Fully Decentralized Twitter (Dewitter) App Scenario: Platform Requirements

Michael Herman

Trusted Digital Web
Hyperonomy Digital Identity Lab
Parallelspace Corporation

#FirstPrinciples Thinking

- *Sometimes called "reasoning from first principles," the idea is to break down complicated problems into basic elements and then reassemble them from the ground up. It's one of the best ways to learn to think for yourself, unlock your creative potential, and move from linear to non-linear results.*
  - First Principles: The Building Blocks of True Knowledge (https://fs.blog/2018/04/first-principles/)

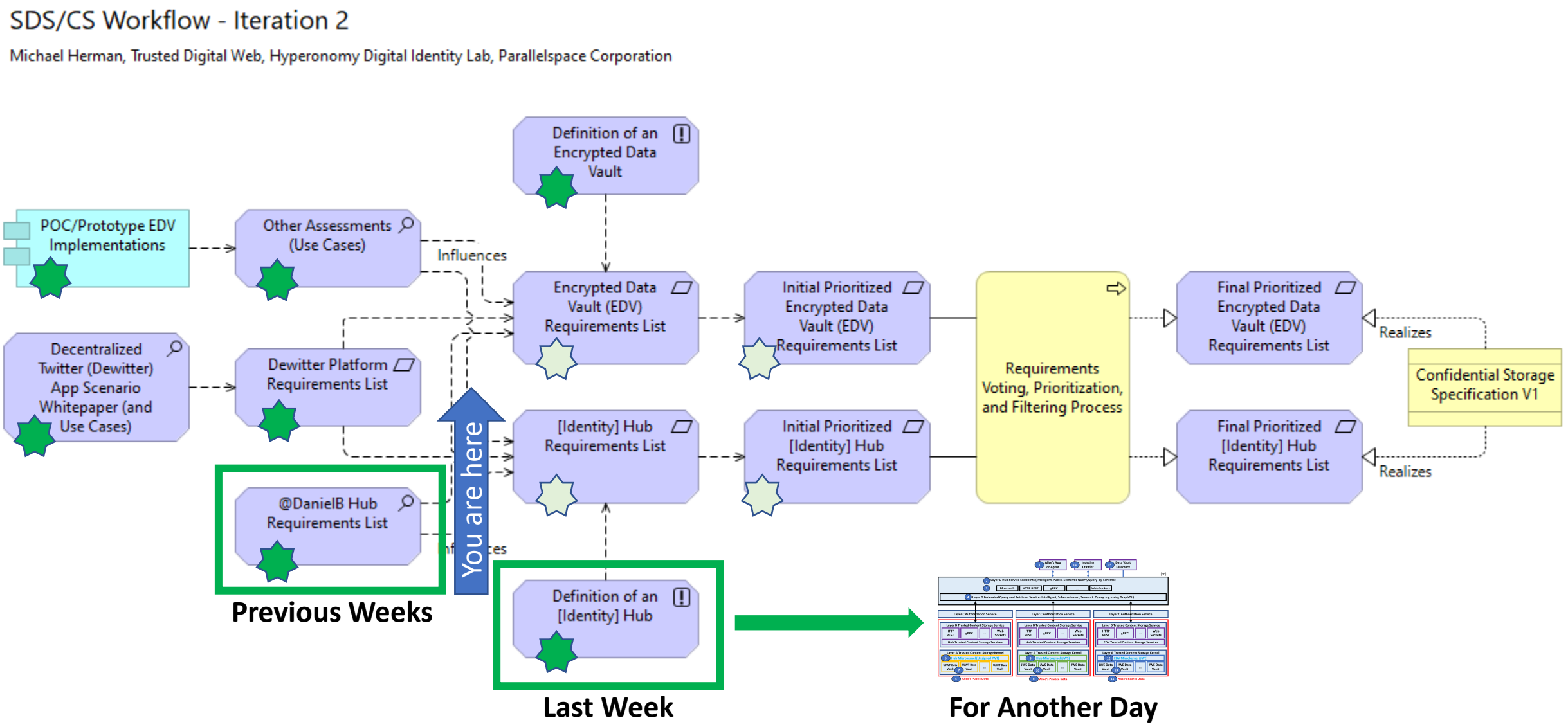
- *I think it is most important to reason from first principles rather than by analogy. One of the ways we conduct our lives is we reason by analogy. We do this because something was like something else that was done or it was like what other people were doing. It's mentally easier to reason by analogy rather than from first principles.*
  - First Principles Method Explained by Elon Musk (https://www.youtube.com/watch?v=NV3sBlRgzTI)

- Example: Why can't the Spring change off of daylight savings time happen in the middle of a workday?
  - …instead of the middle of the night

Discussion Flow

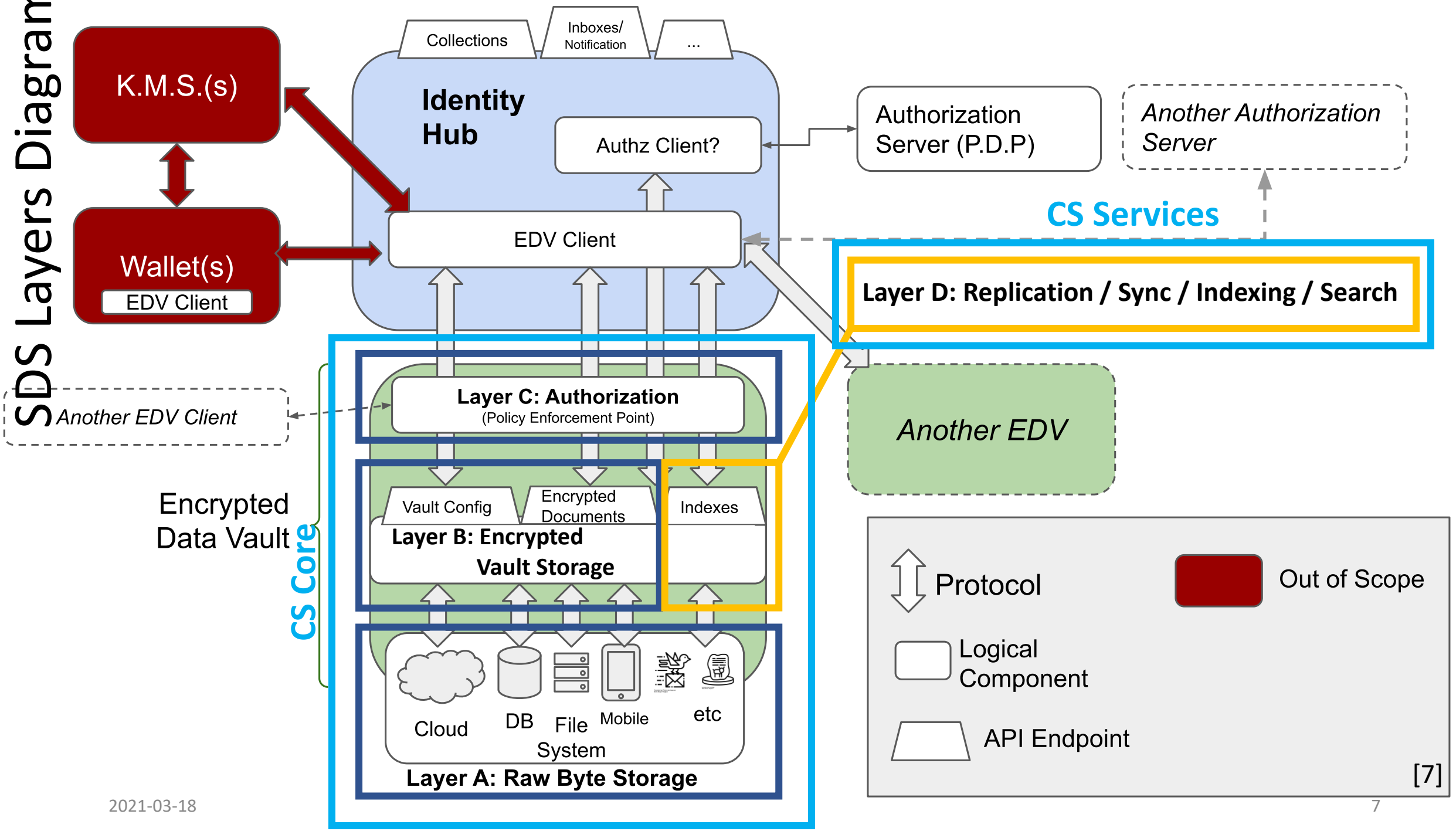
- Where are we and how did we get here?
- Context
  - Confidential Storage Stack (CS Stack) (aka SDS Layers Diagram)
  - Trusted Content Storage Stack (TCS Stack)
  - Dewitter App Scenario
    - Assumptions, Principles, Requirements, and Other Considerations
    - Tweet Item Data Model
    - Personal Agent Protocol Operations
  - Dewitter App Scenario Use Cases
- Dewitter App Scenario Requirements List

