Clustern mit Condor

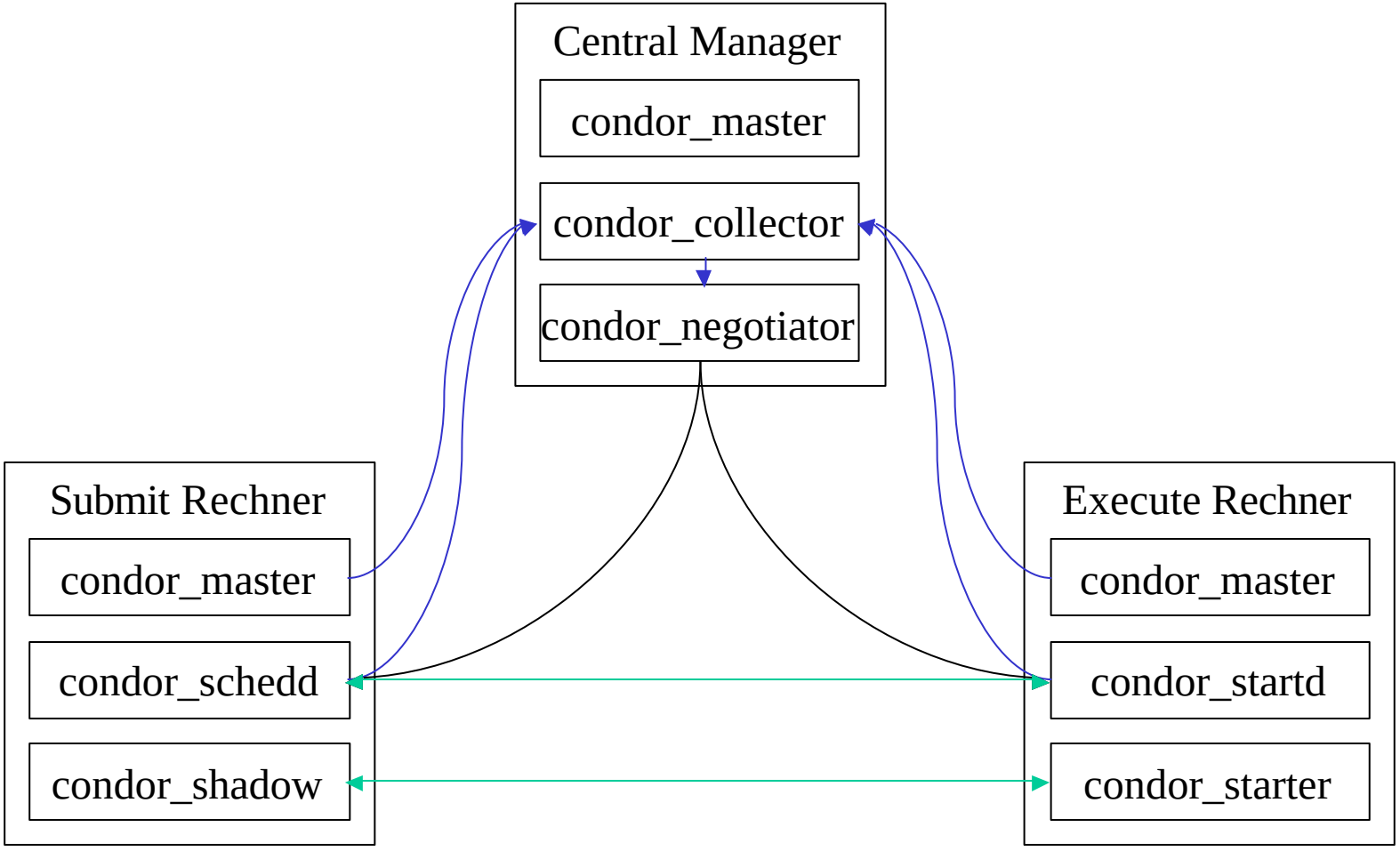
Inhalt:

1. Einleitung
2. Struktur eines Condor Cluster
3. Feature's von Condor
4. Jobs in Auftrag geben
5. Ausführung der Jobs
6. Sicherheit
7. Condor Universes
8. Erweiterung des Condorpool

