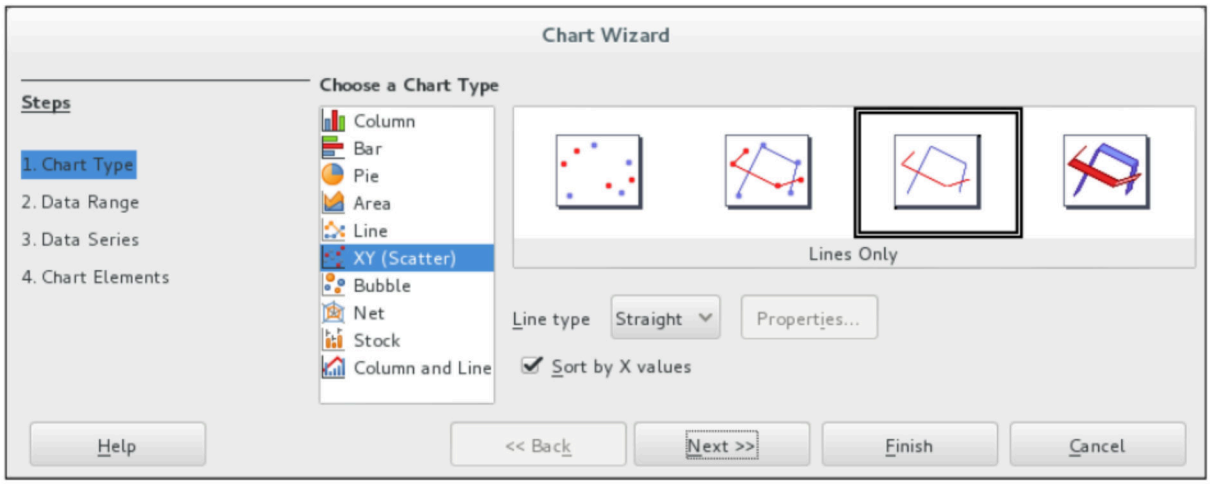
Each row in the file represents one unit of time, either minute or second, with total counts for that time period. The CSV file (for both per-minute and per-second statistics) contains the following columns, in order:

```
Time,time_t,Results,Search,Add,Mod,Modrdn,Delete,Abandon,Connections,SSL
Conns,Bind,Anon Bind,Unbind,Unindexed
```

The CSV file can be manipulated in any spreadsheet program, like LibreOffice Calc, and in many other business applications. The procedures for importing the CSV data and generating charts or other metrics depends on the application itself.

For example, to create a chart in LibreOffice Calc:

1. Open the CSV file.
2. Click the **Insert** menu, and select **Chart**.
3. In the **Chart Type** area, set the chart type to **XY (Scatter)**.
    1. Set the subtype to lines only.
    2. Select the option to sort by X values.



4. Accept the defaults in the other screens (particularly, to use the data series in columns and to set the first row and first column as labels), and create the chart.

19.5. REPLACING LOG FILES WITH A NAMED PIPE

The named pipe log script enables administrators to replace a log file with a named pipe to automatically process the log data. This provides advanced logging features, such as:

- Log only certain events, such as failed binds or connections from certain IP addresses.
- Log only lines that match a regular expression.
- Log only a defined number of lines.
- Send an email or other notification when a defined event is logged.

You can configure a named pipe for logging:

- For testing purposes, see Section 19.5.1, “Temporarily Replacing a Log File with a Named Pipe”

- For permanent usage, see Section 19.5.2, "Creating a New Named Pipe for Logging"

19.5.1. Temporarily Replacing a Log File with a Named Pipe

If you replace a log file with a named pipe, no server modifications are required. With this configuration, you cannot use log viewers, such as in the Admin Console, because they require to read the content a from a file.

To replace a log file with a named pipe:

1. Stop the Directory Server instance:

```
# systemctl stop dirsrv.target
```

2. Remove the log file. For example:

```
# rm -f /var/log/dirsrv/errors
```

3. Configure the named pipe to start with the Directory Server. For details, see Section 19.5.3, "Starting and Shutting Down the Named Pipe with the Directory Server Service".

