CONDOR INSTALLATION GUIDE

1. GENERAL

This document explains how to create and install *condor* from the source. To do this, you will need the capability of becoming super user on all of the machines where you want condor to run. We also assume that you are familiar with local procedures for creating unix[1] accounts, and allocating user and group id numbers. In addition, if you wish to use NFS[2] to share some of the *condor* executables and libraries between machines, you will need to be familiar with exporting and importing NFS file systems.

2. OPERATIONS IN A HETEROGENEOUS ENVIRONMENT

Most local area networks contain a variety of machine and operating system types. To maximize the sharing of resources, *condor* attempts to provide some interoperability between the various machine architectures and operating systems. This does add some complication to the *condor* code, and will require some consideration by the person creating and installing the *condor* executables.

Condor is designed so that all of the various machines may reside in a single ‘‘resource’’ pool. All of the *condor* daemons communicate through XDR[3] library routines, and are thus compatible even between machines which use different byte ordering. Users of a machine of any particular architecture and operating system type will be able to submit jobs to be run on other architecture and operating system combinations. (Of course those jobs will need to be compiled and linked for the appropriate target architecture/operating system combination.)

Supported Architecture and Operating System Combinations

Architecture	Operating System	Description
VAX	ULTRIX	Vax running Ultrix 3.0+
VAX	BSD43	Vax running 4.3BSD
MC68020	SUNOS32	Sun 3 running SunOs3.2
SPARC	SUNOS32	Sun 4 running SunOs3.2
SPARC	SUNOS40	Sun 4 running SunOs4.0
I386	DYNIX	Sequent symmetry running Dynix[4]
MIPS	ULTRIX	DECstation 3100 running Ultrix 3.0+

3. INSTALLATION OVERVIEW

If you use NFS, you can extract the *condor* source on a single machine, and then mount object and executable file trees on machines of the various types for which you wish to compile *condor*. Alternatively, you can distribute the source tree to the appropriate machines using ‘‘rcp’’ or ‘‘rdist’’.

---

[1]UNIX is a trademark of AT&T.

[2]NFS is a trademark of Sun Microsystems.

[3]XDR is a trademark of Sun Microsystems.

[4]Symmetry and Dynix are a trademarks of Sequent Computer Systems.

Once you have compiled executables suitable for the machines you wish to include in your *condor* pool, you will need to make those executables available on the member machines. There are a number of possible ways to do this. When you create the executables they will be placed in a directory called <releasedir>, (details on how to set this pathname will be given later). To include the machine where you did the compilation in the pool, you might set <releasedir> to be a well known directory, or create a symbolic link to a well known directory. For other machines in the pool, you might choose to use the compilation machine as a fileserver; in this case you can use NFS to mount <releasedir> in a well known place. If the machines to be in the *condor* pool already have some of their executables remotely mounted from fileservers, you might want to distribute the executables from <releasedir> to those fileservers using ''rcp'' or ''rdist''.

Each machine which is a member of the *condor* pool will need to have a ''condor'' account. If you use NFS, it will be necessary for all of the ''condor'' accounts to share common user and group id's. *Condor* must have its own group id. Group id's such as ''daemon'' or ''bin'' are not suitable for use by *condor*. Each ''condor'' account will need a home directory containing 3 subdirectories, ''log'', ''spool'', and ''execute''. A script is provided which will create these directories with the proper ownership and mode. If you choose to have these directories remotely mounted, be sure each *condor* machine has its own private version of these directories. Each ''condor'' account will have two files in the home directory called ''condor_config'' and ''condor_config.local''. ''Condor_config'' may be shared via NFS, but ''condor_config.local'' should be private. *Condor* users will need to access the *condor* ''bin'' and ''lib'' directories. These may also be shared with remotely mounted file systems. Finally, each member machine will need to start the *condor* daemons at boot time. This will be done by placing an entry in ''/etc/rc'' or ''/etc/rc.local'' which starts the ''condor_master''. The master will determine which daemons should be run on its machine, and will monitor their health, restarting them and mailing system personnel about problems as necessary.

In addition to the member machines, you will need to designate one machine to act as the ''central manager''. The central manager will run two extra daemons which communicate with and coordinate the daemons on all member machines. These daemons will also be started and monitored by the master.

