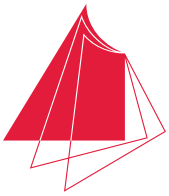

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Analyse einer Referenzimplementierung des ActivityPub Standards mit Blick auf die Sicherheit

Bachelor Thesis of

Armin Kunkel

at the Faculty of Computer Science and Business Information Systems
Subject Area Distributed Systems (VSYS)

Reviewer: Prof. Dr. rer. nat. Christian Zirpins
Advisor: M. Sc. Robert Schäfer

01. Dezember 2018 – 31. März 2019

Hochschule Karlsruhe Technik und Wirtschaft
Fakultät für Informatik und Wirtschaftsinformatik
Moltkestr. 30
76133 Karlsruhe

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

PLACE, DATE

Please replace with actual values

. . . . . .

(Armin Kunkel)

Inhaltsverzeichnis

1. Einleitung **1**
- 1.1. Motivation . . . 1
- 1.2. Unterschied zentraler sozialer zu dezentralen sozialen Netzwerken . . . 2

2. Grundlagen zur sicheren Umsetzung dezentraler sozialer Netzwerke **3**
- 2.1. ActivityPub Standard . . . 3
  - 2.1.1. Bestandteile des Protokolls . . . 3
  - 2.1.2. Zugehörige Standards . . . 4
- 2.2. Client-zu-Server Kommunikation . . . 4
  - 2.2.1. Client Teil . . . 4
  - 2.2.2. Server Teil . . . 5
- 2.3. Server-zu-Server Kommunikation . . . 6
- 2.4. Authentifizierung und Datenintegrität . . . 6

3. Analyse bestehender Ansätze **7**
- 3.1. Vergleich bestehender Implementierungen . . . 7
- 3.2. Verwandte Protokolle . . . 7

4. Entwurf einer Lösung für sichere Server-zu-Server Interaktion mit ActivityPub **9**
- 4.1. Entwurfsentscheidung . . . 9
- 4.2. Technische Architektur . . . 9

5. Implementierung eines ActivityPub Prototyps **11**
- 5.1. Server-zu-Server Protokoll . . . 11

6. Evaluation **13**
- 6.1. Anwendungsbeispiel . . . 13
- 6.2. (Performanzmessungen) . . . 13
- 6.3. Disskusion von Vor- und Nachteilen der Lösung . . . 13

7. Zusammenfassung und Ausblick **15**

A. Appendix **17**
- A.1. First Appendix Section . . . 17

Abbildungsverzeichnis

2.1. Interaktionen des Client mit dem Server 4
2.2. Schnittstellen des ActivityPub Protokolls 6

A.1. A figure 17

1. Einleitung

- Rechtfertigung der Arbeit

Viele Portale berichten heutzutage davon, dass zentrale soziale Netzwerke dezentralisiert werden sollen.→ (*Quellen*!!!)*Auerdemliestmanimmerhufigerdarber*, *dassActivityPubdie„Zukunftder*

- Ziel der Arbeit

Um die Arbeit erfolgreich abzuschließen soll eine Implementierung des verteilten Server-zu-Server Protokolls als Prototyp und eventuell ein Konzept zum sicheren Einsatz des Protokolls vorliegen. Darüber hinaus soll die Arbeit einen Überblick über die Grundlagen des Protokolls geben und wie die relevanten Bestandteile zu verstehen sind.

- Abgrenzung des Themas und Themenbezogenen Definitionen

- Geschichte und Stand der Forschung:

Der ActivityPub[1] Standard wurde am 23 Januar 2018 von der W3C empfohlen und von einer Arbeitsgruppe des W3C, der Social Web Working Group(SWWG)[2][3], entwickelt. Diese Gruppe war vom 21. Juli 2014 bis zum 13 Februar 2018 aktiv[2] und entwickelte unter anderem ActivityPub, Activity Streams Core[4], Activity Streams Vocab[5].→ *berwissenschaftlichePuklikatione*

1.1. Motivation

Um der Zentralisierung von Daten entgegenzuwirken ist es sinnvoll die Daten zu dezentralisieren. Mail Server sind zum Beispiel dezentral ausgelegt, sodass nicht ein Mail Server alle E-Mails verwaltet sondern jeder Mail Server die Mails für die eigene Domain. → (*Richtig*???)*AuchdasInternetistdezentralaufgebautumnichteinenFlaschenhalszuhaben*, *vondemdieganz*

Protokollewie„DiasporaFederationundOStatus“ bietendenZugriff auf dezentralgespeichertesozialeInh (*Quelleangeben*!!!).