1. Einleitung

- Ziel ist die Ausnutzung nicht benötigter Rechenzeit von Workstations
- gedacht für High Throughput Computing, also Anwendungen wo Rechenleistung pro Woche/Monat/Jahr von Interesse und nicht pro Sekunde
- Verwaltet Jobqueue mit Aufträgen der User und verteilt diese auf die zur Verfügung stehenden Rechner
- wird an der University of Wisconsin in Madison entwickelt
- binarys frei verfügbar für Linux, Windows, HPUX, Solaris und Digital Unix
- http://www.cs.wisc.edu/condor

2. Struktur eines Condor Cluster



3. Feature's von Condor

Job Managment

- es können viele Jobs gleichzeitig in Auftrag gegeben werden
- wenn Rechner die einen Job ausführen verschwinden werden die Jobs automatisch neu gestartet
- Jobs können Anforderungen an Rechner stellen und werden möglichst optimal auf die Rechner verteilt
- User die noch keine Ressourcen belegen haben Vorrang vor Usern die bereits viele Ressourcen belegen

Input / Output Umleitung

- stdout und stderr werden in Files umgeleitet
- stdin wird aus einem File umgeleitet

Checkpointing

- Checkpoints sind komplette Abbilder des laufenden Jobs
- enthalten auch welche Dateien gerade geöffnet sind und Zeiger in den Dateien
- ermöglichen das Fortsetzen des Jobs an der Stelle der Abarbeitung
- werden während der Abarbeitung der Jobs regelmäßig angelegt
- werden auch angelegt wenn der Job den Rechner freigeben muss

Einschränkungen

- können nicht angelegt werden solange Sockets geöffnet sind
- es kann zu Fehlern bei Ausgabedateien kommen wenn der Job an einem früheren Abarbeitungspunkt fortgesetzt wird

Remote System Calls

- Dateizugriffe der Jobs werden vom ausführenden Rechner an den auftraggebenden Rechner gesandt und dort ausgeführt
- Job hat die selbe Dateisystem Umgebung wie auf dem Rechner des Users

Datei Übertragung

- für Jobs bei denen keine Remote System Calls möglich sind und wo kein shared Filesystem vorhanden ist
- angegebene Dateien werden beim Starten bzw. Beenden des Jobs vom Rechner des Users auf den ausführenden Rechner übertragen und umgekehrt

4. Jobs in Auftrag geben

Submit Files für Jobs

Beschreiben den auszuführenden Job bzw. eine Anzahl von Jobs

Bsp:

```
Executable = job
Arguments = -c500 –s100
Universe = standard
Output = job.outbut
Error = job.error
Input = job.input
Initialdir = jobdir

Queue
```

Requirements

- definiert Anforderungen des Jobs an den ausführenden Rechner
- dabei alle Attribute der Maschine ClassAd‘s verwendbar (z.B. Architektur, Ram und Festplatten Speicher, Filesystem Domäne)
- wenn nicht anders definiert wird automatisch die Architektur und das OS des auftraggebenden Rechners hinzugefügt

Bsp.:

```
Requirements = ( FileSystemDomain ==
                      "prakinf.tu-ilmenau.de")
                  && ( Memory >= 64 )
```

Rank

- über den Rank Ausdruck wird eine Bewertung der Rechner für den Job vorgenommen
- erlaubt bei mehreren freien Rechnern Wahl des für den Job besten Rechners
- wie bei den Requirements alle Werte des Maschine ClassAd erlaubt
- wenn nicht definiert ist der Rank = 0
- Beispiel:

```
Rank = (memory * 1000) + kflops
```

Mehrfachausführung

- es können mehrere ähnliche Jobs auf einmal gestartet werden
- durch einzelne Angabe der Änderungen

```
Executable = job
Universe = standard
Output = job.outbut
Error = job.error
Input = job.input
Initialdir = firstdir
Queue

Initialdir = secounddir
Queue
```

- durch Verwendung von $(Process) welches immer die Nummer der Ausführung enthält

```
Executable = job
Universe = standard
Output = job.outbut
Error = job.error
Input = job.input
Initialdir = jobdir.$(PROCESS)

Queue 100
```

Ausführung auf verschiedenen Architekturen

- es ist möglich einen Job für mehrere Alternative Architekturen in Auftrag zu geben

```
Executable = job.$$(Arch).$$(Opsys)
Requirements =
   ((Arch=="INTEL")&&(OpSys=="LINUX)) ||
   ((Arch=="SUN4u")&&(OpSys=="SOLARIS28"))
Queue
```

