The Leading Open Source Backup Solution

Bacula® Miscellaneous Guide

Kern Sibbald

June 29, 2018
This manual documents Bacula version 9.0.8 (28 May 2018)

Contents

Chapter 1

Python Scripting

You may be asking what Python is and why a scripting language is needed in Bacula. The answer to the first question is that Python is an Object Oriented scripting language with features similar to those found in Perl, but the syntax of the language is much cleaner and simpler. The answer to why have scripting in Bacula is to give the user more control over the whole backup process. Probably the simplest example is when Bacula needs a new Volume name, with a scripting language such as Python, you can generate any name you want, based on the current state of Bacula.

1.1 Python Configuration

Python must be enabled during the configuration process by adding a --with-python, and possibly specifying an alternate directory if your Python is not installed in a standard system location. If you are using RPMs you will need the python-devel package installed.

When Python is configured, it becomes an integral part of Bacula and runs in Bacula's address space, so even though it is an interpreted language, it is very efficient.

When the Director starts, it looks to see if you have a **Scripts Directory** Directive defined (normal default **/etc/bacula/scripts**, if so, it looks in that directory for a file named **DirStartUp.py**. If it is found, Bacula will pass this file to Python for execution. The **Scripts Directory** is a new directive that you add to the Director resource of your bacula-dir.conf file.

Note: Bacula does not install Python scripts by default because these scripts are for you to program. This means that with a default installation with Python enabled, Bacula will print the following error message:

```
09-Jun 15:14 bacula-dir: ERROR in pythonlib.c:131 Could not import
Python script /etc/bacula/scripts/DirStartUp. Python disabled.
```

The source code directory **examples/python** contains sample scripts for DirStartUp.py, SDStartUp.py, and FDStartUp.py that you might want to use as a starting point. Normally, your scripts directory (at least where you store the Python scripts) should be writable by Bacula, because Python will attempt to write a compiled version of the scripts (e.g. DirStartUp.pyc) back to that directory.

When starting with the sample scripts, you can delete any part that you will not need, but you should keep all the Bacula Event and Job Event definitions. If you do not want a particular event, simply replace the existing code with a **noop = 1**.

1.2 Bacula Events

A Bacula event is a point in the Bacula code where Bacula will call a subroutine (actually a method) that you have defined in the Python StartUp script. Events correspond to some significant event such as a Job Start, a Job End, Bacula needs a new Volume Name, ... When your script is called, it will have access to all the Bacula variables specific to the Job (attributes of the Job Object), and it can even call some of the Job methods (subroutines) or set new values in the Job attributes, such as the Priority. You will see below how the events are used.

1.3 Python Objects

There are four Python objects that you will need to work with:

The Bacula Object The Bacula object is created by the Bacula daemon (the Director in the present case) when the daemon starts. It is available to the Python startup script, **DirStartup.py**, by importing the Bacula definitions with **import bacula**. The methods available with this object are described below.

The Bacula Events Class You create this class in the startup script, and you pass it to the Bacula Object's **set_events** method. The purpose of the Bacula Events Class is to define what global or daemon events you want to monitor. When one of those events occurs, your Bacula Events Class will be called at the method corresponding to the event. There are currently three events, JobStart, JobEnd, and Exit, which are described in detail below.

The Job Object When a Job starts, and assuming you have defined a JobStart method in your Bacula Events Class, Bacula will create a Job Object. This object will be passed to the JobStart event. The Job Object has a has good number of read-only members or attributes providing many details of the Job, and it also has a number of writable attributes that allow you to pass information into the Job. These attributes are described below.

The Job Events Class You create this class in the JobStart method of your Bacula Events class, and it allows you to define which of the possible Job Object events you want to see. You must pass an instance of your Job Events class to the Job Object set_events() method. Normally, you will probably only have one Job Events Class, which will be instantiated for each Job. However, if you wish to see different events in different Jobs, you may have as many Job Events classes as you wish.

The first thing the startup script must do is to define what global Bacula events (daemon events), it wants to see. This is done by creating a Bacula Events class, instantiating it, then passing it to the **set_events** method. There are three possible events.

JobStart This Python method, if defined, will be called each time a Job is started. The method is passed the class instantiation object as the first argument, and the Bacula Job object as the second argument. The Bacula Job object has several built-in methods, and you can define which ones you want called. If you do not define this method, you will not be able to interact with Bacula jobs.

JobEnd This Python method, if defined, will be called each time a Job terminates. The method is passed the class instantiation object as the first argument, and the Bacula Job object as the second argument.

Exit This Python method, if defined, will be called when the Director terminates. The method is passed the class instantiation object as the first argument.

Access to the Bacula variables and methods is done with:

import bacula

The following are the read-only attributes provided by the bacula object.

Name

ConfigFile

WorkingDir

Version string consisting of "Version Build-date"

A simple definition of the Bacula Events Class might be the following:

```
import sys, bacula
class BaculaEvents:
  def JobStart(self, job):
     ...
```

Then to instantiate the class and pass it to Bacula, you would do:

```
bacula.set_events(BaculaEvents()) # register Bacula Events wanted
```

And at that point, each time a Job is started, your BaculaEvents JobStart method will be called.

Now to actually do anything with a Job, you must define which Job events you want to see, and this is done by defining a JobEvents class containing the methods you want called. Each method name corresponds to one of the Job Events that Bacula will generate.

A simple Job Events class might look like the following:

```
class JobEvents:
  def NewVolume(self, job):
     ...
```

Here, your JobEvents class method NewVolume will be called each time the Job needs a new Volume name. To actually register the events defined in your class with the Job, you must instantiate the JobEvents class and set it in the Job **set_events** variable. Note, this is a bit different from how you registered the Bacula events. The registration process must be done in the Bacula JobStart event (your method). So, you would modify Bacula Events (not the Job events) as follows:

```
import sys, bacula
class BaculaEvents:
  def JobStart(self, job):
     events = JobEvents()         # create instance of Job class
     job.set_events(events)       # register Job events desired
     ...
```

When a job event is triggered, the appropriate event definition is called in the JobEvents class. This is the means by which your Python script or code gets control. Once it has control, it may read job attributes, or set them. See below for a list of read-only attributes, and those that are writable.