Where are we and how did we get here?

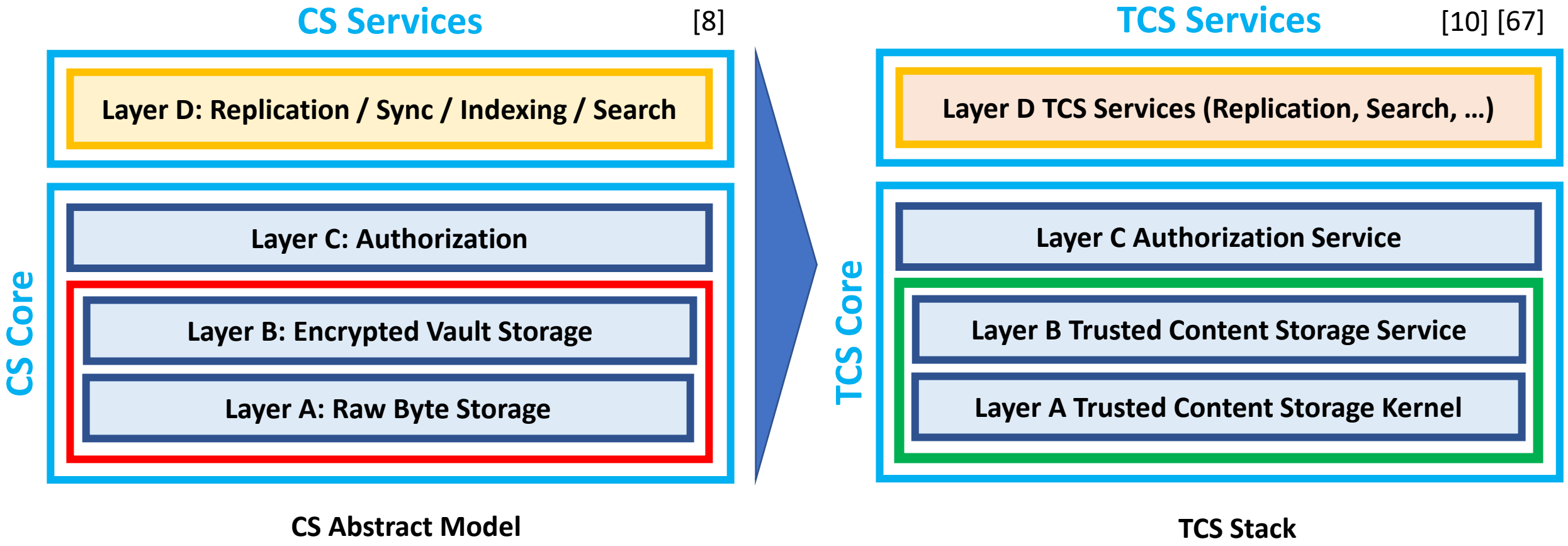


Confidential Storage Stack (CS Stack)
and
Trusted Content Storage Stack (TCS Stack)

SDS Layers Diagram



Confidential Storage Stack (CS Stack) mapped to the Trusted Content Storage Stack (TCS Stack)



Dewitter App Scenario: Assumptions, Tweet Item Data Model, & Personal Agent Protocol Operations

NOTE: Every assumption, detail, protocol operation, and use case has a unique sequential, numeric identifier.

Assumptions

1. General Assumptions
   a. *Fully decentralized* => no centralized servers
   b. Each actor is assumed to have a local device (i.e. smartphone, laptop, and/or tablet)

- Out of Scope
   a. Distributed Twitter (Distwitter) – discussed briefly in the App Scenario whitepaper
   b. Addition of centralized servers (e.g. EDV hosting, etc.)
   c. Offline Personal Agents
   d. Offline personal Local EDV Server Instances
   e. Data Metrics and Analysis
   f. Synchronization of Account (Identity) Containers across Dewitter Data Vaults attached to different Local EDV Server Instances
   g. #hashtags

2. Actors

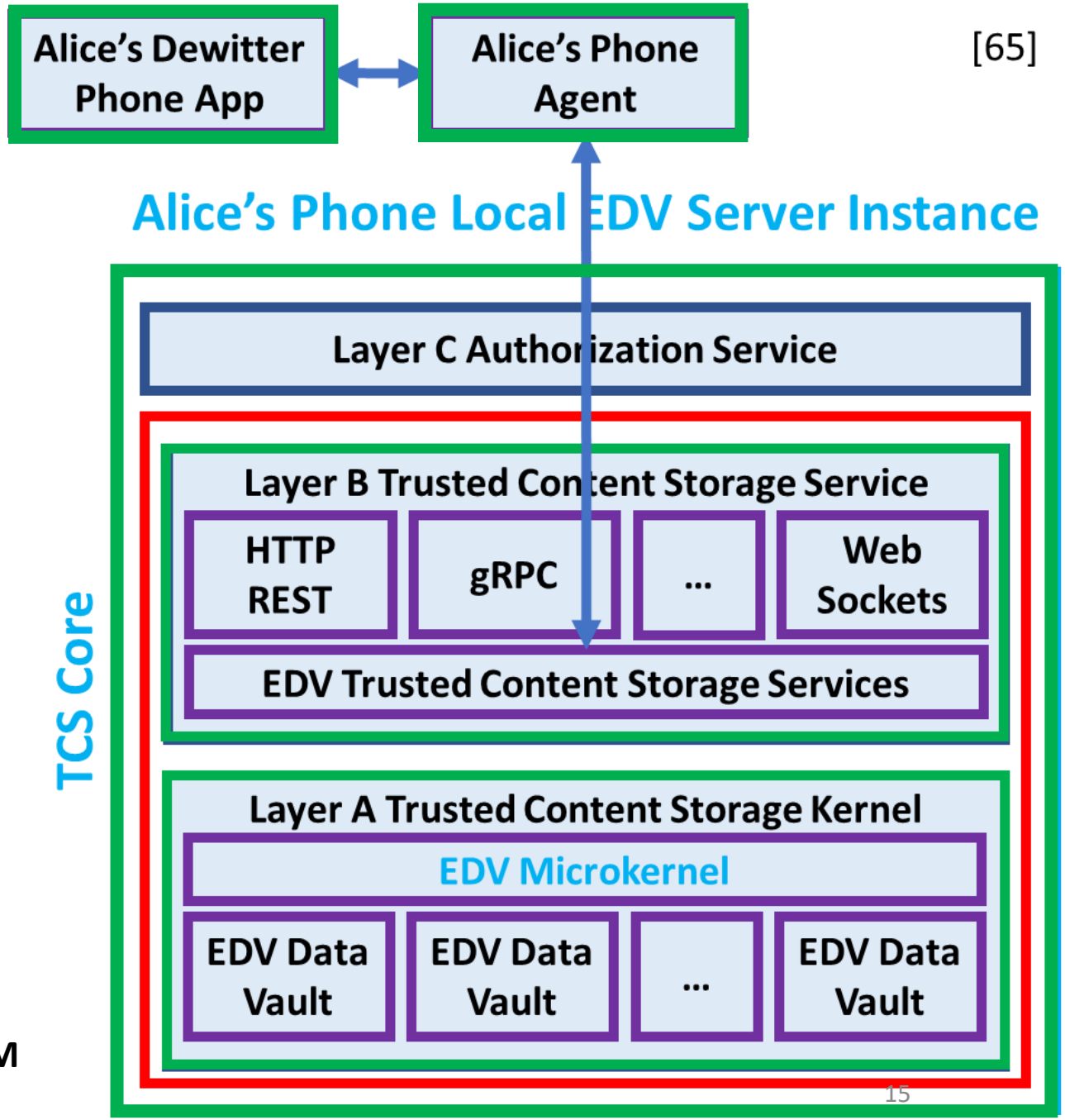
Actor	Description
2.1. Alice	Alice is a Dewitter user with 2 Dewitter accounts (identities): @alice and @alicecooper. Alice uses Dewitter on 2 devices: Alice's Phone and Alice's Laptop. On Alice's Phone, Alice uses both identities. On Alice's Laptop; Alice only uses the @alice identity. When Alice isn't using Alice's Laptop, Alice closes Alice's Laptop.
2.2. Bob	Bob is a Dewitter user with 1 Dewitter account (identity): @bob. Bob uses Dewitter on 1 device: Bob's Phone. On Bob's Phone, Bob uses the @bob identity.
2.3. Carol	Carol is a Dewitter user with 1 Dewitter account (identity): @carol. Carol uses Dewitter on 1 device: Carol's Tablet. On Bob's Phone, Bob uses the @bob identity.

