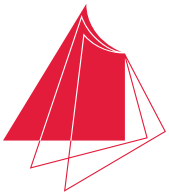


Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Analyse einer Referenzimplementierung des ActivityPub Standards mit Blick auf die Sicherheit

Bachelor Thesis von

Armin Kunkel

an der Fakultät für Informatik und Wirtschaftsinformatik
Fachrichtung Verteilte Systeme (VSYS)

Erstgutachter: Prof. Dr. rer. nat. Christian Zirpins
Betreuer: M. Sc. Robert Schäfer

01. Dezember 2018 – 31. März 2019

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

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Änderungen entnommen wurde.

Karlsruhe, 14. März 2019

. . . . . .

(Armin Kunkel)

Zusammenfassung

Deutsche Zusammenfassung

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

SWWG Social Web Working Group

W3C World Wide Web Consortium

DSA Digital Signature Algorithm

OWP Open Web Platform

RSA Rivest-Shamir-Adleman

API Application Programming Interface

1. Einleitung

Zentralisierung von Daten und das Vertrauen auf einzelne Instanzen ist heutzutage allgegenwärtig. Im Internet gibt es eine Fülle an sozialen Netzwerken wie Facebook, Twitter, Google+, Instagram oder Pinterest die zumeist das Recht an den Daten, die Sie von ihren Nutzer bekommen, behalten und nicht zuletzt diese Daten auch verkaufen für z. B. Werbezwecke. Für Kriminelle sowie Geheimdienste ist es außerdem leichter an die gesamte Datenbank zu kommen, da bei zentralisierten Netzwerken meist auch eine zentrale Datenbank, bzw. ein zentraler Zugriffspunkt, vorhanden ist. Wenn der Zugriff auf diesen zentralen Punkt erreicht ist kann die ganze Datenbank ausgelesen werden.

Was bedeutet nun dezentral? In der Politik wird unter Dezentralisierung „die Übertragung zentral staatlicher Aufgaben auf subnationale oder subsidiäre Ebene(n) verstanden“[13]. In der Energiewirtschaft spricht man von einer „dezentralen Stromerzeugung“, wenn der Strom an den Stellen wo er verbraucht auch erzeugt wird. Ein Beispiel hierfür wäre ein Wasserkraftwerk, dass den Strom für die umgebenen Dörfer oder Städte liefert[12]. In der Informatik versteht man unter Dezentralisierung das verteilen von u. A. Daten über mehrere unabhängige Server hinweg.

Durch die Verteilung der Aufgaben auf sub-nationale oder subsidiäre Ebene wird die zentral staatliche Ebene entlastet und hat somit mehr Ressourcen für andere Aufgaben. Beim Errichten von Wasserkraftwerken nahe an Dörfern und Städten entfällt der Transport des Stroms über größere Distanzen und somit verringert sich der Verlust beim Transport über das Stromnetz. Dezentralisieren von sozialen Netzwerken bring verschiedene Vorteile mit sich. Ein Vorteil ist die Skalierbarkeit bei dezentralen sozialen Netzwerken. Durch das hinzufügen weiterer Instanzen können dem Netzwerk neue Ressourcen bereitgestellt werden. Ein weiterer Vorteil liegt darin, dass keine zentrale Kontrollinstanz vorhanden ist welche korrumpiert werden kann, sei es durch Kriminelle, wirtschaftliche Interessenvertreter oder staatliche Autoritäten.

Ziel der Arbeit ist es die Implementierung des verteilten Server-zu-Server Protokolls als Prototyp vorzunehmen und die bis dato von der World Wide Web Consortium (W3C) Gemeinschaft empfohlenen Sicherheitsstandards für das ActivityPub Protokoll zu prüfen. Es soll die Frage geklärt werden ob HTTP- und Linked Data Signaturen bestmöglich dafür geeignet sind, die sichere Kommunikation zu gewährleisten anhand eines Vergleichs der beiden Verfahren. Darüber hinaus soll die Arbeit einen Überblick über die Grundlagen des Protokolls geben und wie die relevanten Bestandteile zu verstehen sind. In diesem Zusammenhang ist mit „sicher“ die authentische und manipulationsfreie Übertragung der Server untereinander gemeint.

1.1. Motivation

Bei zentralen sozialen Netzwerken, z.B. Facebook, besitzt die Kontrolle über Daten das Unternehmen. Durch die Dezentralisierung bleiben Daten bei den einzelnen Instanzen des sozialen Netzwerkes. Die Datenbanken bei zentralen Netzwerken können zwar verteilt sein und auch das Netzwerk könnte dezentral angelegt sein, trotzdem gehören die Daten einem Unternehmen. Außerdem können bei zentralen sozialen Netzwerken nicht ohne weiteres Inhalte mit anderen Netzwerken ausgetauscht werden. Mit einem Protokoll wie ActivityPub wird das Verteilen sowie der Zugriff auf Inhalte und der Austausch von Aktivitäten über mehrere soziale Netzwerke hinweg ermöglicht.

ActivityPub besteht aus zwei Teilen. Dem Client-zu-Server Protokoll, auch „Social API“ genannt, und dem förderierten Server-zu-Server Protokoll. Durch die ActivityPub W3C Empfehlung Anfang 2018 und die Social Web Working Group (SWWG) Arbeitsgruppe des W3C sowie weiteren, wird die Verbreitung des Protokolls vorangebracht.

