CENTER FOR
HIGH THROUGHPUT
COMPUTING

HTCondor Input File Transfer via Squid Caching Proxy

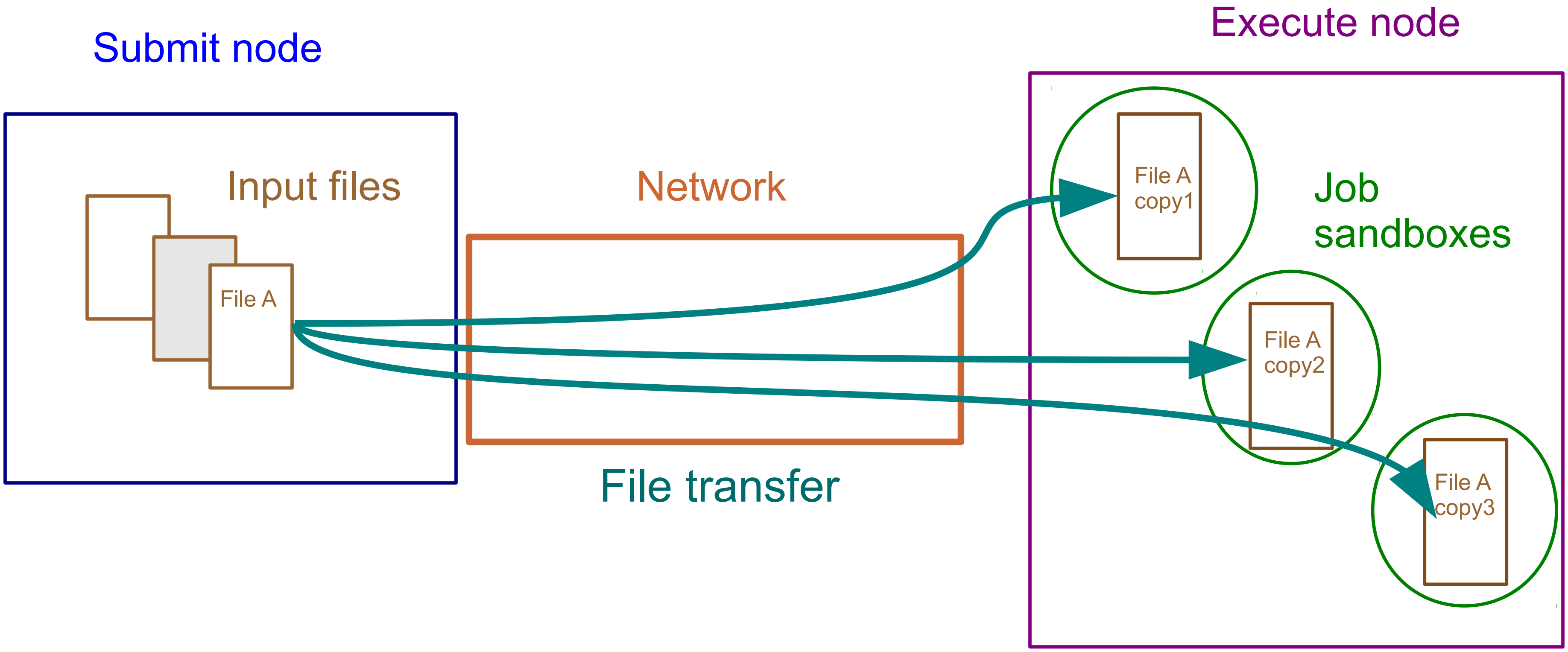
Carl Vuosalo
CHTC

Outline

- Introduction
- Basic Design
- Configuration
- Explanatory Notes
- Review

HTCondor Input File Transfer

Before HTCondor jobs start, their input files must be copied from submit node to execute node



New Input File Caching

- Input files are copied across network from submit to execute node
- If large number of jobs starts at once, network traffic from submit node may be heavy as many files are copied
- If same file is used for multiple jobs, caching becomes possible to avoid repeatedly transferring same file across network
- Such caching will be introduced in HTCondor 8.4.0

