glideinWMS factory Internals

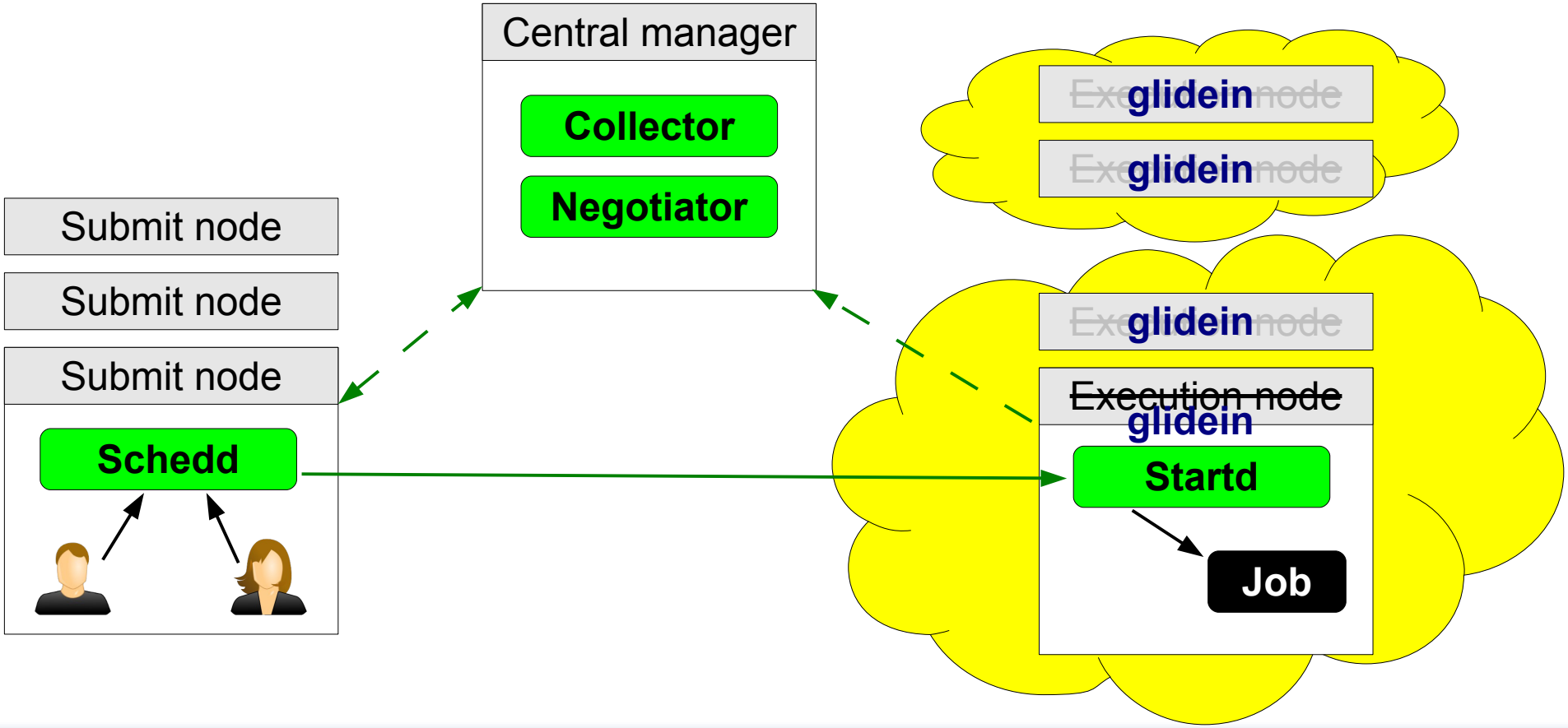
by Igor Sfiligoi and Jeff Dost (UCSD)

