Application 2023-04-169 data shell received

The following submission was recorded by NLnet. Thanks for your application, we look forward to learning more about your proposed project.

Contact

name ▶	Alexander Praetorius
phone ▶	+44 7458 611 224
email ▶	a.praetorius@serapath.de
organisation name ▶	playproject.io / Vision Baker Ltd.
country ▶	Germany/UK
consent ▶	Erase my details when no longer needed

Project

code ▶	2023-04-169
project name ▶	data shell
fund ▶	Entrust_Fund
requested amount ▶	€ 25500
website ▶	• https://github.com/playproject-io/datashell

synopsis

```
DataShell - A Peer-to-Peer Prototyping
Environment for Web Apps with User-Owned Data
Vaults
```

Moving data to the cloud solved users' early issues with backups and data loss, but today Big Tech companies own most of our data and control how it is used, leaving users vulnerable to privacy violations and data manipulation.

As a response to this unhealthy dependency on Big Tech, Dat Ecosystem projects have been building a neutral common p2p infrastructure over the past 10 years. But cross-platform packaging, standards on how to build p2p apps, which give users sovereignity over their data, and educating developers how to build and bring p2p apps to the users still remains a challenge.

To address this problem, DataShell aims to create a new open standard for user-owned, GDPR compliant data vaults. Furthermore, we aim to provide developers with an open in-browser prototyping environment which they can use (or self-host) to build web apps based on the vault standard in order to create an open resilient app ecosystem that gives users full control, data portability and the ability to self-authenticate, verify data and permissionlessly combine it in new ways.

experience

Over the years, our interdisciplinary team has actively contributed to the Dat ecosystem by

providing feedback, testing, design, communication efforts, submitting pull requests, and creating related modules. Additionally, we are active members of the Dat consortium, where we collaborate with other teams on interoperability standards.

Most recently, we have been organizing and managing a grant provided by the 501c3 foundation "Code for Science and Society", for the Dat ecosystem. During this project, we identified the need for a data vault standard and browser environment that can facilitate the prototyping of apps that separate the app logic from the user data to put users in full control over their data.

In addition to our involvement in the Dat ecosystem, we also frequently participate in online calls with teams from Agoric, Endo, and Metamask to explore Lavamoat and SES standards for in-process sandboxing. Our work follows the object capability network paradigm and draws inspiration from Mark Miller's whitepapers on JavaScript proxy-based membranes.

Furthermore, we are enthusiastic volunteers in the WizardAmigos community. As part of this community, we teach developers how to build Dat-based applications, organize meetups, workshops, hackathons, and unconferences to help foster a vibrant P2P e-learning community.

Part of our team is focused on DatDot, which is another member project within the Dat ecosystem. DatDot is building a peer-to-peer seeding service that ensures Dat-based datasets remain available even when regular peers who have a copy of the dataset are offline. Our long-term goal is to enable use cases such as GitHub, Wikipedia, or YouTube to function entirely in a peer-to-peer setting, without the need for large tech corporations or cloud services. We are also actively pursuing grant opportunities to further support this project.

Our team also has the experience in building developer environments. For three years, we worked as a contracting agency with the Ethereum Foundation to help them create Remix, an in-browser development environment for building smart contracts and dApps. We also developed a tool for visualizing abstract syntax trees for programming in the Solidity language.

usage

Data Shell is a new project that has not yet received any funding. It is a software project that will run client-side only on the systems of users and developers. The only hosting required is static web hosting, which we plan to fulfill mostly through Github pages for the time being.

Our team is fully remote and spread out across

the world. We have all the necessary tools to collaborate online, and as such, the only cost factor for this project is the time and effort that developers spend on planning and building the system.

To achieve all of our milestones, we require a senior peer-to-peer engineer, a senior JavaScript engineer, and a part-time frontend developer. Our team will execute a milestone-based plan, which includes writing documentation and blog articles to explain and document all aspects of the project and our mentioned deliv-erables.

35 EUR/h * 80h a month * 3 months * 2 senior en-gineers (16.800 EUR)
30 EUR/h * 80h a month * 1.5 month * 1 frontend engineers (3.600 EUR)
+ 25 % for the administrative work, reporting, accounting, rent for office space, electricity, internet, amortized computing hardware costs, pay for statutory holidays etc.

MILESTONES:
* **v0.1 - Core Environment**
  - Core: portal (web server)
    - web server configuration standard
    - includes index.html with single script tag
    - includes shim.js to import and boot kernel script (loaded via script tag)