Disclaimer

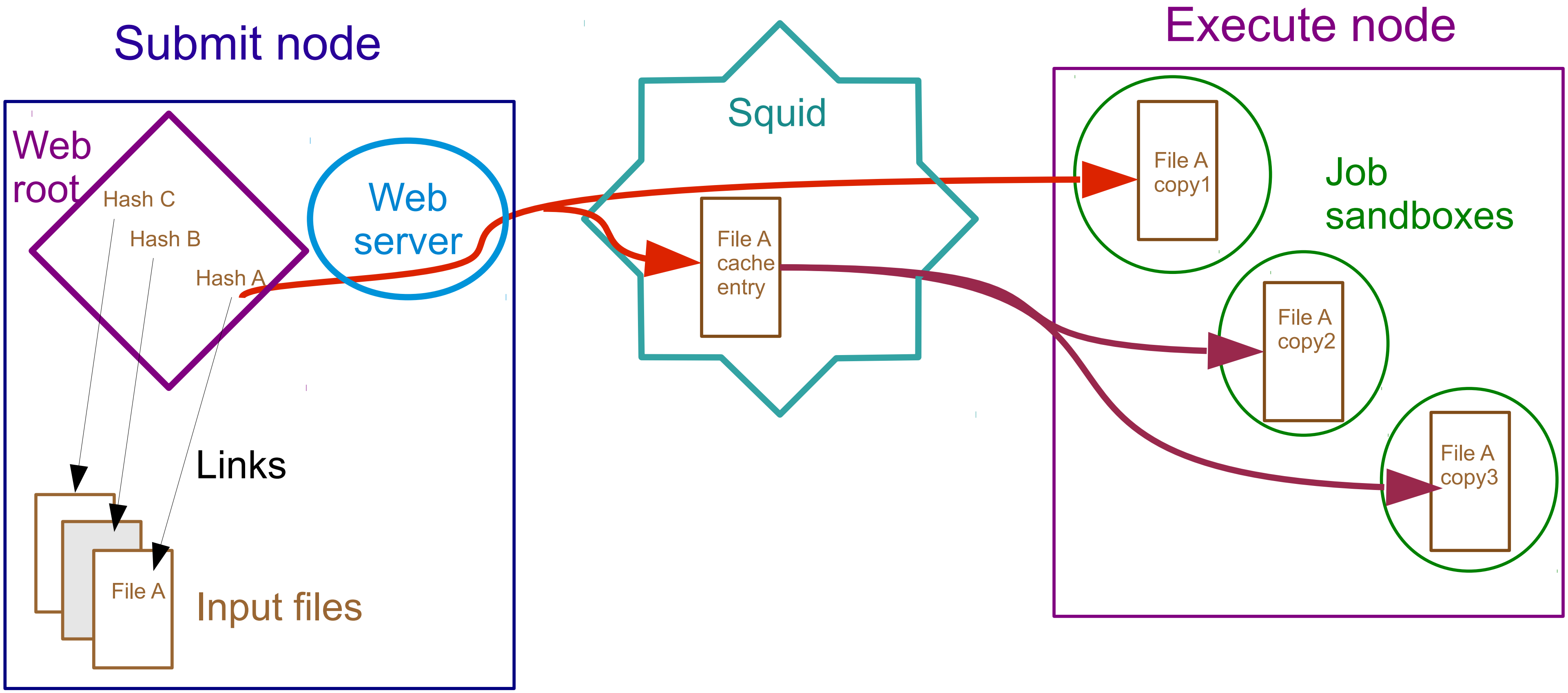
- This presentation is a preview
- These new features are still undergoing testing and review, and documentation is still being written
- Minor details may change prior to release
- Please consult HTCondor manual about these features when they are released rather than relying on this presentation