In addition, the Bacula **job** object in the Director has a number of methods (subroutines) that can be called. They are:

set_events The set_events method takes a single argument, which is the instantiation of the Job Events class that contains the methods that you want called. The method names that will be called must correspond to the Bacula defined events. You may define additional methods but Bacula will not use them.

run The run method takes a single string argument, which is the run command (same as in the Console) that you want to submit to start a new Job. The value returned by the run method is the JobId of the job that started, or -1 if there was an error.

write The write method is used to be able to send print output to the Job Report. This will be described later.

cancel The cancel method takes a single integer argument, which is a JobId. If JobId is found, it will be canceled.

DoesVolumeExist The DoesVolumeExist method takes a single string argument, which is the Volume name, and returns 1 if the volume exists in the Catalog and 0 if the volume does not exist.

The following attributes are read/write within the Director for the **job** object.

Priority Read or set the Job priority. Note, that setting a Job Priority is effective only before the Job actually starts.

Level This attribute contains a string representing the Job level, e.g. Full, Differential, Incremental, ... if read. The level can also be set.

The following read-only attributes are available within the Director for the **job** object.

Type This attribute contains a string representing the Job type, e.g. Backup, Restore, Verify, ...

JobId This attribute contains an integer representing the JobId.

Client This attribute contains a string with the name of the Client for this job.

NumVols This attribute contains an integer with the number of Volumes in the Pool being used by the Job.

Pool This attribute contains a string with the name of the Pool being used by the Job.

Storage This attribute contains a string with the name of the Storage resource being used by the Job.

Catalog This attribute contains a string with the name of the Catalog resource being used by the Job.

MediaType This attribute contains a string with the name of the Media Type associated with the Storage resource being used by the Job.

Job This attribute contains a string containing the name of the Job resource used by this job (not unique).

JobName This attribute contains a string representing the full unique Job name.

JobStatus This attribute contains a single character string representing the current Job status. The status may change during execution of the job. It may take on the following values:

- **C** Created, not yet running
- **R** Running
- **B** Blocked
- **T** Completed successfully
- **E** Terminated with errors
- **e** Non-fatal error
- **f** Fatal error
- **D** Verify found differences
- **A** Canceled by user
- **F** Waiting for Client
- **S** Waiting for Storage daemon
- **m** Waiting for new media
- **M** Waiting for media mount
- **s** Waiting for storage resource

- **j** Waiting for job resource
- **c** Waiting for client resource
- **d** Waiting on maximum jobs
- **t** Waiting on start time
- **p** Waiting on higher priority jobs

Priority This attribute contains an integer with the priority assigned to the job.

CatalogRes tuple consisting of (DBName, Address, User, Password, Socket, Port, Database Vendor) taken from the Catalog resource for the Job with the exception of Database Vendor, which is one of the following: MySQL, PostgreSQL, SQLite, Internal, depending on what database you configured.

VolumeName After a Volume has been purged, this attribute will contain the name of that Volume. At other times, this value may have no meaning.

The following write-only attributes are available within the Director:

JobReport Send line to the Job Report.

VolumeName Set a new Volume name. Valid only during the NewVolume event.

1.4 Python Console Command

There is a new Console command named **python**. It takes a single argument **restart**. Example:

```
python restart
```

This command restarts the Python interpreter in the Director. This can be useful when you are modifying the DirStartUp script, because normally Python will cache it, and thus the script will be read one time.

1.5 Debugging Python Scripts

In general, you debug your Python scripts by using print statements. You can also develop your script or important parts of it as a separate file using the Python interpreter to run it. Once you have it working correctly, you can then call the script from within the Bacula Python script (DirStartUp.py).

If you are having problems loading DirStartUp.py, you will probably not get any error messages because Bacula can only print Python error messages after the Python interpreter is started. However, you may be able to see the error messages by starting Bacula in a shell window with the **-d1** option on the command line. That should cause the Python error messages to be printed in the shell window.

If you are getting error messages such as the following when loading DirStartUp.py:

```
Traceback (most recent call last):
  File "/etc/bacula/scripts/DirStartUp.py", line 6, in ?
    import time, sys, bacula
ImportError: /usr/lib/python2.3/lib-dynload/timemodule.so: undefined
symbol: PyInt_FromLong
bacula-dir: pythonlib.c:134 Python Import error.
```

It is because the DirStartUp script is calling a dynamically loaded module (timemodule.so in the above case) that then tries to use Python functions exported from the Python interpreter (in this case PyInt_FromLong). The way Bacula is currently linked with Python does not permit this. The solution to the problem is to put such functions (in this case the import of time into a separate Python script, which will do your calculations and return the values you want. Then call (not import) this script from the Bacula DirStartUp.py script, and it all should work as you expect.

