Extending the OAuth2 Workflow to Audit Data Usage for Users and Service Providers in a Cooperative Scenario

Marius Politze[1], Bernd Decker[2]

Abstract: The increasing amount and heterogeneity of devices demands changes in IT infrastructure. Many web service architectures used to meet these demands use the OAuth2 workflow to secure their interfaces. These implementations usually tightly couple web services and an OAuth2 authorization service. The presented extension to the OAuth2 workflow is capable handling authorizations for multiple attached services and therefore combines existing services of a central IT service provider but also allows other services running in a cooperative model with only a single instance of the authorization server. Based on auditing parameters it is possible to present access per resource or per method giving service providers and application developers more insight in how their services are used and show users by whom their personal data is used.

Keywords: authorization, micro services, mobile, OAuth2, privacy, security, SOA, transparency

1 Introduction

Globalization and digitalization pose continuous challenges to established processes in government and administration. The standardization of existing processes to improve cooperation between institutions, reduce overall costs or increase efficiency is a routine at several levels: global, national and local. Even at the local level compatibility and reusability of available components is key to meet these challenges.

Due to Increased mobility and the rising number of students the universities have to standardize existing processes, improve cooperation between institutions, reduce overall costs or increase efficiency. In addition, students' demands on universities and their employees have changed. Previously, central IT service providers introduced various processes and supporting IT infrastructure and applications to meet these demands. Not only IT infrastructure and applications are becoming more important to the universities' processes and employees but to students and their daily lives [Ju09]. This leads to increased competition among universities to present most appealing services to their students. Several research groups and IT service providers have recognized this and IT service providers and lead to improvement of existing services [Mi12] as well as new methods in designing future IT services [KL16].

[1] IT Center RWTH Aachen University, Seffenter Weg 23 52074 Aachen, politze@itc.rwth-aachen.de
[2] IT Center RWTH Aachen University, Seffenter Weg 23 52074 Aachen, decker@itc.rwth-aachen.de

1.1 Problem statement

Modern process supporting information systems usually do not consist of monolithic structures but of loosely coupled services. Each of these components is responsible only for certain steps within the supported processes. From the software engineering point of view, this kind of micro service architecture has several advantages like easier maintainability, expendability and replaceability of the components used.

The processes base on personal data passed between different services within the infrastructure. As the coupling of services decreases, so does the control of transferred, processed or saved date in the different parts of the process. With the OAuth2 workflow service providers and users can identify a subset of information that they allow be exchanged between the different steps of the process. However, it remains opaque which information the application actually uses.

The OAuth2 workflow forms an integral part of most of these processes. The management of authorizations already collects granted authorizations for users and services as well as metadata, like the application that requested the authorization. This allows using the OAuth2 workflow as a starting point to gain more transparency on how applications use personal data for service providers and users.

1.2 The OAuth2 Workflow

The OAuth2 workflow is described in RFC 6749 [Ha12] allows secured, personalized access to web services or resources and handles the users' authorization without supplying credentials to the application itself. This also paves the way for third party developers accessing central IT services. Generally, it follows the steps 1-4 shown in Figure 1.

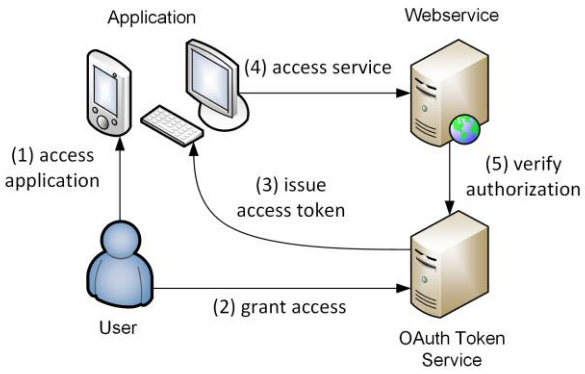

Figure 1: Schematic of the OAuth2 workflow

At first, the user accesses the application (1). To access the web service resources for the user, the application needs an access token. Therefore, the application directs the user to the token service where the user logs in and grant access for the application (2). The token service then issues the access token (3). The application can now use the token to access the web service resources (4). The web service now has to verify the authorization (5). Most implementations tightly couple web services and token services, so this step remains internal. To decrease coupling of the services, explicitly modelling of this step is a key part of the cooperative workflow proposed in the following sections.