Der Netzwerkeffekt ist ein aus der Volkswirtschaftslehre stammender Begriff. Als Beispiel sei das Telefonnetz genannt. Umso mehr Leute ein Telefon besitzen, umso höher ist der Nutzen für jeden einzelnen Telefon Besitzer[10]. Übertragen auf ActivityPub bedeutet das soviel wie: „Je mehr Netzwerke den Standard implementieren, desto höher ist der Nutzen für ein einzelnes Netzwerk und im Endeffekt für die einzelnen Nutzer“.

Protokolle wie „Diaspora Federation und OStatus“ bieten den Zugriff auf dezentral gespeicherte soziale Inhalte und ermöglichen das veröffentlichen von sozialen Inhalten. Der im März letzten Jahres vom W3C empfohlene Standard namens „ActivityPub“ wurde auf Basis des Wissens im Umgang mit dem OStatus und Pump.io Protokoll entwickelt[5].

Quelle für dezentralen Zugriff

ActivityPub ist wie „OStatus“ ein Protokoll für dezentrale soziale Netzwerke und wird in dieser Bachelor Arbeit untersucht.

2. Verwandte Protokolle

Zu Beginn dieses Kapitels wird das Wort Fediverse etwas näher definiert und erläutert worum es sich dabei handelt. Im Anschluss werden mehrere Protokolle die, sowie auch ActivityPub, zum Fediverse gehören vorgestellt.

Als erstes soll der Begriff „förderiertes Netzwerk“ von Förderation hergeleitet werden. Unter einer Förderation versteht man den Verbund von etwas. Deutschland ist z. B. ein förderierter Bundesstaat, welcher 16 Bundesländern verbindet. Ein Verbund von Netzwerken wird demnach als Netzwerkverbund, oder auch „förderiertes Netzwerk“, bezeichnet.

„Fediverse“ ist ein Kofferwort aus „Federated“ und „Universe“. Historisch gesehen beinhaltete der Begriff nur Microblogging Plattformen die das OStatus Protokoll unterstützen. Mehr zu OStatus **S. 2.1**. Mittlerweile beinhaltet das Fediverse mehr als nur Microblogging Plattformen wie Mastodon[1] und GNU Social[2], sondern auch soziale Netzwerke, sowie „Video hosting“ (PeerTube[3]),„Content publishing“ Plattformen, wie Wordpress, und viele weitere.[3]

Im „Fediverse“ dreht sich alles um freie (Open Source) Software anstelle von kommerziellen Produkten. Zudem kann ausgesucht werden welchem Administrator man die Kontrolle über seine Daten geben möchte, anstatt auf eine einzelne Instanz vertrauen zu müssen.[3] Ein weiterer wichtiger Aspekt ist das verbünden von Netzwerken über Protokolle wie OStatus oder Pump.io. Somit kann ein verteiltes soziales Netzwerk durch die Implementierung eines dieser Protokolle zum förderierten sozialen Netzwerk werden.

indirektes online Zitat!!!

Laut dem Fediverse Netzwerkreport 2018, welcher online zur Verfügung steht, hat sich die Gesamtanzahl der erreichbaren Instanzen von 2.756 auf 4.340 erhöht. Dies entspricht einem Wachstum von 58%. Die Nutzeranzahl ist von 1.786.036 auf 2.474.601 gestiegen (39%). Weitere Details sind im Netzwerk Report nachzulesen. Es ist hinzuzufügen das der Netzwerkreport unvollständig ist und Fehler beinhalten kann.[4]

Vergleich der Vor- und Nachteile der Protokolle

[1] `https://mastodon.social/about`
[2] `https://gnu.io/social/`
[3] `https://joinpeertube.org/en/`

2.1. OStatus

Der OStatus Standard ist eine Sammlung von verschiedenen Protokollen wie Atom, Activity Streams, WebFinger und weiteren. Ausgelegt ist der Standard auf das empfangen und versenden von Status updates für förderierte Microblogging Dienste wie z. B. Mastodon und Friendica. Das Zusammenspiel mehrerer Protokolle ermöglicht den Austausch von Status updates in fast Echtzeit.

2.2. Diaspora

Diaspora ist ein freier Web Server mit einer integrierten Implementierung eines verteilten sozialen Netzwerkes. In diesem Netzwerk wird ein Knoten als Pod bezeichnet. Die einzelnen Pods sind das, was das Diaspora Netzwerk ausmacht. Durch die integrierte Funktionalität zum sozialen Netzwerken kann auf dem Web Server aufsetzend eine eigene Applikation entwickelt werden.

2.3. Pump.io

Der ActivityPub Standard wurde mit der gesammelten Erfahrung im Umgang mit dem Pump.io Protokoll entwickelt.

3. Grundlagen zur sicheren Umsetzung dezentraler sozialer Netzwerke

Allgemeine Grundlagen zu Sozialen Netzwerken/Social Media und deren Sicherheitsaspekte

Welche Klassen von Anwendungen und Implementierungen gibt es allgemein bei sozialen Netzwerken? Einordnen von ActivityPub!!

Dieses Kapitel gibt einen Überblick über allgemeine Grundlagen von sozialen Netzwerken. Es wird grob erläutert für was soziale Netzwerke gedacht sind; Der Unterschied zwischen zentralen, verteilen, dezentralen und förderierten sozialen Netzwerken herausgestellt sowie auf Sicherheitsaspekte eingegangen.

Im Anschluss wird ein Kryptographie Unterkapitel eingeführt in dem die benötigten Verfahren für die sichere Umsetzung erläutert werden. Begonnen wird mit einer kurzen Einführung in Kryptographie. Darauf folgend wird kurz das RSA Verfahren sowie eine allgemeine Beschreibung von Signatur Algorithmen vorgenommen. HTTP Signaturen werden etwas ausführlicher beschrieben, da diese bei der Implementierung des Prototypen verwendet wurden.