1.6 Python Example

An example script for the Director startup file is provided in **examples/python/DirStartup.py** as follows:

```
#
# Bacula Python interface script for the Director
#

# You must import both sys and bacula
import sys, bacula

# This is the list of Bacula daemon events that you
#  can receive.
class BaculaEvents(object):
  def __init__(self):
     # Called here when a new Bacula Events class is
     #  is created. Normally not used
     noop = 1

  def JobStart(self, job):
     """
       Called here when a new job is started. If you want
       to do anything with the Job, you must register
       events you want to receive.
     """
     events = JobEvents()         # create instance of Job class
     events.job = job             # save Bacula's job pointer
     job.set_events(events)       # register events desired
     sys.stderr = events          # send error output to Bacula
     sys.stdout = events          # send stdout to Bacula
     jobid = job.JobId; client = job.Client
     numvols = job.NumVols
     job.JobReport="Python Dir JobStart: JobId=%d Client=%s NumVols=%d\n" % (jobid,client,numvols)

  # Bacula Job is going to terminate
  def JobEnd(self, job):
     jobid = job.JobId
     client = job.Client
     job.JobReport="Python Dir JobEnd output: JobId=%d Client=%s.\n" % (jobid, client)

  # Called here when the Bacula daemon is going to exit
  def Exit(self, job):
      print "Daemon exiting."

bacula.set_events(BaculaEvents()) # register daemon events desired

"""
  These are the Job events that you can receive.
"""
class JobEvents(object):
  def __init__(self):
     # Called here when you instantiate the Job. Not
     # normally used
     noop = 1

  def JobInit(self, job):
     # Called when the job is first scheduled
     noop = 1

  def JobRun(self, job):
     # Called just before running the job after initializing
     #  This is the point to change most Job parameters.
     #  It is equivalent to the JobRunBefore point.
     noop = 1

  def NewVolume(self, job):
     # Called when Bacula wants a new Volume name. The Volume
     #  name returned, if any, must be stored in job.VolumeName
     jobid = job.JobId
     client = job.Client
     numvol = job.NumVols;
     print job.CatalogRes
     job.JobReport = "JobId=%d Client=%s NumVols=%d" % (jobid, client, numvol)
     job.JobReport="Python before New Volume set for Job.\n"
```

```
      Vol = "TestA-%d" % numvol
      job.JobReport = "Exists=%d TestA-%d" % (job.DoesVolumeExist(Vol), numvol)
      job.VolumeName="TestA-%d" % numvol
      job.JobReport="Python after New Volume set for Job.\n"
      return 1

   def VolumePurged(self, job):
      # Called when a Volume is purged. The Volume name can be referenced
      #  with job.VolumeName
      noop = 1
```

Chapter 2

Variable Expansion

Please note that as of version 1.37, the Variable Expansion is deprecated and replaced by Python scripting (not yet documented).

Variable expansion is somewhat similar to Unix shell variable expansion. Currently (version 1.31), it is used only in format labels, but in the future, it will most likely be used in more places.

2.1 General Functionality

This is basically a string expansion capability that permits referencing variables, indexing arrays, conditional replacement of variables, case conversion, substring selection, regular expression matching and replacement, character class replacement, padding strings, repeated expansion in a user controlled loop, support of arithmetic expressions in the loop start, step and end conditions, and recursive expansion.

When using variable expansion characters in a Volume Label Format record, the format should always be enclosed in double quotes (**"**).

For example, **${HOME}** will be replaced by your home directory as defined in the environment. If you have defined the variable **xxx** to be **Test**, then the reference **${xxx:p/7/Y/r}** will right pad the contents of **xxx** to a length of seven characters filling with the character **Y** giving **YYYTest**.

2.2 Bacula Variables

Within Bacula, there are three main classes of variables with some minor variations within the classes. The classes are:

Counters Counters are defined by the **Counter** resources in the Director's conf file. The counter can either be a temporary counter that lasts for the duration of Bacula's execution, or it can be a variable that is stored in the catalog, and thus retains its value from one Bacula execution to another. Counter variables may be incremented by postfixing a plus sign (+ after the variable name).

Internal Variables Internal variables are read-only, and may be related to the current job (i.e. Job name), or maybe special variables such as the date and time. The following variables are available:

Year – the full year

Month – the current month 1-12

Day – the day of the month 1-31

Hour – the hour 0-24

Minute – the current minute 0-59

Second – the current second 0-59

WeekDay – the current day of the week 0-6 with 0 being Sunday

Job – the job name

general – the Director's name

Level – the Job Level

Type – the Job type

JobId – the JobId

JobName – the unique job name composed of Job and date

Storage – the Storage daemon's name

Client – the Client's name

NumVols – the current number of Volumes in the Pool

Pool – the Pool name

Catalog – the Catalog name

MediaType – the Media Type

Environment Variables Environment variables are read-only, and must be defined in the environment prior to executing Bacula. Environment variables may be either scalar or an array, where the elements of the array are referenced by subscripting the variable name (e.g. **${Months[3]}**). Environment variable arrays are defined by separating the elements with a vertical bar (—), thus **set Months="Jan—Feb—Mar—Apr—..."** defines an environment variable named **Month** that will be treated as an array, and the reference **${Months[3]}** will yield **Mar**. The elements of the array can have differing lengths.

2.3 Full Syntax

Since the syntax is quite extensive, below, you will find the pseudo BNF. The special characters have the following meaning:

```
::=     definition
( )     grouping if the parens are not quoted
|       separates alternatives
'/'     literal / (or any other character)
CAPS    a character or character sequence
*       preceding item can be repeated zero or more times
?       preceding item can appear zero or one time
+       preceding item must appear one or more times
```