4. Decentralized Twitter (Dewitter) ARM

[65]

Based on the Detailed TCS Stack

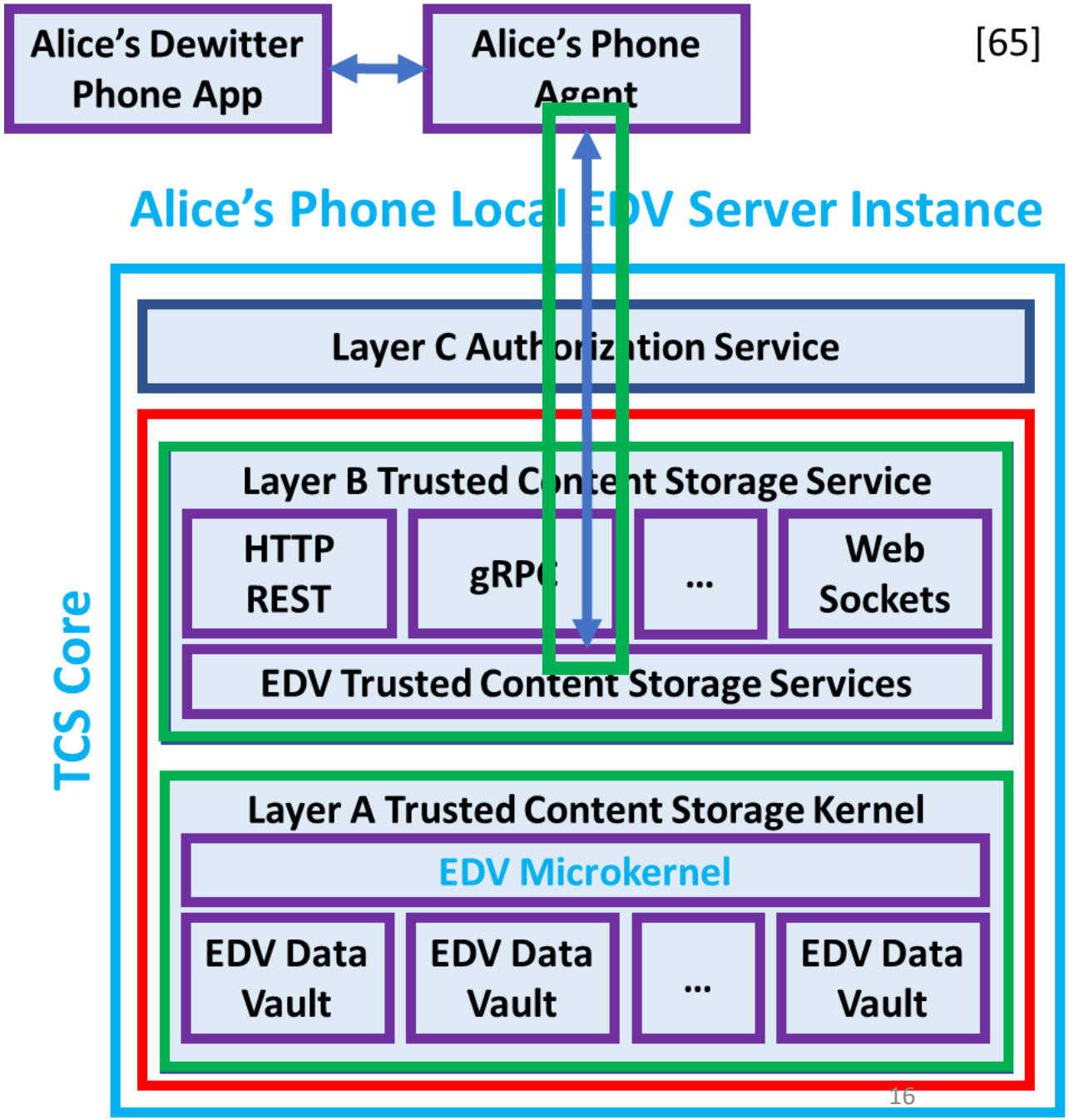
1. Personal Local EDV Server Instance
   - TCS Stack model
2. Personal Agent
   - Aries Agent/DIDCOMM model
3. Dewitter Phone/Laptop/Tablet App
   - Your favorite mobile app platform