Darauf folgend wird der ActivityPub Standard eingeführt sowie eingeordnet, Bestandteile des Protokolls beschrieben, die Funktionsweise der Client-zu-Server sowie Server-zu-Server Kommunikation und zugehörige Standards kurz erläutert. Desweiteren wird auf die Authentifizierung und Datenintegrität bei ActivityPub eingegangen um damit folgende Fragen zu klären:

- Wie authentifiziert sich ein Benutzer gegenüber dem Server?

- Wie stellt man sicher, dass die Übertragenen Daten unverändert angekommen sind?

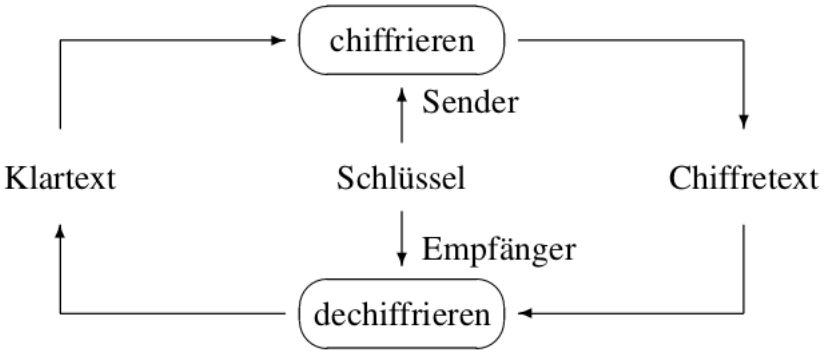
3.1. Allgemeine Grundlagen sozialer Netzwerke

3.1.1. Unterschied zentraler zu dezentralen sozialen Netzwerken

3.1.2. Sicherheitsaspekte sozialer Netzwerke

3.2. Kryptographie

„Unter dem Begriff Kryptographie ist die Wissenschaft vom geheimen Schreiben zu verstehen“[1]. Man spricht von symmetrischer Verschlüsselung wenn eine Nachricht im Klartext vom Sender mit einem geheimen Schlüssel, welcher beiden Parteien bekannt ist, verschlüsselt und vom Empfänger, mit demselben Schlüssel, entschlüsselt wird[2].



Quelle: (Wätjen, 2018, S.1)

Abbildung 3.1.: Symmetrische Ver- und Entschlüsselung

Von asymmetrischer Verschlüsselung ist die Rede, wenn die Teilnehmer anstatt einen gemeinsamen Schlüssel zu haben, der im vor hinein ausgetauscht werden muss, jeder ein Schlüsselpaar, bestehend aus öffentlichem und privatem Schlüssel, besitzt.

3.2.1. RSA

Das Rivest-Shamir-Adleman (RSA) Verfahren ist ein solches asymmetrisches Verschlüsselungsverfahren welches von den drei namens gebenden Mathematikern 1977 entwickelt wurde. Das RSA Verfahren ist wohl das am häufigsten verwendete Public-Key-Kryptosystem.

Beispiel 1. Die Gesprächspartner Alice und Bob, welche beide ein Schlüsselpaar besitzen, miteinander kommunizieren. Alice verschlüsselt einen Nachrichtentext mit dem öffentlichen Schlüssel von Bob und sendet die verschlüsselte Nachricht an Bob. Dieser

[1] Dietmar Wätjen, 2018, S.1
[2] Vgl. Dietmar Wätjen, 2018, S.1

kann seinerseits mit seinem privaten Schlüssel die Nachricht entschlüsseln[3].

3.2.2. Signaturen

Für die Sicherstellung der Authentizität können sogenannte Signaturen verwendet werden. Das oben kurz erläuterte RSA Verfahren kann nicht nur zum Ver- und Entschlüsseln von Nachrichten benutzt werden, sondern auch zum signieren.
Dabei wird statt des öffentlichen Schlüssels, der private Schlüssel benutzt um eine Signatur zu erzeugen. Diese kann dann mit dem öffentlichen Schlüssel des zugehörigen privaten Schlüssels verifiziert werden.

Ein zugehöriges Beispiel wäre das folgende:
Beispiel 2. Bob möchte eine Nachricht an Alice schicken und sichergehen, dass diese auf dem Weg nicht verändert wurde. Er verwendet seinen privaten Schlüssel und wendet diesen auf eine Nachricht an um eine Signatur zu erzeugen. Beides übermittelt er an Alice. Mit dem öffentlichen Schlüssel von Bob kann die fehlerfreie Übertragung der Nachricht verifiziert werden.

3.2.3. HTTP Signaturen

Eine *HTTP Signatur* wird verwendet um die Authentizität sicherzustellen. Diese wird als Wert einer „Signature“ Kopfzeile eingetragen und besteht aus mehreren Teilen:

- ***keyId***=keyId=”https://example.org/activitypub/users/leamain-key”
- ***algorithm***=”rsa-md4”
- ***headers***=”(request-target) date host content-type”
- ***signature***=”DHeEH0Okmtf1ec/lbM1/F5FiLVfQfbWuoFf9t/TzNZiZ7ak”

Über die *keyId*, was eine Referenz auf einen Schlüssel darstellt, kann der öffentliche Schlüssel angefragt werden. Dies wird beim verifizieren einer Signatur benötigt um die Übertragenen Daten auf ihre Authentizität hin zu prüfen.