And the pseudo BNF describing the syntax is:

```
input       ::= ( TEXT
                | variable
                | INDEX_OPEN input INDEX_CLOSE (loop_limits)?
                )*
variable    ::= DELIM_INIT (name|expression)
name        ::= (NAME_CHARS)+
expression  ::= DELIM_OPEN
                (name|variable)+
                (INDEX_OPEN num_exp INDEX_CLOSE)?
                (':' command)*
                DELIM_CLOSE
command     ::= '-' (TEXT_EXP|variable)+
              | '+' (TEXT_EXP|variable)+
              | 'o' NUMBER ('-'|',') (NUMBER)?
              | '#'
              | '*' (TEXT_EXP|variable)+
              | 's' '/' (TEXT_PATTERN)+
                    '/' (variable|TEXT_SUBST)*
                    '/' ('m'|'g'|'i'|'t')*
```

```
             | 'y' '/' (variable|TEXT_SUBST)+
                   '/' (variable|TEXT_SUBST)*
                   '/'
             | 'p' '/' NUMBER
                   '/' (variable|TEXT_SUBST)*
                   '/' ('r'|'l'|'c')
             | '%' (name|variable)+
                   ('(' (TEXT_ARGS)? ')')?
             | 'l'
             | 'u'
num_exp     ::= operand
             | operand ('+'|'-'|'*'|'/'|'%') num_exp
operand     ::= ('+'|'-')? NUMBER
             | INDEX_MARK
             | '(' num_exp ')'
             | variable
loop_limits ::= DELIM_OPEN
               (num_exp)? ',' (num_exp)? (',' (num_exp)?)?
               DELIM_CLOSE
NUMBER      ::= ('0'|...|'9')+
TEXT_PATTERN::= (^('/'))+
TEXT_SUBST  ::= (^(DELIM_INIT|'/'))+
TEXT_ARGS   ::= (^(DELIM_INIT|')'))+
TEXT_EXP    ::= (^(DELIM_INIT|DELIM_CLOSE|':'|'+'))+
TEXT        ::= (^(DELIM_INIT|INDEX_OPEN|INDEX_CLOSE))+
DELIM_INIT  ::= '$'
DELIM_OPEN  ::= '{'
DELIM_CLOSE ::= '}'
INDEX_OPEN  ::= '['
INDEX_CLOSE ::= ']'
INDEX_MARK  ::= '#'
NAME_CHARS  ::= 'a'|...|'z'|'A'|...|'Z'|'0'|...|'9'
```

2.4 Semantics

The items listed in **command** above, which always follow a colon (:) have the following meanings:

```
-    perform substitution if variable is empty
+    perform substitution if variable is not empty
o    cut out substring of the variable value
#    length of the variable value
*    substitute empty string if the variable value is not empty,
     otherwise substitute the trailing parameter
s    regular expression search and replace. The trailing
     options are: m = multiline, i = case insensitive,
                  g = global,     t = plain text (no regexp)
y    transpose characters from class A to class B
p    pad variable to l = left, r = right or c = center,
     with second value.
%    special function call (none implemented)
l    lower case the variable value
u    upper case the variable value
```

The **loop_limits** are start, step, and end values.

A counter variable name followed immediately by a plus (+) will cause the counter to be incremented by one.

2.5 Examples

To create an ISO date:

```
DLT-${Year}-${Month:p/2/0/r}-${Day:p/2/0/r}
```

on 20 June 2003 would give **DLT-2003-06-20**

If you set the environment variable **mon** to

```
January|February|March|April|May|...
File-${mon[${Month}]}/${Day}/${Year}
```

on the first of March would give **File-March/1/2003**

Chapter 3

Using Stunnel to Encrypt Communications

Prior to version 1.37, Bacula did not have built-in communications encryption. Please see the **TLS** chapter (chapter 38 on page 339) of the Bacula Community Main Manual if you are using Bacula 1.37 or greater.

Without too much effort, it is possible to encrypt the communications between any of the daemons. This chapter will show you how to use **stunnel** to encrypt communications to your client programs. We assume the Director and the Storage daemon are running on one machine that will be called **server** and the Client or File daemon is running on a different machine called **client**. Although the details may be slightly different, the same principles apply whether you are encrypting between Unix, Linux, or Win32 machines. This example was developed between two Linux machines running stunnel version 4.04-4 on a Red Hat Enterprise 3.0 system.

3.1 Communications Ports Used

First, you must know that with the standard Bacula configuration, the Director will contact the File daemon on port 9102. The File daemon then contacts the Storage daemon using the address and port parameters supplied by the Director. The standard port used will be 9103. This is the typical server/client view of the world, the File daemon is a server to the Director (i.e. listens for the Director to contact it), and the Storage daemon is a server to the File daemon.

3.2 Encryption

The encryption is accomplished between the Director and the File daemon by using an stunnel on the Director's machine (server) to encrypt the data and to contact an stunnel on the File daemon's machine (client), which decrypts the data and passes it to the client.

Between the File daemon and the Storage daemon, we use an stunnel on the File daemon's machine to encrypt the data and another stunnel on the Storage daemon's machine to decrypt the data.

As a consequence, there are actually four copies of stunnel running, two on the server and two on the client. This may sound a bit complicated, but it really isn't. To accomplish this, we will need to construct four separate conf files for stunnel, and we will need to make some minor modifications to the Director's conf file. None of the other conf files need to be changed.

3.3 A Picture

Since pictures usually help a lot, here is an overview of what we will be doing. Don't worry about all the details of the port numbers and such for the moment.

```
File daemon (client):
               stunnel-fd1.conf
                 |===========|
Port 29102  >----| Stunnel 1 |-----> Port 9102
                 |===========|
               stunnel-fd2.conf
                 |===========|
Port 9103   >----| Stunnel 2 |-----> server:29103
                 |===========|
Director (server):
               stunnel-dir.conf
                 |===========|
Port 29102  >----| Stunnel 3 |-----> client:29102
                 |===========|
               stunnel-sd.conf
                 |===========|
Port 29103  >----| Stunnel 4 |-----> 9103
                 |===========|
```

3.4 Certificates

In order for stunnel to function as a server, which it does in our diagram for Stunnel 1 and Stunnel 4, you must have a certificate and the key. It is possible to keep the two in separate files, but normally, you keep them in one single .pem file. You may create this certificate yourself in which case, it will be self-signed, or you may have it signed by a CA.

If you want your clients to verify that the server is in fact valid (Stunnel 2 and Stunnel 3), you will need to have the server certificates signed by a CA (Certificate Authority), and you will need to have the CA's public certificate (contains the CA's public key).

Having a CA signed certificate is **highly** recommended if you are using your client across the Internet, otherwise you are exposed to the man in the middle attack and hence loss of your data.