2 Current Implementation

In order to access the services currently available, third party developers have to perform a simple registration process. Apart from a contact name and email address, they have to supply the use cases their app covers and what data it needs from the services to perform these tasks. The register of all applications is publicly visible to all users. Third party developers extend the functionality of existing IT systems and applications using the OAuth2 workflow.

Furthermore, the implemented workflow is capable of handling authorizations for multiple attached services. Central IT service providers but also for other services running in the university context use it cooperatively. Allowing not only the reuse of already established infrastructure but also the use of OAuth2 authorization for inter service communication while providing full transparency to the user.

2.1 Existing Service Oriented Architectures

Initially RWTHApp [PD14] demanded an infrastructure to make data provided by legacy systems accessible from smart devices in a secure and consistent fashion. Nevertheless, some rather traditional web applications are already taking advantage of the centralized implementation of the infrastructure. Even though these services centralize access to legacy systems, it builds upon some of the paradigms introduced by micro services: primarily the loose coupling of functional units to access the independent legacy systems. These functional units are centrally published and appear as a single web service instance using consistent access and naming conventions.

Apart from security and consistency, the design of the infrastructure also tries to increase availability and speed of the legacy systems by introducing automated and semi-automated caching layers in order to increase overall performance and improve user experience. The cache uses a probabilistic, proactive model to predict future service calls [PSD16]. While most legacy systems require some kind of integrated authentication infrastructure to handle authorization and data access, most modern systems offer services that are more flexible and allow delegation of authorization. This allows integration into the OAuth2 workflow using the cooperative model. Using this general architecture, several services in the field of campus management, student lifecycle, e-learning and other university services like canteen menus and university library are already available.

2.2 The Cooperative OAuth2 Model

Since most application developers use OAuth2 to authorize a client application to run in a user context, also implementations of the OAuth2 workflow focus on client side authorization. Many major IT companies like Google, Facebook and GitHub offer to authorize the use of their services using an OAuth2 workflow. However, each of these companies offer their own authorization service that depends on the services offered and vice versa.

To allow authorization within a cooperation of various service providers the OAuth2 workflow needs to offer the means to verify that a certain access token is valid and to identify the application and the user that requested the token. The *4*-tuples of validity, application, identity, and service provider form the context in which the token may be used. The main features of the cooperative model are:

- there is only a single instance of the authorization server
- all service providers are known to the authorization server
- the identity information needed by the service providers is known to the authorization server

Provided a token by the application, the service provider can therefore resolve the context of the token from the authorization server. It is furthermore easy to add new service providers to the cooperation to extend the features of the interfaces offered to the users. It is however important that tokens issued before adding a service remain invalid for the new service until the user authorizes a token for the service.

This model relates to the authentication via authorization workflow used by many current mobile and internet applications. However, there is a major difference: The authentication via authorization workflow usually uses some sort of user information service to identify the user. Consequently, the user authorizes the application only to access this service and not the actual services offered by the application vendor. This leads to major security flaws [YLL16]. The cooperative model resolves these issues by explicitly authorizing specific services within the cooperation.

2.3 Implementation of the Cooperative Model

The initial OAuth2 workflow uses four endpoints to perform the authorization of the tokens:

- *Authorize*
  The endpoint is used to authorize tokens for server side and web applications.

- *Code*
  The endpoint is used to request codes that can be shown to the user in order to authorize an installed application.
- *Token*
  The endpoint is used to manipulate access tokens during and after the authorization process. The manipulations include extending the lifetime of the token or invalidating a token.
- *TokenInfo*
  The endpoint supplies information about a token. Applications can use the endpoint to verify that the token is valid and actually belongs to the application.

