Update on Token Operations using Vault

Dave Dykstra, Fermilab
October 18, 2022
Workshop on Token-Based Authorization & Authentication
Background

- At 2 previous WoTBAn&Az, FNAL discussed htgettoken & Vault
  - Vault stores Oauth refresh tokens, and issues its own tokens to access them
  - Vault takes the place of MyProxy in our job submission architecture, and is also the Oauth client
  - htgettoken is the Vault client
  - htvault-config is the Vault configurator
- We use CILogon as the token issuer
- We use htgettoken for both submitting jobs to HTCondor and for direct file transfers
  - We want to share the token, so the same authentication can be used for both
- The use of htgettoken & Vault is integrated into HTCondor
What’s new with htgettoken/Vault usage

- No VO is yet in full production, but advances in preparation have been made
  - The Fermilab authentication services group has set up HA vault services, one of which is expected to be used for production
    - 3 VMs in each service
    - Using a configuration generator (more on that next)
  - Fermilab has set up a “managed token service” to make things easier for automation by experiments (more on that too)
  - LIGO is closest to production use
    - Now working on including support for “Robot” scripts
  - JLab has this year also set up an infrastructure using an HA vault service, htgettoken, and CILogon as the token issuer
Vault configuration generator

- Fermilab has a custom database and interface to it (FERRY) that describes all our experiments, the members, and roles people are authorized for, and the token scopes each role is authorized to receive
- `htvault-config` works with yaml files describing the “issuers” and roles
- `htvault-gen` reads information from FERRY frequently (twice an hour) and creates most of the yaml files `htvault-config` uses
  - Not a general purpose tool, but the concept could be useful for others
  - Reads from its own smaller yaml files describing information not in FERRY such as token issuer URL, client ids, secrets, and also the URLs of FERRY instances.
  - Avoids having to ask the operators to do frequent manual configuration changes
  - Only a ~160 line bash script (uses yaml to bash converter that’s in `htvault-config`)

**Managed token service**

- Fermilab currently has a “Managed proxy service” that automatically renews getting X.509 proxies for multiple automated processes in multiple experiments, including managing the longer-lived proxies in MyProxy for job submission.
- We have now also developed the corresponding “Managed token service”
  - The operators are given permission by CILogon (via FERRY writing to LDAP) to request the refresh token for each managed account.
  - They run `htgettoken` via the HTCondor vault-integration script `condor_vault_storer` so the corresponding long-lived Vault token gets stored in HTCondor’s `credd`.
    - They log in on their web browser and approve the operation.
  - They also obtain a Kerberos keytab corresponding to the storage location in Vault for each experiment’s shared accounts.
  - They are given permission to write to a location in the shared accounts via `ssh` on each of the job submission machines.
  - Then an automated script sends updated Vault tokens to those shared accounts, which are used to get access tokens for submitting jobs, transferring files, etc.
- Overall security design: limit hosts holding unlimited life credentials.
Token flow with Managed Token Service

- **V** = vault tokens (7 or 28 day)
- **R** = refresh tokens (effectively unlimited)
- **A** = access tokens (3 hours)
- **K** = kerberos keytab (unlimited)

**Managed Token Server**
- condor_vault_storer
- htgettoken

**Token Issuer**

**Identity Provider**
- condor_submit
- condor_vault_storer
- htgettoken

**Job Submission node**
- condor_submit
- condor_vault_storer
- htgettoken

**Worker node**
- condor_schedd

**Operator**
- condor_credd
- condor_credmon_vault

**HTCondor server**
- condor_schedd

**User/script**

**Policy DB**
Links

● Vault: https://www.vaultproject.io/
● htvault-config: https://github.com/fermitools/htvault-config
● htgettoken: https://github.com/fermitools/htgettoken