Wallet Attached Storage Project Explainer

August 12, 2025

About the DCC WAS Project

Wallet Attached Storage (WAS) is a cloud storage protocol designed to extend the functionality of W3C Verifiable Credential (VC) wallets while maintaining user privacy and control. WAS leverages the cryptographic key management that already exists in credential wallets to solve complex problems of cross-domain authentication and access control. Over a year-long project funded by Walmart, the DCC has been working with DID.Coop and RIPL to develop the WAS specification and develop a proof of concept for WAS functionality implemented in RIPL's DOORs platform.

The goal of the Wallet Attached Storage (WAS) project is to deliver:

1. An 0.1 version of the Wallet Attached Storage specification: a general-purpose permissioned cloud storage HTTP API (with authorization mediated by VC wallets)
2. A test storage server and client implementation of the spec
3. An example integration of WAS with DCC's open source Learner Credential Wallet (LCW). Additionally, as part of this work, the DCC has also updated VerifierPlus.org to enable the verification of raw .json files hosted on URLs.
4. Example integration with third party applications that consume or interact with credentials. In this case, the RIPL DOORs platform, a data driven tool that connects job seekers with careers and recommendations for education, training, and supportive services.

The project team is in the final stages of completing this work. You can view the specification here as well as technical documentation on the DCC Github. Sign up to the DCC mailing list for forthcoming information, including blog posts and an invitation to a public webinar sharing the outcomes and learnings of this exciting project!

Why do we need WAS?

The Digital Credentials Consortium's (DCC) work is centered around promoting the effective use and adoption of W3C VCs and aligned standards like Open Badges 3.0 (OBv3). This is because of the many advantages a standards-based approach offers to credential holders, issuers and verifiers (Learn more about VCs and Open Badges 3.0 on the DCC wiki and blog). However, while mobile VC wallets that only store a user's data on their personal device offer significant benefits in terms of data privacy and portability, they do have several limitations:

- Many credential sharing mechanisms (such as being able to add a credential to one's LinkedIn profile) require the creation of web-accessible public links. In general, sharing a credential is much

easier via sharing a URL (vs uploading the full VC as an attachment).
This means that to offer that sharing functionality, each wallet vendor needs to operate their own proprietary storage servers, increasing operating costs and liabilities.
- The creation of some VCs, such as OBv3 skill credentials for skill portfolios or resumes often requires additional cloud storage for uploading and linking to files and documents to serve as supporting evidence, or recommendations and endorsements.
- Synchronization of VC wallet contents between multiple devices also requires a cloud storage component.
- General purpose VC authoring and curation apps that are not locked into any particular platform could significantly benefit from being able to interact with user's wallets via cloud based APIs, as opposed to the limited options currently available to a mobile app when communicating with a website.

WAS provides a solution to these limitations, with "Bring Your Own Storage" functionality enabling users to connect credential wallets to cloud storage that is under their own control and allowing them to share credentials and create public links, upload supporting documents, synchronize credentials across devices and different wallets apps, and more. This "Bring Your Own Storage" approach transforms cloud storage for the credential ecosystem in much the same way that email standards transformed electronic messaging, providing wallet vendors with a cost-effective alternative to operating their own proprietary storage servers while giving users the freedom to choose and control their own storage infrastructure.

For vendors/providers of VC wallets, WAS:

- extends the functionality of mobile only wallets while retaining desirable privacy characteristics and lack of platform lock-in
- lowers the cost and liabilities threshold for new wallet providers
- encourages synchronization and interoperability between multiple wallet providers
- allows multi-device synchronization for any given wallet app
- allows for easier sharing of verifiable credentials (whether in public or access-controlled use cases)

How does it work?

Existing proprietary commercial cloud storage providers like Google Drive, Dropbox, OneDrive, etc. offer essentially the same set of features, However, they are completely incompatible with one another's APIs and permission mechanisms. Additionally, there is significant implementation friction, such as API rate limits, complex approval mechanisms, access control limitations when using existing cloud providers. Much in the same way that Email standardized and transformed the sending of electronic messages between proprietary platforms, WAS can do this for cloud storage.

WAS can significantly *extend* the functionality of wallets. But more importantly, mobile VC wallets facilitate the standardization and transformation of cloud storage by leveraging something credential wallets already have: cryptographic keys and DIDs that prove who you are.

At its core, WAS is a **protocol** and **data model** for performing familiar cloud storage operations. Specifically, creation and access/querying of:

- Spaces (similar to logical disk drive partitions in desktop storage)
- Collections (similar to file Folders, database Tables, or document store Collections)

- Resources (files and binary blobs, text, JSON objects, database rows, etc)

There have been many previous attempts at establishing a standardized general purpose storage API (from the venerable WebDAV protocol, to Tim Berners-Lee's Solid Project, to every database's essentially identical but incompatible REST API). The critical challenge that each of those projects ran into is the fact that cross-domain authentication, that is, anything more advanced than "usernames and passwords" and access control are really hard problems to solve. And that most discovered solutions to those problems involved cryptographic key management -- itself a very hard problem.

Because of this, Wallet Attached Storage is only possible since it builds upon many decades of specification and implementation work that went into Decentralized Identifiers (DIDs), Verifiable Credentials (VCs), and Authorization Capabilities (zCaps).

WAS takes advantage of the fact that in order to have a VC wallet, you need an app that creates one or more DIDs for a user, which requires key management under the hood. Any given DID contains keys for many different purposes, such as for signing VCs, for authentication, and for authorization and delegation. WAS uses the same DID technology that is used for VC operations, but instead uses it for client authentication and capability delegation for general purpose cloud storage.

WAS uses many existing specifications and standards, including:

- HTTP REST API for storage operations (although other protocol bindings, such as JSON-RPC, gRPC, DIDComm, are possible as future work)
- RFC 9421 - HTTP Message Signatures for client authentication (using wallet-managed DIDs) and capability invocation
- Authorization Capabilities (zCaps) for access control delegation

As a wallet user, what can I *do* with WAS?

At the moment, WAS allows a VC wallet user to do the following:

- **Onboard** to a WAS service provider using a compatible wallet such as LCW. A user selects a storage provider that speaks the WAS protocol, or uses the default one provided by the wallet vendor.
- **Connect** to a WAS space. The wallet creates a new Space using an existing DID as the authenticated controller, to store VCs, supporting documents.
- **Share/Unshare** a VC. When a user goes to Share a VC (or 'Create Public Link'), the wallet writes the VC to the connected WAS, sets up appropriate access control options (readable by all, or by specific parties), and produces a regular https URL usable for sharing.
- **Interface** with third-party apps and wallets, only with the user's consent.
- **Export** the contents of a connected Space to a convenient .tarball.
- **Offboard/Disconnect** from a given space or storage provider and erase all of the data stored there.

Next Steps

The scope of the DCC WAS project includes a proof of concept implementation that demonstrates the feasibility of sharing VCs with a third party application via their own storage server. However, there are a number of opportunities to explore with future support and resources. This could include:

- Advance & automate platform to storage syncing functionality
- Infrastructure that could support multiple kinds of backends including Google Drive, Dropbox, AWS, and personal servers
- Adding storage and access capabilities for Open Badges evidence files
- Exploration of how WAS can be used for recovery or backup of phone contents.

For more information about this project contact us at digitalcredentials@mit.edu. To stay informed on updates to this work sign up for the DCC mailing list here.