With the exception of the Authorize endpoint, all endpoints are REST web service methods that are called using a HTTPS POST request. In the cooperative model, the new context endpoint extends this set:

- *Context*
  The endpoint is used to resolve the context of a token for a certain service provider.

Again the context endpoint is a REST web service that is called using HTTPS POST. If the token is valid for the service provider calling the method, it returns the *4*-tuple of the context.

3 Data Usage Audits in the OAuth2 Workflow

In order to achieve more transparency for the user, the cooperative OAuth2 workflow as shown in Figure 2 needs further extensions. When verifying the context of the token, the service providers may add auditing information to their request. The authorization server then processes and aggregates the information. While most service providers are very well capable of collecting such data, they rarely make it available to users or application developers. This workflow therefore presents a user centric way to provide more transparency on the usage of personal data.

3.1 Auditing in the Cooperative Workflow

Service providers have to call the context endpoint to validate the token essentially every time an application requests a resource. At this point, it is possible to log the number of occurrences of the *4*-tuples that describe the context. Information from this audit log, allows producing an overview for users on which applications and services were active due to their authorization.

Service providers and application developers may also access auditing information. This does not only allow the deduplication of information but also removes some necessity to save personal data on remote locations. After all enforcement of laws and best practices concerning personal data and privacy on a single central system serving the OAuth2 workflow is easier.

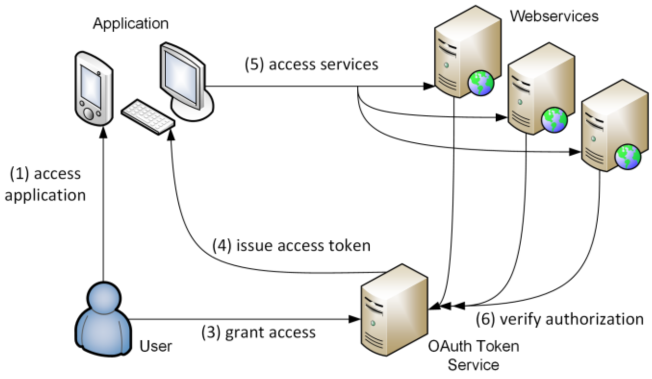

Figure 2: Schematic of the cooperative workflow

3.2 Further Extending the Cooperative Workflow

In the initial model, the context does not include the actual resource accessed by the user. To make auditing more expressive, the context endpoint is extended. Service providers may therefore append additional auditing data to context requests. Basing on the *4*-tuple of the context this information is added. While it is generally possible to extend the *4*-tuple to an *n*-tuple of arbitrary length, the additionally logged fields for auditing are limited to preserve performance of the actual authorization workflow. The auditing system allows appending the following auditing parameters, all of which are optional:

- *Resource*
  The actual resource requested by the user. For most RESTful web services this is equivalent to the URL requested by the application.
- *Operation*
  The kind of operation performed on the resource. For most RESTful web services this is equivalent to the HTTP method used by the application.
- *Cost*
  The cost caused by the call in terms of computing resources. It is best practice that the service provider checks the authorization before taking any actions. Even though real costs are generally unknown a priori, this allows the service provider to supply an estimate based on the resource and the operation.

This extension allows service providers and users to gain insight on how applications use resources and personal data. Again all auditing information collected by the authorization server are subject to personal data and privacy laws and operators can and should strictly enforce them this point.

4 Auditing Usage of Personal Data

As previously shown both variants of the cooperative workflow generally deliver auditable information. However, it is important to note that insights for the users as well as service providers and application developers are more valuable if additional auditing information is available. It is therefore important that the service providers join in and supply the additional information needed.

4.1 For Service Providers

The auditing information presented to the service provider aim at giving an overview of how their services and resources are used. The basic variant of the cooperative workflow however does not allow resolving information for individual resources but can only distinguish between different applications accessing the resources.

