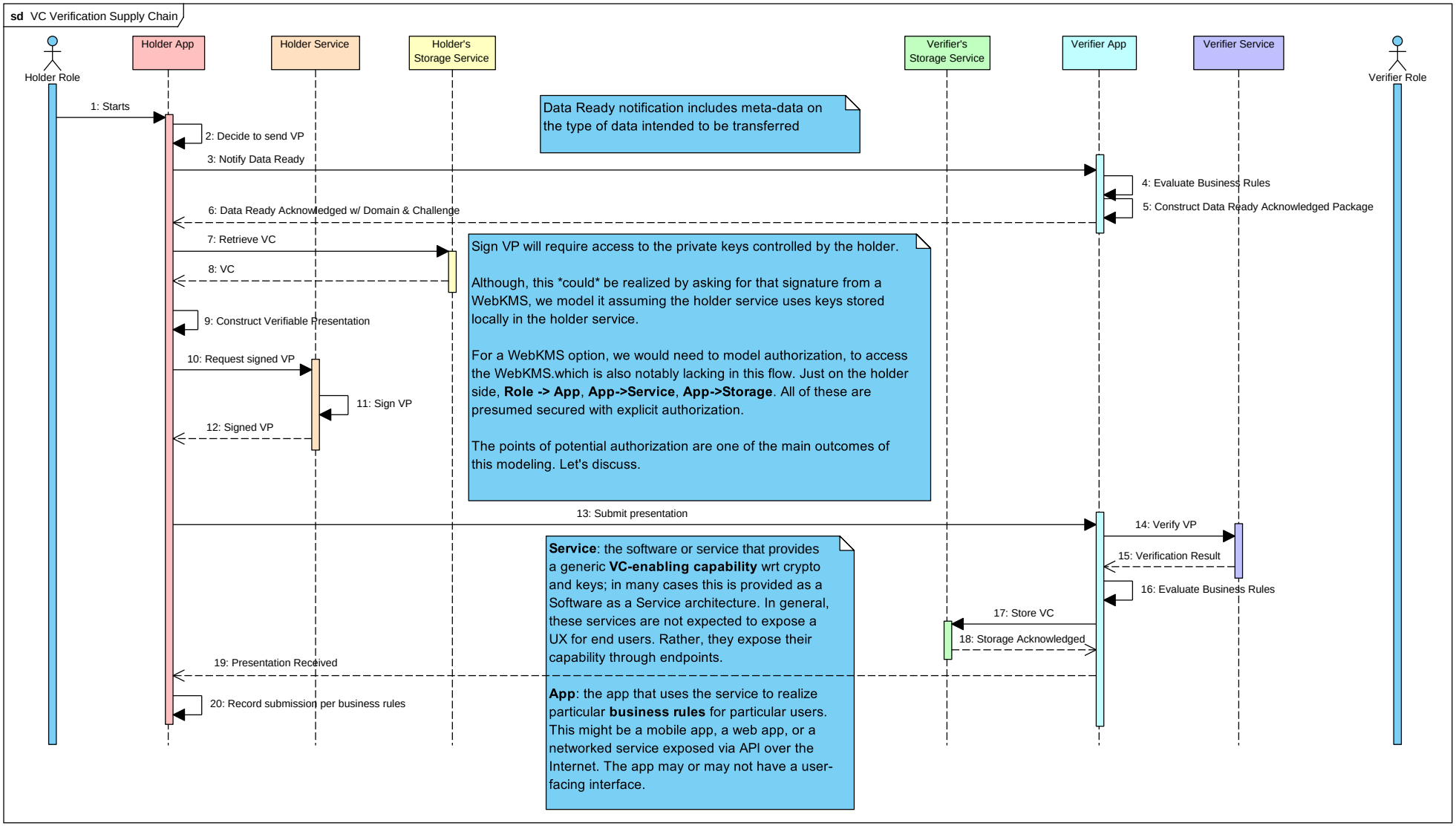
sd VC Verification Supply Chain
Holder Role
Holder App
Holder Service
Holder's Storage Service
Verifier's Storage Service
Verifier App
Verifier Service
Verifier Role
1: Starts
2: Decide to send VP
3: Notify Data Ready
Data Ready notification includes meta-data on the type of data intended to be transferred
4: Evaluate Business Rules
5: Construct Data Ready Acknowledged Package
6: Data Ready Acknowledged w/ Domain & Challenge
7: Retrieve VC
8: VC
Sign VP will require access to the private keys controlled by the holder.
Although, this *could* be realized by asking for that signature from a WebKMS, we model it assuming the holder service uses keys stored locally in the holder service.
For a WebKMS option, we would need to model authorization, to access the WebKMS.which is also notably lacking in this flow. Just on the holder side, Role -> App, App->Service, App->Storage. All of these are presumed secured with explicit authorization.
The points of potential authorization are one of the main outcomes of this modeling. Let's discuss.
9: Construct Verifiable Presentation
10: Request signed VP
11: Sign VP
12: Signed VP
13: Submit presentation
14: Verify VP
15: Verification Result
16: Evaluate Business Rules
17: Store VC
18: Storage Acknowledged
19: Presentation Received
20: Record submission per business rules
Service: the software or service that provides a generic VC-enabling capability wrt crypto and keys; in many cases this is provided as a Software as a Service architecture. In general, these services are not expected to expose a UX for end users. Rather, they expose their capability through endpoints.
App: the app that uses the service to realize particular business rules for particular users. This might be a mobile app, a web app, or a networked service exposed via API over the Internet. The app may or may not have a user-facing interface.