CERN Kerberos and HTCondor Integration Proposal

Overview

This document describes the objectives and proposed plan for interfacing HTCondor with CERN's NextGenerationBatchAuth service, referred to as "Credential Service" throughout the rest of this document.

Users of CERN's system use Kerberos for authentication. When submitting jobs to a batch system such as HTCondor, it is important for the executing jobs to be able to authenticate to various services (e.g. AFS) during the job's execution.

The high-level goal is for users to submit jobs to HTCondor by authenticating to HTCondor with their Kerberos credentials and have those credentials be available to the job when the job executes on a remote resource. It is important to note that jobs may exist in the queue in an idle state for quite some time ("days") before they are scheduled for execution, whereas the typical Kerberos credential is only valid for a much shorter time period ("hours") by default. As such, the challenge here is preserving a chain of custody for some credential over the entire lifetime of the job from submission to completion such that when a job is executing it has the ability to authenticate to various other services.

Touchpoints

This section describes the exact callouts made by HTCondor to any external services.

The submit side:

1) condor_submit must obtain the uberticket. It will do this by executing an external program specified in the condor_config file as SEC_CREDENTIAL_PRODUCER. This program takes no arguments, and writes its output to stdout. condor_submit will capture this output and use it as the uberticket. The program must exit with status zero on success and non-zero status on failure. condor_submit will send the uberticket to the condor_credd daemon, and will block for a configurable amount of time until the condor_credd signals that everything is ready.
2) The condor_credd runs on the same machine as the condor_schedd. The condor_master on that machine will launch the Credential Monitor as root to maintain the user's credentials on the submit side. There will be one Credential Monitor per machine that is shared by all users. The Credential Monitor takes a directory as input and monitors all credentials in that directory. The condor_master will find the program specified in the condor_config as SEC_CREDENTIAL_MONITOR and launch it as root. The one

command line flag to that program is "<directory_to_monitor>". If the Credential Monitor exits for any reason, it will be restarted by the condor_master after a short delay. The exit status of the Credenital Monitor is logged but is otherwise ignored. The Credential Monitor must handle a SIGHUP signal which informs it that the contents of the directory it is monitoring have changed and it should rescan the directory and perform whatever actions are necessary.

3) HTCondor will determine the directory in which to store ubertickets using the directory specified in the condor_config as SEC_CREDENTIAL_DIRECTORY. The files in this directory will be owned by the user 'root' and have permissions 0600 or 0400. All files written into this directory must be written atomically. Files with the extension .tmp should be created first and then rename(2)ed into place.
4) The condor_credd will atomically place credentials into that directory when the user has jobs in the queue that need to run, and will remove credentials from that directory when a given user has no more jobs. The ubertickets will be named "<username>.cred". The Credential Monitor will notice the new uberticket, either periodically or upon receiving SIGHUP, and obtain a TGT and atomically place it in a krb5 credential cache in the credential directory under the filename "<username>.cc". HTCondor will know it has a valid TGT and AFS token for the user when the file "<user>.cc" is present in that directory. If the file "<username>.cc" is not present, HTCondor will assume that user does not have valid credentials and it should NOT try to perform any actions on that user's behalf. The Credential Monitor does not need to do anything when an uberticket is removed from the credential directory.

The execute side:

5) The condor_master on the execute machine will launch the Credential Monitor as root to maintain the user's credentials on the execute side. There will be one Credential Monitor per machine shared by all users. The Credential Monitor takes a directory as input and monitors all credentials in that directory. The condor_master will find the program specified in the condor_config as SEC_CREDENTIAL_MONITOR and launch it as root. The one command line flag to that program is "<directory_to_monitor>". If the Credential Monitor exits for any reason, it will be restarted by the condor_master after a short delay. The exit status of the Credenital Monitor is logged but is otherwise ignored. The Credential Monitor must handle a SIGHUP signal which informs it that the contents of the directory it is monitoring have changed and it should rescan the directory and perform whatever actions are necessary.
6) HTCondor will determine the directory in which to store ubertickets using the directory specified in the condor_config as SEC_CREDENTIAL_DIRECTORY. The files in this directory will be owned by the user 'root' and have permissions 0600 or 0400. All files written into this

directory must be written atomically. Files with the extension .tmp should be created first and then rename(2)ed into place.

7) The condor_starter will atomically place credentials into that directory when the user has jobs scheduled to run on that execute machine, and will remove credentials from that directory when a given user has no more jobs for that execute machine. The uberticket will be named "<username>.cred". The Credential Monitor will notice the uberticket, either periodically or upon receiving SIGHUP, and will obtain a TGT and atomically place it in a krb5 credential cache in the credential directory under the filename "<username>.cc". HTCondor will know it has a valid TGT and AFS token for the user when the file "<user>.cc" is present in that directory. If the file "<username>.cc" is not present, HTCondor will assume that user does not have valid credentials and it should NOT try to perform any actions on that user's behalf. The Credential Monitor does not need to do anything when an uberticket is removed from the credential directory.
8) When HTCondor executes the job, it will copy the user's credential cache into the job sandbox and set the KRB5CCNAME environment variable to point to the credential cache. The condor_starter will also monitor the .cc file in the credential directory and place fresh copies into the job sandbox as needed.

Both sides:

When the Credential Monitor is to be shut down, HTCondor will send the program SIGTERM. It will be the responsibility of HTCondor to clean up any credentials. If the Credential Monitor does not exit after some amount of time (10 seconds) then HTCondor will send the program a SIGKILL.

If the Credential Monitor crashes or exits for any reason, it will be restarted by the condor_master after a small delay.