Decentralized Twitter (Dewitter) ARM

4. Decentralized Twitter (Dewitter) ARM

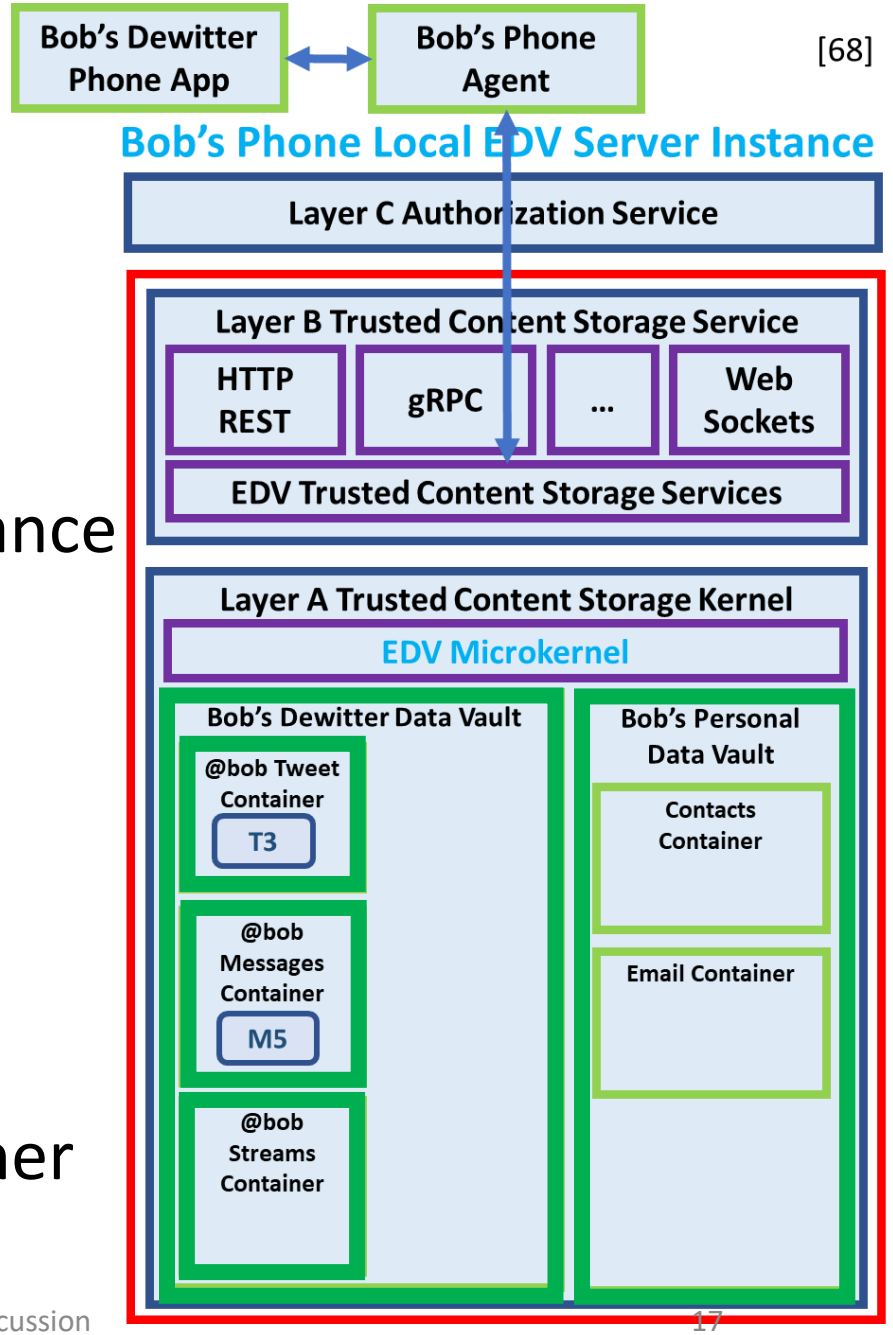
- Personal Agent to Local EDV Server Instance Communications
  - Direct API
  - Direct access to Resources stored in EDV Data Vault Containers
  - Not using a remote protocol



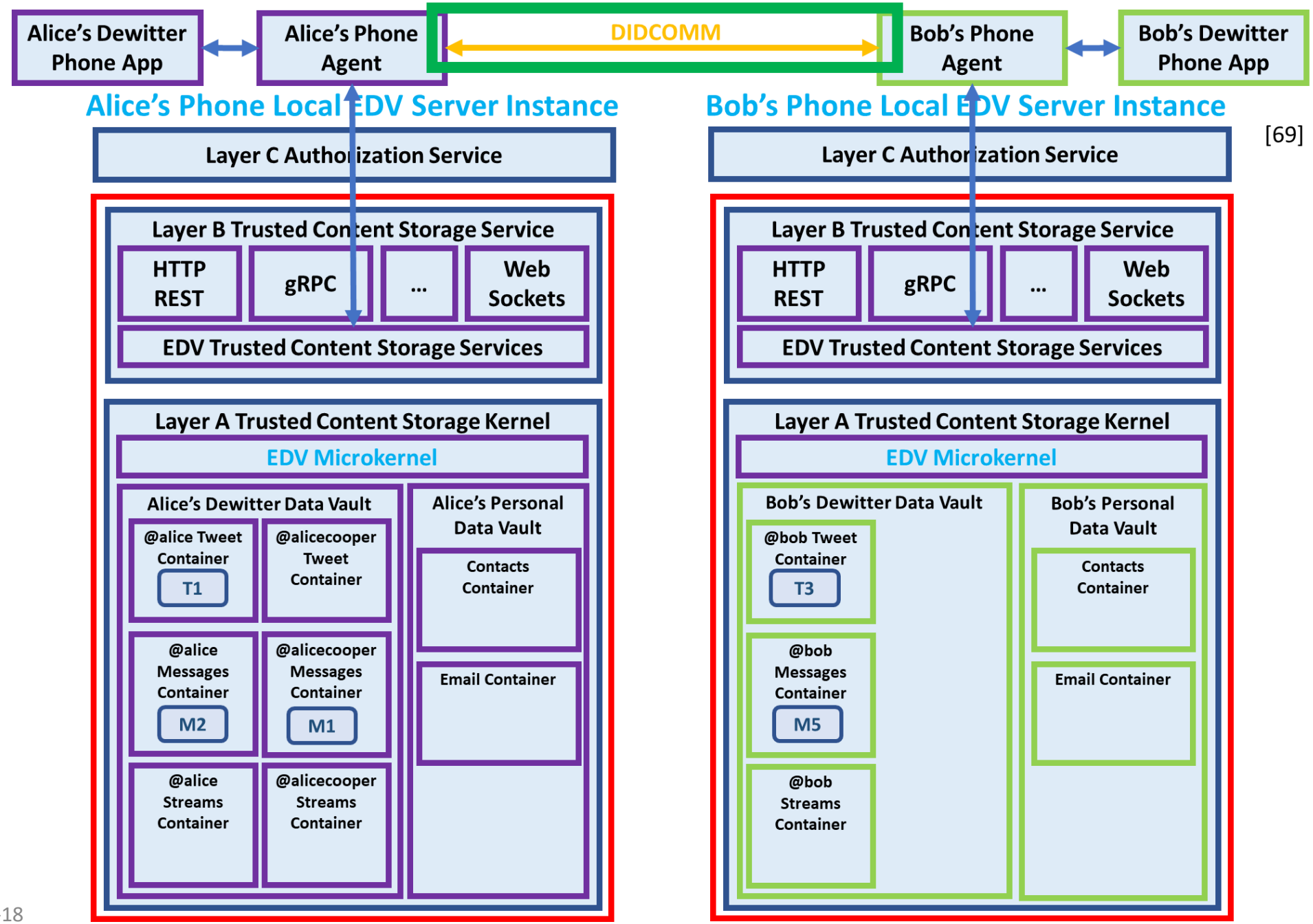
Decentralized Twitter (Dewitter) ARM

4. Decentralized Twitter (Dewitter) ARM

1. Personal Local EDV Server Instance
   - Bob's Phone Local EDV Server Instance
2. 1+ EDVs attached to each Local EDV Server Instance
   - Bob's Dewitter Data Vault
   - Bob's Personal Data Vault
3. 1+ Containers per Data Vault
   - @bob's Tweet Container
   - @bob's Messages Container
   - @bob's Streams Container
4. A Resource is created and managed in a Container



Decentralized Twitter (Dewitter) ARM: EDV Design Details



[69]

Decentralized Twitter (Dewitter) ARM: Personal Agent to Personal Agent Communication