See below for how to create a self-signed certificate.

3.5 Securing the Data Channel

To simplify things a bit, let's for the moment consider only the data channel. That is the connection between the File daemon and the Storage daemon, which takes place on port 9103. In fact, in a minimalist solution, this is the only connection that needs to be encrypted, because it is the one that transports your data. The connection between the Director and the File daemon is simply a control channel used to start the job and get the job status.

Normally the File daemon will contact the Storage daemon on port 9103 (supplied by the Director), so we need an stunnel that listens on port 9103 on the File daemon's machine, encrypts the data and sends it to the Storage daemon. This is depicted by Stunnel 2 above. Note that this stunnel is listening on port 9103 and sending to server:29103. We use port 29103 on the server because if we would send the data to port 9103, it would go directly to the Storage daemon, which doesn't understand encrypted data. On the server machine, we run Stunnel 4, which listens on port 29103, decrypts the data and sends it to the Storage daemon, which is listening on port 9103.

3.6 Data Channel Configuration

The Storage resource of the bacula-dir.conf normally looks something like the following:

```
Storage {
  Name = File
  Address = server
  SDPort = 9103
  Password = storage_password
  Device = File
  Media Type = File
}
```

Notice that this is running on the server machine, and it points the File daemon back to server:9103, which is where our Storage daemon is listening. We modify this to be:

```
Storage {
  Name = File
  Address = localhost
  SDPort = 9103
  Password = storage_password
  Device = File
  Media Type = File
}
```

This causes the File daemon to send the data to the stunnel running on localhost (the client machine). We could have used client as the address as well.

3.7 Stunnel Configuration for the Data Channel

In the diagram above, we see above Stunnel 2 that we use stunnel-fd2.conf on the client. A pretty much minimal config file would look like the following:

```
client = yes
[29103]
accept = localhost:9103
connect = server:29103
```

The above config file does encrypt the data but it does not require a certificate, so it is subject to the man in the middle attack. The file I actually used, stunnel-fd2.conf, looked like this:

```
#
# Stunnel conf for Bacula client -> SD
#
pid = /home/kern/bacula/bin/working/stunnel.pid
#
# A cert is not mandatory here. If verify=2, a
#  cert signed by a CA must be specified, and
#  either CAfile or CApath must point to the CA's
#  cert
#
cert = /home/kern/stunnel/stunnel.pem
CAfile = /home/kern/ssl/cacert.pem
verify = 2
client = yes
# debug = 7
# foreground = yes
[29103]
accept = localhost:9103
connect = server:29103
```

You will notice that I specified a pid file location because I ran stunnel under my own userid so I could not use the default, which requires root permission. I also specified a certificate that I have as well as verify level 2 so that the certificate is required and verified, and I must supply the location of the CA (Certificate Authority) certificate so that the stunnel certificate can be verified. Finally, you will see that there are two lines commented out, which when enabled, produce a lot of nice debug info in the command window.

If you do not have a signed certificate (stunnel.pem), you need to delete the cert, CAfile, and verify lines.

Note that the stunnel.pem, is actually a private key and a certificate in a single file. These two can be kept and specified individually, but keeping them in one file is more convenient.

The config file, stunnel-sd.conf, needed for Stunnel 4 on the server machine is:

```
#
# Bacula stunnel conf for Storage daemon
#
pid = /home/kern/bacula/bin/working/stunnel.pid
#
# A cert is mandatory here, it may be self signed
#  If it is self signed, the client may not use
#  verify
#
cert   = /home/kern/stunnel/stunnel.pem
client = no
# debug = 7
# foreground = yes
[29103]
accept = 29103
connect = 9103
```

3.8 Starting and Testing the Data Encryption

It will most likely be the simplest to implement the Data Channel encryption in the following order:

- Setup and run Bacula backing up some data on your client machine without encryption.
- Stop Bacula.
- Modify the Storage resource in the Director's conf file.
- Start Bacula
- Start stunnel on the server with:

  ```
  stunnel stunnel-sd.conf
  ```

- Start stunnel on the client with:

  ```
  stunnel stunnel-fd2.conf
  ```

- Run a job.
- If it doesn't work, turn debug on in both stunnel conf files, restart the stunnels, rerun the job, repeat until it works.

3.9 Encrypting the Control Channel

The Job control channel is between the Director and the File daemon, and as mentioned above, it is not really necessary to encrypt, but it is good practice to encrypt it as well. The two stunnels that are used in this case will be Stunnel 1 and Stunnel 3 in the diagram above. Stunnel 3 on the server might normally listen on port 9102, but if you have a local File daemon, this will not work, so we make it listen on port 29102. It then sends the data to client:29102. Again we use port 29102 so that the stunnel on the client machine can decrypt the data before passing it on to port 9102 where the File daemon is listening.

3.10 Control Channel Configuration

We need to modify the standard Client resource, which would normally look something like:

```
Client {
  Name = client-fd
  Address = client
  FDPort = 9102
  Catalog = BackupDB
  Password = "xxx"
}
```

to be:

```
Client {
  Name = client-fd
  Address = localhost
  FDPort = 29102
  Catalog = BackupDB
  Password = "xxx"
}
```

This will cause the Director to send the control information to localhost:29102 instead of directly to the client.