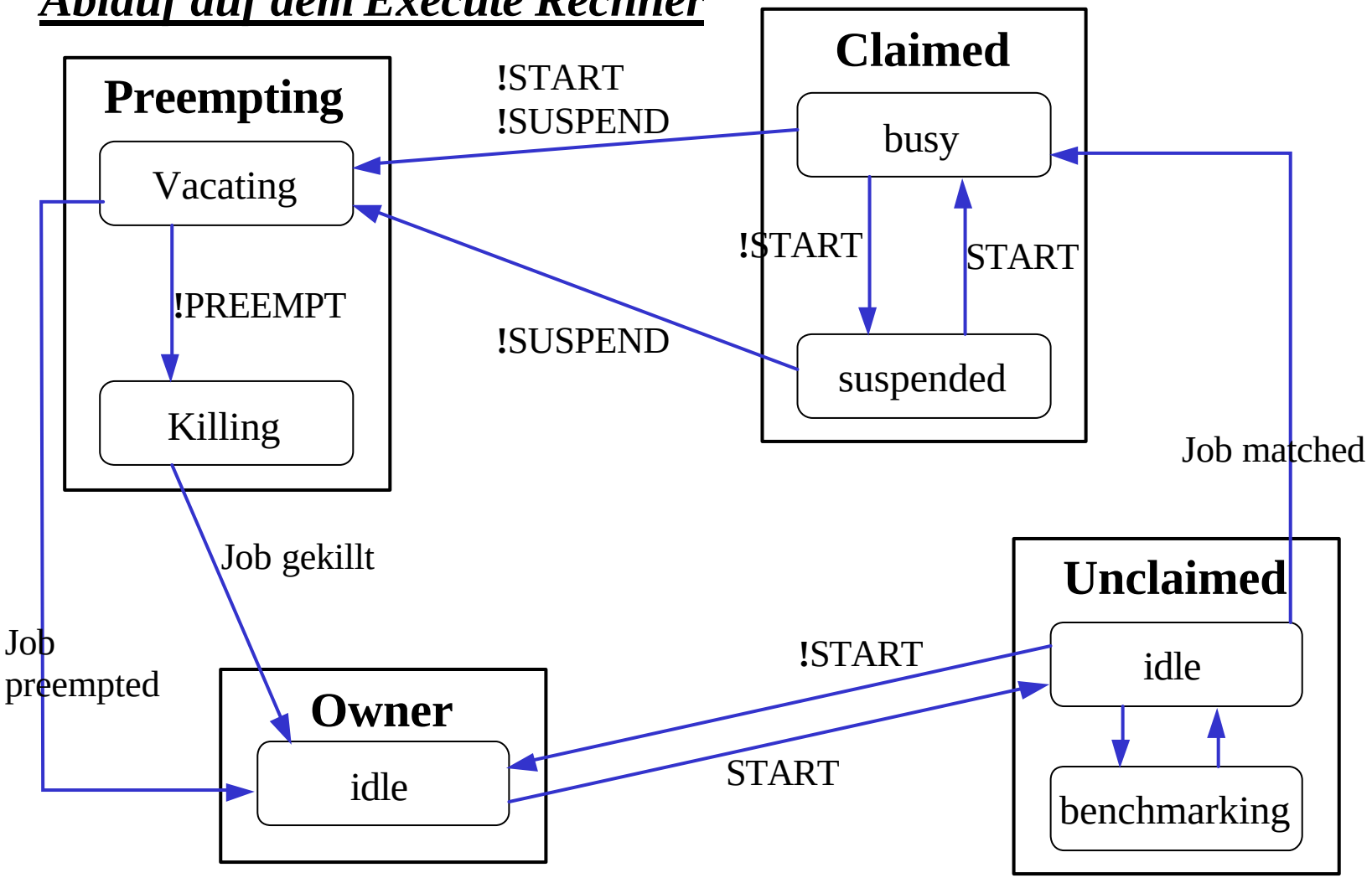
- je nach Rechner auf dem der Job dann ausgeführt wird, wird dann als binary *job.INTEL.LINUX* oder *job.SUN4u.SOLARIS* verwendet

5. Ausführung der Jobs

Ressourcen im Condor Pool

- über Start Ausdruck kann Nutzer festlegen wann Jobs auf seinem Rechner ausgeführt werden dürfen
- Requirements Ausdruck definierbar der TRUE ergeben muss ehe der Job auf den Rechner gestartet wird
- Rank Ausdruck bewertet Jobs welchen der Rechner lieber ausführen würde
- Rank und Requirements können auf alle Attribute des Jobs zugreifen
- damit Jobs einzelner Nutzer auf Rechnern ausschließbar oder priorisierbar

