBI) with a shared acronym. For the sake of clarity, unless otherwise qualified, OOBI is used to mean this special case of an introduction and not the general case.

Moreover, because IP infrastructure is not trusted by KERI, a KERI OOBI by itself is considered insecure with respect to KERI and any OOBI must therefore be later proven and verified using a KERI BADA (Best Available Data Acceptance) mechanism. The principal use case for an OOBI is to jump start the discovery of a service endpoint for a given AID. To reiterate, the OOBI by itself is not sufficient for discovery because the OOBI itself is insecure. The

OOBI merely jump starts authenticated discovery.

Using IP and DNS infrastructure to introduce KERI AIDs which AIDs are then securely attributed allows KERI to leverage IP and DNS infrastructure for discovery. KERI does not therefore need its own dedicated discovery network, OOBIs as URLs will do.

The simplest form of a KERI OOBI is a namespaced string, a tuple, a mapping, a structured message, or structured attachment that contains both a KERI AID and a URL. The OOBI associates the URL with the AID. In tuple form this abstractly:
`(url, aid)`
and concretely
`("http://8.8.5.6:8080/oobi", "EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM")`

An OOBI itself is not signed or otherwise authenticatible by KERI but may employ some other Out-Of-Band-Authentication (OOBA) mechanism (non-KERI).

The OOBI is intentionally simplistic to enable very low byte count introductions such as a QR code or Data matrix or the like.

A recipient of an OOBI authenticates authorized endpoints for the AID in the OOBI by querying the OOBI URL. The URL resource responds with supporting BADA reply messages that are KERI authenticatable.

Multi-OOBI (MOOBI)

An OOBI may include a list of URLs thus simultaneously making an introductory association between the AID and multiple URLs. This would be a multi-OOBI (MOOBI). In general we may refer to a multi-OOBI as a special case of an OOBI without making a named distinction.

OOBI as URL (iurl)

URLs provide a namespace which means that the mapping between URL and AID can be combined into one namespaced URL where the AID is in the path component and any other hints such as roles or names are in the query component of the URL. This would be a type of self-describing OOBI URL.

For example suppose the `aid` is `EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM`. This may be included as a path component of the `url` such as:

```
http://8.8.5.6:8080/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM
```

This is called an OOBI URL or `iurl` for short.
This means that all that is needed to bootstrap discovery of a KERI AID is an `iurl`. KERI can leverage the full IP/DNS infra-structure for discovery bootstrap of an `aid` by providing an `iurl` with that `aid` for lookup.

The aid may act in any of the KERI roles such as `watcher`, `witness`, `juror`, `judge` or `registrar` but is usually a `controller`. In the later case the url may be a service endpoint provided by one of the supporting components for a given controller. Thus the `aid` in an OOBI may be either a controller id, `cid` or an endpoint provider id, `eid`. The resource at that URL in the OOBI is ultimately responsible for providing that detail but an OOBI as URL may contain hints in the query string for the URL such as a `role` or `name` designation.

```
http://8.8.5.6:8080/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM?role=watcher&name=eve
```

Other examples of `iurls` (OOBI URLs).

```
https://example.com/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM?role=witness
```

When the role is provided in the `iurl`, the AID (EID) of the the endpoint provider for that role would be discovered via the proof returned by querying the URL. The proof returned may indicate a different URL for that role. Thus a self-describing OOBI URL may act as a forwarding mechanism.

To clarify, the minimum information in an OOBI is pair, `(url, aid)`. A compact representation of an OOBI leverages the namespacing of the URL itself to provide the AID. Furthermore the query string in the URL namespace may contain other information or hints such as the role of the service endpoint represented by the URL or a user friendly name.

Other Forms of OOBIs

Well-Known

An OOBI may be returned as the result of a get request to an IETF RFC 5785 well-known URL. For example:

```
 /.well-known/keri/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM
```

Where `EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM` is the AID
and the result of the request is either target URL or a redirection to

the target URL where the target URL is something like

`https://example.com/witness/witmer`

or

`http://8.8.5.5:8080/witness/witmer`

or

`http://10.0.5.15:8088/witness/witmer`

The resultant target URL may be in a different domain or IP address from the well known resource.

Full CID and EID

A more verbose version would also include the endpoint role and the AID (EID) of the endpoint provider in the self describing OOBI URL.

```
https://example.com/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM/witness/BrHLayDN-mXKv62DAjFLX1_Y5yEUe0vA9YPe_ihiKYHE
```

or

```
http://8.8.5.6/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM/witness/BrHLayDN-mXKv62DAjFLX1_Y5yEUe0vA9YPe_ihiKYHE
```

Where `EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM` is the AID (CID) of the controller and `BrHLayDN-mXKv62DAjFLX1_Y5yEUe0vA9YPe_ihiKYHE` is the AID (EID) of the endpoint provider for that controller in the role of `witness`.