3.11 Stunnel Configuration for the Control Channel

The stunnel config file, stunnel-dir.conf, for the Director's machine would look like the following:

```
#
# Bacula stunnel conf for the Directory to contact a client
#
pid = /home/kern/bacula/bin/working/stunnel.pid
#
# A cert is not mandatory here. If verify=2, a
#  cert signed by a CA must be specified, and
#  either CAfile or CApath must point to the CA's
#  cert
#
cert   = /home/kern/stunnel/stunnel.pem
CAfile = /home/kern/ssl/cacert.pem
verify = 2
client = yes
# debug = 7
# foreground = yes
[29102]
accept = localhost:29102
connect = client:29102
```

and the config file, stunnel-fd1.conf, needed to run stunnel on the Client would be:

```
#
# Bacula stunnel conf for the Directory to contact a client
#
```

```
pid = /home/kern/bacula/bin/working/stunnel.pid
#
# A cert is not mandatory here. If verify=2, a
#  cert signed by a CA must be specified, and
#  either CAfile or CApath must point to the CA's
#  cert
#
cert   = /home/kern/stunnel/stunnel.pem
CAfile = /home/kern/ssl/cacert.pem
verify = 2
client = yes
# debug = 7
# foreground = yes
[29102]
accept = localhost:29102
connect = client:29102
```

3.12 Starting and Testing the Control Channel

It will most likely be the simplest to implement the Control Channel encryption in the following order:

- Stop Bacula.
- Modify the Client resource in the Director's conf file.
- Start Bacula
- Start stunnel on the server with:

```
stunnel stunnel-dir.conf
```

- Start stunnel on the client with:

```
stunnel stunnel-fd1.conf
```

- Run a job.
- If it doesn't work, turn debug on in both stunnel conf files, restart the stunnels, rerun the job, repeat until it works.

3.13 Using stunnel to Encrypt to a Second Client

On the client machine, you can just duplicate the setup that you have on the first client file for file and it should work fine.

In the bacula-dir.conf file, you will want to create a second client pretty much identical to how you did for the first one, but the port number must be unique. We previously used:

```
Client {
  Name = client-fd
  Address = localhost
  FDPort = 29102
  Catalog = BackupDB
  Password = "xxx"
}
```

so for the second client, we will, of course, have a different name, and we will also need a different port. Remember that we used port 29103 for the Storage daemon, so for the second client, we can use port 29104, and the Client resource would look like:

```
Client {
  Name = client2-fd
  Address = localhost
  FDPort = 29104
  Catalog = BackupDB
  Password = "yyy"
}
```

Now, fortunately, we do not need a third stunnel to on the Director's machine, we can just add the new port to the config file, stunnel-dir.conf, to make:

```
#
# Bacula stunnel conf for the Directory to contact a client
#
pid = /home/kern/bacula/bin/working/stunnel.pid
#
# A cert is not mandatory here. If verify=2, a
#  cert signed by a CA must be specified, and
#  either CAfile or CApath must point to the CA's
#  cert
#
cert   = /home/kern/stunnel/stunnel.pem
CAfile = /home/kern/ssl/cacert.pem
verify = 2
client = yes
# debug = 7
# foreground = yes
[29102]
accept = localhost:29102
connect = client:29102
[29104]
accept = localhost:29102
connect = client2:29102
```

There are no changes necessary to the Storage daemon or the other stunnel so that this new client can talk to our Storage daemon.

3.14 Creating a Self-signed Certificate

You may create a self-signed certificate for use with stunnel that will permit you to make it function, but will not allow certificate validation. The .pem file containing both the certificate and the key can be made with the following, which I put in a file named **makepem**:

```
#!/bin/sh
#
# Simple shell script to make a .pem file that can be used
# with stunnel and Bacula
#
OPENSSL=openssl
   umask 77
   PEM1="/bin/mktemp openssl.XXXXXX"
   PEM2="/bin/mktemp openssl.XXXXXX"
   ${OPENSSL} req -newkey rsa:1024 -keyout $PEM1 -nodes \
       -x509 -days 365 -out $PEM2
   cat $PEM1 > stunnel.pem
   echo ""   >>stunnel.pem
   cat $PEM2 >>stunnel.pem
   rm $PEM1 $PEM2
```

The above script will ask you a number of questions. You may simply answer each of them by entering a return, or if you wish you may enter your own data.

3.15 Getting a CA Signed Certificate

The process of getting a certificate that is signed by a CA is quite a bit more complicated. You can purchase one from quite a number of PKI vendors, but that is not at all necessary for use with Bacula.

To get a CA signed certificate, you will either need to find a friend that has setup his own CA or to become a CA yourself, and thus you can sign all your own certificates. The book OpenSSL by John Viega, Matt Mesier & Pravir Chandra from O'Reilly explains how to do it, or you can read the documentation provided in the Open-source PKI Book project at Source Forge: http://ospkibook.sourceforge.net/docs/OSPKI-2.4.7/OSPKI-html/ospki-book.htm . Note, this link may change.

3.16 Using ssh to Secure the Communications

Please see the script **ssh-tunnel.sh** in the **examples** directory. It was contributed by Stephan Holl.

Chapter 4

Bacula Projects

Once a new major version of Bacula is released, the Bacula users will vote on a list of new features. This vote is used as the main element determining what new features will be implemented for the next version. Generally, the development time for a new release is between four to nine months. Sometimes it may be a bit longer, but in that case, there will be a number of bug fix updates to the currently released version.

For the current list of project, please see the projects page in the CVS at: http://cvs.sourceforge.net/viewcvs.py/*checkout*/bacula/bacula/projects see the **projects** file in the main source directory. The projects file is updated approximately once every six months.

Separately from the project list, Kern maintains a current list of tasks as well as ideas, feature requests, and occasionally design notes. This list is updated roughly weekly (sometimes more often). For a current list of tasks you can see **kernstodo** in the Source Forge CVS at http://cvs.sourceforge.net/viewcvs.py/*checkout*/bacula/bacula/kernstodo .

Chapter 5

The internal database is not supported, please do not use it.

5.1 Internal Bacula Database

Previously it was intended to be used primarily by Bacula developers for testing; although SQLite is also a good choice for this. We do not recommend its use in general.

This database is simplistic in that it consists entirely of Bacula's internal structures appended sequentially to a file. Consequently, it is in most cases inappropriate for sites with many clients or systems with large numbers of files, or long-term production environments.