Even though they are not the primary target audience for the auditing, service providers need to be convinced to provide the additional information. Using the extended workflow, service provider gain additional value by partitioning the information further. Based on the auditing parameters it is now possible to present access per resource or per method as shown in Figure 3 giving service providers more insight in how their services are used. Furthermore, the cost parameter allows identifying applications that put more excessive load on the services than others and therefore may provide an indication if applications are abusing the services offered.

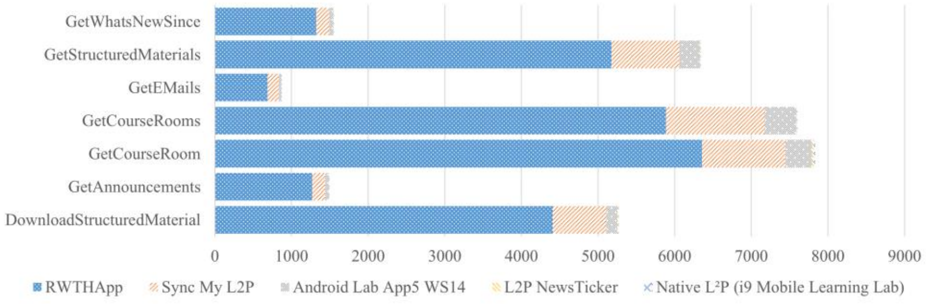

Figure 3: Selection of endpoints offered by the eLearning system used by different applications.

4.2 For Application Developers

In general, it is very common for application developers to retrieve information on how their applications are actually used. Application platforms like Android and iOS already offer detailed reports on the kind and number of devices running the application. In addition, most of these platforms offer usage reports that allow advanced analysis of user behavior.

Even though these tools are sophisticated, their use is debatable due to privacy concerns by the users. To overcome the need to include such tools in their application and protect the users' privacy, application developers can access some insights from the auditing. This especially includes the number of users who used the application as well as the number of requests issued to the different endpoints and service providers as shown in Figure 4.

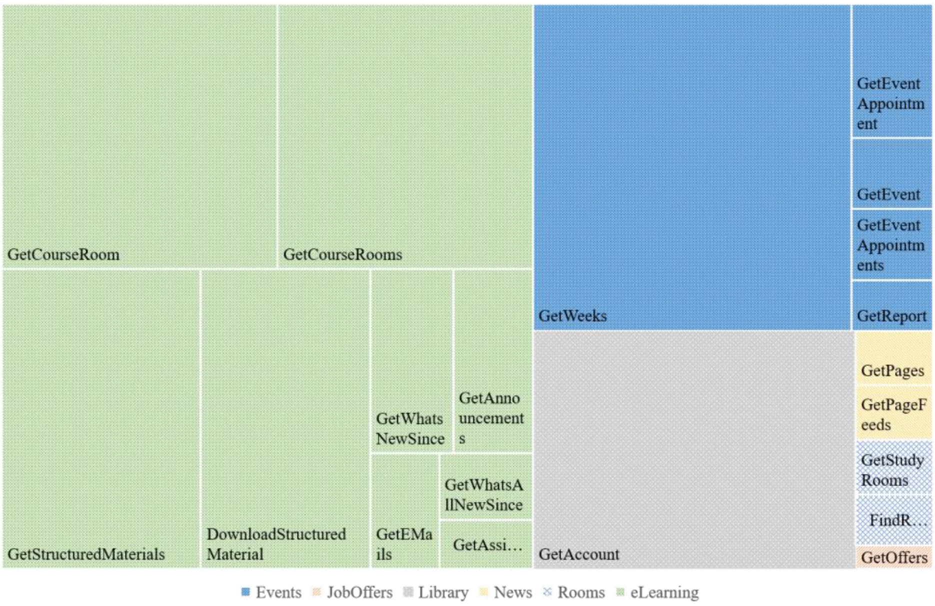

Figure 4: Treemap of a selection of endpoints and service providers called by RWTHApp. The area of the rectangles is relative to the frequency of calling the endpoint.