Welcher Hashing-Algorithmus bei der Erstellung verwendet wurde, kann über das *algorithm* Feld der Signatur nachgeschlagen werden. Zudem muss der Algorithmus bei Erstellung in dieses Feld zum Nachschlagen eingetragen werden.

Um die eigentliche Signatur zu erzeugen werden die in *headers* angegebenen Kopfzeilen der HTTP Anfrage verwendet. Somit kann eingeschränkt werden welche Metadaten in die Signierung einfließen.

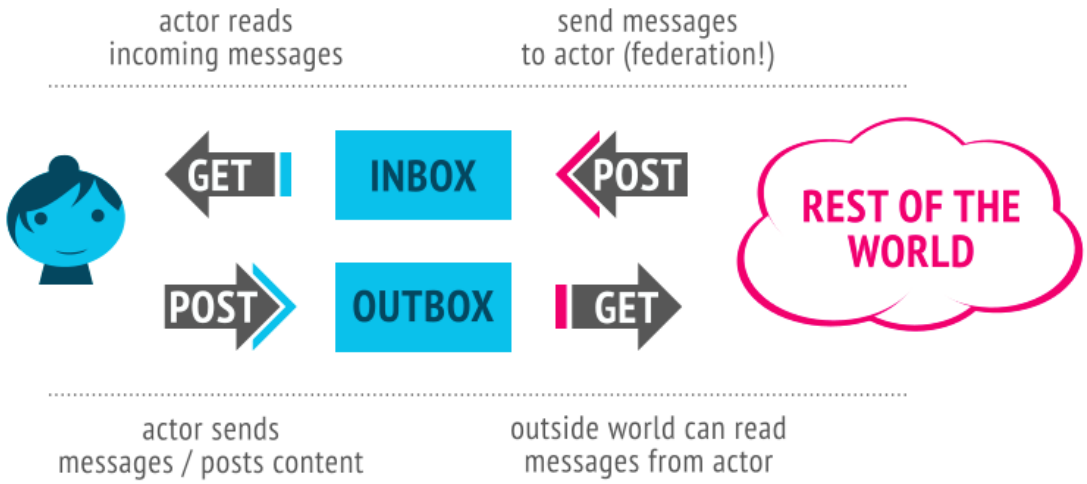
[3]Vgl. Dietmar Wätjen, 2018, S.73 f.

Die bei der Signierung mit gegebenem Hashing-Algorithmus und Kopfzeilen erzeugte Signatur wird in das *signature* Feld der HTTP Signatur eingetragen sowie die HTTP Signatur an sich als „Signature“ Kopfzeile der HTTP Anfrage gesetzt.[**http-signature**]

3.3. ActivityPub Standard

Der ActivityPub Standard wurde am 23 Januar 2018 von der W3C empfohlen[5] und von einer Arbeitsgruppe des W3C, der SWWG[1, 2], entwickelt. Diese Gruppe war vom 21. Juli 2014 bis zum 13 Februar 2018 aktiv[1] und entwickelte unter anderem ActivityPub, ActivityStreams Core[9] und ActivityStreams Vocab[8]. Die SWWG war eine Arbeitsgruppe des W3C mit dem Ziel neue Protokolle, Vokabulare und Application Programming Interface (API)’s zu definieren für den Zugriff auf soziale Inhalte der sogenannten Open Web Platform (OWP)[11].

ActivityPub definiert zwei Protokollschichten, sowie Konzepte, Sammlungen und Interaktionen für dezentrale soziale Netzwerke. Eine Protokollschicht ist das Client-zu-Server Protokoll (Social API), um Clients den Zugriff auf die neusten an sie gesendeten Inhalte zu ermöglichen sowie zum entgegennehmen von Anfragen die vom Client abgesetzt wurden[5].



Quelle: ActivityPub 2018 - Overview

Abbildung 3.2.: Schnittstellen des ActivityPub Protokolls

Die zweite Protokollschicht besteht aus dem förderierten Server-zu-Server Protokoll (Federation Protocol), welches den einzelnen Instanzen von dezentralen sozialen Netzwerken den Austausch von Inhalten untereinander gestattet. ActivityPub setzt auf bereits bestehende Empfehlungen des W3C auf, welche teilweise auch von der SWWG entwickelt

wurden wie zB. ActivityStreams Core und ActivityStreams Vocab[5]. Die zwei Protokollschichten können unabhängig voneinander implementiert werden.

Auch andere Technologien wie JSON Linked Data werden genutzt um u. A. die Erweiterbarkeit zu gewährleisten. Über neue Ontologien und Vokabulare können weitere syntaktische Definitionen und semantische Beschreibungen zu den bestehenden hinzugefügt werden[5]. Diese Vokabulare können im Kontext des JSON Linked Data Objektes, angegeben werden. Bei ActivityPub wird das ActivityStreams 2.0 Vokabular verwendet welches durch ActivityStreams Vocab erweitert wird.

3.3.1. Bestandteile des Protokolls

Die Hauptbestandteile des ActivityPub Standards sind die folgenden:

- Aktoren
- Objekte
- Sammlungen
- Aktivitäten

In ActivityPub werden Benutzer als „Aktoren“(actors) dargestellt. Diese können nicht nur Personen, sondern auch Applikationen, Organisationen, Gruppen und Services sein[9]. Jedes Aktoren Objekt muss eine „Inbox“ und „Outbox“, welche geordnete Sammlungen sein müssen, sowie eine ID und ein Typ besitzen[5]. Die ID muss global einzigartig sein. Dies kann garantiert werden durch eine Domänen und Protokoll bezogene URI oder IRI wie z. B. „https://example.org/users/alice“oder „https://example.org/users/álicê“. Der Typ eines Aktor (z. B. ”type”: ”Person”) kann variieren zwischen den fünf oben genannten.

