JWT Mapping to Unix User

CNAF Multilateral

Carmelo Pellegrino

Outline

- Why? - The problem
- Proposed solution
- Possible applications

Why?

Because... the initial use-case

- HTCondor support for JWTs:
  - https://htcondor.com/htcondor-ce/v5/configuration/authentication/#scitokens

SciTokens

To allow clients with SciToken or WLCG tokens to submit jobs to your HTCondor-CE, add lines of the following format:

```
SCITOKENS /<TOKEN ISSUER>,<TOKEN SUBJECT>/ <USERNAME>
```

Replacing `<TOKEN ISSUER>` (escaping any `/` with `\/`, `<TOKEN SUBJECT>`, and `<USERNAME>` with the token issuer (`iss`), token subject (`sub`), and the unix account under which the job should run, respectively. For example, to map any token from the `OSG` VO regardless of the token `sub`, add the following line to a `*.conf` file in `/etc/condor-ce/mapfiles.d/`:

```
SCITOKENS /^https:\/\/scitokens.org\/osg-connect,.*/ osg
```

Why?!?!?

why is it done in this way?

- Big collaborations do make use of a Workload Management System (WMS, like DIRAC, PANDA, AliEn, etc...)
  - Few (iss, sub) pairs to map per big VO
- SCITOKENS => OSG use case in mind
  - Each OSG-supported VO has one dedicated token issuer
  - VO <=> issuer

```
SCITOKENS /^https:\/\/scitokens.org\/osg-connect,.*/ osg
```

Why?

pitfalls

1. Difficult to put in production:
   - has to be filled by hand
   - static file
2. Medium/Small collaborations not using a WMS => tons of hand-made mapping in the future?
3. Mapping entire OSG collaborations with one single Unix user
4. Complete lack of VO/group handling in HTCondor. In IAM:
   - VO => (iss, sub, **[wlcg.]groups**). *Given the VO, a token can be created*
   - not "<=>". *Given a token, cannot automatically associate to a VO (hence uid)*
   - What if a token has multiple valid groups in its claims?

Proposed solution

Token to Unix User - t2u2

- Written in C++14
- HTTP-based
- Easy to configure
- Very small code base
- Very small resource consumption and fast
- https://baltig.infn.it/budda/t2u2

- Few dependencies:
  - openssl 1.1.1k
  - Crow (C++ HTTP framework)
  - boost 1.69.0 (dep of Crow)
  - libcurl
  - yaml-cpp (configuration file)
  - jwt-cpp