4. CREATING AND DISTRIBUTING EXECUTABLES

(1) Create a user account for ''condor'', and set up a condor group as well. Change directory to the condor home directory and run as ''condor'', e.g. ''su condor''.

(2) Extract the ''CONDOR'' directory from the distribution file, e.g. ''uncompress Condor.tar.Z'', then ''tar xf Condor.tar''.

(3) If you wish to make executables for an architecture/operating system other than the machine where you have extracted the tape, you will need to either copy the files to the ''compilation'' machine, or preferably remotely mount them via NFS. In any case, all the condor files should have owner ''condor'', and group ''condor''. You should always be running as ''condor'' when you make condor executables.

(4) ''Cd'' to the ''CONDOR'' directory, and edit ''GENERIC/condor_config''. Set ''CONDOR_HOST'' to the hostname where the central manager daemons will run. We recommend setting this up as a network alias, so you can easily change which machine runs the central manager daemons later. Set ''CONDOR_ADMIN'' to the electronic mailing address of the person responsible for maintenance of *condor* at your site. We recommend setting this to a mailing list, again so you can easily change who the actual recipient is at a later date. Set ''RELEASEDIR'' to the pathname where you want condor users to access the executables and libraries. This should probably not be the condor source directory.

(5) Create an object tree for the specific architecture/operating system type for which you are creating executables. Do this by running ''NewArch <ARCH> <OPSYS>'' in the ''CONDOR'' directory. The ''ARCH'' and ''OPSYS'' arguments must be chosen from the above table. This will create a directory called ''<ARCH>_<OPSYS>'' containing subdirectories for all of the *condor* object files.

(6) ''Cd'' to the ''<ARCH>_<OPSYS>'' directory and compile all of the *condor* programs by typing ''make opt=release''.

(7) If you want to make the executables accessible via a symbolic link, or by distributing to a fileserver, you can edit the template ‘‘INSTALL’’ script to do this for your situation. If you will be using the compilation machine as a fileserver for *condor* executables, you should grant the necessary permissions in ‘‘/etc/exports’’.

(8) You will also need to install the *condor* man pages. These will be found in ‘‘CONDOR/doc/man/{manl,catl}’’. The exact commands will vary somewhat depending on the situation at your site. If you mount your man pages on a shared fileserver, they may look something like this:

```
rcp manl/* <fileserver>:/usr/man/manl
rcp catl/* <fileserver>:/usr/man/catl
```

(9) To make and install executables for other architecture/operating system types, go back to step 3.

5. ADDING MEMBER MACHINES TO THE CONDOR POOL

Complete the following steps on each machine you want to add to the condor resource pool. Add the machine which will act as the ‘‘central manager’’ first. **N.B. all of the steps in setting up a member of the condor pool will require you to operate as the super user.**

(1) Create an account for ‘‘condor’’ on the member machine. Be sure to use the same user and group id’s on all member machines.

(2) If you are planning to access the condor executables on this machine via a remotely mounted file system, make sure that file system is currently mounted, and that there is an appropriate entry in ‘‘/etc/fstab’’ so that it will get mounted whenever the machine is booted.

For example on a Sun 3 running SunOs 3.2 the fstab entry might look something like:

```
<fileserver>:<some path>/MC68020_SUNOS32/release <RELEASEDIR> nfs ro 0 0
```

(3) Run the script ‘‘Condor.init’’. This will link ‘‘condor_config’’ to a site specific version of that file, and create the ‘‘log’’, ‘‘spool’’, and ‘‘execute’’ directories with correct ownership and permissions.

(4) Run the script ‘‘Condor.on’’. This will create and edit ‘‘condor_config.local’’ setting ‘‘START_DAEMONS’’ to ‘‘True’’ so that the condor daemons are able to run, then it will actually start them.

(5) At this point the member machine should be fully operational. On all machines you should find the ‘‘condor_master’’, ‘‘condor_startd’’, and ‘‘condor_schedd’’ running. Machines which run the X window system, should also be running the ‘‘condor_kbdd’’. Additionally the ‘‘central manager’’ machine should be running the ‘‘condor_collector’’ and ‘‘condor_negotiator’’. You can check to see that the proper daemons are runing with

```
ps -ax | egrep condor
```

You should also run ‘‘condor_status’’ to see that the new machine shows up in the resource pool. If you wish to run some trivial jobs to check operation of all the *condor* software, example user programs and a ‘‘job description’’ file have been compiled and are provided in the ‘‘<ARCH>_<OPSYS>/client’’ directory.