3.3.2. Zugehörige Standards und Komponenten

ActivityPub benutzt die ActivityStreams Daten Syntax und das Vokabular. Zusätzlich kann ein weiteres Sicherheitsvokabular[4] benutzt werden um Definitionen zum Bereitstellen eines öffentlichen Schlüssels, Signaturen sowie Verschlüsselten Inhalten u.v.m. zu haben. Am 22 April 2016 hat die „W3C Community Group“ einen Entwurfsbericht herausgebracht. Durch diesen wird neue Syntax und Semantik definiert um Internet basierten Applikationen das Verschlüsseln, Entschlüsseln sowie digitale Signieren und Verifizieren von verlinkten Daten (Linked Data) zu ermöglichen. Es enthält auch Vokabeln für die Erstellung und Verwaltung einer dezentralen Public-Key-Infrastruktur über das Internet[7]. Ein Anwendungsfall ist das holen des öffentlichen Schlüssels eines Nutzers, über dessen Aktoren Objekt, um eine von Nutzer gesendete Nachricht zu verifizieren.

[4]Eine Ontologie die Sicherheitsaspekte definiert wie öffentliche Schlüssel, Signaturen u.v.m.

„ActivityStreams 2.0“ beinhaltet Modelle für Aktoren, Aktivitäten, Intransitiven Aktivitäten, Objekte, Links, Sammlungen, Natürliche Sprachwerte (Strings) und für Internationalisierung. Das Kernvokabular von ActivityStreams 2.0 wird durch ActivityStreams Vocab erweitert. Dazu gehören verschiedene Aktivitätstypen wie z.B. „Accept“,„Add“,„Remove“,„Delete“ und „Create“[5], um Aktorentypen wie „Person“, „Application“ und „Group“[6] sowie um verschiedenste Objekttypen wie „Article“, „Event“, „Note“ und „Relationship“[7].

JSON Linked Data ist eine Erweiterung des JSON Formates um verlinkte Daten zu Repräsentieren. JSON an sich, ist ein Format welches im Web häufig Anwendung findet um Daten auszutauschen. Im Kern sind ActivityStreams 2.0 auch JSON Linked Data Objekte. Der ActivityStreams 2.0 Kontext definiert verschiedene Klassen und Eigenschaften, von denen nicht alle benutzt werden. Typische Klassen sind „Activity“, „Link“ und „OrderedCollection“. Ein Beispiel ActivityStreams 2.0 Objekt sieht wie folgt aus:

Listing 3.1: Beispiel ActivityStreams 2.0 Objekt

3.3.3. 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 Bearer Tokens“ benutzt, zum anderen auf der Server Seite „HTTP“ oder „Linked Data Signatures“zur Sicherstellung der Datenintegrität.

Bei„OAuth 2.0 Bearer Tokens“ handelt es sich um eine Methode um auf geschützte Ressourcen zugreifen zu können[6]. ActivityPub nutzt diese für jegliche Interaktionen mit dem Server.

„Die Datenintegrität umfasst Maßnahmen damit geschützte Daten während der Verarbeitung oder Übertragung nicht durch unautorisierte Personen entfernt oder verändert werden können. Sie stellt die Konsistenz, die Richtigkeit und Vertrauenswürdigkeit der Daten während deren gesamten Lebensdauer sicher und sorgt dafür, dass die relevanten Daten eines Datenstroms rekonstruierbar sind“[**data-integrity**].

Um sicherzustellen das HTTP Anfragen beim Transport nicht verändert wurden, können HTTP Signaturen verwendet werden. Diesen verwenden einen kryptografischen Algorithmus um aus ausgewählten Kopfzeilen einer HTTP Anfrage einen kryptischen Zeichenfolge zu generieren. Auf der Empfängerseite kann die Zeichenfolge mit mitgeliefer-

[5] activity-types https://www.w3.org/TR/activitystreams-vocabulary/
[6] actor-types https://www.w3.org/TR/activitystreams-vocabulary/
[7] object-types https://www.w3.org/TR/activitystreams-vocabulary/

ten und auch nachschlagbaren Information verifiziert werden.

Wenn ein Objekt nicht nur vom Client zum Server gesendet, sondern auch zwischen Servern untereinander weitergeleitet werden soll wird zum Sicherstellen der Datenintegrität ein anderes Verfahren benötigt als HTTP Signaturen. Die „Best Practices“ empfehlen für solche Fälle „Linked Data Signatures“. Der größte Unterschied zwischen HTTP Signaturen und „Linked Data Signatures“ besteht darin, welche Daten zum Erstellen der Signatur verwendet werden. Bei HTTP Signaturen sind es die Kopfzeilen. Mit „Linked Data Signatures“ kann auch das Objekt selbst, also der Payload einer HTTP Anfrage, anstatt nur die Kopfzeilen, zum signieren verwendet werden.

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

Zu Beginn dieses Kapitels sollte sich die Frage gestellt werden wie die Architektur sicher gestaltet wird.

Eine Trennung der ActivityPub Komponenten in einen eigenen Service würde für eine isolierbare Ausführung sorgen. Es

4.1. Entwurfsentscheidung

Für die Nachfolgende Architektur wurde sich in dieser Arbeit entschieden, da sie einen Modularen Charakter hat. Dies ist gegeben durch die Kapselung des Services in eine Express Middleware. Sowohl das alleinstehende betreiben des Services ist möglich als auch die Integration in einen bestehen Express Server.