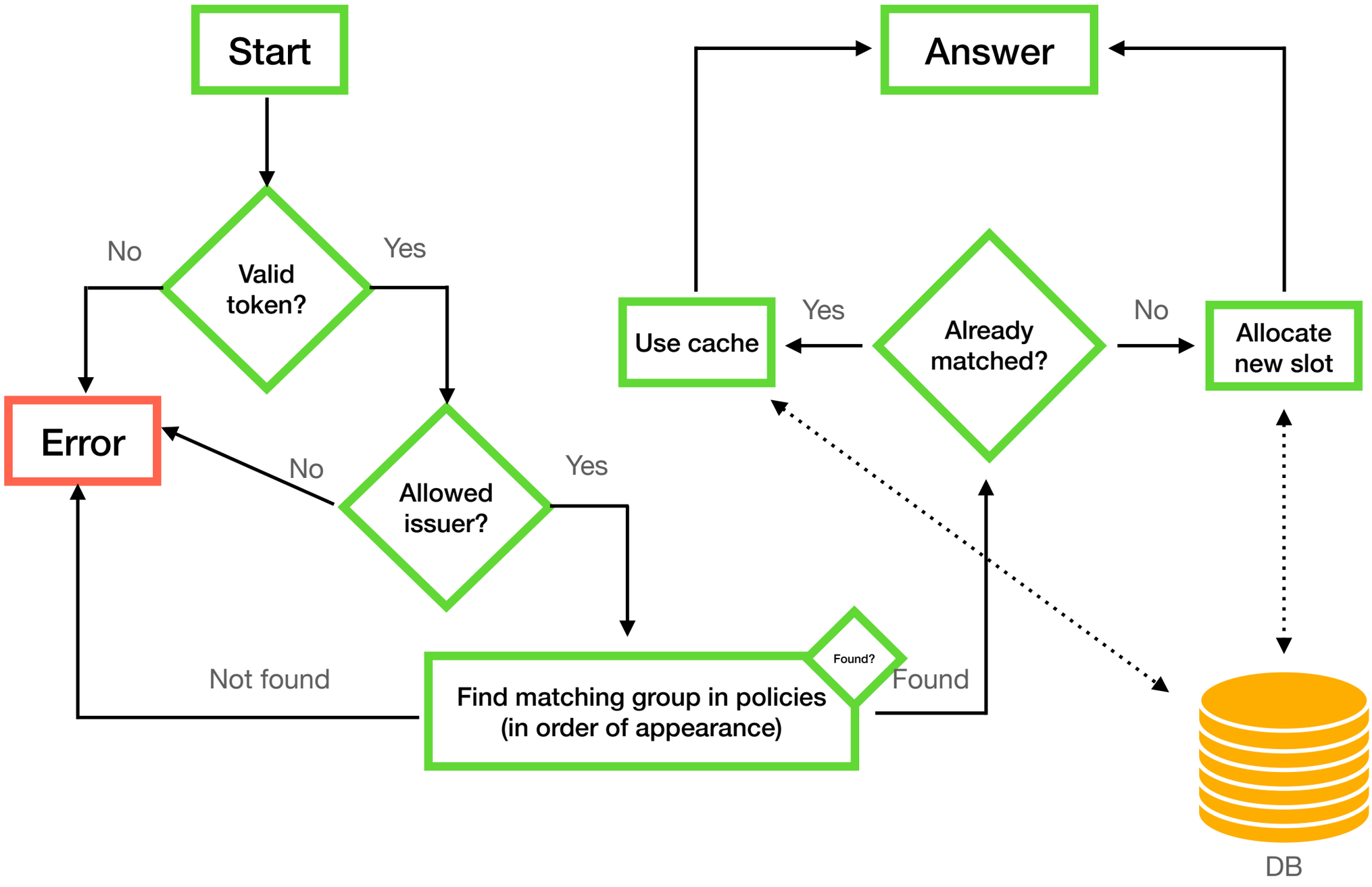
Proposed solution

Policy definition

```yaml
policies:
  allow_untrusted_issuer: false                    # default: false
  trusted_issuers:                                 # trusted IAMs
    - https://iam-t1-computing.cloud.cnaf.infn.it
    - https://wlcg.cloud.cnaf.infn.it
  groups:                                          # dictionary of IAM groups
    wlcg:
      reuse_users: true                            # default: false
      users:                                       # local Unix users
        - wlcg001
        - wlcg002
  dteam:
    users:
      pattern: "dteam%03d"                         # pattern like in `man 3 printf`
      range: [1, 100]
  km3net:
    users:
      pattern: "km3net%03d"
      range: [1, 50]
```

Proposed solution

Request workflow



Proposed solution

Mapping algorithm

```
preferred_group := http::headers::preferred_group

if preferred_group {
  if preferred_group in (jwt::claims::groups or jwt::claims::wlcg.groups) {
    group := preferred_group
  } else {
    error();
  }
} else {
  if not empty(jwt::claims::groups) {
    group := jwt::claims::groups[0]
  } else if not empty(jwt::claims::wlcg.groups) {
    group := jwt::claims::wlcg.groups[0]
  } else {
    error();
  }
}

user := group in policies
```

Examples of usage

- Run the executable:

```
$  ./t2u2 [--configfile <file.yml=/etc/t2u2/config.yml>]
```

- Query the server:

```
$  curl -H "Authorization: Bearer $TOKEN" https://t2u2.example.com/map
   myuserntof
$  curl -H "Authorization: Bearer $TOKEN" -H 'X-Preferred-Group: litebird' https://t2u2.example.com/map
   myuserlitebird
```

Possible weak points

Token to Unix User - t2u2

- Home-made solution
  - Needs maintenance
- DB is currently a local text file + in-memory copy
  - No distributed DB => scalability problem?
  - ~4.3 connections/s at each CNAF CE (6 in total)
  - average response delay ~O(ms)

Possible applications

- HTCondor-CE mapping
  - needs support in HTCondor for a callout (as per GSI Auth with ARGUS)
  - HTCondor devs would like to receive a pull request to work on
- StoRM mapping (?)
  - Discussion

Backup

Configuration file

```yaml
log:
  level: debug
ssl:
  disable: false                # default
  cert: /etc/t2u2/cert.pem
  key: /etc/t2u2/key.pem

db: cache.db
address: 127.0.0.1              # address to bind
port: 9090                      # port to bind
policies:
allow_untrusted_issuer: false # default
trusted_issuers:
  - https://iam-t1-computing.cloud.cnaf.infn.it
  - localhost:8080
groups:                         # IAM groups
  wlcg:
    reuse_users: false          # default
    users:
      - wlcg001                 # Unix users
      - wlcg002
  dteam:
    users:
      pattern: "dteam%03d"
      range: [1, 100]
  km3net:
    users:
      pattern: "km3net%03d"
      range: [1, 3]
```