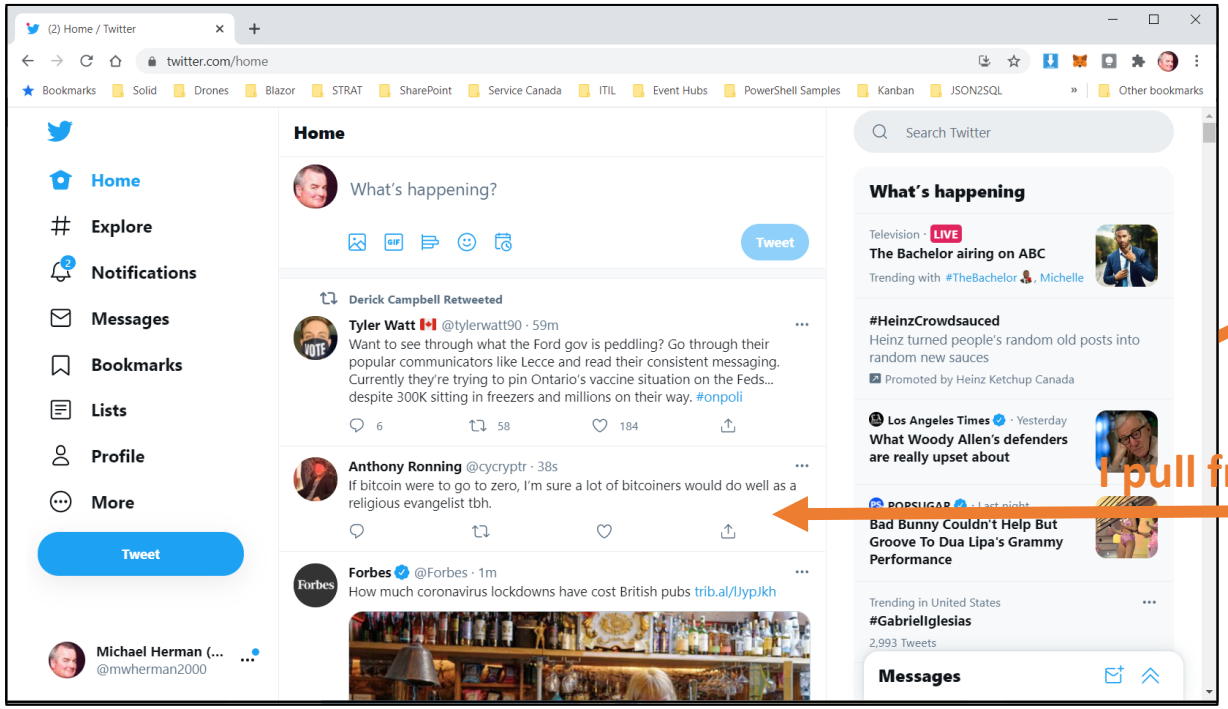
4. Decentralized Twitter (Dewitter) ARM

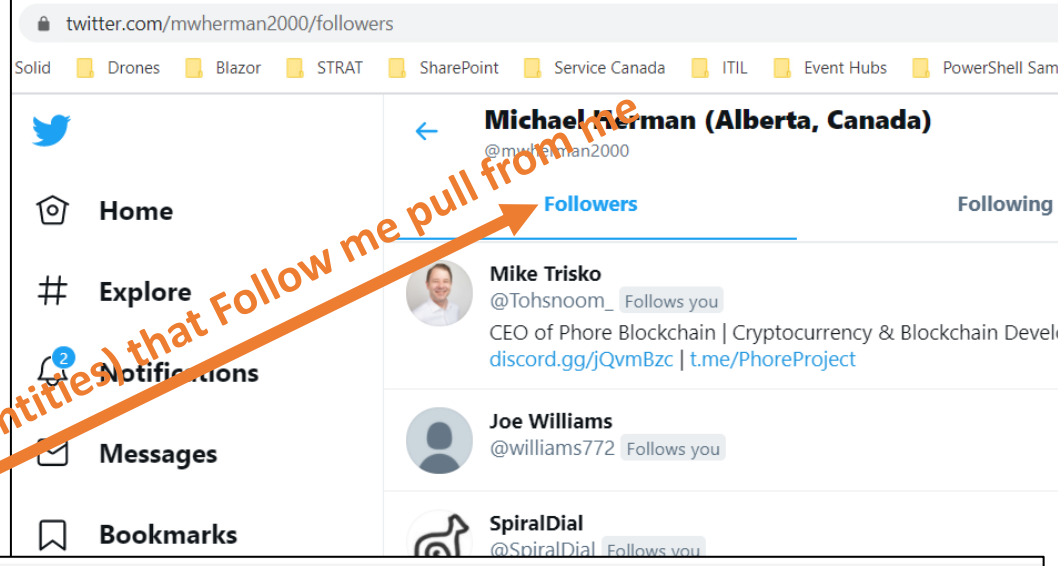
Clockwise from top-left…

1. Alice’s Phone
2. Bob’s Phone
3. Carol’s Tablet
4. Alice’s Laptop

Decentralized Twitter (Dewitter) ARM

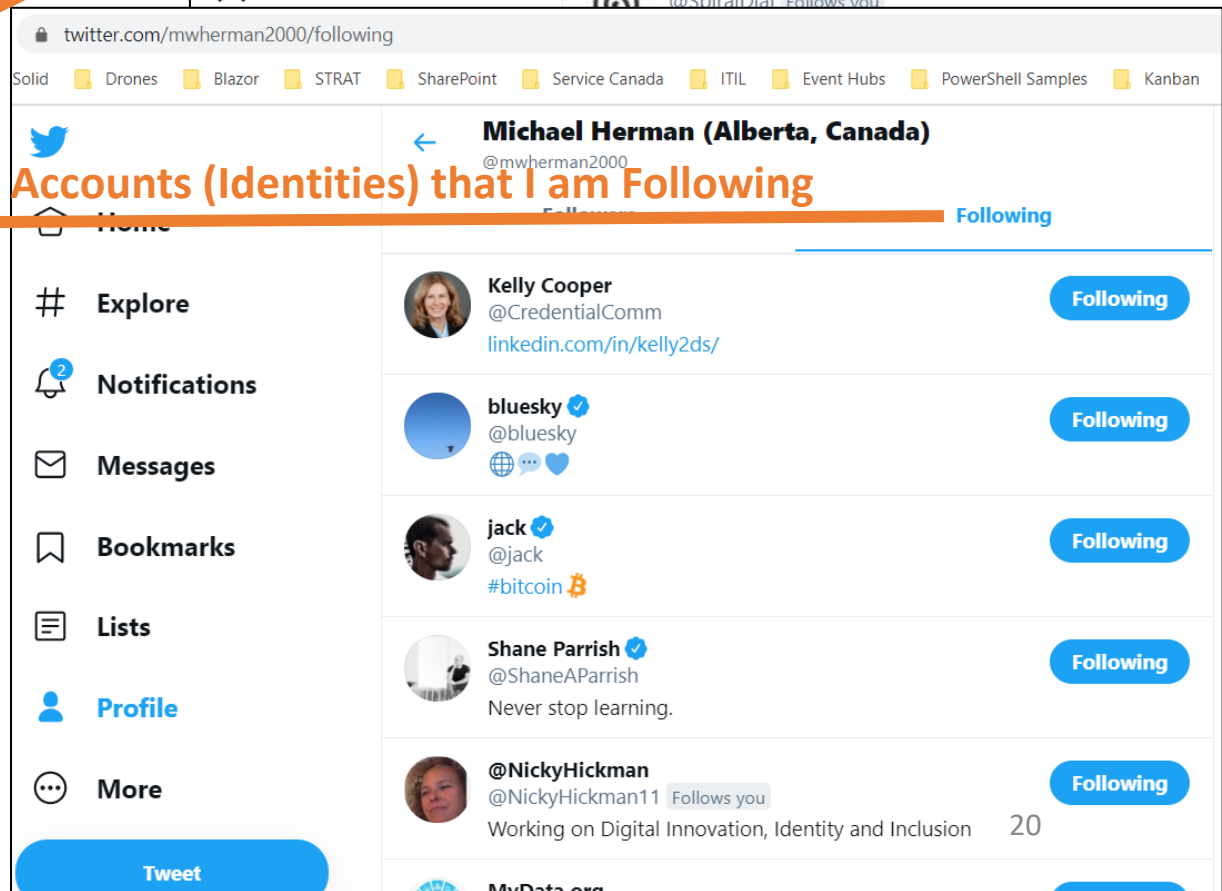
3. Friends and Followers: *Neighborhoods*