Overview

- Refresher
- Factory architecture
- Entry logic
- Security considerations

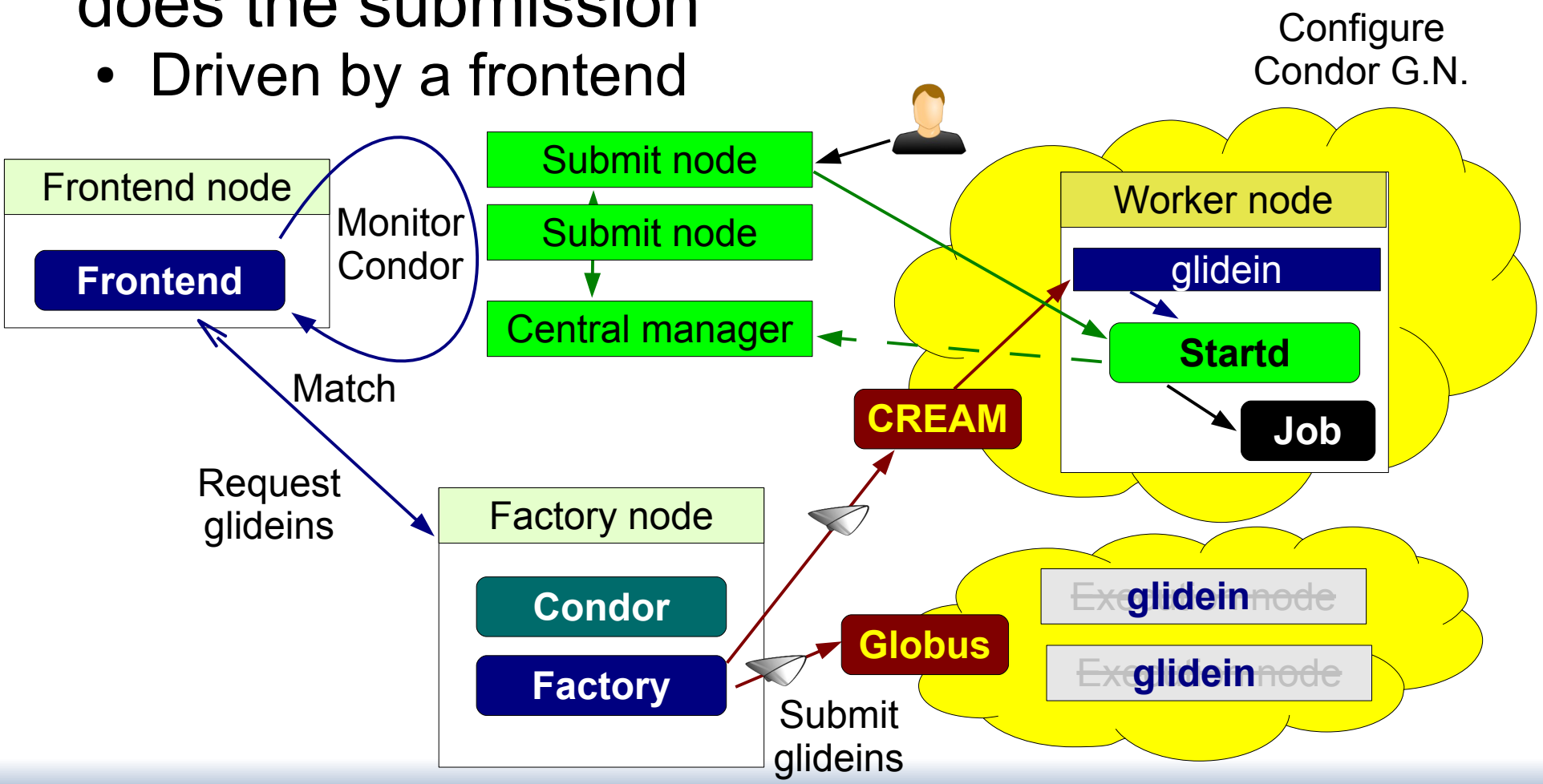
Refresher - What is a glidein?

