NAME OF THE LAYOUT
Title slide

ng image
. Select the
idebuilder' and

' and image

itton on the image

o to the tab
click on
mage itself with
age frame with

click
Inspiration

NAME OF THE LAYOUT

Text #3

t level-buttons to

click

Inspiration

Outline

- **Why?**
- **How?**
- **Demo**
- **Your feedback**

- **Draft spec: https://github.com/SURFnet/surf-token-based-access/**

SURF

We're defining an API for adding scope configuration to an Authorization Server

Where we stand now:
- We have a demo (smoke and mirrors)
- First draft of a specification
- There are still open questions

Looking for feedback. Is this more generally useful?

NAME OF THE LAYOUT

Text #3

t level-buttons to

click

Inspiration

Why?

- SURF – Dutch National Research and Education Network
- Hosting SAML Authentication federation with OIDC Provider
- Hosting Authorization and Access management services for Researchers
- Group based access management, provisioning LDAP and SCIM. non-web
- Authorization beyond what attribute based and role based can provide
- Research infrastructure proxies are being created internationally. E.g. https://aarc-project.eu/

SURF

We work for SURF (https://surf.nl), the Dutch national education and research network. Members are Dutch Universities, higher education institutions, research institutes and related entities. SURF literally has a network as-in part of the internet. SURF also provides many other IT services to Dutch Universities and research institutions. From the national supercomputer to IT procurement.

SURF has been hosting a authentication federation since 2007, now SAML 2.0 based (SURFconext). Have been offering a OpenID Provider for many years now. We have always been offering this using a central proxy.

A few years ago we added a authentication and authorization proxy hosted as-a-service specifically for researchers (SRAM) that offers:
- Group/authorization management
- Provisioning using SCIM and LDAP
- Many system integration tools typically needed for research like SSH , PAM to deal with non-web use

Good news: all new developments in the research space are moving to some

token based architecture/design. Bad news: these can still be very different.

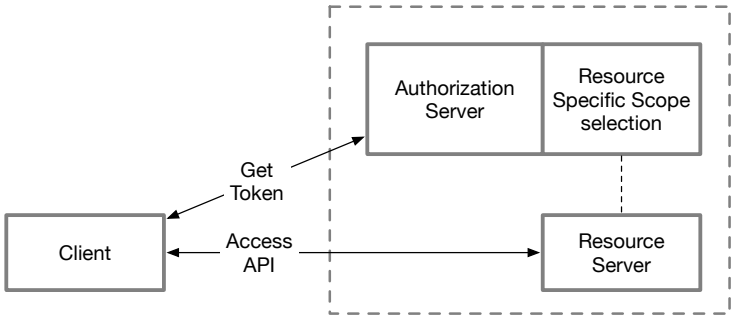
These kind of infrastructure proxies are being created more and more, and efforts are ongoing to standardise how these work, and can work together. See the AARC project: https://aarc-project.eu/ (full disclaimer, SURF is a participant in AARC)

One thing that has been especially challenging has been authorization. Authorization requires all participants to be aligned, otherwise the authorization information that they exchange is meaningless. The more fine-grained the authorization, the more difficult this becomes.

We are looking at fine-graind authorization here. E.g. a user giving access to a particular (set of) directories on a storage server. This is not something that can be easily solver using role (scope) or attribute based authorization. And certainly not in a user friendly way. The typical “bad” way to do this is:
1. User goes to the storage server to setup the required access rights and assigns it to a scope. The user somehow knows where and how to do this. User writes down the name of the scope.
2. User enters the name of scope from the previous in the client so the client can the request this scope. The user somehow knows how and where to do this.

NAME OF THE LAYOUT
Title only #1

Close relationship between Authorization Server and Resource Server



he authorization server needs to know how to present scope selection to the user, this requires the authorization server to have knowledge of the resource server and for fine-grained authorization, to be able to have access to the users resources.

There may also be a dependence of the Resource Server on the access token format used by the Authorization server

When the AS and the RS are being operated by the same entity this is not an issue. But what if this is not the case, as it is in our use case where we host the AS as a service and many different external resources servers want to use this service?

