SciTokens and Credential Management

Alex Withers alexw1@illinois.edu
Derek Weitzel dweitzel@unl.edu
Jim Basney jbasney@illinois.edu

NSF Cyber Summit 2019

This material is based upon work supported by the National Science Foundation under Grant No. 1738962. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.
SciTokens Project

• The SciTokens project aims to:

  • Introduce a **capabilities-based authorization infrastructure** for distributed scientific computing
  
  • Provide a **reference platform**, combining CILogon, HTCondor, CVMFS, and XRootD

  • **Implement specific use cases** to help our science stakeholders (LIGO and LSST) better achieve their scientific aims
SciTokens Uses Standards

- SciTokens model utilizes OAuth2 workflows to issue the tokens.
- Uses \texttt{-formatted access tokens} (a growing trend).
- The use of common protocols and workflows that we have a large number of battle-tested libraries we can leverage.

- RFC 6749: OAuth 2.0 Authorization Framework
  - token request, consent, refresh
- RFC 7519: JSON Web Token (JWT)
  - self-describing tokens, distributed validation
- RFC 8414: OAuth 2.0 Authorization Server Metadata
  - token signing keys, policies, endpoint URLs
- OAuth 2.0 Token Exchange (IETF OAuth WG I-D)
  - token delegation, drop privileges
The decoded token contains multiple scopes - basically filesystem authorizations.

The audience narrows who the token is intended for.

The issuer identifies who created the token; value used to locate the public keys needed to validate signature.

The subject is an opaque identifier for the resource owner. In this case, it also happens to be the identity.

The expiration is a Unix timestamp when the token expires. A typical lifetime is 10 minutes.
Capabilities versus Impersonation

• If GSI took over the world, an attacker could use a stolen grid proxy to make withdrawals from your bank account.

• With capabilities, a stolen token only gets you access to a specific authorization.

• SciTokens is following the principle of least privilege distributed scientific computing.
A common grid computing scenario

Scientist submits a compute job:

• This compute job is scheduled and ultimately starts running on some server out in the grid, or HPC center.
• The job requests to read and/or write data from some remote data storage service.

How should the storage service validate the job's request to access the data?
Identity & Impersonation-based Authorization Infrastructure w/ Certs

• Common grid solution used today: identity and impersonation via X.509 certificates.
  • Each user is assigned a grid certificate providing you with a globally-recognized identification.
  • The grid proxy, shipped with the job, allows a third party to impersonate you, (ideally) on your behalf.
  • The remote service maps your identity to some set of locally defined authorizations.

• Not ideal for a few reasons: Not *least privilege* (what if identity is stolen?), global identity complicates life...
Submit Server ➔ Compute Node ➔ Data Server

ID: I'mAlex!

JOB

ID: I'mAlex!

JOB

ID: I'mAlex!
• We want to change the infrastructure to focus on capabilities!
  • The tokens passed to the remote service describe what authorizations the bearer has.
  • For traceability purposes, there may be an identifier that allows tracing of the token bearer back to an identity.
  • Identifier != identity. It may be privacy-preserving, requiring the issuer (VO) to provide help in mapping.
• "The bearer of this piece of paper is entitled to read from /data/awithers".
SciTokens Model

- Integrating an OAuth2 client on the HTCondor submit host
- Enhancing HTCondor to manage token refresh and delivery to
- Enhancing CILogon to support OAuth2 with VO-defined scopes
- Enhancing services (e.g. CVMFS, Apache/NGINX, Xrootd) to allow read/writes using tokens instead of grid proxies
The world uses capabilities!

