Web Push Notifications from Solid Pods

Christoph H.-J. Braun[0000−0002−5843−0316] and Tobias Käfer[0000−0003−0576−7457]

Institute AIFB, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
braun@kit.edu, tobias.kaefer@kit.edu

Abstract. Our demo showcases how a Solid Pod, i. e. a web server that adheres to the Solid Protocol, can be extended to support Web Push Notifications for Progressive Web Applications (PWAs). For a user's perspective, we present a PWA where a user can choose to receive Web Push Notifications when a message is posted to her Solid Pod's inbox.

Website http://uvdsl.solid.aifb.kit.edu/conf/2022/icwe/demo
Demo https://km.aifb.kit.edu/services/solid-web-pwa/
Code https://github.com/uvdsl/solid-web-push

1 Introduction

The Solid Project[1], Tim Berners-Lee's endeavor to re-decentralise the Web, aims to decouple data from consuming applications while ensuring data privacy. The Solid Protocol [2] connects mature Web technologies like the Resource Description Framework (RDF) and RESTful HTTPS APIs and an adaptation of Open ID Connect: Users store data in personal online data storages (Pods) and define access control on their resources as desired.

The Solid Protocol focuses on establishing server-side technologies while relying on generic client-side libraries (for HTTP and RDF) to create Web applications. For example, if developers need to react on updates of resources, they currently need to rely on polling, which is battery-draining on mobile, or WebSocket-based subscriptions, which are not standardised. Yet, nowadays live updates and notifications are a commonly expected feature of an application.

To live up to user expectations, Progressive Web Applications (PWAs) offer a rich feature set including Web Push Notifications, i. e. a notification scheme where push messages are delivered to Web applications via the browsers' messaging service. Such notifications can even be received when the Web application is closed. We thus ask: What if the Solid Protocol supported Web Push Notifications from Solid Pods to help developers create modern Web applications easily? To this end, we showcase:

- A notification scheme for Web Push Notifications from Solid Pods.
- A data model for Solid Web Push subscriptions using RDF.

[1] https://solidproject.org

– An extension module for Solid Pods to support the notification scheme.
– A PWA that uses these Solid Web Push Notifications.

This paper is structured as follows: First, we give a short overview on related work. Next, outline the system architecture. Then, we touch on our data model. Hereafter, we provide a walkthrough of our demo.

2 Related Work

We briefly survey related work in the realm of the Solid Protocol [2] and its notification schemes. An early description about Solid is provided in [3].

The current state of notification schemes in Solid are comprised of two alternatives: Either the client relies on polling to detect updates on a resource independent of the Pod's implementation, or the client uses a WebSocket Subscription as defined in version v0.7.0 of the Solid Protocol[2]. As the WebSocket notification scheme is currently supported by the most used Pod implementations we rely on this scheme in our demo as a fallback when the user chooses not to receive push notifications.

Recently, the Solid Community Notification Panel started reworking Solid's notification protocol. Four notification schemes are currently proposed[3]: (1) The WebSocket Subscription defines a notification scheme for bi-directional communication between client and Pod using WebSockets. (2) The EventSource Subscription defines a notification for uni-directional server-sent events from Pod to client using HTTP/2. (3) The Linked Data Notification Subscription defines a notification scheme relying on the Solid Protocol itself for sending Pod-to-Pod notifications. (4) The WebHook Subscription defines a notification scheme for pod-to-server callbacks using HTTP.

Web Push Notifications have been proposed in an IETF Draft for Generic Event Delivery using HTTP Push [6] and have been picked up by the W3C in the Push API [1]. With this demo, we propose an additional notification scheme of Solid Web Push Notifications in Pod-to-PWA fashion.

3 System Architecture

With Solid, each user is identified by a WebID, a URI which derefenced yields the user's Solid profile document. This profile document is stored on the user's personal online data storage (Pod) where the user can store any data under self-defined access control. On the Pod, there exists an inbox where messages to the user can be posted to. Using some app, the user can access this data after logging in to the app with her WebID.

To receive notifications even when the app is closed, we extend the Solid Pod with an additional module to support Web Push Notifications for Pod-stored resources. We implement the corresponding client-side functionality in our PWA.

[2] `https://github.com/solid/solid-spec/blob/master/api-websockets.md`
[3] `https://solid.github.io/notifications/protocol`

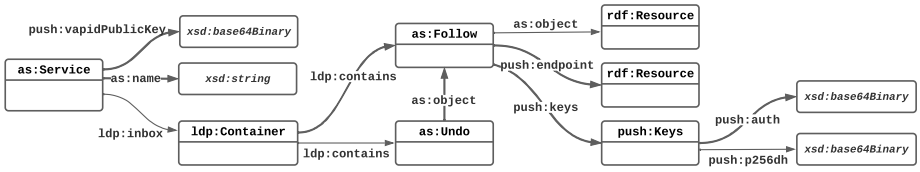

Fig. 1. The RDF datamodel underlying our Solid Web Push Notification scheme. We use CURIEs, abbreviated URIs, with prefixes from `http://prefix.cc/`. We use *push* as prefix that is short for `https://purl.org/solid-web-push/vocab#`.