NAME OF THE LAYOUT

Text #3

t level-buttons to

click

Inspiration

Design choices

- No changes required to Resource Server and Clients
- Heavy lifting is done by the Authorization Server
- User experience without dead-ends
- Structured scopes

SURF

No changes required to Resource Server and Client, so existing libraries and implementations can be used as-is

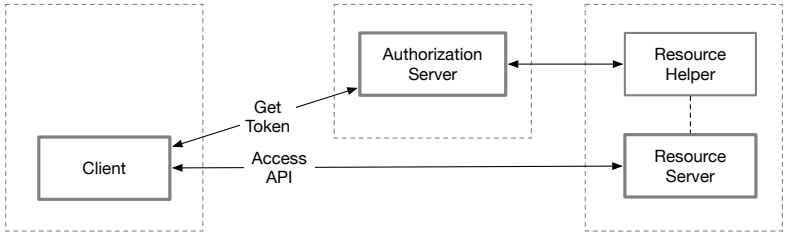
Heavy lifting is done by the authorization server, there is only one AS. There will be many clients and Resource Servers. Changing these is a burden and the entities hosting operating them are often not the ones the develop the software.

Usability - User flow without dead-ends, guide the user though the entire process. No “having to know where to go next”

Structured scopes - i.e. not strings but JSON blobs that describe the authorization in a way that (only) the RS understand

NAME OF THE LAYOUT
Title only #1

Loose relationship between Authorization Server and Resource Server



SURF

Our solution

Add a helper along-side the resource server that handles the resource specific scope selection. This allows the Authorization server to externalize the scope selection by forwarding the user to the Resource Helper using a standardised API. We want to keep the API between the Authorization Server and the Resource Helper as simple as possible. We want to keep it as simple as possible to implement a Resource Helper.

The Resource Helpers and the associated Resource Servers are registered at the AS beforehand.

We were Inspired by structured scopes and the intent lodging pattern used in payment transactions and the Federated Authorization for User-Managed Access (UMA) for communicating the selected scope back from the RH to the AS

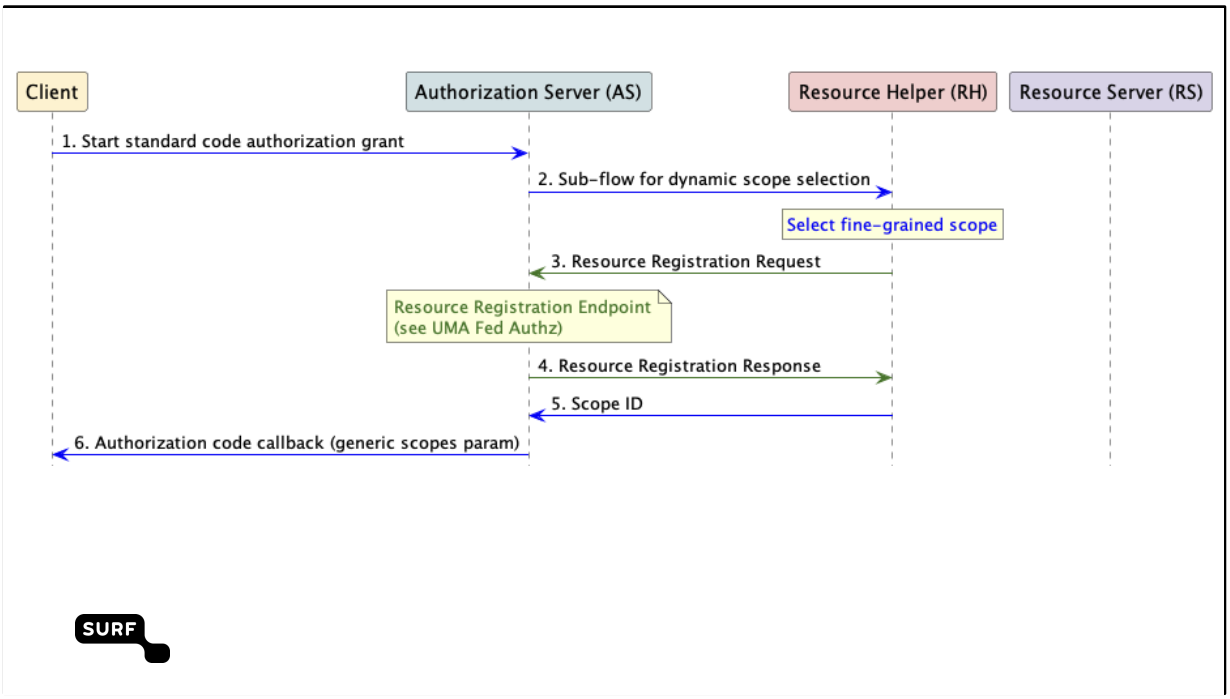
Difference: in our case the client is not involved in creating the structured scope, instead the Resource Helper creates the structured scope. The resource helper sends back: information (JSON) for in the access token, information to be returned by the