Below, you will find a table comparing the features available with SQLite and MySQL and with the internal Bacula database. At the current time, you cannot dynamically switch from one to the other, but must rebuild the Bacula source code. If you wish to experiment with both, it is possible to build both versions of Bacula and install them into separate directories.

Feature	SQLite or MySQL	Bacula
Job Record	Yes	Yes
Media Record	Yes	Yes
FileName Record	Yes	No
File Record	Yes	No
FileSet Record	Yes	Yes
Pool Record	Yes	Yes
Client Record	Yes	Yes
JobMedia Record	Yes	Yes
List Job Records	Yes	Yes
List Media Records	Yes	Yes
List Pool Records	Yes	Yes
List JobMedia Records	Yes	Yes
Delete Pool Record	Yes	Yes
Delete Media Record	Yes	Yes
Update Pool Record	Yes	Yes

Implement Verify	Yes	No
MD5 Signatures	Yes	No

In addition, since there is no SQL available, the Console commands: **sqlquery**, **query**, **retention**, and any other command that directly uses SQL are not available with the Internal database.

Chapter 6

Bacula Copyright, Trademark, and Licenses

There are a number of different licenses that are used in Bacula. If you have a printed copy of this manual, the details of each of the licenses referred to in this chapter can be found in the online version of the manual at http://www.bacula.org .

6.1 CC-BY-SA

The Creative Commons Attribution-ShareAlike 4.0 International License (CC-BY-SA) is used for this manual, which is a free and open license. Though there are certain restrictions that come with this license you may in general freely reproduce it and even make changes to it. However, rather than distribute your own version of this manual, we would much prefer if you would send any corrections or changes to the Bacula project.

The most recent version of the manual can always be found online at http://www.bacula.org .

6.2 GPL

The vast bulk of the source code is released under the Affero GNU General Public License version 3..

Most of this code is copyrighted: Copyright ©2000-2016, Kern Sibbald.

Portions may be copyrighted by other people. These files are released under different licenses which are compatible with the Bacula AGPLv3 license.

6.3 LGPL

Some of the Bacula library source code is released under the GNU Lesser General Public License. This permits third parties to use these parts of our code in their proprietary programs to interface to Bacula.

6.4 Public Domain

Some of the Bacula code, or code that Bacula references, has been released to the public domain. E.g. md5.c, SQLite.

6.5 Trademark

Bacula® is a registered trademark of Kern Sibbald.

6.6 Fiduciary License Agreement

Developers who have contributed significant changes to the Bacula code should have signed a Fiduciary License Agreement (FLA), which guarantees them the right to use the code they have developed, and also ensures that the Free Software Foundation Europe (and thus the Bacula project) has the rights to the code. This Fiduciary License Agreement is found on the Bacula web site at:

http://www.bacula.org/en/FLA-bacula.en.pdf

and if you are submitting code, you should fill it out then sent to:

Kern Sibbald
Cotes-de-Montmoiret 9
1012 Lausanne
Switzerland

When you send in such a complete document, please notify me: kern at sibbald dot com.

6.7 Disclaimer

NO WARRANTY

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

6.8 Authors

The following people below have contributed to making this document what it is today:

Alexandre Baron ¡balexfr at users dot sourceforge dot net¿ Arno Lehmann ¡arnol at users dot sourceforge dot net¿ Bastian Friedrich ¡bastian dot friedrich at collax dot com¿ Christopher S dot Hull ¡csh at raidersolutions dot com¿ Dan Langille Davide Franco ¡dfranco at dflc dot ch¿ Dirk H Bartley ¡dbartley at schupan dot com¿ Eric Bollengier ¡eric.bollengier at baculasystems dot com¿ Frank Sweetser James Harper bendigoit dot com dot au¿ Jeremy C dot Reed ¡jeremy-c-reed at users dot sourceforge dot net¿ Jose Herrera ¡herrerajs at yahoo dot com¿ Jo Simoens Juan Luis Francis ¡indpnday at users dot sourceforge dot net¿ Karl Cunningham ¡karlec at users dot sourceforge dot net¿ Kern Sibbald ¡kern at sibbald dot com¿ Landon Fuller ¡landonf at opendarwin dot org¿ Lucas Di Pentima Ludovic Strappazon Meno Abels Nicolas Boichat Peter Buschman Philippe Chauvat ¡philippe.chauvat at baculasystems dot com¿ Philipp Storz Richard Mortimer ¡richm at oldelvet dot org dot uk¿ Robert Nelson ¡robertn at the-nelsons dot org¿ Scott Barninger Sebastien Guilbaud Thomas Glatthor Thomas Mueller ¡thomas at chaschperli dot ch¿ Thorsten Engel ¡thorsten dot engel at matrix-computer dot com¿ Victor Hugo dos Santos ¡victorhugops at users dot sourceforge dot net