[1] https://www.w3.org/TR/activitypub/
[2] https://www.w3.org/wiki/Socialwg
[3] https://www.w3.org/blog/news/archives/3958
[4] https://www.w3.org/TR/activitystreams-core/
[5] https://www.w3.org/TR/activitystreams-vocabulary/

ActivityPubistwie„Diaspora“ und„OStatus“ einProtokoll f rdezentralesozialeNetzwerkeundwirdindieserBa

1.2. Unterschied zentraler sozialer zu dezentralen sozialen Netzwerken

2. Grundlagen zur sicheren Umsetzung dezentraler sozialer Netzwerke

„ActivityStreams Core“ActivityStreams Core beinhaltet Modelle für Aktoren, Aktivitäten, Intransitiven Aktivitäten, Objekte, Links, Sammlungen, Natürliche Sprachwerte (Strings) und für Internationalisierung per „@language“ Eigenschaft.

2.1. ActivityPub Standard

ActivityPub definiert zwei Protokollschichten, sowie Konzepte, Sammlungen und Interaktionen für dezentrale soziale Netzwerke. Es setzt auf bereits bestehenden Empfehlungen des W3C auf, welche teilweise auch von der SWWG entwickelt wurden wie zB. Activity Streams Core und Activity Streams Vocab.

Auch andere Technologien wie JSON-LD[1] werden benutzt um die Erweiterbarkeit zu gewährleisten. Über neue Ontologien (Vokabulare) können weitere syntaktische Definitionen und semantische Beschreibungen zu den bestehenden hinzugefügt werden. Diese Vokabulare können im Kontext des JSON-LD Objektes angegeben werden.
(Stimmt das mit den Ontologien???) (Beispiel Context Abbildung mit erweitertem Context???)

2.1.1. Bestandteile des Protokolls

Im Grunde besteht der ActivityPub Standard aus zwei Protokollen:

- ActivityPub Client-zu-Server Protokoll (Social API)
- ActivityPub verteiltes Server-zu-Server Protokoll (Federation Protocol)

Die beiden Protokolle können unabhängig voneinander implementiert werden. Das Client-zu-Server Protokoll besteht aus einem Client und Server Teil.

In ActivityPub werden Benutzer als „Aktoren“(actors) dargestellt. Jedes Aktoren Objekt muss eine „Inbox“ und „Outbox“, welche geordnete Sammlungen sein müssen, sowie eine ID und ein Typ besitzen[2]. Die ID muss global einzigartig sein und der Typ kann variieren zwischen „Person“, „Application“ und weiteren.[3]

[1] `https://www.w3.org/TR/json-ld/`
[2] actor-objects https://www.w3.org/TR/activitypub/
[3] actors https://www.w3.org/TR/activitystreams-core/

Listing 2.1: Beispiel Aktoren Objekt

2.1.2. Zugehörige Standards

ActivityPub benutzt die ActivityStream Daten Syntax und das Vokabular. Zusätzlich kann das Sicherheitsvokabular von Webpayments benutzt werden um Syntax und Semantik für das Beschreiben eines öffentlichen Schlüssels o.ä. zur Verfügung zu stellen.

2.2. Client-zu-Server Kommunikation

Der Client empfängt Nachrichten über zwei Schnittstellen. Er kann sich per HTTP GET Anfrage auf seine eigene „Inbox“ die neusten an ihn adressierten Inhalt holen. Außerdem kann eine HTTP GET Anfrage an die „shared Inbox“ des Servers gesendet werden um eine Repräsentation aller Inhalte der Instanz zu bekommen.
(Oder alle Inhalte des bekannten Netzwerks???)

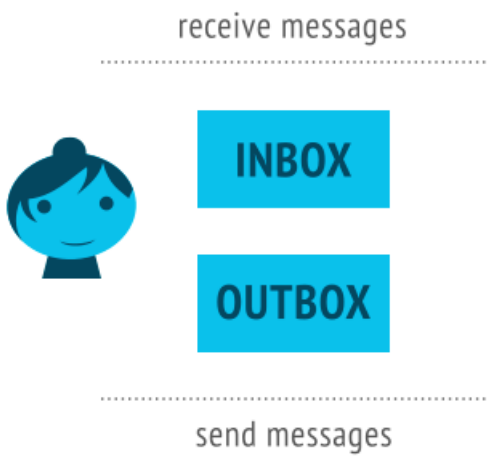

Abbildung 2.1.: Interaktionen des Client mit dem Server

Um neue Aktivitäten an seine Folgenden, direkt an einzelne Personen oder sichtbar innerhalb der Domain zu veröffentlichen kann der Client eine HTTP POST Anfrage an seine „Outbox“ senden mit einer Aktivität oder einem ActivityStreams 2.0(AS2) Objekt als Inhalt.

