CENTER FOR HIGH THROUGHPUT COMPUTING

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

(Sci)Tokens Authentication in HTCondor-CE

EU HTCondor Workshop 2023

Jaime Frey

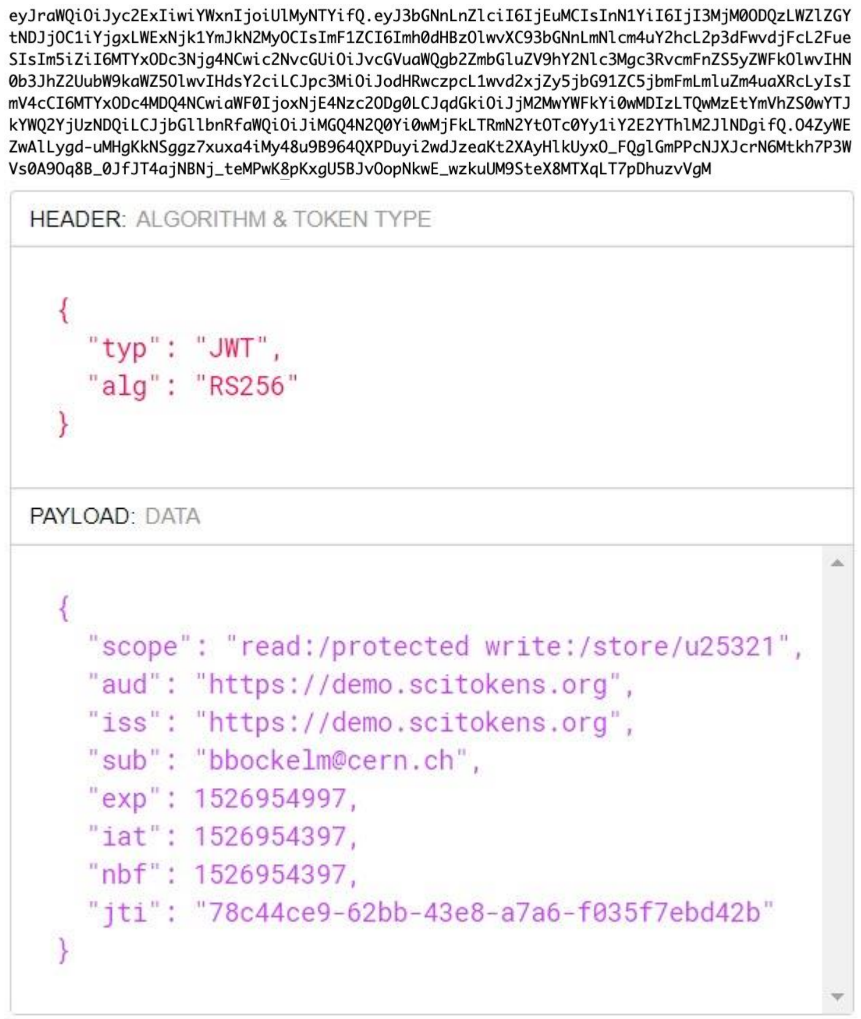
HTCondor Software Suite

Why Move to Tokens?

- HTC has used X.509 credentials for user authentication
  - Proposed by grid community with extensions
- Tokens are more widely supported
  - Proxies never embraced by industry
- Tokens allow a better security model
  - Authorization via capability instead of identity

JSON Web Tokens (JWT)

- A set of key-value pairs…
- Signed by an issuer
- Some keys are standardized
  - iss: Token Issuer
  - exp: Expiration time
  - scope: List of authorizations
  - aud: Service token can be used at



```
eyJraWQiOiJyc2ExIiwiYWxnIjoiUlMyNTYifQ.eyJ3bGNnLnZlciI6IjEuMCIsInN1YiI6IjI3MjM0ODQzLWZlZGY
tNDJjOC1iYjgxLWExNjk1YmJkN2MyOCIsImF1ZCI6Imh0dHBzOlwvXC93bGNnLmNlcm4uY2hcL2p3dFwvdjFcL2Fue
SIsIm5iZiI6MTYxODc3Njg4NCwic2NvcGUiOiJvcGVuaWQgb2ZmbGluZV9hY2Nlc3Mgc3RvcmFnZS5yZWFkOlwvIHN
0b3JhZ2UubW9kaWZ5OlwvIHdsY2ciLCJpc3MiOiJodHRwczpcL1wvd2xjZy5jbG91ZC5jbmFmLmluZm4uaXRcLyIsI
mV4cCI6MTYxODc4MDQ4NCwiaWF0IjoxNjE4Nzc2ODg0LCJqdGkiOiJjM2MwYWFkYi0wMDIzLTQwMzEtYmVhZS0wYTJ
kYWQ2YjUzNDQiLCJjbGllbnRfaWQiOiJiMGQ4N2Q0Yi0wMjFkLTRmN2YtOTc0Yy1iY2E2YThlM2JlNDgifQ.O4ZyWE
ZwAlLygd-uMHgKkNSggz7xuxa4iMy48u9B964QXPDuyi2wdJzeaKt2XAyHlkUyxO_FQglGmPPcNJXJcrN6Mtkh7P3W
Vs0A9Oq8B_0JfJT4ajNBNj_teMPwK8pKxgU5BJvOopNkwE_wzkuUM9SteX8MTXqLT7pDhuzvVgM
```