Accounts (Identities) that Follow me pull from me

I pull from Accounts (Identities) that I am Following



- Dewitter App Principles
  - "Pull"/ Query from accounts (identities) you're Following
    - No push (or writing) to another Personal Agent
    - Except for Direct Messages (DM)
    - No "pushing" supported (e.g. Advertising, News, …)
  - Focus: Create Tweets – all types
  - Focus: Query and Retrieve new Tweets from your Following
  - Blocking is supported

Tweet Item Data Model (1/3)

- 6. Tweet Items and Tweet Notifications are stored in the *Dewitter Tweet Container* in the account (identity)'s Dewitter Data Vault
- 9. Dewitter Tweet Item Types
  i. General Tweets – with/without Stream Items
  ii. General Tweets – with/without @Mentions
  iii. Replies
  iv. Retweets without Comments
  v. Retweets with Comments
  ~~vi. News Tweets~~
  ~~vii. Promoted Tweets~~
  viii. Direct Messages

Scope: Tweet Item Data Model (2/3)

- 10. Dewitter Tweet Key. A key consisting of pair of integer values:
  a. DateTimeKey: 64-bit integer value representing the time-of-day in Ticks
  b. DiscriminatorKey: 128-bit binary value representing a specific random GUID
- 12. Dewitter Tweet Notifications
- 13. Dewitter Tweet Query
- 14. Dewitter Like Counter (~~and Dewitter Like Account List~~)

Scope: Tweet Item Data Model (3/3)

- 6. Stream Items are stored in the *Dewitter Stream Container* in the account (identity)'s Dewitter Data Vault
- 15. Dewitter Stream Item Type
  i. Image
  ii. Audio
  iii. Video
- 16. Dewitter Stream Key = Dewitter Tweet Key
- 17. Dewitter Stream Query

Dewitter Personal Agent Protocol Operations

Twelve (12) Operations with Multiple Subparts

24. **Create/Update a Tweet Item or Stream Item in a Personal Local EDV Service Instance**
25. Query a Personal Local EDV Service Instance for a List of Tweet Keys or a Collection of Tweet Items
26. **Query Another Personal Agent or List of Personal Agents for a List of Tweet Keys**
27. **Query Another Personal Agent or List of Personal Agents for a Collection of Tweet Items**
28. **Block a Query from Another Personal Agent**
29. Query a Personal Local EDV Service Instance for a Stream Item
30. Query Another Personal Agent or List of Personal Agents for a Stream Item
31. Send/Receive/Accept/Block Single Tweet Item to Another Personal Agent
32. Send/Receive/Accept/Block Batch of Tweet Items to Another Personal Agent
33. **Broadcast/Receive/Accept/Block a Batch of Tweet Items to Multiple Personal Agents**
34. Send/Broadcast/Receive/Accept/Block a Batch of Tweet Notifications to Multiple Personal Agents
35. **Update a Tweet Item in Another Personal Local EDV Service Instance**

- NOTE: Need clearer separation: Dewitter App <> Personal Agent and Personal Agent <> Personal Agent

Dewitter App Scenario
Use Cases

Dewitter Use Cases

Sixteen Use Cases with Multiple Subparts – each linked back to 1+ Personal Agent Protocol Operations

36. Create (“Tweet”)
37. Notify Your Followers
38. **Follow Your Following – query and retrieve recent Tweet Items from the accounts (identities) you follow**
39. Read a Specific Tweet
40. Reply
41. Retweet (RT) without Comment
42. Retweet (RT) with Comment
43. **Direct Message (DM)**
44. **Reply to a Direct Message (DM)**
45. Search Local Personal Dewitter Data Vault
46. Search Personal Dewitter Data Vault connected to Another Personal Agent
47. Photos and Image Resources
48. **Like a Tweet***
49. Blocking
50. Muting
51. Notifications

48. Like a Tweet*

a. Read the Tweet to be Liked using operation 39 (Read a Specific Tweet).

b. Increment the Tweet's Like Counter by 1

c. Using protocol operation 35a, update the Tweet's Like Counter in the Dewitter Tweet Container in the personal Dewitter Data Vault attached to the personal Local EDV Server Instance.

d. *NOTE: The "increment and update the Tweet's Like Counter" needs to be an atomic operation executed remotely while possibly multiple Personal Agents are also trying to increment the same Tweet's Like Counter at the same time.*

e. *NOTE: Like a Tweet is very difficult to do in a decentralized system – almost impossible – without the ability to define, deploy, and remotely execute a Layer A Trusted Content Storage Kernel-level stored procedure (or the equivalent) to perform the Like counter increment operation directly on a locked resource in EDV Data Vault.*

f. *NOTE: The Tweet Like Account List is not implemented in this version of the Dewitter app scenario.*

Dewitter App Scenario Requirements List

NOTE: Every requirement has a unique sequential, alphabetic identifier.

Dewitter App Scenario Platform Requirements (1/9)

- General Assumptions

  Dewitter Use Cases: #1

  A. User Case 1b. No centralized servers are required to support Encrypted Data Vaults for phone, tablet, or laptop-based applications.
     - EDV Server Instance runs on local device.
     - Personal agent uses a direct API to access EDVs attached to the EDV Server Instance.

- Actors, Roles (Personas), and Roles (Followers, Following, Neighborhoods)

  Dewitter Use Cases: #2-4

  B. Use Case 2. Users of an EDV-based application can have multiple accounts (identities) associated with themselves and their EDV Apps.

Dewitter App Scenario Platform Requirements (2/9)

- EDV Server Instances, EDV Data Vaults, and Containers

  Dewitter Use Cases: #5

  C. Use Case 5a. Local EDV Service Instances are capable of running standalone on a personal device (e.g. phone, laptop, or tablet).

  D. User Case 5c. Each Local EDV Service Instance can host (have attached to it) 1 or more EDVs

  E. Use Case 5d. Each EDV has the capability of storing resources into a separate and secure resource Container (or the equivalent). An EDV can contain 1 or more Containers.

  F. Use Case 5x. Local EDV Service Instances automatically start and restart (after the device resumes from hibernation).

Dewitter App Scenario Platform Requirements (3/9)

- Personal Data Vaults and Containers

  Dewitter Use Cases: #6

  G. Use Case 6d. A resource is stored securely in a named Container in a personal local EDV.

- Personal Agents

  Dewitter Use Cases: #7

  H. Use Case 7e. Personal Agents can talk to each other and exchange messages (e.g. using DIDCOMM).

  I. Use Case 7f. A Personal Agent (or the equivalent) mediates all access to a personal Local EDV Service Instance, the attached EDVs, and the Containers of resources contained in the EDVs.

  J. Use Case 7h. Personal Agents automatically start and restart (after the device resumes from hibernation).

- Dewitter Data Vault

  Dewitter Use Cases: #8

  No requirement items

Dewitter App Scenario Platform Requirements (4/9)

- Dewitter Data Model Definitions

  This section describes how Tweet Items are stored in Dewitter's fully decentralized architecture.

  Dewitter Use Cases: #9-18 (Use Cases: 19-22 Reserved)

  K. Use Case 10. Any resource, optionally, can have plain-text properties associated with the resource for the purpose of querying and retrieving a single resource or a collection of resources (Simple Resource Key).

  L. Use Case 10. A resource property (Simple Resource Key) can have sub-properties (Resource Subkeys) – called a Resource Complex Key.

  M. Use Case 10. A specific Container can be queried for a single resource or a collection of resources given a Simple Resource Key or a Complex Resource Key.

  N. Use Case 14. A Local EDV Service Instance has an atomic, transaction-safe capability for updating the value of a property on a resource in a Container (e.g. incrementing the value of "counter" property on a resource).

