#460

XRootD/SciToken-Based Access to Lustre Storage at GSI

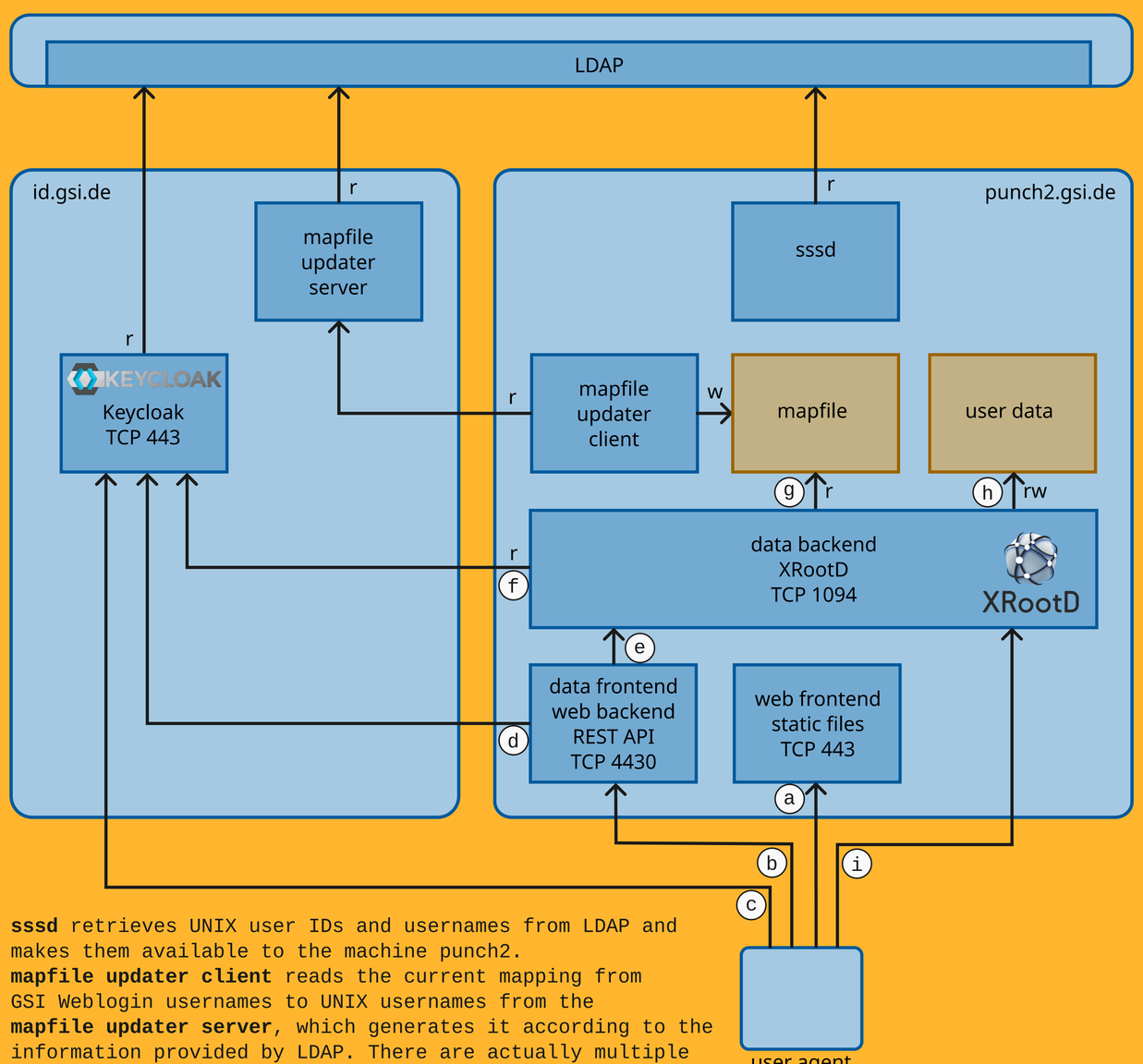
A First Step Toward Data Federation for FAIR

Mohammad Al-Turany, **Sören Fleischer**, Thorsten Kollegger, Anar Manafov, **Rouven Spreckels**

Goal

- Make reading+writing files available via both CLI and web browser.
- Implement authentication using GSI Weblogin and SciTokens.
  Weblogin is an account at GSI intended for employees, apprentices, external companies, guests, guest scientists.
- Utilize POSIX ownership and permissions to manage access rights.