Creative Commons Attribution-ShareAlike 4.0 International

```
    Attribution.

        If You Share the Licensed Material (including in modified form), You must:
            retain the following if it is supplied by the Licensor with the Licensed Material:
                identification of the creator(s) of the Licensed Material and any others designated
                a copyright notice;
                a notice that refers to this Public License;
                a notice that refers to the disclaimer of warranties;
                a URI or hyperlink to the Licensed Material to the extent reasonably practicable;
            indicate if You modified the Licensed Material and retain an indication of any previous
            indicate the Licensed Material is licensed under this Public License, and include the te
        You may satisfy the conditions in Section 3(a)(1) in any reasonable manner based on the medi
        If requested by the Licensor, You must remove any of the information required by Section 3(a
    ShareAlike.

    In addition to the conditions in Section 3(a), if You Share Adapted Material You produce, the fo
        The Adapters License You apply must be a Creative Commons license with the same License Elem
        You must include the text of, or the URI or hyperlink to, the Adapter's License You apply. Y
        You may not offer or impose any additional or different terms or conditions on, or apply any

Section 4  Sui Generis Database Rights.

Where the Licensed Rights include Sui Generis Database Rights that apply to Your use of the Licensed

    for the avoidance of doubt, Section 2(a)(1) grants You the right to extract, reuse, reproduce, a
    if You include all or a substantial portion of the database contents in a database in which You
    You must comply with the conditions in Section 3(a) if You Share all or a substantial portion of

For the avoidance of doubt, this Section 4 supplements and does not replace Your obligations under t

Section 5  Disclaimer of Warranties and Limitation of Liability.

    Unless otherwise separately undertaken by the Licensor, to the extent possible, the Licensor off
    To the extent possible, in no event will the Licensor be liable to You on any legal theory (incl

    The disclaimer of warranties and limitation of liability provided above shall be interpreted in

Section 6  Term and Termination.

    This Public License applies for the term of the Copyright and Similar Rights licensed here. Howe

    Where Your right to use the Licensed Material has terminated under Section 6(a), it reinstates:
        automatically as of the date the violation is cured, provided it is cured within 30 days of
        upon express reinstatement by the Licensor.
    For the avoidance of doubt, this Section 6(b) does not affect any right the Licensor may have to
    For the avoidance of doubt, the Licensor may also offer the Licensed Material under separate ter
    Sections 1, 5, 6, 7, and 8 survive termination of this Public License.

Section 7  Other Terms and Conditions.

    The Licensor shall not be bound by any additional or different terms or conditions communicated
    Any arrangements, understandings, or agreements regarding the Licensed Material not stated herei

Section 8  Interpretation.

    For the avoidance of doubt, this Public License does not, and shall not be interpreted to, reduc
    To the extent possible, if any provision of this Public License is deemed unenforceable, it shal
    No term or condition of this Public License will be waived and no failure to comply consented to
    Nothing in this Public License constitutes or may be interpreted as a limitation upon, or waiver
```

Creative Commons is not a party to its public licenses. Notwithstanding, Creative Commons may elect

Creative Commons may be contacted at creativecommons.org.

Affero GNU General Public License

Version 3, 19 November 2007

The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, our General Public Licenses are intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License which gives you legal permission to copy, distribute and/or modify the software.

A secondary benefit of defending all users' freedom is that improvements made in alternate versions of the program, if they receive widespread use, become available for other developers to incorporate. Many developers of free software are heartened and encouraged by the resulting cooperation. However, in the case of software used on network servers, this result may fail to come about. The GNU General Public License permits making a modified version and letting the public access it on a server without ever releasing its source code to the public.

The GNU Affero General Public License is designed specifically to ensure that, in such cases, the modified source code becomes available to the community. It requires the operator of a network server to provide the source code of the modified version running there to the users of that server. Therefore, public use of a modified version, on a publicly accessible server, gives the public access to the source code of the modified version.

An older license, called the Affero General Public License and published by Affero, was designed to accomplish similar goals. This is a different license, not a version of the Affero GPL, but Affero has released a new version of the Affero GPL which permits relicensing under this license.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

   "This License" refers to version 3 of the GNU Affero General Public License.

   "Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

   "The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

   To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

   A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

   The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

   The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

   Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
   (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
   (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
   (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

   A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

   You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

   (a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
   (b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
   (c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

(d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

(e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

(a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

(b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

(c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

(d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

(e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

(f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

   You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

 A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

 A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

 Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

 In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

 If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

 If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

 A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

 Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

 If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Remote Network Interaction; Use with the GNU General Public License.

    Notwithstanding any other provision of this License, if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a computer network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software. This Corresponding Source shall include the Corresponding Source for any work covered by version 3 of the GNU General Public License that is incorporated pursuant to the following paragraph.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the work with which it is combined will remain governed by version 3 of the GNU General Public License.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU Affero General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU Affero General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU Affero General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future versions of the GNU Affero General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

    Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <textyear>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU Affero General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If your software can interact with users remotely through a computer network, you should also make sure that it provides a way for users to get its source. For example, if your program is a web application, its interface could display a "Source" link that leads users to an archive of the code. There are many ways you could offer source, and different solutions will be better for different programs; see section 13 for the specific requirements.

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU AGPL, see `http://www.gnu.org/licenses/`.

GNU Lesser General Public License

image of a Philosophical GNU [English — Japanese]

- Why you shouldn't use the Lesser GPL for your next library
- What to do if you see a possible LGPL violation
- Translations of the LGPL
- The GNU Lesser General Public License as a text file
- The GNU Lesser General Public License as a Texinfo file

This GNU Lesser General Public License counts as the successor of the GNU Library General Public License. For an explanation of why this change was necessary, read the Why you shouldn't use the Lesser GPL for your next library article.

6.9 Table of Contents

6.10 GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

```
Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA  02110-1301  USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
[This is the first released version of the Lesser GPL.  It also counts
 as the successor of the GNU Library Public License, version 2, hence
 the version number 2.1.]
```

6.11 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

6.12 TERMS AND CONDITIONS

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- **a)** The modified work must itself be a software library.
- **b)** You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- **c)** You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- **d)** If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

  (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

  These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

  Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- **a)** Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

- **b)** Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- **c)** Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- **d)** If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- **e)** Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- **a)** Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- **b)** Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License.

If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

6.13 How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
{\it one line to give the library's name and an idea of what it does.}
Copyright (C) {\it year}  {\it name of author}
This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.
This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Lesser General Public License for more details.
You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
USA
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in
the library "Frob" (a library for tweaking knobs) written
by James Random Hacker.
{\it signature of Ty Coon}, 1 April 1990
Ty Coon, President of Vice
```

That's all there is to it! Return to GNU's home page .

FSF & GNU inquiries & questions to gnu@gnu.org . Other ways to contact the FSF.

Comments on these web pages to webmasters@www.gnu.org , send other questions to gnu@gnu.org .

Updated: 27 Nov 2000 paulv