4. Start the Directory Server instance:

```
# systemctl start dirsrv.target
```

IMPORTANT

When the log files are rotated, the named pipe is replaced with a regular file. Use this procedure only as a temporary solution. For a permanent solution, see Section 19.5.2, "Creating a New Named Pipe for Logging"

19.5.2. Creating a New Named Pipe for Logging

To log to a named pipe and additionally be able to use log viewers, such as in the Admin Console, configure the named pipe to use a different name than the name of your log file:

1. Configure the named pipe to start with Directory Server. To enable log viewers, additionally redirect the output of the **ds-logpipe.py** command to a file. For example:

```
# python /usr/bin/ds-logpipe.py ... >
/var/log/dirsrv/slapd-instance/errors &
```

For details, see Section 19.5.3, "Starting and Shutting Down the Named Pipe with the Directory Server Service".

2. Update the Directory Server configuration to log to the named pipe. For example, to send the access log to the **/var/log/dirsrv/slapd-instance/access.pipe** named pipe:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h
server.example.com -x
dn: cn=config
```

```
changetype: modify
replace: nsslapd-accesslog
nsslapd-accesslog: /var/log/dirsrv/slapd-instance/access.pipe
```

Optionally, you can also set:

- The ***nsslapd-errorlog*** parameter for error events.
- The ***nsslapd-auditlog*** parameter for audit events. Note that audit logging is disabled by default. To enable it, additionally set the ***nsslapd-accesslog-logging-enabled*** parameter to **on**.
- The ***nsslapd-auditfaillog*** parameter for failed audit events. Note that logging of failed audit events is disabled by default. To enable it, additionally set the ***nsslapd-accesslog-logging-enabled*** parameter to **on**.

NOTE

Updating the parameters takes effect immediately. However, you must start the named pipe manually or restart the Directory Server instance.

3. Disable buffering and log rotation for the event you configured the named pipe for. For example, to disable the features for the access log:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h
server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-accesslog-logbuffering
nsslapd-accesslog-logbuffering: off
-
replace: nsslapd-accesslog-maxlogsperdir
nsslapd-accesslog-maxlogsperdir: 1
-
replace: nsslapd-accesslog-logexpirationtime
nsslapd-accesslog-logexpirationtime: -1
-
replace: nsslapd-accesslog-logrotationtime
nsslapd-accesslog-logrotationtime: -1
```

To disable the parameters for the error log, update:

- ***nsslapd-errorlog-logbuffering***
- ***nsslapd-errorlog-maxlogsperdir***
- ***nsslapd-errorlog-logexpirationtime***
- ***nsslapd-errorlog-logrotationtime***

To disable the parameters for the audit log, update:

- ***nsslapd-auditlog-logbuffering***
- ***nsslapd-auditlog-maxlogsperdir***

- ***nsslapd-auditlog-logexpirationtime***
- ***nsslapd-auditlog-logrotationtime***

To disable the parameters for the audit faillog, update:

- ***nsslapd-auditfaillog-logbuffering***
- ***nsslapd-auditfaillog-maxlogsperdir***
- ***nsslapd-auditfaillog-logexpirationtime***
- ***nsslapd-auditfaillog-logrotationtime***

4. Restart the Directory Server instance to start the pipe.

```
# systemctl restart dirsrv.target
```

19.5.3. Starting and Shutting Down the Named Pipe with the Directory Server Service

To start and shut down the named pipe with the Directory Server instance:

1. Open the **/etc/sysconfig/dirsrv-*instance*** instance configuration file.

WARNING

Do not edit the **/etc/sysconfig/dirsrv** file.