2.2.1. Client Teil

Der Client Teil des Client-zu-Server Protokolls beinhaltet mehrere Aufgaben:

- Auffinden der URL über das Profil

- Serialisieren der Daten in eine Aktivität oder AS2 Objekt
- Adressieren von Aktivitäten oder AS2 Objekten
- Eine HTTP POST Anfrage mit „Content-Type: application/ld+json: profile="https://w3.org/ns/activ „Content-Type: application/activity+json: profile="https://w3.org/ns/activitystreams"“ an seine „Outboxg“ absetzen
- Bei den Aktivitätstypen CREATE, UPDATE, DELETE, FOLLOW, ADD, REMOVE, LIKE, BLOCK, UNDO wird die „object“ Eigenschaft bereitgestellt
- Die „target“ Eigenschaft wird bei den Aktivitätstypen ADD und REMOVE bereitgestellt
- Anfragen müssen mit den Zugangsdaten des Nutzer, zu dem die „Outbox“ gehört, authentifiziert werden

Zusammengefasst muss sich der Client um das Entdecken von Endpunkten - Inbox/Outbox - über Profile, das Serialisieren von Daten zu Aktivitäten oder AS2 Objekten sowie um das setzen der entsprechenden Kopfzeilen (Mittlerer Teil der obigen Auflistung) und absenden der HTTP POST Anfrage kümmern. Außerdem muss bei HTTP GET Anfragen an den Server die „ACCEPT: application/ld+json“Kopfzeile gesetzt sein.

2.2.2. Server Teil

Zu den Aufgaben des Serverteils gehören das Annehmen von HTTP POST Anfragen auf die „Outbox“ eines Nutzers und das verifizieren ob der Nutzer berechtigt ist diese Anfrage zu tätigen. Weiter werden die „Inbox“ Endpunkte bereitgestellt um authentifizierten Nutzern den Zugriff auf die neusten Inhalte zu geben.

(Ist das richtig oder bestehen die Aufgaben des Serverteils auch im weiterleiten zu anderen Instanzen, weil das ist ja eigentlich der Server-zu-Server Teil???) (Ist der Server Teil ein interner Speicher Mechanismus der Instanz oder ein delivery Mechanismus für alle Instanzen des Netzwerk oder für alle Instanzen mit einer Federated Implementierung insgesamt??? Ich verstehe es wie oben beschrieben, also ohne Server-to-Server Teil alles innerhalb einer Instanz)

Angenommen der Server Part beinhaltet nicht nur das annehmen und abspeichern innerhalb einer Instanz, sonder auch das weiterleiten an andere Instanzen:

Dann müsste ein Nutzer alleine durch die Implementierung des Server Teils des Client-zu-Server Protokolls in der Lage sein Inhalt mit anderen Servern auszutauschen. Den Unterschied zu einem förderierten Server liegt dann im Empfangen von öffentlichen Nachrichten (5.6 Public Addressing | Shared Inbox Delivery [4]) und dem Weiterleiten (7.1.2 Forwarding from Inbox) von auf dieser Domäne

[4] sharedInbox https://www.w3.org/TR/activitypub/

nicht zustellbaren Empfängern.

Im Endeffekt ist die Frage ob der Server Teil des Client-zu-Server Protokolls eine Teilmenge des Server-zu-Server Protokolls ist???