KERI Reply Messages as OOBIs

A more verbose expression for an OOBI would be a KERI reply message `rpy` that is unsigned. The route specifies that it is an OOBI so the recipient knows to apply OOBI processing logic to the message. A list of URLs is provided so that it may provide multiple introductions. For example:

```
{
          "v" : "KERI10JSON00011c_",
          "t" : "rpy",
          "d": "EZ-
i0d8JZAoTNZH3ULaU6JR2nmwyvYAfSVPzhzS6b5CM",
          "dt": "2020-08-22T17:50:12.988921+00:00",
          "r" : "/oobi/witness",
          "a" :
          {
             "urls":  ["http://example.com/watcher/
watson", "http://example.com/witness/wilma"]
             "aid":  "EaU6JR2nmwyZ-
i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM"
          }
}
```

A service endpoint location reply message could also be re-purposed as an OOBI by using a special route path that includes the AID being introduced and optionally the role of the service endpoint provider as follows:

```
{
          "v" : "KERI10JSON00011c_",
          "t" : "rpy",
          "d": "EZ-
i0d8JZAoTNZH3ULaU6JR2nmwyvYAfSVPzhzS6b5CM",
          "dt": "2020-08-22T17:50:12.988921+00:00",
          "r" : "/oobi/EaU6JR2nmwyZ-i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM/watcher",
          "a" :
          {
             "eid": "BrHLayDN-mXKv62DAjFLX1_Y5yEUe0vA9YPe_ihiKYHE",
             "scheme": "http",
             "url":  "http://example.com/watcher/wilma",
          }
}
```

This more verbose approach includes the AID (EID) of the service endpoint provider which may allow a short cut to authenticating the service endpoint.

Self and Blind Introductions

A bare URL but no AID may be used as a bare OOBI for blind or self introductions. Querying that bare OOBI may return or result in a default target OOBI or default target endpoint reply. This provides a mechanism for self-introduction, i.e. self OOBI (SOOBI). Consider a bare OOBI as follows:

`http://8.8.5.7:8080/oobi`

or

`http://localhost:8080/oobi`

or

`http://8.8.5.7:8080/oobi?role=controller&name=eve`

or

```
http://localhost:8080/oobi?role=controller&name=eve
```

By default the result of get request to this OOBI URL could be another OOBI with an AID that is the `self` AID of the node providing the bare OOBI endpoint or the actual authenticatable `self` endpoint with its AID or a default set of authenticatable endpoints.

This may be especially useful to bootstrap components in an infrastructure where the target URLs do not use a public DNS address but use instead something more secure like an explicit public IP address or a private IP or private DNS address. A self introduction provides a bootstrap mechanism similar to a hostname configuration file with the exception that in the OOBI case the AID is not in the configuration file just the bare OOBI URL and the given node queries that bare OOBI to get the target endpoint AID. This allows bootstrap using bare IP addresses in systems where the IP infrastructure is more securely managed than public DNS or where some other Out-Of-Band-Authentication (OOBA) mechanism is used in concert. Because the OOBI itself does not contain an AID the association of the resultant AID is not provided by the OOBI and the resultant AID's association must be secured by some other mechanism.

For example a given indirect mode controller is identified by its AID (CID). The controller must also create witness hosts with endpoints. This means first spinning up witness host nodes and creating witness AIDs (WIDs) for those nodes. Given that these WIDs must be eventually designated in the KEL for the CID, the controller of the CID can confirm using its KEL that the signed endpoint reply provided by a bare OOBI request is indeed signed by the corresponding private keys for a WID designated in its KEL. This means that the only place that the WID must appear is in the KEL and not in all the config files used to boostrap communications between the CID host and its designated WID hosts. Bare OOBIs will do. Redundant configuration information may be a vector for a type of DDOS attack where corrupted inconsistent redundant configuration information results in a failure to boot a system that must be manually fixed. Redundancy for security is best applied in the context of a self-healing or resilient threshold structure that explicitly manages the redundancy as a security mechanism not as un-managed inadvertent redundancy.

Equivalently a bare OOBI (no AID) provides a mechanism for blind introductions, i.e. a blind or bare OOBI (BOOBI). Because the OOBI does not expose an AID, the the resultant response when querying the OOBI may depend on other factors such as the source IP of the querier (requester) and/or another out-of-band-authentication (OOBA) mechanism. This supports private bootstrap of infrastructure. Of course one could argue that this is just kicking the can down the road but IP addresses are correlatable and a blind OOBI can leverage IP infrastructure for discovery when useful in combination with some other OOBA mechanism without unnecessary correlation.