4.2. Technische Architektur

Der technischen Architektur liegt das Express Middleware Framework zugrunde. Der ActivityPub Service wird als Express Middleware, sowie als allein stehender Express Web Server implementiert. Bei der letzteren Variante wird anstatt die Middleware in ein bestehenden Express Server zu integrieren, ein eigener Express Server gestartet.

In der oben gezeigten Abbildung wurde der ActivityPub Service in einen bestehenden Express Server einer GraphQL API integriert.

Benutzer der API kommunizieren mit dem Web Server über das HTTP Protokoll. Der Web Server leitet die Anfragen dann an den entsprechenden Router weiter, welcher wiederum den Anfrage Inhalt transformiert und „Handler“ Funktionen des ActivityPub Service mit entsprechenden Parametern aufruft.

Die obige Abbildung zeigt eine detailliertere Variante des ActivityPub Service Diagramms aus Abb. 4.1.
Es wird außerdem ersichtlich, dass, um eine maßgeschneiderte Version für ein weiteres Projekt zu erstellen, das Interface IDataSource implementiert werden muss.

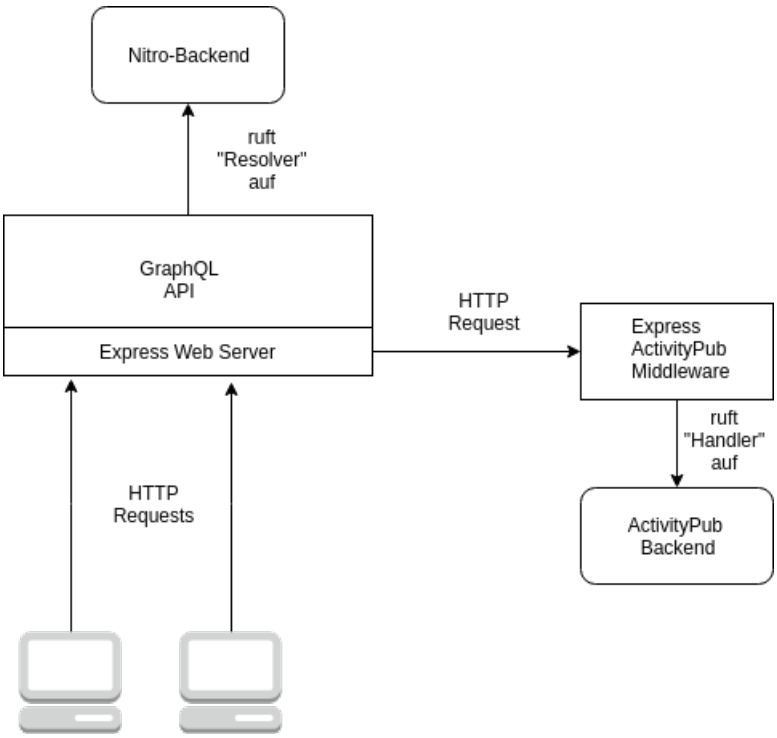


Abbildung 4.1.: Technische Architektur ActivityPub

Sonst wird bei der Architektur weitestgehend darauf Wert gelegt, dass die meisten ActivityPub konformen Inhalte „on the fly“generiert werden können. Damit wird versucht für weitere Projekte die auf diesem Prototypen aufbauen wollen einen einfachen Einstieg zu bieten um andere Datenbanken o.ä. anbinden zu können.

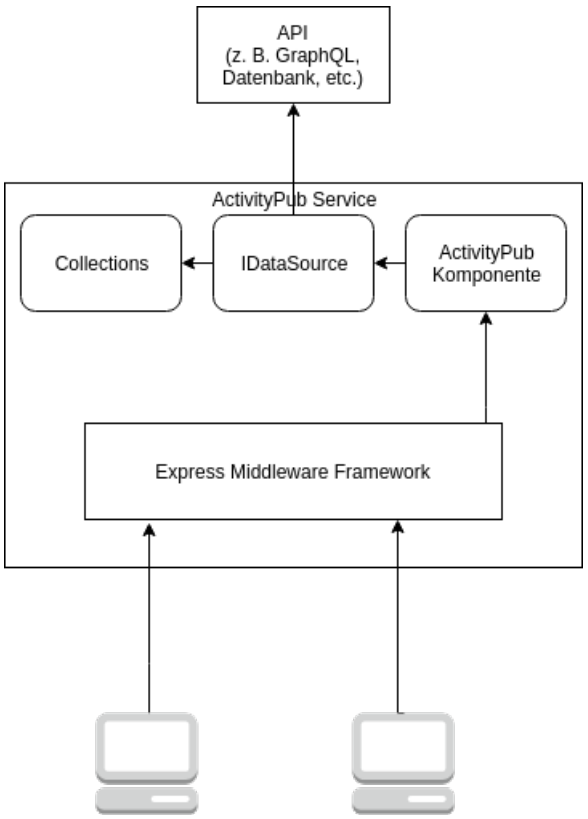


Abbildung 4.2.: Technische Architektur als allein stehender Server

5. Implementierung eines ActivityPub Prototyps

Die IDataSource Implementierung wird für das Unternehmen angefertigt in der die Bachelor Thesis bearbeitet wird. Im Falle man möchte den Service mit einer anderen Datenquelle versorgen, kann das IDataSource Interface implementiert und so angepasst werden, dass eine andere Datenbank Verwendung findet.

