

Decentralized Platforms: Comprehensive Comparative Analysis

Section 5 – Platform Deep Profile: Solid (Data Pods)

Version 3.0 | 16 August 2026 | Segment 05

5.1 Overview

Attribute	Value
Full name	Solid (Social Linked Data)
Type	Specification + ecosystem for personal online data stores (Pods)
Primary identity	WebID (HTTP URI → profile document)
Core concept	User-controlled Pod + interoperable applications that request access
Official site	https://solidproject.org
Specifications	https://solidproject.org/TR/ • github.com/solid/specification
Primary design goal	Decouple data from applications; user control over data and access

Solid is the purest expression in this survey of user-controlled data ownership. Applications do not own the user's data; they request permission to read or write specific resources in the user's Pod. The user can revoke access or move the Pod.

5.2 History and Design Philosophy

Initiated by Sir Tim Berners-Lee and collaborators. Core idea: restore user agency by separating Identity (WebID), Storage (Pod), and Applications (which request access). Data lives in a user-controlled Pod; applications receive fine-grained grants; data uses open standards (RDF, LDP concepts) for interoperation. Governance involves the Open Data Institute and W3C Solid Community Group. Solid26 (2026) packages stable core specs for implementers. Linked Web Storage WG activity treats Solid Protocol as input.

5.3 Architecture



Roles: Pod (personal store), WebID (agent identity), Application (client requesting access), Identity Provider (authentication). Pods and IdPs can be separate services.

5.4 Identity: WebID

A WebID is an HTTP URI that identifies an agent and dereferences to a profile document (RDF/Turtle or JSON-LD). Example: <https://alice.example/profile/card#me>. Profile contains basic info, Pod links, trusted IdPs, and keys. Authentication is primarily Solid-OIDC. WebIDs also work for autonomous agents (backend services).

5.5 Data Model and Protocol

Builds on Linked Data Platform concepts and the Solid Protocol. Resources addressed by HTTP URIs; containers collect resources; standard HTTP methods; RDF preferred for structured data and interoperation. Protocol maturity continues to evolve through the Community Group.

5.6 Authorization: WAC and ACP

Web Access Control (WAC): classic ACL resources; agents (WebIDs) granted Read/Write/Append/Control; widely deployed.

```
@prefix acl: . <#auth1> a acl:Authorization; acl:agent ; acl:accessTo ; acl:mode acl:Read, acl:Write, acl:Control.
```

Access Control Policy (ACP): more expressive; preferred in some enterprise deployments (e.g. Inrupt ESS); still maturing relative to WAC. Many servers support one or both.

5.7 Implementations (mid-2026)

Implementation	Type	Notes
Community Solid Server (CSS)	Open source	Configurable; WAC + ACP; TypeScript; research & self-hosting
Node Solid Server (NSS)	Open source	Original popular server; WAC; file-system based
Inrupt ESS	Commercial	Microservices; ACP; Access Grants; notifications; scaling

5.8 Application Ecosystem & Current Status

Promise is application interoperability on user-owned data. Ecosystem still smaller and less polished than Nostr/Matrix/Fediverse. Existing apps: profiles, notes, viewers, research pilots, consent UIs. Gap between architectural ideal and day-to-day UX remains a central challenge.

Status Aug 2026: Protocol ~0.11; Solid-OIDC primary; WAC more widely deployed in OSS; CSS main open-source server; Inrupt commercial; Solid26 packaging; end-user adoption still limited relative to ambition.

5.9 Strengths and Limitations

Strengths: cleanest separation of identity/storage/apps; true grant/revoke and Pod portability; open standards; WebID flexible; self-hostable; good foundation for consent-heavy sharing.

Limitations: immature consumer app ecosystem; broader/more complex protocol surface than Nostr; inconsistent advanced feature support; demanding UX for permissions and migration; weaker network effects; RDF learning curve.

5.10 Comparative Position

Dimension	vs Peergos	vs Nostr	vs ActivityPub	vs Matrix
Data ownership purity	Closest pure form	Stronger	Stronger	Stronger
Default storage encryption	Weaker	N/A (events)	N/A	Weaker than E2EE rooms
App / client diversity	Moderate	Much lower	Lower	Lower
Identity	WebID (URI)	Keypair	Actor + WebFinger	Matrix ID
Access control	WAC/ACP expressive	None native	Server-mediated	Power levels + E2EE
Interop ambition	Highest	Low	Medium	Medium

Solid optimizes for data ownership and application interoperation. It is best understood as a data-layer and authorization substrate rather than a complete social or encrypted-storage product.

5.11 Key Primary Sources

- <https://solidproject.org> • <https://solidproject.org/TR/> • <https://github.com/solid/specification>
- Solid Protocol: <https://solidproject.org/TR/protocol> • Solid-OIDC: <https://solidproject.org/TR/oidc>
- Community Solid Server and Inrupt documentation; WAC and ACP specifications