Block Diagram

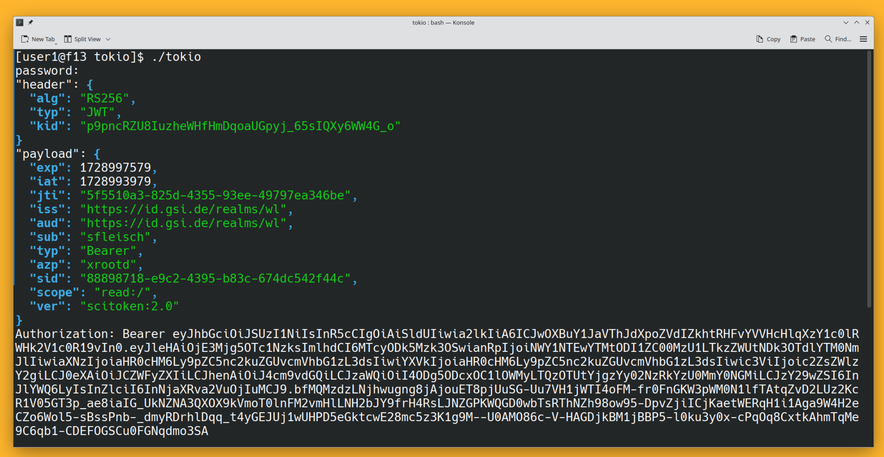


sssd retrieves UNIX user IDs and usernames from LDAP and makes them available to the machine punch2.
mapfile updater client reads the current mapping from GSI Weblogin usernames to UNIX usernames from the **mapfile updater server**, which generates it according to the information provided by LDAP. There are actually multiple **LDAP** machines involved. In this illustration, they are depicted as one block.
Keycloak is configured to require 2FA.

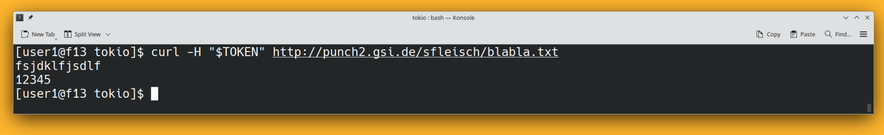
User Workflows

CLI

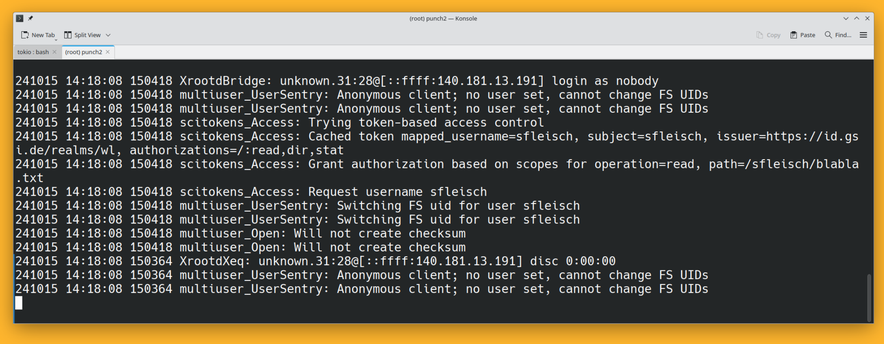
The user initially does not have a token.
(c) The user sends an HTTPS request to id.gsi.de, specifying their GSI Weblogin username and password. If everything is correct, the server responds with a token.



