"Expand: High Performance Storage System for HPC and Big Data Environments" (TED2021-131798B-I00)

High Performance Storage Sytems for HPC and Big Data (Expand)

D 2.2
Installation and user manual

Universidad Carlos III de Madrid

June, 2025

CONTENTS

1. DOWNLOAD AND INSTALLATION

The Expand Parallel File System repository is available at:

- Source code: `https://github.com/xpn-arcos/xpn`
- Documentation: `https://expand-arcos.github.io`

In order to deploy Expand, you have to follow the next steps:

1. **Install required software** (pre-requisites).
2. **Download and configure** Expand.
3. **Compile and install** Expand.

1.1. Install the required software

To install the required software, please follow the next steps:

1. Please ensure you have already installed the following development software:

   ```
   sudo apt-get install autoconf automake gcc g++ make build-essential
   ```

2. Install MPICH (at least version 3.4.2 or compatible version):

   ```
   sudo apt-get install mpich libmpich-dev mpich-doc
   ```

1.2. Download and configure Expand

1. Clone the last version of the Expand source code:

   ```
   git clone https://github.com/xpn-arcos/xpn.git
   ```

2. Access into the source code directory, which from now on we will denote as <xpn_path>:

   ```
   cd xpn
   ```

3. Build configure script:

   ```
   ACLOCAL_FLAGS="-I /usr/share/aclocal/" autoreconf -v -i -s -W all
   ```

4. Configure Expand, usually:

   ```
   ./configure --prefix=<install_path> --enable-mpi_server=<mpicc_path> --enable-sck_server
   ```

The `--prefix` is used to change the default directory in which Expand will be installed. This switch is optional, and it common used if your UNIX user do not have enough permissions on the default directory.
The `--enable-mpi_server` switch is used to enable the the MPI communication connector module of Expand. If MPICH has not been installed in its default installation path (e.g.: `/usr/bin`), it is necessary to specify the path in which it has been installed (`--enable-mpi_server=<mpich_path/bin>`).
The `--enable-sck_server` switch is used to enable the socket communication conector module of Expand.

1.3. Compile and install Expand

1. Compile Expand:

   ```
   make -j
   ```

2. Install Expand:

   ```
   make install
   ```

The key elements of Expand are now available in the following paths:

- Header files: `<install_path>/include/`
  - Expand client API: `<install_path>/include/xpn_client/xpn.h`
- Libraries: `<install_path>/lib/`
  - Expand client library: `<install_path>/lib/libxpn.a`
  - Expand system call interception library: `<xpn_path>/lib/xpn_bypass.so`
- Binaries: `<install_path>/bin/`
  - Expand server: `<install_path>/bin/xpn_server`

2. RUNNING EXPAND

The Expand File System is based on the client-server paradigm. We need to run the server first, then run the application with the Expand client(s).

2.1. Running the Expand server

To start the Expand server, the following steps must be followed:

1. Include the machines to be used in the <hostfile_server> file, one per line.
2. Start the Expand server(s) (with N processes distributed on the machines listed in the <hostfile_server> file) by running:

```
export WORK_DIR=<shared directory among hostfile computers, $HOME for example>
export NODE_DIR=<local directory to be used on each node, /tmp for example>

<xpn_path>/xpn -v \
               -n <N> \
               -l $WORK_DIR/hostfile_server \
               -w $WORK_DIR \
               -x $NODE_DIR \
               start
```

2.2. Running the Expand client

To start the Application with the Expand client(s), the following steps must be taken:

1. Create the configuration file for the Expand client (it there is not an existing one already).
   This file must include a partition section for each storage space the client use to store files and directories.
   For each partition section, user must provide the partition name (e.g.: `expand1`),
   the partition block size (e.g.: `512k`),
   and the Expand server nodes (e.g.: `mpi_server://<server>/<path>`) associated.

   An example of a configuration file for the Expand client is the following one:

```
[partition]
bsize = 512k
replication_level = 0       # 0 means RAID-0 at partition level
partition_name    = expand1 # partition name is expand1

# list of servers:
# * first one is a socket server,
server_url = sck_server://hostname_1/tmp/xpn/data
# * second one is a mqtt socket server,
server_url =  mq_server://hostname_2/tmp/xpn/data
# * and the third one is a MPI server
server_url = mpi_server://hostname_3/tmp/xpn/data
```

Where the label partition enables the definition of the structure of a storage partition. For each partition, we define each one of the servers used to form the partition. The servers are defined by a URL which is unique within a partition, not being able to use the same URL more than once per partition. The current prototype only allows the definition of one only partition.