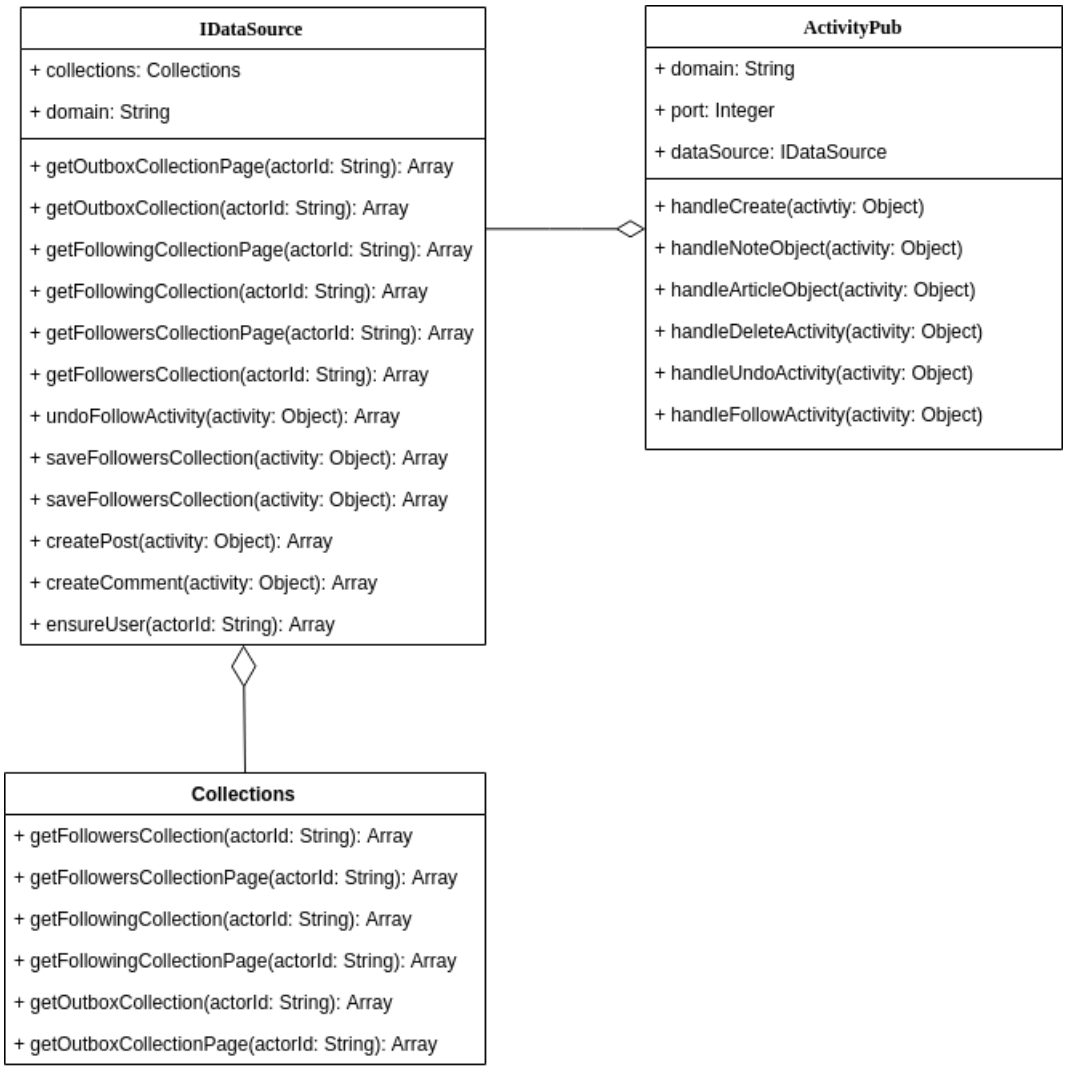


Abbildung 5.1.: Hauptkomponenten des förderierten Servers

Obig abgebildetes Klassendiagramm zeigt die Hauptkomponenten des ActivityPub Services und die enthaltenen Methoden.
In der Collections Klasse sind Weiterleitungen zu entsprechenden Methoden der Datenquelle; Sie dient ausschließlich der Lesbarkeit für den Entwickler.
Bei der ActivityPub Klasse handelt es sich den Eingangspunkt der Express Router. Die entsprechenden Anfrage-Handler wandeln die Anfrage in einen Service Aufruf um. Diese Klasse enthält nicht nur Handler für eingehende, sondern auch Methoden zum senden von Anfragen.

Um eine möglichst Modulare Implementierung zu gewährleisten wurden alle Datenbankspezifischen Methoden in das IDataSource Interface ausgelagert. Durch die Implementierung dieses Interfaces können Aktivitäten ActivityPub konform empfangen werden. Das senden wiederum muss jedes Backend selbst implementieren. Dafür stehen in der ActivityPub Klasse Methoden zum senden von Aktivitäten bereit. Durch einfaches importieren der ActivityPub Klasse erhält man eine Instanz. Darüber sind die Methoden verfügbar.

Es ist auch möglich andere Datenquellen als die in der Bachelor Arbeit verwendete GraphQL Schnittstelle zu nutzen, wie z. B. MySQL oder MongoDB.