2.3. Server-zu-Server Kommunikation

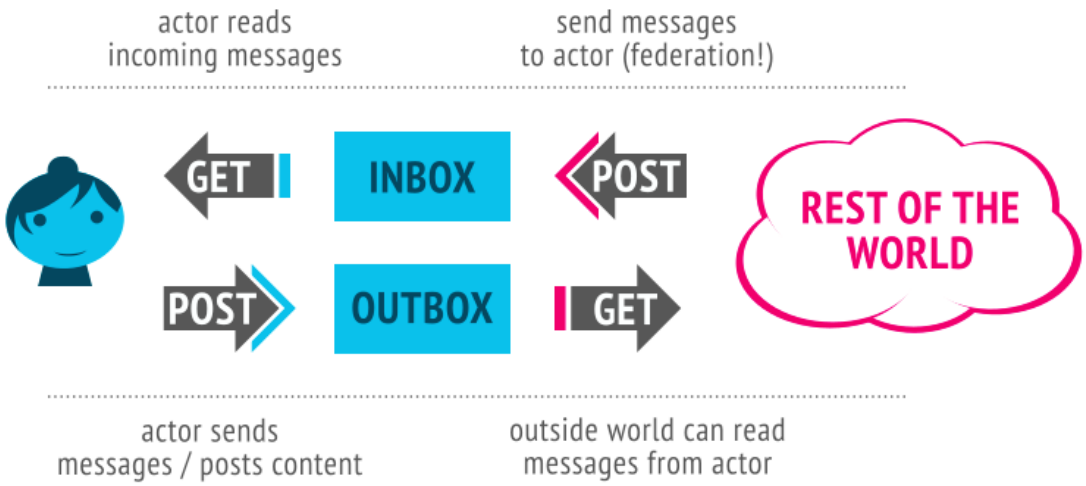

Abbildung 2.2.: Schnittstellen des ActivityPub Protokolls

Bei der Server-zu-Server Kommunikation bestehen die Hauptaufgaben im annehmen und zustellen von HTTP POST Anfragen auf die „Inboxen“ und „Outboxen“ der Nutzer und im weiterleiten von Aktivitäten die der Server nicht zustellen kann. Dazu kommt das Bereitstellen der „Outbox“ Sammlungen.

2.4. Authentifizierung und Datenintegrität

Für die Authentifizierung und zum sichern der Datenintegrität definiert der Standard keine Mechanismen. Es gibt allerdings „Best Practices“ für die Umsetzung dieser Anforderungen.

Zum einen werden bei der Client-zu-Server Authentifizierung „OAuth 2.0“ Tokens benutzt, zum anderen auf der Server Seite HTTP Signaturen oder „Linked Data Signatures“zur Sicherstellung der Datenintegrität. Letztere Signatur wird eher verwenden wenn Objekte übertragen und weitergeleitet werden, da damit die Integrität des Objekts an sich verifiziert werden kann.[5]

[5] https://www.w3.org/wiki/SocialCG/ActivityPub/Authentication_Authorization

3. Analyse bestehender Ansätze

3.1. Vergleich bestehender Implementierungen

3.2. Verwandte Protokolle

- OStatus
- Diaspora Federation

4. Entwurf einer Lösung für sichere Server-zu-Server Interaktion mit ActivityPub

4.1. Entwurfsentscheidung

4.2. Technische Architektur

5. Implementierung eines ActivityPub Prototyps

5.1. Server-zu-Server Protokoll

6. Evaluation

6.1. Anwendungsbeispiel

6.2. (Performanzmessungen)

6.3. Disskusion von Vor- und Nachteilen der Lösung

7. Zusammenfassung und Ausblick

A. Anhang

A.1. First Appendix Section

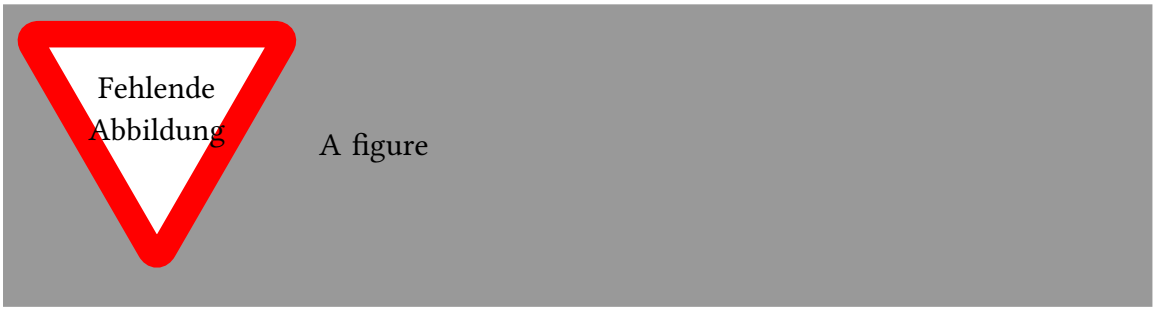

Abbildung A.1.: A figure

...