2 Run the client binary (with M processes distributed on the machines in the <hostfile_client> file) by using:

```
export WORK_DIR=<shared directory among hostfile computers, $HOME for example>

mpiexec -np               <M> \
        -hostfile         $WORK_DIR/<hostfile_client> \
        -genv XPN_CONF    $WORK_DIR/xpn.conf \
        -genv LD_PRELOAD  <INSTALL_PATH>/xpn/lib/xpn_bypass.so:$LD_PRELOAD \
        <full path to client binary>/<client_binary>
```

2.a If the client binary use the Expand API directly then we do not need to intercept the POSIX system calls. In this case you can execute it by running:

```
export WORK_DIR=<shared directory among hostfile computers, $HOME for example>

mpiexec -np               <M> \
        -hostfile         $WORK_DIR/<hostfile_client> \
        -genv XPN_CONF    $WORK_DIR/xpn.conf \
        <full path to client binary>/<client_binary>
```

2.3. Expand name space to be used in the Expand client

By using the following configuration file:

```
[partition]
bsize = 512k
replication_level = 0       # 0 means RAID-0 at partition level
partition_name    = expand1 # partition name is expand1

# list of servers
server_url = sck_server://hostname_1/tmp/xpn/data
server_url =  mq_server://hostname_2/tmp/xpn/data
server_url = mpi_server://hostname_3/tmp/xpn/data
```

If the name used for a file in Expand is, for example:

`/tmp/expand/expand1/dirA/filename`

Then, this file will be distributed in the partition `expand1` using three servers: `hostname_1`, `hostname_2`, and `hostname_3`. In each server the subfile used for this Expand file is:

`/tmp/xpn/data/dirA/filename`

2.4. Running Expand server and Application from a SLURM job

An example of SLURM job might be:

```
#!/bin/bash

#SBATCH --job-name=test
#SBATCH --output=$HOME/results_%j.out
#SBATCH --nodes=8
#SBATCH --ntasks=8
#SBATCH --cpus-per-task=8
#SBATCH --time=00:05:00

export WORK_DIR=<shared directory among hostfile computers, $HOME for example>
export NODE_DIR=<local directory to be used on each node, /tmp for example>

############### Running the Expand server ###############

# 2.- to create a hostfile
scontrol show hostnames ${SLURM_JOB_NODELIST} > $WORK_DIR/hostfile

# 3.- to launch the Expand MPI servers
<xpn_path>/xpn -v \
               -w $WORK_DIR -x $NODE_DIR \
               -n <number of XPN processes> \
               -l $WORK_DIR/hostfile start
sleep 2

############### Running the Expand client ###############

```

```
# 2.- to launch the XPN client (app. that will use Expand)
mpiexec -np <number of client processes> \
        -hostfile $WORK_DIR/hostfile \
        -genv XPN_CONF    $WORK_DIR/xpn.conf \
        -genv LD_PRELOAD  <INSTALL_PATH>/xpn/lib/xpn_bypass.so:$LD_PRELOAD \
        <full path to the app.>

############### Stopping the Expand server ###############

<xpn_path>/xpn -v -d $WORK_DIR/hostfile stop
sleep 2
```

3. EXPAND INSTALLATION FOR BIG DATA SUPPORT

To deploy the Expand parallel file system with the Expand connector for Apache Spark, the following steps must be followed:

1. **Install required software** (pre-requisites).
2. **Download and configure** Expand.
3. **Compile and install** Expand.

3.1. Install the required software

In order to use the Expand connector for Apache Spark, the following software must be installed:

1. Please ensure you have already installed the following development software:

   ```
   sudo apt-get install build-essential libtool autoconf automake git gcc g++ make
   ```

2. Install MPICH (at least version 3.4.2 or compatible version):

   ```
   sudo apt-get install mpich libmpich-dev mpich-doc
   ```

3. Install mosquitto:

   ```
   sudo apt-get install mosquitto mosquitto-clients mosquitto-dev libmosquitto-dev
   ```

4. Java (version Java 8 or compatible)

   ```
   sudo apt-get install openjdk-8-jdk
   ```

5. Apache Maven (version 3.9.x or compatible)

   ```
   sudo apt install maven
   ```

3.2. Download and configure Expand

1. Clone the last version of the Expand source code:

   ```
   git clone https://github.com/xpn-arcos/xpn.git
   ```

2. Access into the installation directory, which from now on we will denote as <xpn_path>:

   ```
   cd xpn
   ```

3. Build configure script:

   ```
   ACLOCAL_FLAGS="-I /usr/share/aclocal/" autoreconf -v -i -s -W all
   ```

4. Configure Expand, usually:

```
./configure --prefix=<install_path> --enable-mpi_server=<mpicc_path> --enable-sck_server
↪ --enable-mosquitto
```

The `--prefix` changes the default directory in which Expand will be installed. This switch is optional, and it commonly used if your UNIX user does not have enough permissions on the default directory. The `--enable-mpiserver` switch is used to enable the MPI communication connector module of Expand. If MPICH has not been installed in its default installation path (e.g.: `/usr/bin`), it is necessary to specify the path in which it has been installed (`--enable-mpiserver=<mpich_path>`).
The `--enable-sck_server` switch is used to enable the socket communication connector module of Expand.
The `--enable-mosquitto` switch is used to enable the mosquitto-based socket communication connector module of Expand.