OOBI Forwarding

In every case an OOBI may result in a proof for a different URL than that provided in the OOBI itself. The allows OOBI forwarding so that introductions produced as hard copies such as QR codes do not necessarily become stale. The recipient of the OOBI may choose to accept that proof or not. Ultimately the recipient only treats URLs as valid endpoints when they are fully KERI authenticated. Given that an OOBI result is always KERI authenticated before use in a given role, the worst case from a security perspective is that an OOBI may be part of a DDOS attack but not as part of a service endpoint cache poison attack.

OOBI KERI Endpoint Authentication (OKEA)

Upon acceptance of an OOBI the recipient queries the provided URL for proof that the URL is an authorized endpoint for the given AID. The proof format may depend on the actual role of the endpoint. A current witness for an AID is designated in the current key state's latest establishment event in the AID's KEL. Therefore merely replying with the Key State or KEL may serve as proof for a witness introduced by an OOBI. Other roles are not part of key state (i.e. are not designated in KEL establishment events) and therefore must be authorized by another mechanism. This typically will be a signed /end/role/ reply message. So the query of the OOBI URL could return as proof an associated authorizing reply message. For example:

```
{
          "v" : "KERI10JSON00011c_",
          "t" : "rpy",
          "d": "EZ-
i0d8JZAoTNZH3ULaU6JR2nmwyvYAfSVPzhzS6b5CM",
          "dt": "2020-08-22T17:50:12.988921+00:00",
          "r" : "/end/role/add",
          "a" :
          {
             "cid":  "EaU6JR2nmwyZ-
i0d8JZAoTNZH3ULvYAfSVPzhzS6b5CM",
             "role": "watcher",
             "eid": "BrHLayDN-
mXKv62DAjFLX1_Y5yEUe0vA9YPe_ihiKYHE",
          }
 }
```

OOBI with MFA

An OOBI may be augmented with one or more Out-Of-Band Authentications (OOBAs) to minimize the likelihood of a DDOS OOBI attack. A given recipient may require as a precondition to accepting an OOBI one or more OOBA mechanisms such as text messages, emails, etc that together provide some degree of non-KERI based security to the OOBI. Thus an OOBI could employ out-of-band (with respect to KERI) multi-factor-authentication (MFA) to preclude any OOBI based DDOS attacks on KERI.

KERI OOBI Use in Installation Configuration

OOBI Discovery

The main value of an OOBI is that it is compact and is not encumbered by authentication proofs but may be used to kick-start the process of authentication (proving).

One way to pre-configure a vacuous KERI installation is to provide OOBIs in a configuration file. The bootstrap process of the installation then queries the associated URLs to retrieve the KERI authentication proofs (BADA) that then are used to populate its database securely. This simplifies the configuration file.

In contrast, an alternative would be to populate the configuration file with the KERI authentication proofs. But these proofs may be quite verbose and cumbersome and may make the config file somewhat difficult to manage in human readable/writable form. Furthermore if one already had the proofs one could just pre-populate the database with those proofs. Therefore OOBI based configuration fiels may be advantageous as either easier to manage and as a viable option when the proofs are not yet available at configuration time.

Furthermore a clean clone replay restart of a given KERI component is designed to fix any unverified corruption of its associated KELs.
If each component uses OOBIs to retrieve the authentication proofs from other components then all the components will have the cleaned proofs instead of stale proofs.

OOBI Response

Each KERI installation may also optionally provide an OOBI permissioning record list associated with each habitat to indicate which OOBI queries it will respond to. This may also be inited with a config file.

Addendum

This is an attempt to provide configuration information to an agent to enhance the current OOBI resolution process with a well known verification process for multi-factor authentication of an AID provided by an OOBI in the configuration file. This process will be used in the vLEI ecosystem configuration files to bootstrap resolution and authentication of the GLEIF RoOT and GLEIF External AIDs for all agents in the ecosystem.

The proposed approach would be to add a new top level key to the configuration file (murls or wurls) that contains a list of well-known OOBIs with AIDs. The OOBI resolution process would be updated such that after the successful resolution of endpoints for an AID from an OOBI URL in the iurls section, the system would scan the well-known URLs in array in this new key for all URLs that contain the AID just resolved. The system would then resolve each URL by making a successful HTTP request against that URL. A successful HTTP request would be one that returns either a 2xx or 3xx HTTP status code. After successfully resolving all URLs for the AID, the system would make the contact created for the new AID with a new authenticated status.

Currently, the Keep software sets a verified field to True after a successful 12-word challenge response has been received from the AID that was just resolved with an OOBI. To accommodate this new method of multi-factor authentication, we would change the field name on the contact to authenticated and change the value from boolean to a status value representing the type of authentication that was performed. In addition, we would add an authenticated_date field to the contact to indicate the date and time on which any MFA was performed.