- A glidein is just a properly configured execution node submitted as a Grid job



Refresher – Glidein factory

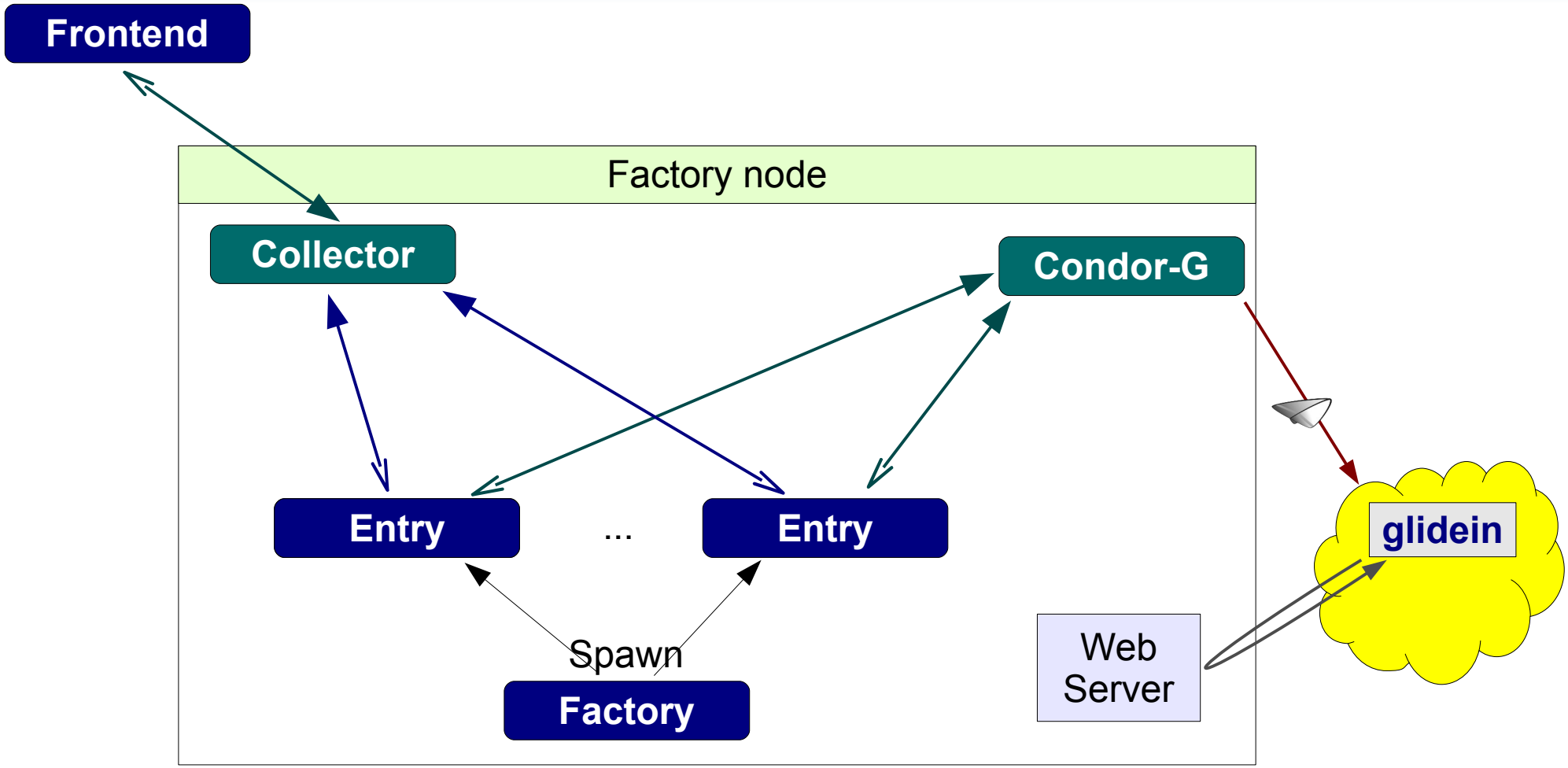
- The glidein factory knows about the sites and does the submission
  - Driven by a frontend