Ablauf auf dem Execute Rechner



User Prioritäten

RUP – Real User Priority

- spiegelt den Ressourcenverbrauch des Users wieder
- startet bei 0,5 und nähert sich der Anzahl der vom User belegten Rechner an
- baut sich mit einer definierbaren Halbwertzeit ab wenn der User weniger Ressourcen verwendet

EUP – Effectiv User Priority

- gibt Priorität an die für Ressourcenvergabe verwendet wird
- ergibt sich durch den Prioritätsfaktor des Users aus der RUP
- Multiplikation mit höherem Faktor wenn der Job mit nice Priorität oder per flocking aus einem andern Pool submittet wurden

6. Sicherheit

Access Levels

- verschiedene Access Levels für Unterschiedliche Zugriffe
- alle Sicherheitseinstellungen lassen sich für die Access Levels getrennt konfigurieren
  - ADMINISTRATOR
  - OWNER
  - READ
  - WRITE
  - NEGOTIATOR
  - CONFIG

Host/User based

- Filterung der Zugriffe nach Hosts oder nach Usern von bestimmten Hosts
- Allow und Deny Regeln wobei Ausdrücke ohne Wildcards und Deny Regeln Vorrang haben

Nachrichtenverschlüsselung

- als erforderlich, bevorzugt, möglich oder nie definierbar
- unterstützte Methoden:
  - 3DES
  - BLOWFISH

Nachrichtenintegrität

- Überprüfung mittels md5 Checksummen

Server / Client Authentifizierung

- für die Access Levels getrennt aktivierbar
- erlauben mehrerer Methoden möglich
- Mögliche Methoden:
  - FS / FS_REMOTE
  - CLAIMTOBE
  - NTSSPI
  - KERBEROS
  - GSS_AUTHENTIFICATION

7. Condor Universes

Standart Universe

- unterstützt Remote System Calls und Checkpointing

Anforderungen an den Job

- Jobs müssen mit Condor Libraries neu gelinkt werden
- keine Kernel-level Threads (kein fork, exec, system)
- keine Interprozesskomunikation
- keine Alarme,Timer oder Sleeps
- Files sollten nur read oder write only öffnen
- keine graphische Oberfläche
- geringe Netzwerkkommunikation

Vanilla Universe

- für Jobs die nicht neu gelinkt werden können
- unterstützt kein Checkpointing und keine Remote System Calls
- File Transfer für Eingabe und Ausgabedateien ist möglich
- Threads, Interprozesskomunikation erlaubt
- jegliche Netzwerkkomunikation erlaubt
- graphischen Oberfläche erlaubt

Java Universe

- ermöglicht Ausführung von Java Programmen
- kein Checkpointing und keine Remote System Calls möglich
- Remote File Transfer möglich

PVM Universe

- PVM ist ein Softwaresystem zum Aufbau eines Parallelrechners aus normalen PC‘s
- PVM binarys können von Condor ausgeführt werden
- es sind nur Programme nach dem Master-Worker Paradigma möglich
- Master wird auf dem Rechner des Users gestartet
- freie Rechner aus dem Condor Pool werden als Worker zugeteilt
- PVM Support muss als extra Modul für den Pool installiert werden

Globus Universe

- ermöglicht es mit dem Condor Interface Jobs in einem Globus Grid in Auftrag zu geben

8. Erweiterung des Condorpool

Flocking

- erlaubt es Nutzern Jobs in weiter Pools zu stellen
- Definition der alternativen Pools erfolgt auf dem Rechner des Nutzers
- Jobs werden automatisch in anderen Pools ausgeführt wenn im eigenen keine Ressourcen frei sind
- Flocking muss im Zielpool für die fremden Rechner erlaubt werden
- Jobs die durch Flocking in einen Pool kommen haben geringere Priorität

Condor_G

- bietet die Möglichkeit Globus Grids als Ressource in den Pool aufzunehmen
- Jobs die auf den Grids laufen sollen können wie normale Condor Jobs gemanged werden und werden von Condor überwacht
- Condor kümmert sich um den Transfer der benötigten Dateien zum Grid

Fragen ?