- Core: dpm OS (default WebApp OS, using web kernel)
    - includes .data/* folder on portal to host all OS customization code
    - defined via package.json dependencies and entry script
    - uses entry script for advanced customization
- Core: web kernel (web runtime patch)
    - installs itself as a service worker and reloads the page
    - serves the OS system html page
    - serves scripts offline to load a visual shell
    - can add more scripts based on package json to customize OS
    - allows user shell code to be executed in runtime
    - provides dpm (data package manager) to be used by user via shell commands
- Core: shell
    - includes a command log so user sees history of interacting with the shell
    - includes a command line to type in javascript and commands for execution
    - indicates vault account user is logged in with (dummy account by default)
    - allows to list all data available on a specific portal (stored just in memory for now)
    - add advanced formatting to all basic types of shell messages for easy debugging
    - allow commonjs based import of data mod-

```
ules in executed code snippets
  - Core: dpm (data package manager)
    - add basic data integrity verification
mechanism
    - implement support for some very limited
basic commands:
      - help command to list all available pack-
age manager commands
      - commands to list all available data
      - commands to download more data
      - commands to create a user project folder
and data inside it
      - commands to download some data from the
browser to the users disk on the device they use
      - commands to start a specific app avail-
able (apps have access to commonjs import mecha-
nism)
  - Core: vault:
    - add vault mode to kernel
    - specify vault message protocol to talk to
kernel and apps
      - based on `comlink` library or similar
logic
      - includes transferable arraybuffers
    - implement vault message protocol to commu-
nicate with kernel and apps
    - add basic permission hooks
    - add basic user permission prompts (all ow
all vs. deny all)
  - Core: sandboxing:
    - adds as basic iframe based sandboxing
mechanism to kernel runtime
```

```
    - allows user shell code to be executed in
such an iframe sandbox
    - add sandbox message protocol to enable
communication with sandbox
    - add commands to list and spawn and shut
down sandboxes
* **v0.2 - Basic UX/UI**
  - vault
    - wireframe basic responive data vault UI
      - list all apps/software installed
      - list all datasets created
      - indicate permissions set for a specific
dataset
      - indicate permissions per portal
    - implement basic iframe view and fullscreen
vault view
    - add UI for user to trigger actions avail-
able via data package manager such as:
      - add actions to change permissions
      - add actions to delete or add datasets
  - dpm OS
    - nav bar
      - user account
      - offline/online indicator
      - open/close/collapse shell
      - data status indicator
    - content area
      - layout design

=> **Initial release**
```

comparison

DataShell Architecture

Our DataShell project has many similarities to the Spritely Project, particularly in terms of technical approach. While Spritely is built within the context of the ActivityPub standards, DataShell is designed for use within the Dat ecosystem and protocols with its peer-to-peer network and unique data structures. Our focus is on creating a browser-based platform that offers developers faster feedback loops, by allowing them to bring apps directly to users, rather than requiring downloads of unknown software to their computers. Additionally, we believe that focusing on the web is advantageous due to the broad availability of JavaScript, web, and front-end technologies, which are widely used and easily accessible for prototyping purposes.

P2P Web Bridges

Several p2p bridges in the form of custom browsers, browser extensions, and self-hosted server-based gateways have been developed within the Dat ecosystem. However, adoption of these bridges has been limited due to various factors.

Custom browsers such as Cliqz, Bunsen, Gateway Browser, Agregore, and Beaker require users to switch to a new browser, which many are hesitant to do. Browser extensions for Chrome and Firefox had issues working due to changes in browser

standards and extension capabilities. Self-hosted server-based gateways, such as the dat-gateway, required users to host their own gateways in the cloud or rely on gateways hosted by others, creating a point of centralization and traffic costs.

Furthermore, most of these bridges have been maintained by volunteers and have not been adequately funded nor regularly updated to the latest breaking changes. As a result, they have not been widely adopted, neither by developers nor users. Our project seeks to address these limitations and provide a user-friendly, reliable, and sustainable p2p bridge that can help accelerate the adoption of the Dat ecosystem.

P2P Prototyping

The Dat project has made several attempts to enable experimentation with Dat. Their initial tool, try-dat, relied on servers and was eventually closed due to growing hosting costs that were diverted from funding protocol development. Later attempts included browsers like Agregore and Beaker, which allowed developers to prototype apps. However, only Beaker Browser focused on a convenient developer experience with the centralized hosting service hashbase.org, but it was limited to hyperdrives, a p2p filesystem abstraction. Although this setup had some success, it was a bit centralizing and it also did not

properly address the issue of giving users control of their data and separating it from the apps, resulting in p2p apps feeling like traditional apps. Since not many users were willing to download a whole new browser for a relatively intangible p2p experience, the project founder eventually moved on and sunsetted the project.

Data Vault

While there are existing solutions like unhosted.org, remotestorage.io, and more recently solidproject.org that encourage app developers to separate app logic from data and encourage users to self-host or find hosting providers for their data backend, none of these solutions focus on p2p networks and data structures or address the need to avoid the need for specific hosting providers in the first place. While the TrustVault Data Wallet white paper from 2022 comes close to our vision, it does not particularly focus on the web and is to some degree coupled with blockchain infrastructure, whereas our focus is on leveraging the existing and thriving p2p dat ecosystem with active apps and users and building on the HTML5 stack for app development.