introspection endpoint (JSON) and a label (string) with a human readable description of the access that was granted.

Doing this proses several interesting questions:
- In many cases the Resource Helper must authenticate the user. This does not have to be the same user as the user that was authenticated to the AS. We can think of mechanisms to transfer the identity (sub) of the user from the AS to the RH, at the cost of complexity. The RH could very well be an Relaying Party, but we do not want to mandate OIDC here.
- What does it mean when the RH sends the scope back to the AS? How should we call this?
- The RH returns a scope label – a human readable description of the scope. Another option would be to have a “display this scope information” interface on the RH.
- We propose to make the label editable by the user, it is there to help them remember what the token stands for (e.g.) when revoking. It must not be used for authorization purposes. This may become a security risk if the label is used for more than informative purposes, or when the authorization process involves multiple persons (GNAP) and the first user sets a misleading label.

NAME OF THE LAYOUT

Title only #1



See draft protocol: https://github.com/SURFnet/surf-token-based-access/blob/main/README.md

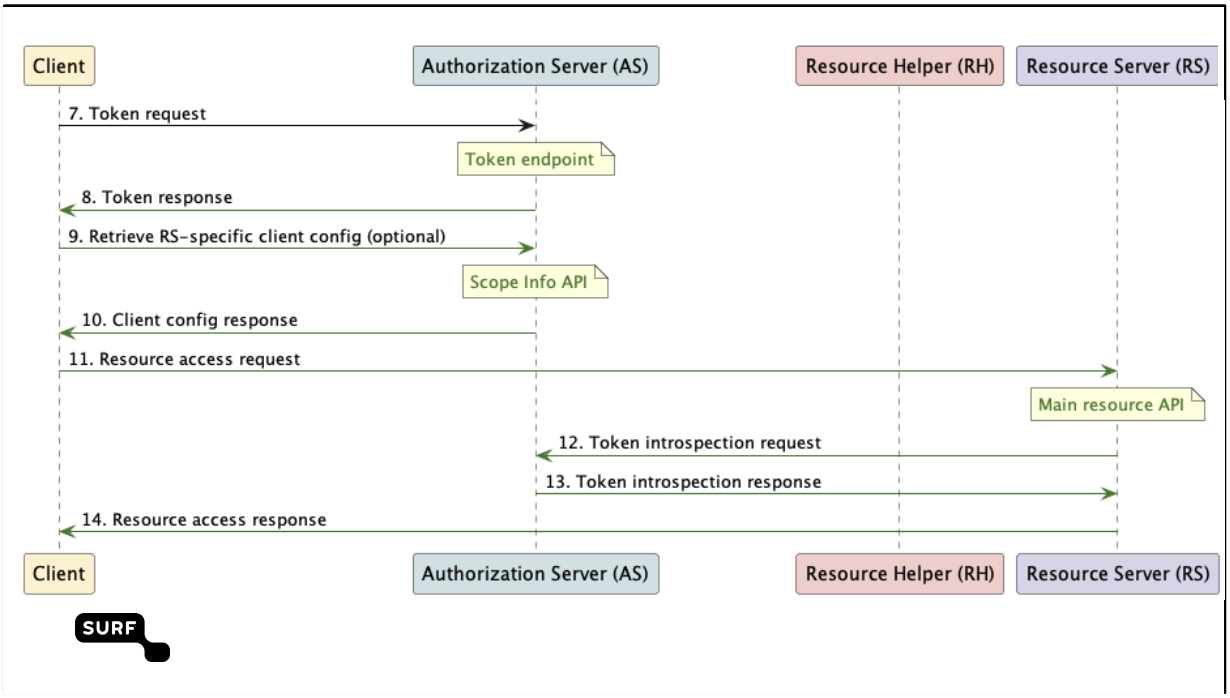
Blue arrows indicate front channel, green arrows indicate back channel.