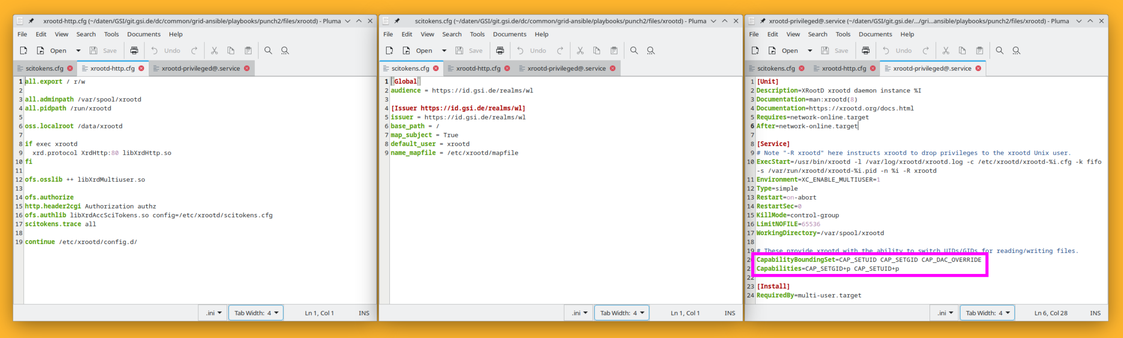
(i) The user sends their file request directly to the XRootD server, specifying their token.



(f) XRootD downloads the public key of id.gsi.de and verifies that the token is signed by id.gsi.de.
(g) XRootD reads the mapfile to determine which UNIX username belongs to the Weblogin username specified in the token.
(h) XRootD performs the actual file I/O on the user data as the UNIX username found in the mapfile. The permission check is then performed by the Linux kernel according to POSIX rules.