The proposed status values are SIGNED_CHALLENGE for the current Keep challenge / response authentication and WELL_KNOWN for the new well-known resolution challenge response.

Given the following configuration file:

```
{
"dt": "2022-01-20T12:57:59.823350+00:00",
"iurls": [
"http://127.0.0.1:5642/oobi/BBilc4-L3tFUnfM_wJr4S4OJanAv_VmF_dJNN6vkf2Ha/controller",
"http://127.0.0.1:5644/oobi/BIKKuvBwpmDVA4Ds-EpL5bt9OqPzWPja2LigFYZN2YfX/controller",
"http://127.0.0.1:5643/oobi/BLskRTInXnMxWaGqcpSyMgo0nYbalW99cGZESrz3zapM/controller",
"http://20.121.171.161:7723/.well-known/keri/oobi/EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG?name=GLEIF Root"
],
"durls": [
"http://127.0.0.1:7723/oobi/EIL-RWno8cEnkGTi9cr7-PFg_IXTPx9fZ0r9snFFZ0nm",
"http://127.0.0.1:7723/oobi/EJEMDhCDi8gLqtaXrb36DRLHMfC1c08PqirQvdPPSG5u",
"http://127.0.0.1:7723/oobi/EDqjl80uP0r_SNSp-yImpLGglTEbOwgO77wsOPjyRVKy",
"http://127.0.0.1:7723/oobi/EOhcE9MV90LRygJuYN1N0c5XXNFkzwFxUBfQ24v7qeEY",
"http://127.0.0.1:7723/oobi/EK0jwjJbtYLIynGtmXXLO5MGJ7BDuX2vr2_MhM9QjAxZ",
"http://127.0.0.1:7723/oobi/ED_PcIn1wFDe0GB0W7Bk9I4Q_c9bQJZCM2w7Ex9PIsta",
"http://127.0.0.1:7723/oobi/ELqriXX1-
lbV9zgXP4BXxqJlpZTgFchll3cyjaCyVKiz"
],
"murls": [
"https://www.gleif.org/.well-known/keri/oobi/EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG",
"https://github.io/WebOfTrust/vLEI/.well-known/keri/oobi/EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG",
"https://github.io/GLEIF-IT/vLEI/.well-known/keri/oobi/EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG",
"https://www.first-roc-member.com/.well-known/keri/oobi/EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG",
"https://www.another-roc-member.com/.well-known/keri/oobi/EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG"
]
}
```

The resolution of the OOBI for AID EL0QKj5uCRRAawr91sVWCVvKM1XNMQ3WJGo_g0O9s-BG from the iurls array (the last one in the list) would result in a redirect to a witness or controller endpoint that would be saved in the database and associated with the contact with the alias “GLEIF Root”. After completion, the system would scan the array at the key murls and find 5 well-known URLs that contain the AID just resolved. If would perform an HTTP GET request on each URL in turn and if every URL results in a 2xx or 3xx status code it would update the contain for the new AID to the following (JSON representation):

```
{
"alias": "GLEIF Root",
"authenticated": "WELL_KNOWN",
"authenticated_date": "2022-10-02T12:57:59.823350+00:00"
}
```

Nothing would prevent the controller of the agent using this configuration file to perform a 12-word challenge / response protocol with the new AID in the future and updating the values in the contact with a new authenticated status and new date in authenticated_date. In this way the controller can leverage the MFA provided in the configuration file but later upgrade the authentication by engaging in a live session challenge.

Consideration was given to allowing for a threshold to be specified for indicating how many URLs in the murls key need to succeed for a given AID for it to be considered authenticated. Since the configuration file is under the control of the distributors of the software, stale values will not affect controllers that have already authenticated the AIDs during their initial start up. There is no need to allow for permanent failures of well known URLs because they can be removed from the configuration file for any new installations of the software. Controllers with older versions will have already performed the initial set up and no longer need a full set of valid well knowns. Temporary failure of well knowns can be accounted for with a simple retry mechanism that is already in place for all OOBI resolutions.

This method can be extended in the future to allow for both a command line version of well known authentication as well as agent based verification. All that would be required is additions to the current kli oobi resolve command to include specifying a list of urls (multi-option collection) as an option to the command as well as an addition key pointing to an array in the body of the POST / oobi endpoint in the agent.

Samuel Smith

10/03/22

Suggest that the authenticated field value could be a list so that both WELL_KNOWN and SIGNED_CHALLENGE types of authentication could be applied to the same AID and not either or. In the future other types of authentication could then be supported.

Samuel Smith

31m

Just noticed that I could not find any documentation of the field labels in the config file .json

I know we discussed these someplace but couldn’t find it.

So we should create a json schema for config files that in the description field of each label defines

what the label means