  O. Use Case 14. A Local EDV Service Instance supports a general-purpose, atomic, transaction-safe capability for updating the value of a property on a resource in a Container *based on an arbitrary property value changing computation* on the specific resource.

  P. Use Case 15. A Local EDV Service Instance has a capability for storing, updating, querying, and retrieving stream resources from a client app or service (e.g. images, audio streams, video streams).

Dewitter App Scenario Platform Requirements (5/9)

- Personal Agent to Local EDV Server Instance Communications

  Use Cases: #23

  Q. Use Case 23a. Capability for a Personal Agent to directly access a Local EDV Server Instance running on the same device using the Layer B EDV Trusted Content Storage Services API (not via one of the Layer B Trusted Content Storage Service remote access service endpoints (e.g. HTTP, Bluetooth, etc.)).

  R. Use Case 23b. Capability for a Local EDV Server Instance Layer B EDV Trusted Content Storage Services API to talk directly to the Layer A Trusted Content Storage Kernel via an API (which, in turn, talks directly to the EDV Microkernel).

  S. Use Case 23c. An EDV Microkernel securely manages all access and operations against each of the attached EDVs.

Dewitter App Scenario Platform Requirements (6/9)

- Dewitter Protocol Operations

  NOTE: A Compound Query is a query parameter that includes multiple Resource Keys (Simple and Complex) and their key values.

  Use Cases: #24-35

- Create/Update a Tweet Item or Stream Item in a Personal Local EDV Service Instance

  T. Use Case 24a. Capability to *create* a new resource in a Container in a Data Vault attached to the Local EDV Service Instance.

  U. Use Case 24b. Capability to *update* a new resource in a Container in a Data Vault attached to the Local EDV Service Instance.

  V. Use Case 24c. Capability to *tombstone* a new resource in a Container in a Data Vault attached to the Local EDV Service Instance.

  W. Use Case 24x. Capability to ensure no one but the owner can update a resource in a Container in the Data Vault attached to the Local EDV Service Instance (*block*).

- Query a Personal Local EDV Server Instance for a List of Tweet Keys or a Collection of Tweet Items

  X. Use Case 25a. Capability to query a Local EDV Server Instance for a *list of Resource Keys* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

  Y. Use Case 25b. Capability to query a Local EDV Service Instance for a *list of Resource Items* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

Dewitter App Scenario Platform Requirements (7/9)

- Dewitter Protocol Operations

  Use Cases: #24-35

- Query Another Personal Agent or List of Personal Agents for a List of Tweet Keys

  AA. Use Case 26a. Capability to *query another Personal Agent for a list of Resource Keys* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

  BB. Use Case 26b. Capability to *query a List of Personal Agents for a list of Resource Keys* given a list of Simple Resource Keys or a Complex Resource Keys or a Compound Queries.

- Query Another Personal Agent or List of Personal Agents for a List of Tweet Items

  CC. Use Case 27a. Capability to *query another Personal Agent for a list of Resource Items* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

  DD. Use Case 27b. Capability to *query a List of Personal Agents for a list of Resource Items* given a list of Simple Resource Keys or a Complex Resource Keys or a Compound Queries.

- Block a Query from Another Personal Agent

  EE. Use Case 28a. *Capability for a receiving Personal Agent to block the receipt and acceptance of a query* for a resource from another Personal Agent.

Dewitter App Scenario Platform Requirements (8/9)

- Dewitter Protocol Operations

  Use Cases: #24-35

- Query a Personal Local EDV Server Instance for a List of Stream Keys or a Collection of Stream Items

  FF. Use Case 29a. Capability to *query a Local EDV Server Instance for a list of Resource Keys* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

  GG. Use Case 29b. Capability to *query a Local EDV Service Instance for a list of Resource Items* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

- Query Another Personal Agent or List of Personal Agents for a List of Stream Keys

  HH. Use Case 29y. Capability to *query another Personal Agent fora list of Resource Keys* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

  II. Use Case 29z. Capability to *query a List of Personal Agents for a list of Resource Keys* given a list of Simple Resource Keys or a Complex Resource Keys or a Compound Queries.

- Query Another Personal Agent or List of Personal Agents for a List of Stream Items

  JJ. Use Case 30a. Capability to *query another Personal Agent for a list of Resource Items* given a Simple Resource Key or a Complex Resource Key or a Compound Query.

  KK. Use Case 30b. Capability to *query a List of Personal Agents for a list of Resource Items* given a list of Simple Resource Keys or a Complex Resource Keys or a Compound Queries.

- Update a Resource Item in Another Personal Local EDV Service Instance

  LL. Use Case 35. Capability, *given a Resource Key, update an existing Resource Item in a Container in an EDV Data Vault attached to a personal Local EDV Server Instance*.

Dewitter App Scenario Platform Requirements (9/9)

- Dewitter Use Cases

  Use Cases: #36-51

  MM. Use Case 48. *A Local EDV Service Instance has an atomic, transaction-safe capability for updating the value of a property on a resource in a Container* (e.g. incrementing the value of "counter" property on a resource). (Same as Requirement Item N.)

- Dewitter Notes: #1-4

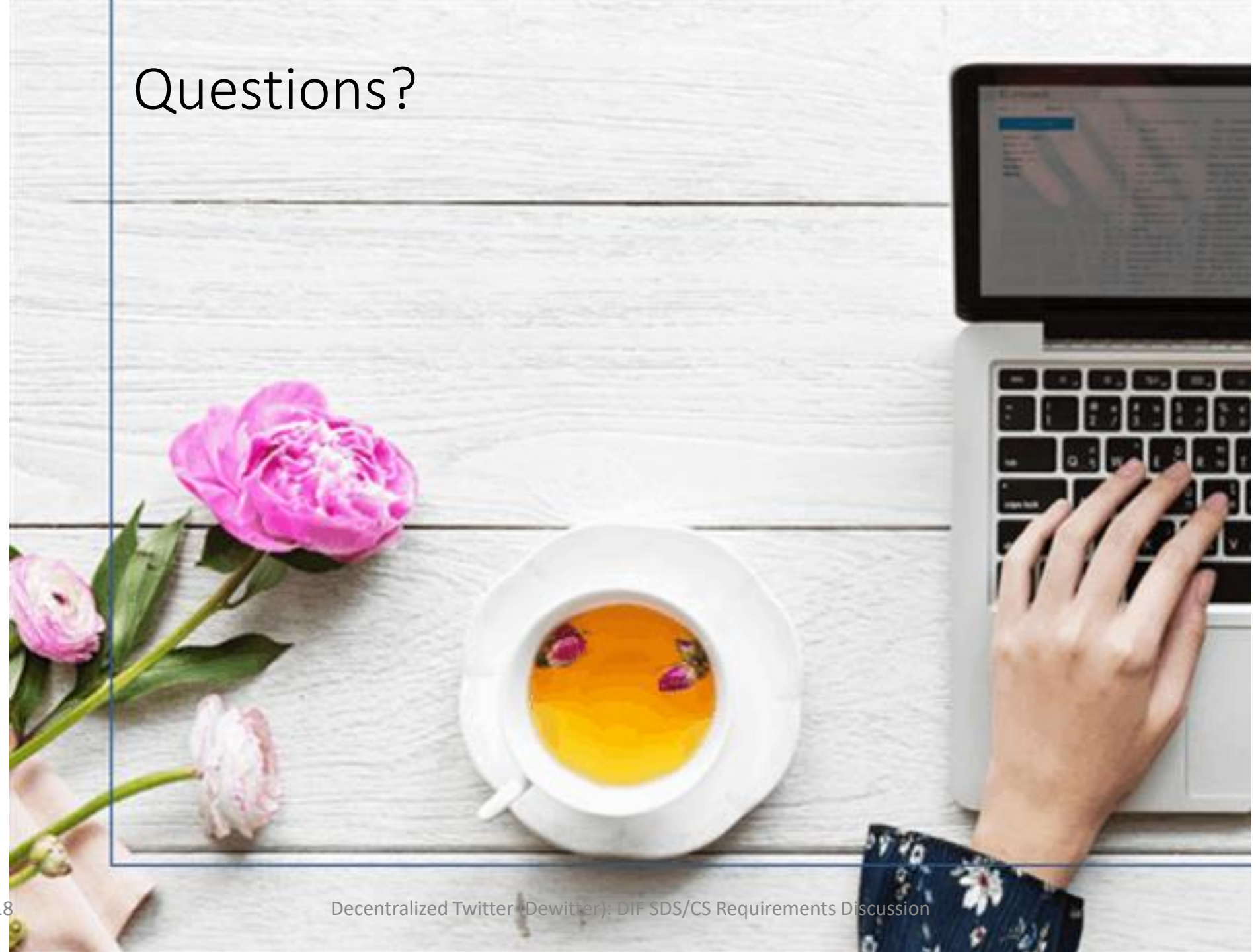
  NN. Note 1. Support for offline Personal Agents.