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "typ": "JWT",
  "alg": "RS256"
}
```

PAYLOAD: DATA

```
{
  "scope": "read:/protected write:/store/u25321",
  "aud": "https://demo.scitokens.org",
  "iss": "https://demo.scitokens.org",
  "sub": "bbockelm@cern.ch",
  "exp": 1526954997,
  "iat": 1526954397,
  "nbf": 1526954397,
  "jti": "78c44ce9-62bb-43e8-a7a6-f035f7ebd42b"
}
```

OAuth2 and OpenID Connect (OIDC)

- Standard frameworks for
  - User to authenticate with issuer and obtain a token
  - Service to validate a token presented by a user
- Widely used in industry
  - E.g. Google, Amazon, Facebook, Microsoft
- Lots of software and language support

SciTokens vs WLCG Tokens

- Both are based on OpenID Connect
  - WLCG tokens follow standard more strictly
- Both define file- and job-based authorizations
  - Additional authorization types are possible
- Format of scope names differs
  - SciTokens: read:/foo CONDOR:/READ
  - WLCG Tokens: storage.read:/foo compute.read
- HTCondor accepts both for job control

Token Discovery

- HTCondor tools look here for a token to use for authorization
  - `$BEARER_TOKEN`: value has token data
  - `$BEARER_TOKEN_FILE`: file has token data
  - `$XDG_RUNTIME_DIR/bt_u<id>`: file has token data
  - `/tmp/bt_u<id>`: file has token data
- First location with a valid token is used

Submitting to an HTCondor-CE

- E.g., the factory use case
  - Using local AP to submit jobs to a remote AP
- Token must be in a file
- Add to your submit file
  - `scitokens_file = <filename>`
- Or if BEARER_TOKEN_FILE in environment, add this
  - `use_scitokens = true`

Token Info in the Job Ad

- Some token claims placed in the job ad
  - `AuthTokenId = "ddb63eca-0aff-4f6b-8bb4-89dc15ec33f7"`
  - `AuthTokenIssuer = "https://demo.scitokens.org"`
  - `AuthTokenScopes = "condor:/READ,condor:/WRITE"`
  - `AuthTokenSubject = "jfrey"`
- Can’t get altered by the user
- Can be used for CE routing

Authorizing SciTokens

- Specify in /etc/condor-ce/mapfiles.d/*
  - Which tokens/issuers to accept
  - Which identity (i.e. user account) to map them to
  - `SCITOKENS <issuer>,<subject> <username>`
- Accept specific token issuer and subject (exact match)
  - `SCITOKENS https://demo.scitokens.org,jfrey jfrey`
- Accept all tokens from an issuer (regular expression)
  - `SCITOKENS /https:\/\/demo.scitokens.org,.*/ jfrey`
- Set audience name of daemons
  - `SCITOKENS_SERVER_AUDIENCE = $(COLLECTOR_HOST)`

SciToken Authorization Plugins

- When issuer and subject aren’t enough
- Delegate authorization/mapping decisions to external program(s)
- SciTokens library still does token validation
- Interface compatible with ARC CE

Plugin Example

› Map file

-
  ```
  # Plugin for specific token issuer
  SCITOKENS /^https:\/\/phys.uz.edu,.*/ PLUGIN:A

  # Plugins for all other token issuers
  SCITOKENS /.*/ PLUGIN:B,C
  ```

› Configuration file

-
  ```
  # Plugin A for specific issuer with fixed mapping result
  SEC_SCITOKENS_PLUGIN_A_COMMAND = $(LIBEXEC)/A.plugin
  SEC_SCITOKENS_PLUGIN_A_MAPPING = physgrp