4.3 For Users

The focus group of the auditing are the users using the services. This is extremely important, as they most certainly have no other means of collecting and auditing how applications access their personal information. Users can therefore gain insight into their full profile. They see when an application requested a certain resource. An example of such a usage profile is shown in Figure 5.

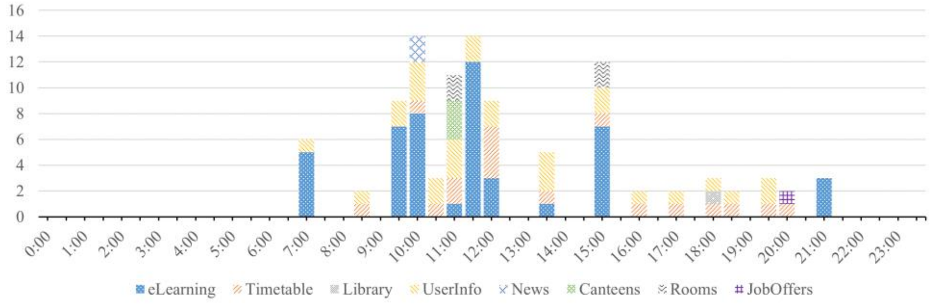

Figure 5: Overview for one user showing the different service providers used by RWTHApp.

To make the information collected for the auditing even more transparent; the users can access to their raw *n*-tuples saved for the auditing process. For concerned users this allows an in depth analysis by whom their data is used. In practice, this information is not saved permanently: To adhere with privacy best practices the data should be anonymized in regular intervals. In the current implementation, this limits users to access their history of the past two weeks.

5 Conclusion

The OAuth2 workflow and the cooperative model allow the introduction of a centralized auditing structure. Apart from providing reports to service providers this approach allows making all information available to the user and therefore leads to a more user centric and transparent auditing. By centrally collecting the auditing information, operators can control the enforcement of data privacy laws and best practices in a better way.

To add more value to reports for users, service providers need to supply additional information. Using the proposed extended cooperative workflow as a standard, lowers hurdles that service providers have to take. It offers additional value by also providing access to reports for service providers.

6 Future Work

The most recent implementation collects auditing data according to the extended cooperative workflow. Monthly reports then condensed the information. However, it is currently not possible for service providers nor users to access their current data. In order to achieve this a more interactive application needs to be built. Dashboards like these are very common in smartphone app stores and analytics tools, a source of information that is, however, usually not available for uses but only for application developers.

The cooperative model only presents one of the possible extensions in the OAuth2 workflow. In future applications may use OAuth2 outside of these boundaries: While application developers are typically from third parties, users and service providers from outside the cooperation cannot access the current infrastructure, per se. As more and more services and service providers are shared across university boundaries, future extensions are needed to transition from the cooperative to a federative model for OAuth2 workflows.

Literature

[Ha12] D. Hardt, The OAuth 2.0 Authorization Framework: RFC Editor, 2012.

[Ju09] W. Juling, Vom Rechnernetz zu e-Science, PIK - Praxis der Informationsverarbeitung und Kommunikation, vol. 32, no. 1, 2009.

[KL16] P. Kupila and U. Lehtonen, Engaging students in building better digital services, in 22nd EUNIS Congress, Thessaloniki, 2016, pp. 126–129.

[Mi14] J. Mincer-Daszkiewicz, We Publish, You Subscribe — Hubbub as a Natural Habitat for Students and Academic Teachers, in 20th EUNIS Congress, Umea, 2014.

[PD14] M. Politze and B. Decker, RWTHApp: from a requirements analysis to a service oriented architecture for secure mobile access to personalized data, 20th EUNIS Congress, Umea, 2014.

[PS16] M. Politze, S. Schaffert, B. Decker. A secure infrastructure for mobile blended learning applications, European Journal of Higher Education IT 2016-1

[YLL16] R. Yang, W. C. Lau, and T. Liu, Signing into One Billion Mobile App Accounts Effortlessly with OAuth2.0, in Black Hat Europe, 2016.