3.3. Compile and install Expand

1. Compile Expand:

   ```
   make -j
   ```

2. Install Expand:

   ```
   make install
   ```

3. Before compiling the Expand connector for Apache Spark, some of the `./src/makerun.sh` file must be modified:

   ```
   export JAVA_INC=<PATH WHERE THE jni.h HEADER IS LOCATED (Usually, the Java Library Path)>
   ```

4. To compile and obtain the necessary libraries for the connector, the following lines must be executed:

```
cd src/connector_spark/
./src/makerun.sh
mvn package
```

   This will generate both the dynamic library `libexpandtoposix.so` and the `jar` package containing the connector.

3.4. Connector usage

A script generator is provided to execute Spark applications using Expand. With this script generator, you can obtain an executable or a SLURM file to execute the Spark application with Expand using `sbatch`. This script will deploy the Expand servers, preload data, deploy Spark cluster, execute Spark application, flush Expand data, and stop Spark and Expand features if needed. You just need to modify the `./script_generator/Config.py` and the `./script_generator/nodes` files following the instructions. After that, you just need to execute:

```
python3 script_generator/launch.py
```

Otherwise, after deploying Expand and Spark features, you can execute Spark applications using Expand adding the following configuration options to the `spark-submit` executable:

1. To specify Spark the use of Expand, the following lines are mandatory:

```
--conf "spark.executorEnv.XPN_CONF=$XPN_CONF"
--conf "spark.executorEnv.PATH=$PATH"
--conf "spark.executorEnv.LD_LIBRARY_PATH=$LD_LIBRARY_PATH"
--conf "spark.hadoop.fs.defaultFS=xpn:///"
--conf "spark.hadoop.fs.xpn.impl=org.expand.hadoop.Expand"
```

   The environment variables must contain at least:

   - `XPN_CONF`: the path to the Expand configuration file.
   - `PATH`: the path to `mpiexec`.
   - `LD_LIBRARY_PATH`: the path to the MPI, Expand and the connector libraries; and the path to the `jar` package containing the connector.

2. To customize the Expand features:

```
--conf "spark.hadoop.xpn.block.size=<XPN BLOCKSIZE>"
--conf "spark.hadoop.xpn.file.buffer.size=<XPN BLOCKSIZE>"
--conf "spark.hadoop.xpn.block.replication=<XPN REPLICATION>"
```

   These features must contain the same values specified to the Expand servers. By default, the blocksize is set to 128 MiB and the replication to 1.

3. To ensure the correct Spark tasks generation the following lines are highly recommended:

```
--conf "spark.hadoop.mapreduce.input.fileinputformat.split.minsize=<XPN BLOCKSIZE>"
--conf "spark.hadoop.mapreduce.input.fileinputformat.split.maxsize=<XPN BLOCKSIZE>"
```

4. An example for a Wordcount application is shown below (note as the blocksize is 128 MiB and the replication is 1):

```
$SPARK_HOME/bin/spark-submit --master spark://$MASTER_NODE:7077 \
--conf "spark.executorEnv.XPN_CONF=$XPN_CONF" \
--conf "spark.executorEnv.PATH=$PATH" \
--conf "spark.executorEnv.LD_LIBRARY_PATH=$LD_LIBRARY_PATH" \
--conf "spark.hadoop.mapreduce.input.fileinputformat.split.minsize=134217728" \
--conf "spark.hadoop.mapreduce.input.fileinputformat.split.maxsize=134217728" \
--conf "spark.hadoop.xpn.block.size=134217728"
--conf "spark.hadoop.xpn.file.buffer.size=134217728"
--conf "spark.hadoop.xpn.block.replication=1"
--conf "spark.hadoop.fs.defaultFS=xpn:///" \
--conf "spark.hadoop.fs.xpn.impl=org.expand.hadoop.Expand" \
--class org.expand.tests.JavaWordCount \
$CONNECTOR_HOME/target/xpn-spark-1.0.jar \
xpn:///xpn/quixote xpn:///xpn/res/quixote-wc
```