  # Plugins B,C for all other tokens
  SEC_SCITOKENS_PLUGIN_B_COMMAND = $(LIBEXEC)/B.plugin
  SEC_SCITOKENS_PLUGIN_C_COMMAND = $(LIBEXEC)/C.plugin -A
  ```

Plugin Interface

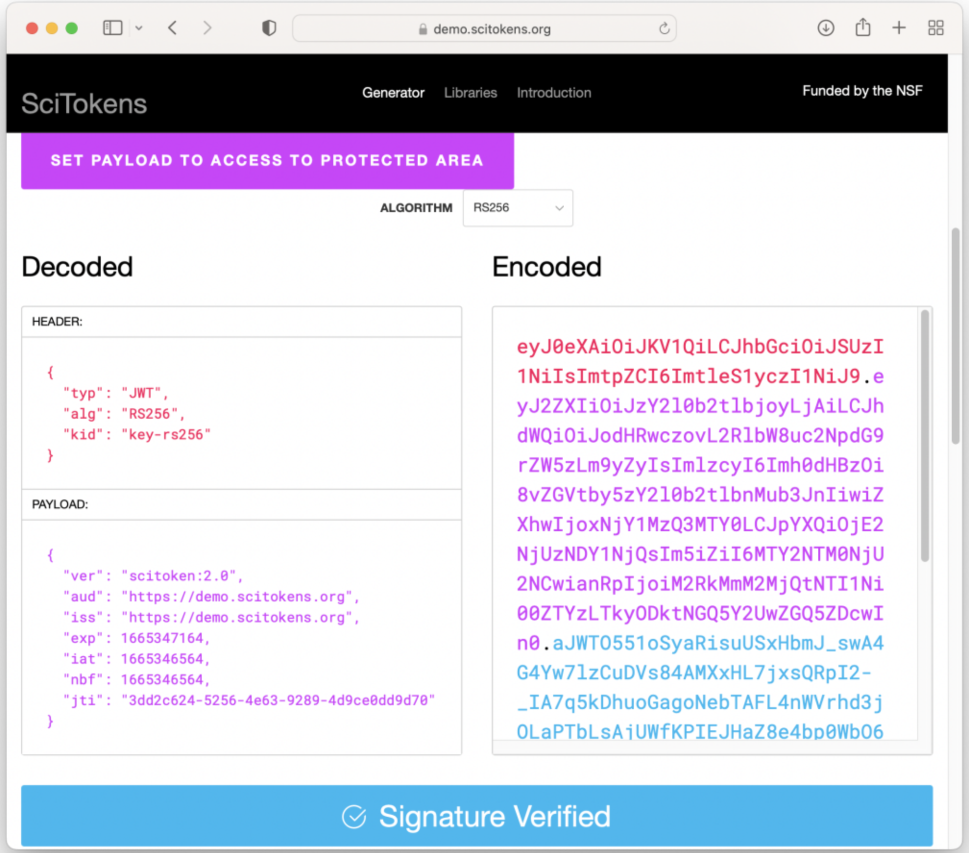
- Token body provided via stdin
- Claims provided in environment variables
  - Same interface as ARC CE
- Exit code
  - 0: accept token
  - 1: decline token, check other plugins
  - default: reject token
- Stdout (needed if `_MAPPING` config not set)
  - Mapping result (HTCondor identifier)

EGI CheckIn Tokens

- Now supported out-of-the-box
  - Tokens need `aud` and `scope` claims
- Plugin available
  - https://github.com/EGI-Federation/check-in-validator-plugin

Simple Testing with SciTokens

- https://demo.scitokens.org
- Will issue any token you want
- Don't use in production!
- Add these keys
  - "aud: "ANY"
  - (Or set to match CE's SCITOKENS_SERVER_AUDIENCE)
  - "sub": "jfrey"
  - "scope": "condor:/READ condor:/WRITE"
- Configure CE to accept this issuer



ALGORITHM RS256

Decoded

HEADER:

```
{
  "typ": "JWT",
  "alg": "RS256",
  "kid": "key-rs256"
}
```

PAYLOAD:

```
{
  "ver": "scitoken:2.0",
  "aud": "https://demo.scitokens.org",
  "iss": "https://demo.scitokens.org",
  "exp": 1665347164,
  "iat": 1665346564,
  "nbf": 1665346564,
  "jti": "3dd2c624-5256-4e63-9289-4d9ce0dd9d70"
}
```

Encoded

eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI
1NiIsImtpZCI6ImtleS1yczI1NiJ9.e
yJ2ZXIiOiJzY2l0b2tlbjoyLjAiLCJh
dWQiOiJodHRwczovL2RlbW8uc2NpdG9
rZW5zLm9yZyIsImlzcyI6Imh0dHBzOi
8vZGVtby5zY2l0b2tlbnMub3JnIiwiZ
XhwIjoxNjY1MzQ3MTY0LCJpYXQiOjE2
NjUzNDY1NjQsIm5iZiI6MTY2NTM0NjU
2NCwianRpIjoiM2RkMmM2MjQtNTI1Ni
00ZTYzLTkyODktNGQ5Y2UwZGQ5ZDcwI
n0.aJWTO551oSyaRisuUSxHbmJ_swA4
G4Yw7lzCuDVs84AMXxHL7jxsQRpI2-
_IA7q5kDhuoGagoNebTAFL4nWVrhd3j
OLaPTbLsAjUWfKPIEJHaZ8e4bp0WbO6

Signature Verified

Advanced Testing

- Impersonate a known issuer
  - `condor_test_token`
    ```
    --issuer https://scitokens.org/osg-connect
    --subject jfrey
    --lifetime 3600
    --scope ‘condor:/READ condor:/WRITE’
    --audience ‘ce.foo.edu’
    ```
- Create fake token that CE will accept for a short period as if it came from the given issuer
- Good for testing authorization configuration

Quitting GSI (and X.509 proxies)

- HTCondor 9.0
  - GSI, WLCG/SciToken authentication
- HTCondor 10.0
  - WLCG/SciToken authentication (no GSI)
- HTCondor 10.X
  - EGI/WLCG/SciToken authentication
  - Token authorization plugins
- (X.509 Proxy for user jobs supported in all versions)

Quitting GSI (and X.509 proxies)

- HTCondor 9.0
  - GSI, WLCG/SciToken, SSL authentication
- HTCondor 10.0
  - WLCG/SciToken, SSL authentication (no GSI)
- HTCondor 10.X
  - EGI/WLCG/SciToken, SSL authentication
  - Token authorization plugins
- (X.509 Proxy for user jobs supported in all versions)

Plain SSL Auth with Grid Proxies

- Client config changes
  - `AUTH_SSL_USE_CLIENT_PROXY_ENV_VAR = True`
  - `AUTH_SSL_CLIENT_CADIR = /etc/grid-security/certificates`
- Client tools must have `X509_USER_PROXY` set in environment

Plain SSL Auth with Grid Proxies

- Server config changes
  - `AUTH_SSL_ALLOW_CLIENT_PROXY = True`
  - Uncomment in /etc/condor-ce/config.d/01-ce-auth.conf:
  - `AUTH_SSL_SERVER_CERTFILE = /etc/grid-security/hostcert.pem`
  - `AUTH_SSL_SERVER_KEYFILE = /etc/grid-security/hostkey.pem`
  - `AUTH_SSL_SERVER_CADIR = /etc/grid-security/certificates`
  - `AUTH_SSL_CLIENT_CERTFILE = /etc/grid-security/hostcert.pem`
  - `AUTH_SSL_CLIENT_KEYFILE = /etc/grid-security/hostkey.pem`
  - `AUTH_SSL_CLIENT_CADIR = /etc/grid-security/certificates`

Plain SSL Auth with Grid Proxies

› Add mappings to CE mapfiles

-
  ```
  # Exact match
  SSL  "/O=condor/OU=CHTC Pool/CN=James Frey"  jfrey
  # regex match
  SSL  /\/O=condor\/OU=CHTC\ Pool\/CN=.*/  jfrey
  ```

› Detailed explanation here:

- https://htcondor-wiki.cs.wisc.edu/index.cgi/wiki?p=HowToUseProxiesWithSsl

SciTokens vs IDTokens

- Both are JWTs and look similar
- SciTokens use asymmetric key
  - Anyone with issuer’s public key can verify
  - Issuer must publish public key via https server
  - Suited for a VO accessing multiple services (including HTCondor pools)
- IDTokens use symmetric key
  - Issuer’s private key required to verify
  - Suited for a single HTCondor pool
  - Doesn’t use OAuth or OIDC

Ongoing and Future Work

- Improving SciTokens C++ library
  - Add non-blocking interface
  - Fetching issuer’s public key delayed by DNS problems
  - Update CE to use new interface
- Improve condor_test_token
  - Impersonate WLCG or EGI tokens

Thank You!

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

This work is supported by NSF under Cooperative Agreement OAC-2030508 as part of the PATh Project. 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 NSF.