Basic Design

- Required software: HTCondor, web server (like Apache) on submit node, Squid caching proxy
- In HTCondor submit file, user lists public_input_files that can be cached
- If properly configured, HTCondor will make HTTP request to web server to transfer these files
- If Squid caching proxy is configured, it will make cache copy of these files as they are transferred
- After first copy of file is transferred across network via web server, all subsequent HTTP requests for that file will be served by Squid, thereby reducing network traffic from submit node

File Transfer with Caching

- First transfer of file creates Squid cache entry
- Later transfers of same file served by Squid cache



HTCondor Configuration

- On submit and execute nodes:
  - `ENABLE_CACHE_TRANSFERS = TRUE`

- On submit nodes:
  - `WEB_ROOT_DIR =` directory on submit node to hold links served by web server. Must be writable by HTCondor

- On execute nodes:
  - `ENABLE_URL_TRANSFERS = TRUE`
  - `WEB_SERVER_PORT` = IP address and port number in standard IP:port format of web server. With `condor_shared_port` daemon, use its address, and it will route HTTP requests to web server
  - `WEB_SERVER_PIPE` = unique name for web server's pipe used by condor_shared_port daemon
  - `STARTD` environment variable `http_proxy` must be set to HTTP address of Squid caching proxy
    - `STARTD_ENVIRONMENT = “http_proxy=http://your.proxy.server:port/”`

Web Server Configuration

- Most modern web servers can be used
- Web server requirements:
  - Fills in “Expires” headers in responses
  - Supports If-Modified-Since queries
  - Follows symbolic links to files
- Web server should be configured to handle expected load of input file requests from jobs starting up
- For Apache web server, set configuration as follows
  - `Options FollowSymLinks`
  - `ExpiresActive On`
  - `ExpiresDefault "modification plus 20 minutes"`
  - Expires time of 20 minutes recommended but another value could be used (see slide 11)

Squid Configuration

- Squid default configuration supports HTCondor input file caching
- No changes necessary
- Squid proxy should be provided with hardware and network resources to handle anticipated loads
  - Consider maximum number of jobs that may start up simultaneously
  - Consider sizes of input files for each job
  - Calculate maximum network throughput that Squid server must support

Squid Cache Expiration Time

- Expiration time directive from web server tells Squid how long before cache entries expire
- If cache entry has been written or checked within this time, Squid serves it without checking
- If entry is older than this time, Squid queries web server to send file if it has been modified since cache entry written
  - If web server sends new version, cache entry updated and sent to client
  - If web server responds that no change has occurred, existing entry is sent to client
- In any case, entry's expiration time stamp is reset, so Squid only queries web server about it at most once in that time period
- User modification of file already used as job input file, with old version in Squid cache, will not propagate to new jobs until after expiration time

Note on public_input_files

- Input files to be cached must be public – should contain no private or sensitive information
- They must be world readable or else error occurs
- Issue is that file will be copied across network and held in Squid cache for indefinite period
  - Cache copy outside of HTCondor's control
  - Encryption not supported for these files
- For protection, names of input files are replaced with opaque hash names, so outsiders unlikely to be able to access them via HTTP requests

Review of Input File Caching

- Jobs submitted with public_input_files included
- HTCondor creates (or updates) soft links to input files with hash names in `WEB_ROOT_DIR`
- Starter sends request for input files to web server at `WEB_SERVER_PORT` via Squid caching proxy at `http_proxy`
- If files in Squid cache, Squid serves them, updating cache entry if necessary
- Otherwise, web server reads files out of `WEB_ROOT_DIR`, which also creates Squid cache entries
- `condor_preen` periodically deletes links in `WEB_ROOT_DIR` that are older than one week or invalid (future feature)

Summary

- Transferring input files for HTCondor jobs can cause heavy network traffic
- If same files are used by multiple jobs, caching those files could reduce network traffic
- HTCondor 8.4.0 will support input file caching
- Requires web server on submit nodes and Squid caching proxy
- File transfer from submit to execute node occurs via web server for first copy of file
  - After that requests for same file served by Squid caching proxy