- The rest of the world uses capabilities for distributed services implemented through OAuth2
  - The authorization service creates a token that describes a certain capability or authorization.
  - Any bearer of that token may present it to a resource service and utilize the authorization.
- When you click “allow access” on the right, the **client** at “OAuth2 Test” will receive a token. This token will permit it to access the listed subset of Google services for your account.
- OAuth2 is used by Microsoft, Facebook, Google, Dropbox, Box, Twitter, Amazon, GitHub, Salesforce (and more) to allow distributed access to their identity services.
WLCG Common JWT Profiles

• https://doi.org/10.5281/zenodo.3460257

• Defines profiles for Group Based Authorization (wlcg.groups) and Capability Authorization (scope)

• cases:
  1) Identity Token with Groups
  2) Access Token Groups
  3) Access Token with Authorization Scopes

• SciTokens supports and helped define #3
• Accomplishments so far:
  • Python, Java, and C++ libraries
  • XRootD token validation plugins
  • Token-based CVMFS access
  • Token-based NGINX and Apache plugin/module for https get/put
  • X509-to-SciToken translation service
  • 3rd-party HTTPS FTS transfers authorized with SciTokens
  • Token authentication method in HTCondor
  • HTCondor support for Box and OneDrive tokens
  • Prototype oauth-ssh accepting SciTokens

https://github.com/scitokens/
In Summary...

• The SciTokens aims to:
  • Introduce a capabilities-based authorization infrastructure for distributed scientific computing
  • provide a reference platform, combining a token library with CILogon, HTCondor, CVMFS, NGINX, and XRootD
  • Deploy this technology to help our science stakeholders better achieve their scientific aims

Note: SciTokens does not do everything... e.g. SciTokens does not manage your identity (still need an identity management solution), nor does SciTokens provide an authorization service. But it will enable taking existing solutions and scale them out of distributed grid infrastructure.
• SciTokens Credmon is installed OSG submit hosts

• It auto-creates tokens for every user that submits jobs and the token to the execution host

• The can be used to write output back to storage
• Example script
  • Loads the stashcp tool, which knows how to use a scitoken
  • Shows the contents of a scitoken

```
#!/bin/sh -x

# Load the stashcache module for the stashcp tool.
module load stashcache

# Show the structure of the credential directory
ls .condor_creds/
cat .condor_creds/*

# Copy back the unique .job.ad back to the storage using the scitoken
stashcp -d .job.ad stash:///user/dweitzel/jobad
```
SciTokens Demo

• Output

```bash
$ ls .condor_creds/
total 1
-rw------- 1 osg gridusers 458 Oct  9 20:02 scitokens.use
$ cat .condor_creds/*
{"access_token":
  "eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOiJiNTQyZGVkMi1jNDEzLTQ4ZDYtOWU3Mi0yOWVmYmEyYmU3M2YiLCJzdWIiOCIsImF1dF9leGFtcGxlIjoiaHR0cHM6Ly9ydW50ZXJuYWxheXRoZS5jaHJnLmNvbS90eXBlL1N0cml0ZSIsImNyb3QiOjEifQ", "expires_in":1200}
$ stashcp -d .job.ad stash:///user/dweitzel/jobad
```
$ ls .condor_creds/
  total 1
  -rw------- 1 osg gridusers 458 Oct  9 20:02 scitokens.use
$ cat .condor_creds/*
{"access_token":
  "eyJhbGciOiJFU ...
XNZLFaMjW3Kk1i9KVNI ...",
  "expires_in": 1200}
$ stashcp -d .job.ad stash://user/dweitzel/jobad
--upload-file .job.ad -H "Authorization: Bearer eyJhbGciOiJFU ...
XNZLFaMjW3Kk1i9KVNI ...
 工商,exp: 1200}

Uses regular curl command with an additional Authorization header
### HEADER: ALGORITHM & TOKEN TYPE

```json
{
  "alg": "ES256",
  "typ": "JWT",
  "kid": "6804"
}
```

### PAYLOAD: DATA

```json
{
  "jti": "b542ded2-c413-48d6-9e72-29efba2be73f",
  "sub": "dweitzel",
  "exp": 1578652483,
  "iat": 1578651283,
  "iss": "https://scitokens.org/osg-connect",
  "scope": "read:/user/dweitzel write:/user/dweitzel",
  "nbf": 1578651283
}
```

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhZG1pbiJ9
```
Visit
https://scitokens.org/
for more info.