5.1. Server-zu-Server Protokoll

Der Funktionsumfang des förderierten Servers beschränkt sich auf den folgenden:

- Empfangen von ***Article*** und ***Note*** Objekten am Geteilten- und Nutzernachrichteneingang (sharedInbox, Inbox)
- Empfangen von ***Like*** und ***Follow*** Aktivitäten am Geteilten- und Nutzernachrichteneingang
- Empfangen von ***Undo*** und ***Delete*** Aktivitäten für ***Articles*** und ***Notes*** am Geteilten- und Nutzernachrichteneingang
- Bereitstellen eines ***Webfinger*** und ***Aktoren*** Objektes
- Bereitstellen der ***Followers***, ***Following*** und ***Outbox*** Sammlungen

Artikel und Notizen werden bei der in dieser Arbeit angefertigten Implementierung gleich behandelt. Beim empfangen einer Create Aktivität die eine Notiz oder Artikel als Objekt Attribut hat, wird der Artikel über die IDataSource Implementierung erstellt und entsprechende Metadaten zum rekonstruieren der ID's gespeichert. Somit können empfangene Posts, welche an die Öffentlichkeit gerichtet sind, im Netzwerk angezeigt werden.

Wie wird nun sichergestellt, dass ein Nutzer dazu berechtigt ist

Eingehende HTTP POST Anfragen werden auf eine vorhandene Signatur Kopfzeile geprüft. Wird diese nicht gefunden, wird die Anfrage verworfen. Ist sie vorhanden, wird die Signatur geprüft indem das Aktoren Objekt über die zugehörige keyId angefragt wird und die Signatur gegen den öffentlichen Schlüssel geprüft wird. Dies stellt sicher, das der Inhalt der Nachricht beim Transport nicht verändert wurde. Eine Authentifizierung des Clients wird nicht benötigt, da es sich um eine öffentliche Schnittstelle handelt.

6. Evaluation

6.1. Anwendungsbeispiel

6.2. (Performanzmessungen)

6.3. Disskusion von Vor- und Nachteilen der Lösung

7. Zusammenfassung und Ausblick

Literatur

[1] W3 Consortium. *Social Web Working Group*. Date accessed: 12.12.2018. 2018. URL: https://www.w3.org/Social/WG.

[2] W3 Consortium. *W3C launches push for social web application interoperability*. Date accessed: 21.07.2014. 2014. URL: https://www.w3.org/blog/news/archives/3958.

[3] Fediverse Gemeinschaft. *Fediverse*. Date accessed: 17.01.2019. 2019. URL: https://fediverse.party/en/fediverse/.

[4] Fediverse Gemeinschaft. *Fediverse Network Report 2018*. Date accessed: 17.01.2019. 2019. URL: https://fediverse.network/reports/2018.

[5] Christopher Lemmer Webber u. a. *Der ActivityPub Standard*. Date accessed: 12.12.2018. 2018. URL: https://www.w3.org/TR/activitypub/.

[6] D. Hardt M. Jones. *RFC 6750 - OAuth 2.0 Authorization Framework: Bearer Tokens Usage*. Date accessed: 24.01.2019. 2019. URL: https://tools.ietf.org/html/rfc6750.

[7] Inc. Manu Sporny Digital Bazaar. *The Security Vocabulary*. Date accessed: 22.04.2016. 2016. URL: https://web-payments.org/vocabs/security.

[8] James M. Snell und Evan Prodromou. *Activity Vocabulary*. Date accessed: 23.05.2017. 2017. URL: https://www.w3.org/TR/activitystreams-vocabulary/.

[9] James M. Snell und Evan Prodromou. *ActivityStreams 2.0*. Date accessed: 23.05.2017. 2017. URL: https://www.w3.org/TR/activitystreams-core/.

[10] Christian Stobitzer. *Netzwerkeffekt*. Date accessed: 29.12.2018. 1986. URL: http://www.wirtschafts-lehre.de/netzwerkeffekte.html.

[11] W3C. *Social Web Working Group Charter*. Date accessed: 17.12.2018. URL: https://www.w3.org/2013/socialweb/social-wg-charter.

[12] Wikipedia. *Dezentrale Stromerzeugung*. Date accessed: 06.11.2018. 2018. URL: https://de.wikipedia.org/wiki/Dezentrale_Stromerzeugung.

[13] Wikipedia. *Dezentralisierung (Politik)*. Date accessed: 10.08.2018. 2018. URL: https://de.wikipedia.org/wiki/Dezentralisierung_(Politik).

A. Anhang

A.1. First Appendix Section

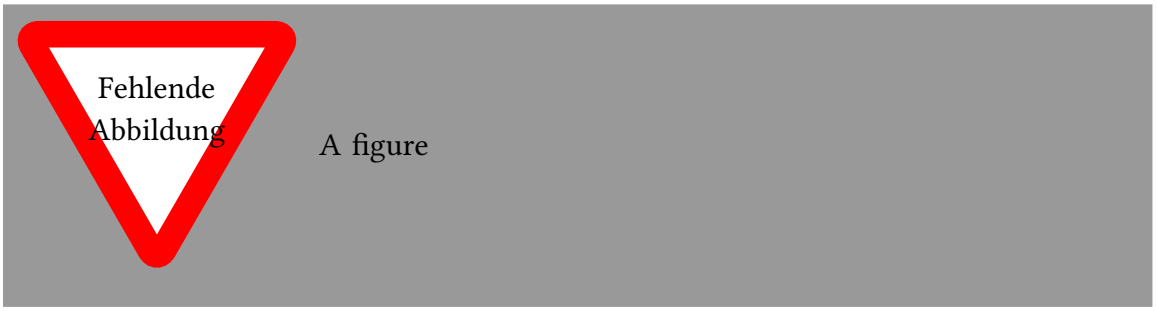


Abbildung A.1.: A figure

...