2. Append the **ds-logpipe.py** commands at the end of the file. For example:

```
# Only keep the last 1000 lines of the error log and
# additionally redirect all log data to the
# /var/log/dirsrv/slapd-instance/errors file
python /usr/bin/ds-logpipe.py
/var/log/dirsrv/slapd-instance/errors.pipe -m 1000 -u dirsrv -s
/var/run/dirsrv/slapd-instance.pid >
/var/log/dirsrv/slapd-instance/errors &

# Only log failed binds
python /usr/bin/ds-logpipe.py
/var/log/dirsrv/slapd-instance/access.pipe -u dirsrv -s
/var/run/dirsrv/slapd-instanec.pid --
plugin=/usr/share/dirsrv/data/failedbinds.py
failedbinds.logfile=/var/log/dirsrv/slapd-instance/access.failedbind
s &
```

For details, see the ds-logpipe.py(1) man page.

IMPORTANT

Make sure that each named pipe command ends with an *&* sign to send the **ds-logpipe.py** process to the background.

19.5.4. Using Plug-ins with the Named Pipe Log

You can call a plug-in to read the log data from the named pipe to perform operations on the log data. When using plug-ins with the named pipe log script, consider the following:

- The plug-in function is called for every line read from the named pipe.
- The plug-in function must be a Python script and use the **.py** suffix.
- Any plug-in arguments are passed in the command line to the **ds-logpipe.py** named pipe log script.
- A **pre** operation function can be called for when the plug-in is loaded.
- A **post** operation function can be called for when the plug-in exits.

19.5.4.1. Loading Plug-ins with the Named Pipe Log Script

There are two options for the **ds-logpipe.py** command to use with plug-ins:

- The **--plugin** option gives the path to the plug-in file.
- The *plugin.arg* option passes plug-in arguments to the named pipe log script.
  - **plugin**: The file name without the **.py** suffix.
  - **arg**: Any argument allowed in the plug-in.

For example:

```
ds-logpipe.py /var/log/dirsrv/slapd-example/errors.pipe --
plugin=/usr/share/dirsrv/data/example-funct.py example-
funct.regex="warning" > /var/log/dirsrv/warnings.txt
```

If there are more than one value passed to the same argument, they are converted into a list of values in the plug-in. For example, this script sets two values for the **arg1** argument:

```
--plugin=/path/to/plugin_name.py plugin_name.arg1=example1
plugin_name.arg1=example2 plugin_name.arg2=demo
```

In the plug-in, this is converted to:

```
{'arg1': ['example1', 'example2'], 'arg2': 'demo'}
```

This is a Python **dictionary** object with two keys. The first key is the string **arg1**, and its value is a Python list object with two elements, the strings **foo** and **bar**. The second key is the string **arg2**, and its value is the string **baz**. If an argument has only a single value, it is left as a simple string. Multiple values for a single argument name are converted into a list of strings.

19.5.4.2. Writing Plug-ins to Use with the Named Pipe Log Script

The **ds-logpipe.py** command supports the following functions in a plug-in:

- **plugin()**: Mandatory. Code in this function is applied to every line of log data received.
- **pre()**: Optional. Code is run when the plug-in is started.
- **post()**: Optional. Code is run when the plug-in exits.

Each function can have any arguments defined for it, and these arguments can then be passed to the script using the *plugin.arg* option. Additionally, each function can have its own return values and actions defined for it.

Example 19.1. Simple Named Pipe Log Plug-in

```
def pre(myargs):
    retval = True
    myarg = myargs['argname']
    if isinstance(myarg, list): # handle list of values
    else: # handle single value
    if bad_problem:
        retval = False
    return retval

def plugin(line):
    retval = True
    # do something with line
    if something_is_bogus:
        retval = False
    return retval

def post(): # no arguments
    # do something
    # no return value
```

19.5.5. Troubleshooting the Named Pipe

19.5.5.1. Directory Server Hangs When Writing to the Named Pipe

If the **ds-logpipe.py** command terminates unexpectedly, the Directory Server hangs while writing to the named pipe. To fix the problem:

Restart the named pipe manually or if configured in the **/etc/sysconfig/dirsrv-*instance*** file, run:

```
# (. /etc/sysconfig/dirsrv-instance)
```

19.6. MONITORING THE LOCAL DISK FOR GRACEFUL SHUTDOWN

When the disk space available on a system becomes too small, the Directory Server process (**slapd**) crashes. Any abrupt shutdown runs the risk of corrupting the database or losing directory data.