Factory architecture

- The factory is composed of:
  - The Condor collector – used for message passing
  - The glideinWMS factory proper
  - Condor-G – does the actual Grid submission
  - Web server – deliver code and data to glideins
    + monitoring
- The glideinWMS factory itself composed of:
  - Entry processes – do the real work
  - Master factory – controls the others and
    aggregates monitoring

Factory arch - picture



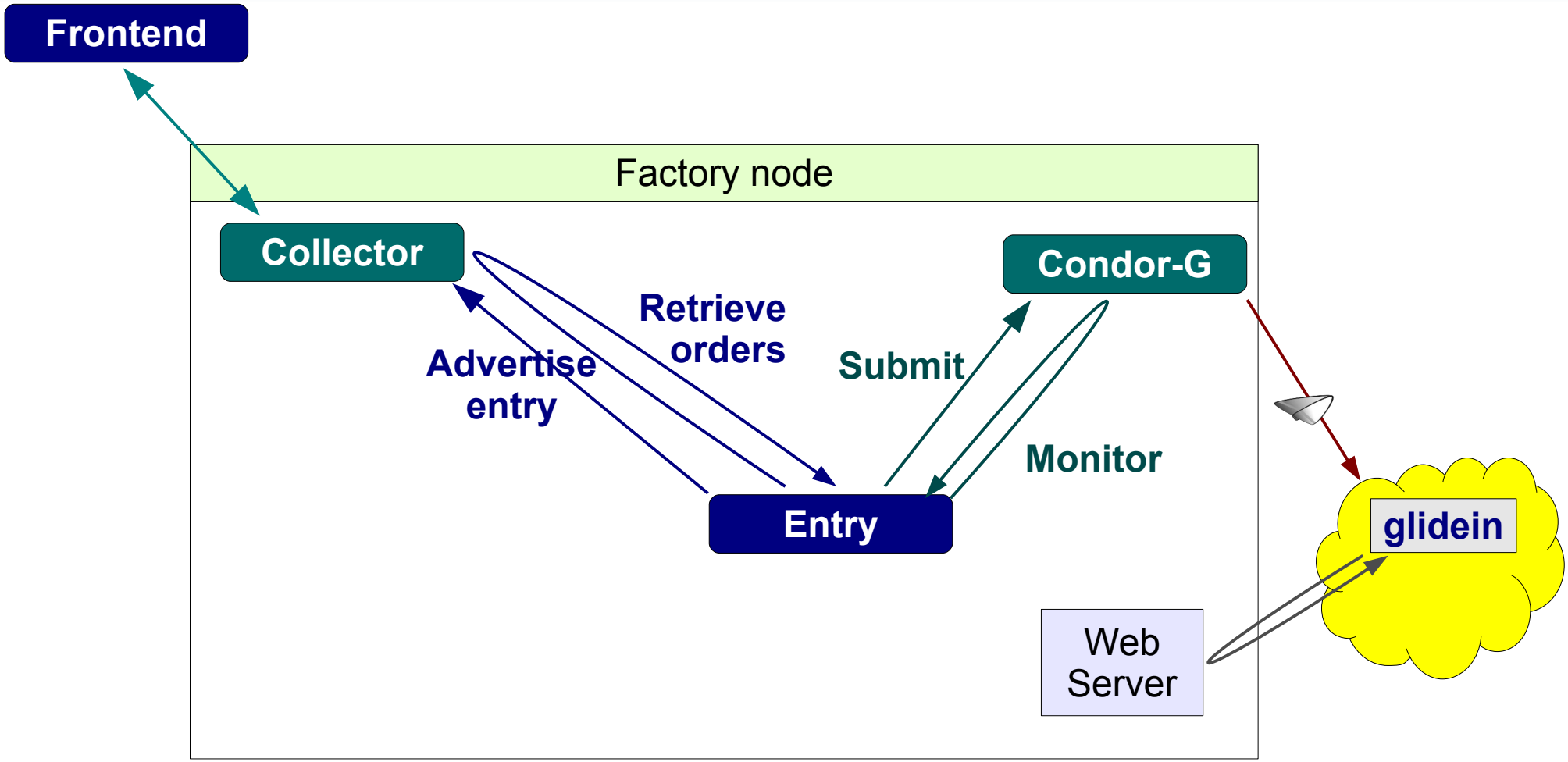
Factory processes

- Real work performed by Entry process
  - `glideFactoryEntry.py`
  - One process x Entry
- They are controlled by master Factory
  - `glideFactory.py`
  - Starts the other processes
  - Aggregates monitoring

Entry logic

- Essentially a slave
  - Will do what a frontend tells it to
- Uses the Collector for communication
  - Advertise own existence and attributes