In-browser Sandboxing

Working with self-authenticating data structures in a p2p network requires security mechanisms

beyond traditional domains, such as CORS and CSPs. The SES and endo project use javascript proxies and lockdown of the environment to create membranes around parts of the code, enabling any kind of policy for any part of the code. Meanwhile, the Lavamoat project focuses on sandboxing individual modules within a single app and allows developers to specify such policies that restrict what each module is allowed to do in the first place. This is crucial for loading and executing code from different p2p origins in the same process environment and allowing users to control data access via their vault without compromising performance or security. Both Metamask and Agoric use these techniques already to build secure crypto wallets and smart contract environments for DeFi on a proof of stake blockchain.

challenges

While we have developed a conceptual architecture for all parts of the project and have a clear roadmap, there are still some key challenges that need to be addressed. These challenges include:

Improving the developer experience to make it more intuitive and easy to understand how to build p2p web apps with a data vault.

Adapting the SES object capability based model to our specific use case and needs in the con-

text of p2p web apps based on a data vault standard.

Enhancing the user experience for access control management of their vault data while also maintaining flexibility for developers.

Integrating all existing p2p data structures used in the dat-ecosystem with the vault model and its permission system.

We are actively working on the challenge and are committed to create a user- and developer-friendly, reliable, and sustainable p2p web bridge.

ecosystem

We've been engaged in tech activism since at least 2015 when we started organizing WizardAmigos events and working on an open source RefugeesWork project, where we together with Trebor Scholz from Platform cooperativism explored ways of making the RefugeesWork co-owned by its users.
This is also the time we started actively participating in the Dat ecosystem and today we are core members of the consortium with two of our ongoing projects (WizardAmigos: teaching p2p technology, DatDot: p2p seeding & availability service).

In the Dat-ecosystem there are many organiza-

tions building solutions very compatible with the DataShell and in need of a standardized data vault solution and we are already in disccus-sions with many of them to explore the detailed needs and opportunities to maximize user bene-fits while fine-tuning the the data vault stan-dard, which is going to enhance all dat-ecosys-tem projects, such as:

0. WizardAmigos will organize community events to teach building p2p apps with a data vault standard.
1. DatDot (the p2p data availability system) and DataShell will provide benefits for app users to control their own data and outsource care for important user datasets' availability to the DatDot network.
2. Decentlabs/PicoStack's Pico hotel needs solu-tions like DataShell for user app data and dataset hosting.
3. Sonar, an NGI-supported p2p search engine, could work in browsers too and help users and their vaults to search all public p2p datasets.
4. Sher's p2p audio broadcasting app can store live broadcasts using DataVault, and keep it available to the network using PicoHotel, or DatDot even after live broadcast ended.
5. SSC's cross-platform packaging toolkit pack-ages our daemon/localhost service for browser-based p2p web apps.
6. Cabal, an NGI-supported project, works on a p2p chat messenger that can be available in reg-

ular web browsers with DataShell.
7. Agregore supports p2p natively, so Dat can be brought to our prototyping environment without an additional daemon.
8. HOP project develops p2p solutions for users to own and control their medical data, and is interested in trying DataShell in their solution.
9. Dat ecosystem's Dat garden program helps developers grow new projects in the ecosystem and can then encourage the data-vault standard too.

With one standard across the dat-ecosystem we could finally enable the first steps towards real data portability and data sovereignity and because of the nature of the data-sharing protocol, users would also get self-authentication out of the box when sharing and collaborating with each other.

Additionally, we have also been in touch with neighbouring ecosystems, including blockchains, where we see the lack of good answer on how to empower their users with full data sovereignity and on how to build decentralized apps that don't just allow financial crypto transactions, which fit on a blockchain, but actually include larger real world datasets, such as the endless streams of audio, videos, photos and other kind of datasets that are too large and expensive to be stored on a blockchain. A standard for p2p web apps based on a data vault could get adopted

by many of these ecosystems.

Through Wizardamigos, we have built connections to many international communities, such as HacktionLab, Karrot, Wikimedia, Open Knowledge Foundation, Co-Up coworking community, Coconat rural Coworking/Coliving, Platform Cooperativism, SSB, Mozilla, g0v Zero, SudoRoom, Astralship HackerSpace, HashedHealth, DWeb Camp, and a few smaller meetup communities in Chiang Mai, Lagos, and Taipei. We know these communities share our open source and data sovereignity values and are potential partners, builders, and users of distributed systems. Our goal is to provide these communities with the necessary tools and standards to try, prototype, and adopt the peer-to-peer (p2p) stack and become part of the wider p2p ecosystem.

pgp

attachments

- Foundational Work.txt

Check

Please check that the above contact details are correct and that any attachments you have included have been uploaded. If you are in doubt, and near a deadline, don't hesitate to resubmit - better safe than sorry. If you want to make changes to the proposal, do the same.

If you experience any technical problems, please contact the webmaster.

I checked the box but did not receive an email

Besides the obvious candidate for undelivered email (check your spam folder if you have it), some people run into their own outdated email configuration. Do you use a legacy forwarding mechanism for your mail, from me@example.com to theactualmailbox@another.example.org? In that case, the final mailserver may toss these out due the use of modern anti-spoofing techniques (notably DMARC, DKIM and SPF) at our side. Essentially, forwarding the original email as was done historically means that you can't satisfy the origin and integrity conditions - and thus our email to you will be discarded...

The structural solution is to do the forwarding with a mechanism like *Sender Rewriting Scheme*. Ask your service provider, or consult the documentation of your software how to do that.