  OO. Note 2. Support for offline Local EDV Server Instances.

  PP. Note 3. For secure logging on an EDV-by-EDV basis: all actions against the Containers in a particular EDV stored in a secure Logging Container for that EDV.

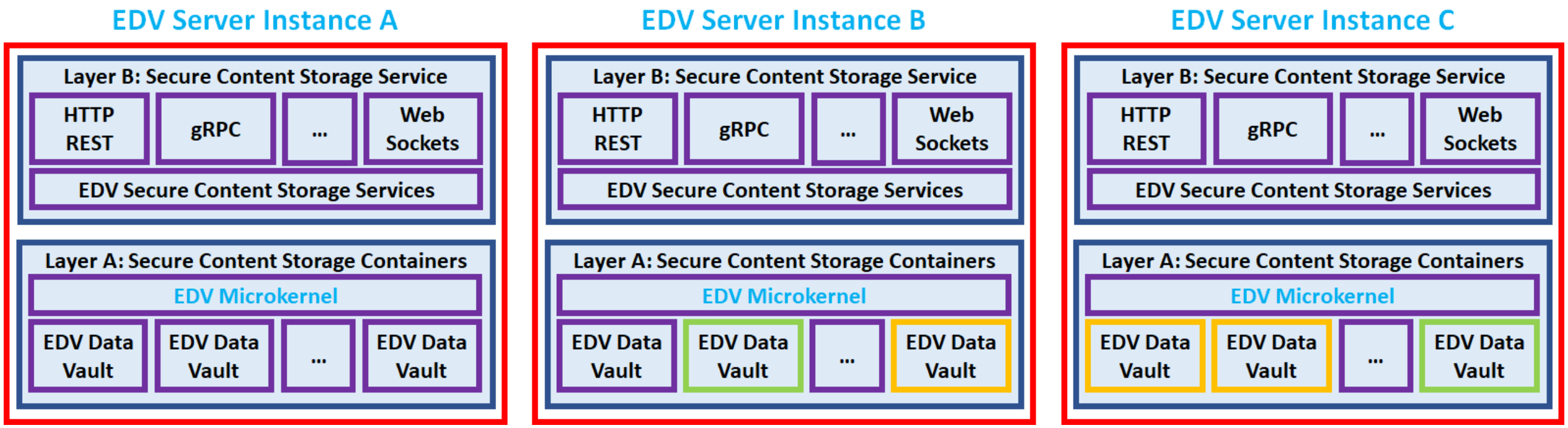
  QQ. Note 3. Support for EDV-to-EDV replication at the Container-by-Container level.

Questions?

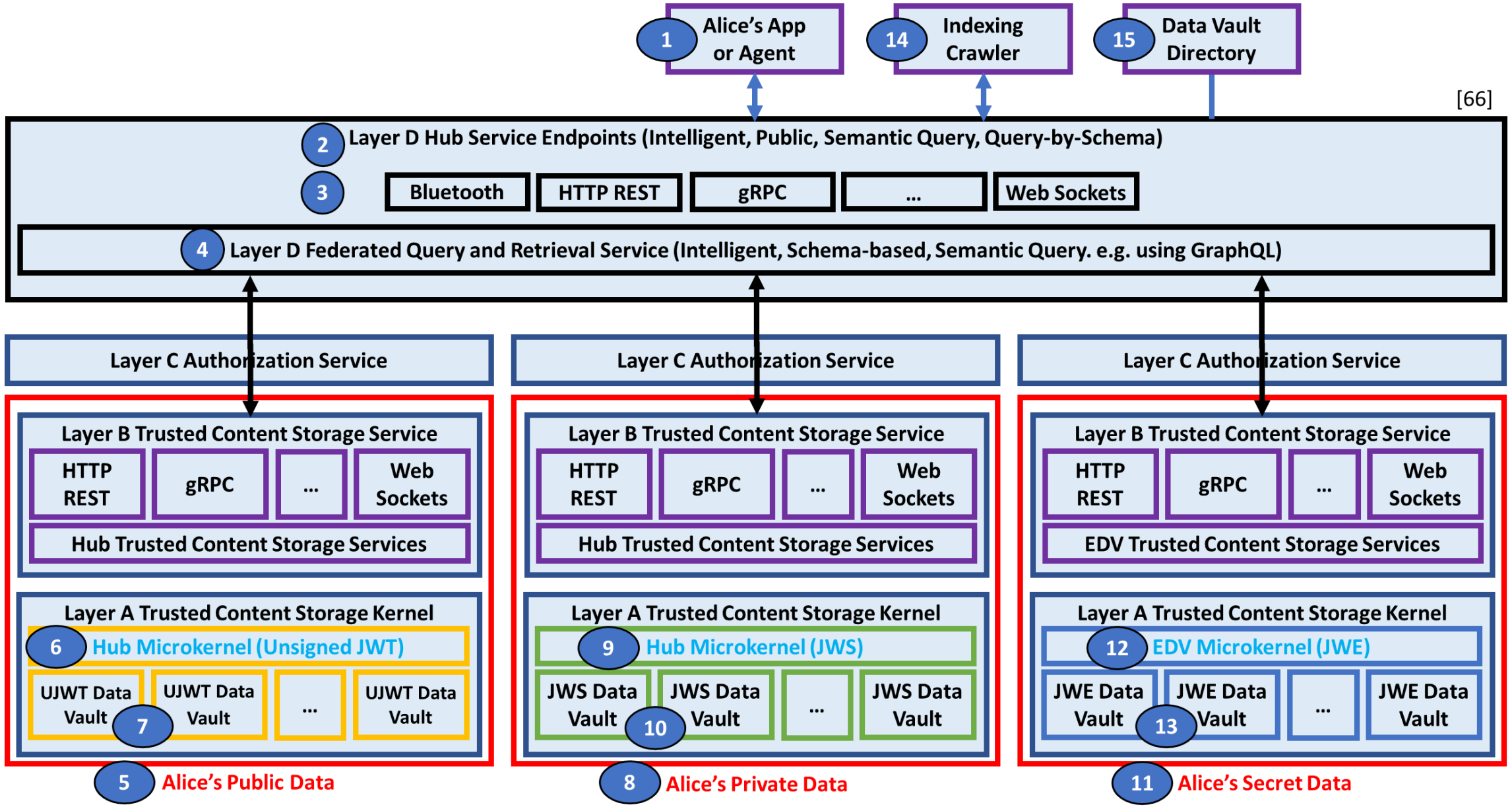


Extra Slides

EDV Architecture Reference Model (EDV-ARM)



Hub Architecture Reference Model (HUB-ARM)



Generic Replication Pipeline