  - Polls the collector for commands
  - Everything ClassAd based
  - All security implemented in the Collector
- Glideins submitted via Condor-G
  - Then just monitors them

Factory Entry - picture



Entry loop

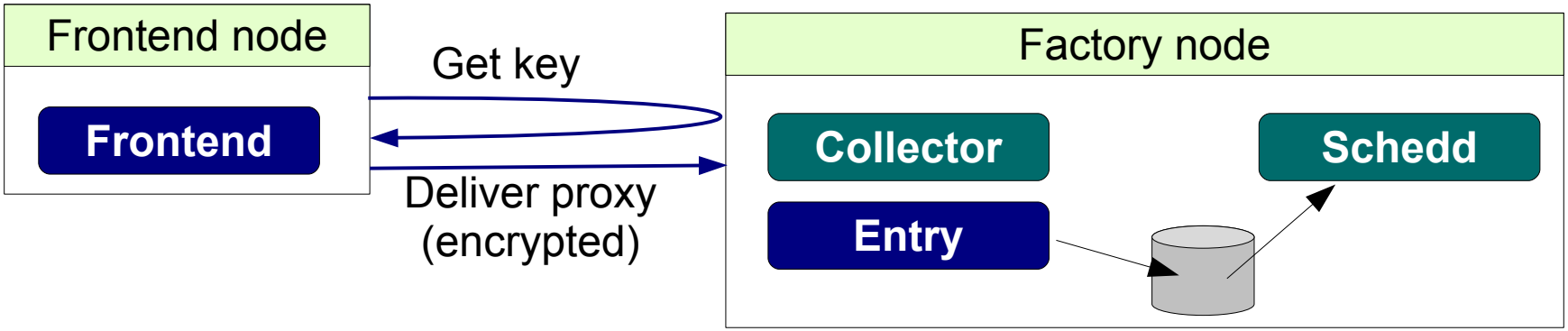
- Entry in continuous loop
  - Advertise → Read → Submit → Monitor → Advertise...
- The monitoring information is stored
  - In internal log files,
  - Web accessible location, and
  - Monitoring ClassAds

Entry Submission Limits

- Limits can be imposed from both the Factory and Frontend for the following:
  - Max Held, Max Idle, Max Running
- When the entry hits these limits it will not submit any new glideins
- This safeguard prevents spamming sites
- Conversely imposing limits can starve sites from new glidein submission
  - Manual intervention must take place when held limits are hit

