khagesh App to App communication

d3c1ffe on Nov 21, 2018

1 contributor

98 lines (62 sloc) 6.27 KB

Inter Edge agent communication

Multiple Edge agents can run inside one device or one or more agents on another device. This document tries to clarify different communication strategies that can be used when edge agents are running on same device and on different devices.

Same device, communication among multiple edge agents

Apps running on Android and iOS devices

We will discuss approaches in which we can communicate with multiple edge agents running inside multiple apps

1. Using Deep Link url
2. Using Content type

Pre-requisite for app's edge agent to app's edge agent communication

Apps which are participating in edge agent communication should define a unique url/scheme of their own. Here are following advantages to this approach

Only for Rich Agents:

- Define a scheme in App's Manifest file (Android) and Info.plist file(iOS) which points to `ssi://`

For both Rich and thin agents:

- Once rich agent has done processing of data and is ready with response, now rich agent needs to send this data back, this unique scheme of another agent would be used directly to pass response back without any need for network or any other hop

- Once discovery is done, and a secure connection is established, we can directly open app with which connection is established.
- We can open apps directly without showing an intermediate UI to user
- Pairwise connection can be used as long as other app remain installed on device

Using Deep Link

Deep link is almost similar to a URL on web. Generally this is how a deep link looks in case of mobile apps. `ssi://data?q=1` . Here the first part of url which is known as scheme (`ssi://`) decides what app should be opened and then app get access to whole deep link through which app was opened. Now app can get path and query params and decides to take action on it. Android, iOS

Message format

- It would have two fields
- `message` , is message that one agent wants to send to another agent. This message format would be same as we are already using in our Agent-to-Agent communication. Once we have message packed data, then we would encode message and that would be value of `message` field
- `return_url` : this is the scheme url to which rich agent sends data back after response to `message` message is ready

Pros:

- No network connection is needed
- Sovrin rich agent enabled app can be opened directly

Cons:

- Cannot transfer binary data directly, it either has to be encoded or a binary file uri has to be passed

Using Content type

Content type in apps can be thought of as same as MIME type on web. The way HTTP response headers define mime type of a content and then browser chooses to use an extension that can handle that mime type content to display it. Apps can also register to handle content on the basis of type. Some details about this on iOS and Android

Message format

It would be a message packed file which will have two fields, same as Deep Link's message format. It would be result of writing message packed data to file and expose the file via ContentProvider (Android) or UIDocumentInteractionController(iOS)

```
msgPackedData = msg_pack({ message: <A2A message>, return_url: "fitbitssi://ssidat
file = writeToFile(msgPackedData)
```

Pros:

- No network connection is needed
- Can transfer binary data directly

Cons:

- Android and ios has different construct to support it

Discovery of rich agent on a device

This describes the way in which agents can probe if a rich agent is available on device or not. We are considering that rich agents have explicitly defined schemes in their Manifest/Info.plist files. Once that is done for each rich agent app, any app can use following for discovery:

- On Android, we can use `packageManager.queryIntentActivities`
- On iOS, we can use `canOpenUrl`

No pairwise connection message communication

This part describes the format of `message` which is A2A encrypted packed message. If an agent does not have a pairwise connection with another edge agent on device, then `message` field would have a bundled message that would have a connection invitation along with message that one agent wanted to pass to another agent. Format for this type of message is still in progress by Doug.

After pairwise connection message communication

Once a connection is established, both agents knows `return_url` scheme and `pairwise DID` . So that messages can be encrypted & packed and data can be passed between agents without further discovery. Message format in this case will be same as our existing A2A messages for credentials, proofs, etc.

Authentication and Authorization for messages

User would see an app being opened from other app. If user wants to share proof or make a connection from rich agent app to the app that user was using, then user can share the data in encrypted format to other app. Both permission for AuthN and AuthZ are given by user in edge agent communication case. Since our approach is dependent on message passing and an app interface to allow responding to message, a rich agent app would display some UI which would authenticate user and would show message from other agent, and would ask user permission to fulfill request and send data back.

Different message type requirement

We can have requirement for different message types. For example:

- Get data specific to one connection
- Get metadata
- Get zero knowledge proof
- Save data to rich agent Almost all of the message types can be satisfied with existing credential offer, credential request, credential, proof request and proof based message which are defined in our A2A communication protocol.

Apps running on IOT devices, or wearables, or similar low spec devices

TODO

Edge agent on Different devices

TODO