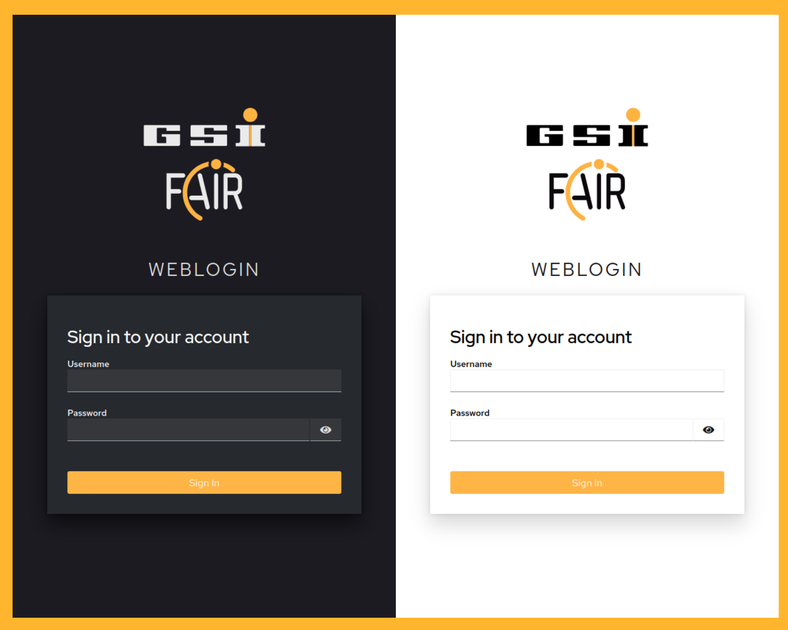


Service Configuration

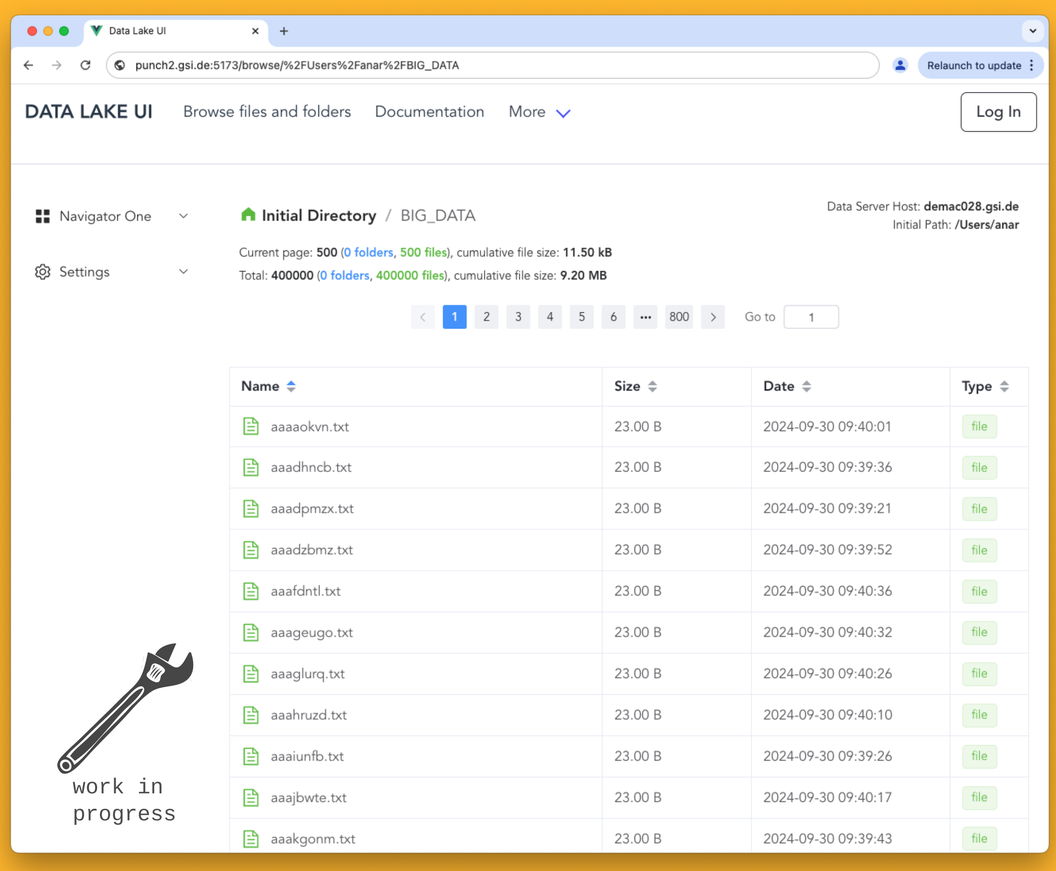


Web Browser

The user initially does not have a token in their browser.
(a) When the user initially browses to the web frontend, the JavaScript code is downloaded to the user's browser.
(b) The code, which is now running in the user's browser communicates with the web backend. Since the user's browser does not yet have a token, they do not see any data, but they have the option to log in.
(c) When the user clicks log in, they browse to id.gsi.de, where they can supply their credentials. If the authentication succeeds, id.gsi.de answers to the user's browser with an HTTP redirect response to the web backend with an ephemeral one-time code in the GET parameters.



(b) The code in the user's browser makes the aforementioned HTTP request with the one-time code to the web backend.
(d) The web backend sends the one-time code to keycloak, which, given correct inputs, returns the tokens. These are returned in an HTTP response to the user's browser.
(b) Subsequents requests from the user's browser to the REST API contain the token.
(e) The web backend requests the list of files in the specified directory from the XRootD server.



(f) XRootD downloads the public key of id.gsi.de and verifies that the user-supplied token is signed by id.gsi.de
(g) XRootD reads the mapfile to determine which UNIX username belongs to the Weblogin username specified in the token.
(i) The links to the actual files are pointing directly to the XRootD server, which is contacted directly by the user's browser when they click on a file.
(h) The XRootD executable performs the actual file I/O on the user data as the UNIX username found in the mapfile. The permission check is then performed by the Linux kernel according to POSIX rules.