The current state of Solid Notification schemes allows us to demonstrate Web Push Notifications for resources stored on any Solid Pod. However, considering security and scope of Web Push Notifications from Solid Pods, we envision a service that is able to send Web Push Notifications only for resources stored in that particular Pod, a Pod-scoped notification service.

An application developer can use the Pod-provided Web Push service and does not need to implement it herself as a server-side component of the application. Instead, the developer may specify an app-specific inbox on the user's Solid Pod where his server posts messages to which in turn can then be forwarded by the Pod's Web Push service. This way, the developer is still in control of the user experience when sending Push Notifications to the user. That said, developers are still free to implement their own Push Notification service.

4 Modeling Solid Web Push Subscriptions

We model a Web Push subscriptions and the corresponding service on the Solid Pod in RDF, as illustrated in Figure 1: To describe Web Push subscriptions, we defined an ontology based on existing definitions from [6]. Next to typical elements of a Web Push subscription such as *endpoint*, *auth* key and *p256dh* key, our subscription model includes the resource which the client requests notifications about. We use terms from ActivityStreams [4] to model the Solid Web Push service and subscription requests from clients, and terms from the Linked Data Platform [5] to model the service inbox which clients post subscriptions to.

5 Demonstrator Walkthrough

To demonstrate Solid Web Push Notifications, we provide a PWA where a user can choose to receive Push Notifications whenever her Pod inbox is updated. Following the Solid Protocol, a user logs in to our demo PWA with her Solid WebID. By default, the PWA relies on Solid's WebSocket Notifications to receive updates on the user's Pod inbox. If the user decides to receive Push Notifications, the user simply clicks the bell icon in the top right corner. This substitutes the WebSocket Notifications for Web Push Notifications from our Solid Web Push service. These notifications will be received even when the PWA is closed.

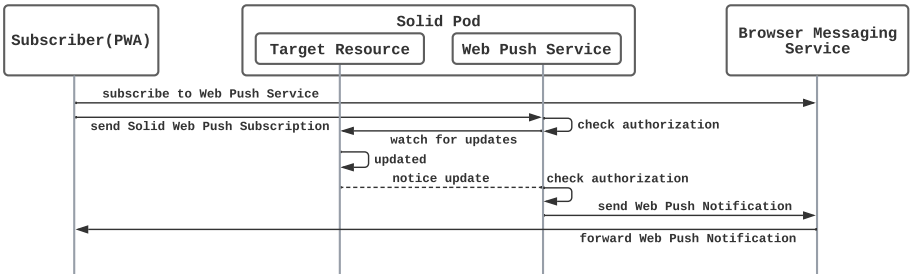

Fig. 2. An excerpt from the Solid Web Push Notification sequence diagram.

Under the hood, the PWA follows the Solid Web Push Notification scheme: A Solid Web Push subscription is sent the Solid Web Push service. The service then acts as a proxy forwarding any Pod-issued update notifications on the resource to the PWA using Web Push as illustrated by Figure 2. When the user chooses not to receive Push Notifications anymore, an undo request for the subscription is sent to the service canceling the Solid Web Push subscription. For an extensive sequence diagram of the Solid Web Push Notification scheme, we recommend the interested reader to take a look at our website which is linked on the first page.

6 Conclusion

In this demo, we showcased an extension to Solid Pods to support Web Push Notifications. We modeled the corresponding Solid Web Push service information and associated subscriptions using RDF. We propose our approach for discussion in the growing Solid ecosystem as Push Notifications are a common feature of today's apps, especially on mobile. Natively supporting Web Push Notifications from Solid Pods could facilitate developing application that live up to user expectations and thus contribute to the wider acceptance of the Solid Protocol.

References

1. Beverloo, P., and Thomson, M.: Push API. W3C Working Draft, `https://www.w3.org/TR/push-api/`. W3C (2021)
2. Capadisli, S., Berners-Lee, T., Verborgh, R., and Kjernsmo, K.: Solid Protocol. Version 0.9.0, `https://solidproject.org/TR/protocol`. W3C Solid CG (2021)
3. Mansour, E., Sambra, A.V., Hawke, S., Zereba, M., Capadisli, S., Ghanem, A., Aboulnaga, A., and Berners-Lee, T.: A Demonstration of the Solid Platform for Social Web Applications. In: Proc. of the 25th WWW. ACM (2016)
4. Snell, J.M., and Prodromou, E.: Activity Streams 2.0. W3C Recommendation, `https://www.w3.org/TR/activitystreams-core/`. W3C (2017)
5. Speicher, S., Arwe, J., and Malhotra, A.: Linked Data Platform 1.0. W3C Recommendation, `https://www.w3.org/TR/ldp/`. W3C (2015)
6. Thomson, M., Damaggio, E., and Raymor, B.: Generic Event Delivery Using HTTP Push. RFC 8030, `http://ietf.org/rfc/rfc8030.txt`. IETF (2016)