1: Standard OAuth authz request, may include scope
2: The forward to the RH
3: Send all the information to the AS
5. Send the user back.
6. Standard Oauth.

Extension:
- In step 1 the client can request a protocol (e.g. webdav), instead of specific resource server. The AS will ask the user which RS to use in that case and forward to the right RH.
- Use scope for asking for a protocol? Or audience? What makes sense?

NAME OF THE LAYOUT

Title only #1



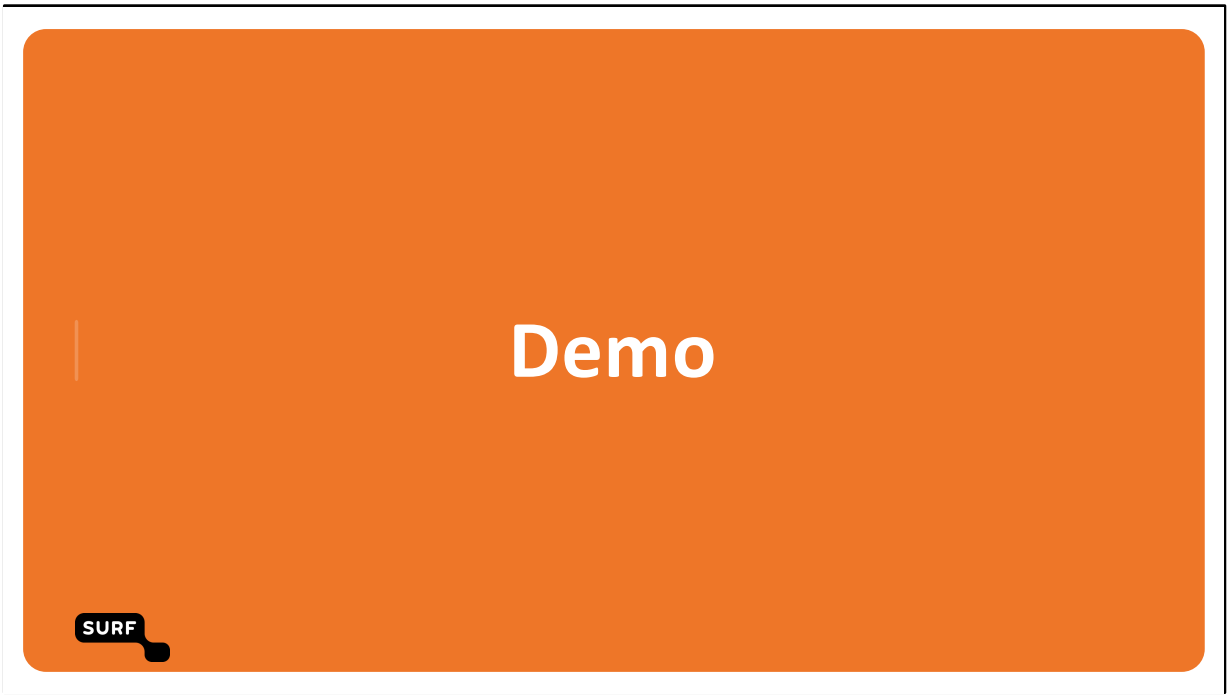
7 & 8: Standard OAth, getting the access token.

A little surprise in steps 9 and 10:

The idea is that instead of asking for a specific resource server (audience) the client in step 1 asks for a protocol. The AS then can show a UI to the user to select an appropriate RS an RH. However, the client will need to retrieve the protocol configuration to use as it does not know about the selected resource.

The idea is to ask for e.g. “webdav”, and than any webdav resource can be used. The client will have to be aware of it, because it need to determine where to send its API calls.

NAME OF THE LAYOUT
Chapter orange



Demoed: https://github.com/SURFnet/surf-token-based-access/tree/main/phase-2/poc-3