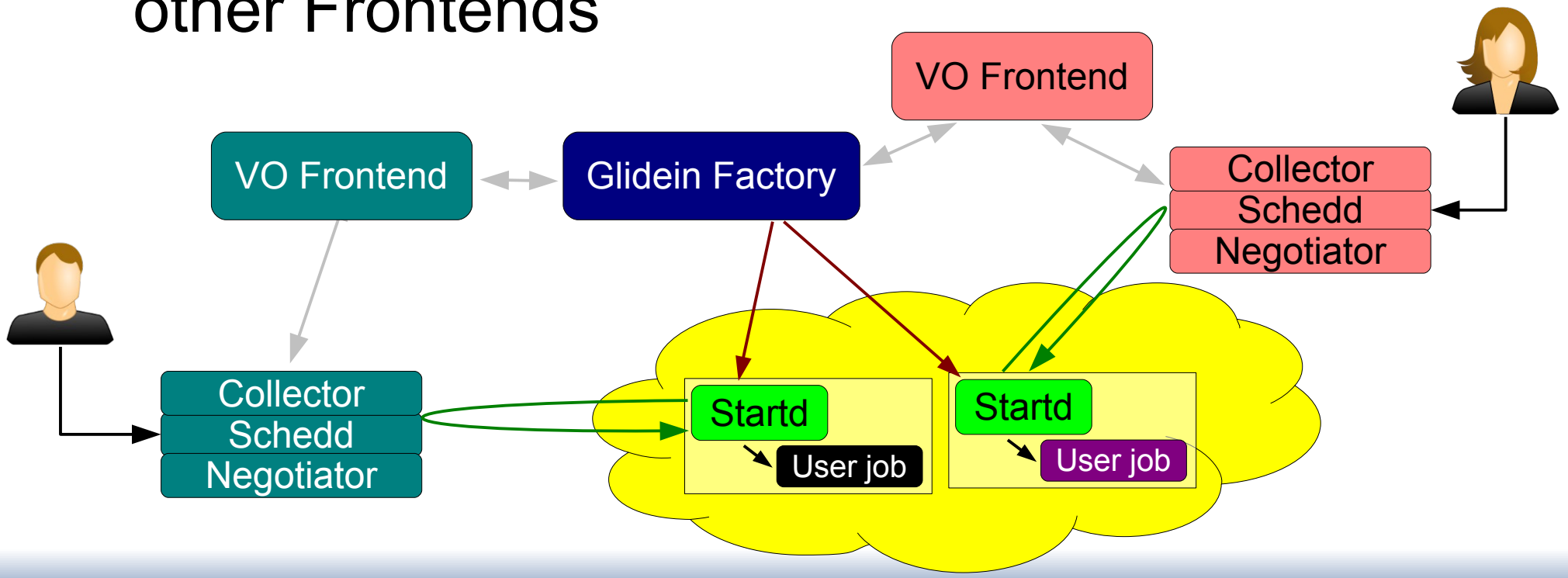
Credentials/Proxy

- **Proxy typically provided by the frontend**
  - Although the factory can provide a default one (rarely used)
- Proxy delivered encrypted in the ClassAd
  - Factory (entry) provides the encryption key (PKI)
- Proxy stored on disk
  - Each VO mapped to a different UID



Security considerations

- An Entry can serve multiple Frontends
  - So it will have multiple proxies
- Frontends trust the factory, but not necessarily other Frontends



User insulation

- Entry must use different UIDs for different Frontends
  - Both for file access, and
  - Condor-G (processes run in the name of that user)
- Entry has a FrontedName → UID map
- Must prevent spoofing
  - i.e. bad Frontend pretending to be a different one
  - Use Collector Auth to get TrustedIdentity, then whilelist FrontendName ↔ TrustedIdentity

Useful Log Files

- Condor-G:
  - stdout and stderr logs, job logs
- Condor logs from glidiens:
  - MasterLog, StartdLog, StarterLog
- Factory Logs
- Factory Condor Logs:
  - CollectorLog, Gridmanager logs

Pointers

- The official project Web page is http://tinyurl.com/glideinWMS
- glideinWMS development team is reachable at glideinwms-support@fnal.gov
- OSG glidein factory at UCSD
  http://hepuser.ucsd.edu/twiki2/bin/view/UCSDTier2/OSGgfactory
  http://glidein-1.t2.ucsd.edu:8319/glidefactory/monitor/glidein_Production_v4_1/factoryStatus.html

Acknowledgments

- The glideinWMS is a CMS-led project developed mostly at FNAL, with contributions from UCSD and ISI
- The glideinWMS factory operations at UCSD is sponsored by OSG
- The funding comes from NSF, DOE and the UC system