(6) Add lines to ‘‘/etc/rc’’ or ‘‘/etc/rc.local’’ which will start ‘‘condor_master’’ at boot time.

The entry will look something like this:

```
if [ -f /usr/uw/condor/bin/condor_master ]; then
      /usr/uw/condor/bin/condor_master; echo -n ' condor'   >/dev/console
fi
```

Note: do not attempt to run this command now, condor is already running.

6. EXAMPLE

Figure 1 presents a simplified version of how *condor* is made and installed at the University of Wisconsin. We keep all of the *condor* source, objects, and the master copies of the executables on ‘‘shorty’’ which is a Sequent Symmetry machine running Dynix. Users on all of our machines access the *condor* executable and library directories as ‘‘/usr/uw/condor/{bin,lib}’’. In some cases this is accomplished by a symbolic link to the <releasedir>, and in other cases it is done by a remote file system mount either directly to shorty, or to a copy on a separate fileserver.

The <releasedir> for the I386_DYNIX version of the executables and libraries is ‘‘~condor/CONDOR/I386_DYNIX/release’’ on shorty. Shorty’s users access those executables and libraries via a symbolic link as ‘‘/usr/uw/condor/{bin,lib}’’.

We use ‘‘Bugs’’ which is a Vax running BSD4.3 to create the *condor* software for vaxen running 4.3 by remotely mounting ‘‘~condor/CONDOR’’ from shorty. From within the ‘‘CONDOR’’ directory on bugs, we were able to run ‘‘NewArch VAX BSD43’’ creating the ‘‘VAX_BSD43’’ directory tree. The <releasedir> for the VAX_BSD43 version of the executables and libraries is ‘‘~condor/CONDOR/VAX_BSD43/release.’’ We distribute that to a fileserver for use by all of the 4.3 vaxen in the department, including bugs. Users on these machines access the *condor* software as ‘‘/usr/uw/condor/{bin,lib}’’ just like on shorty, but this time it is through a remote file system.

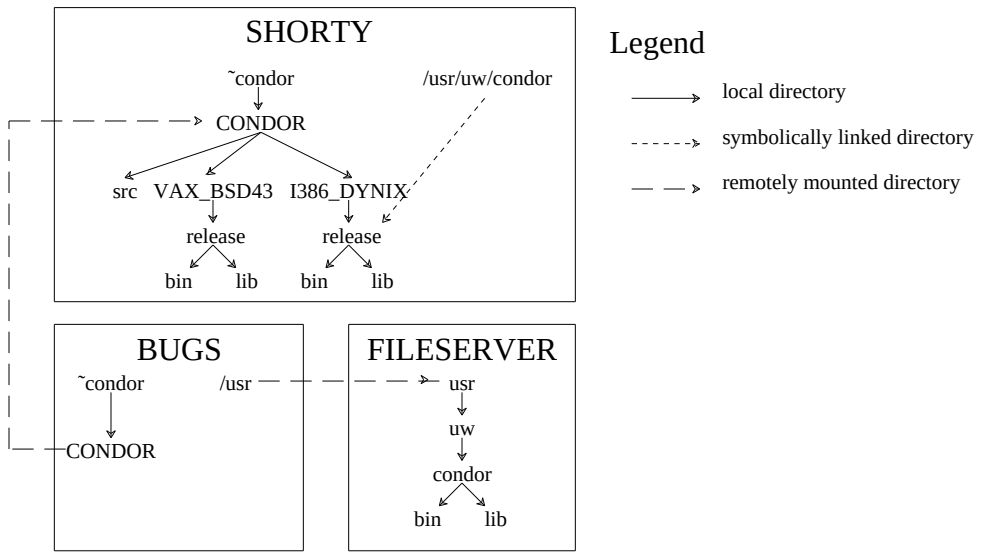

Figure 1

7. Copyright Information

Copyright 1986, 1987, 1988, 1989 University of Wisconsin

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of the University of Wisconsin not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. The University of Wisconsin makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

THE UNIVERSITY OF WISCONSIN DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE UNIVERSITY OF WISCONSIN BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN

CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Authors: Allan Bricker and Michael J. Litzkow,
University of Wisconsin, Computer Sciences Dept.