Internet Backplane Protocol: Server Configuration Options

Alessandro Bassi

Department of Computer Science
University of Tennessee
Knoxville, TN 37996

ibp@cs.utk.edu

Abstract

In this document, we present a description of the IBP Server Configuration.

1 The IBP server

The IBP server manages access to a pool of storage areas that are created, accessed, and managed remotely through the IBP client interface. The IBP server performs no security checks on clients' requests, the only exception being when the configuration file explicitely specifies the clients that can allocate space[1]. A client which connects to the IBP server with the proper capability is granted access to the underlying functionality. The only "protected" operations are those performed through the **IBP_manage()** or the **IBP_status()** call. The first call requires the client to present the management capability that is returned as part of the **IBP_allocate()** call before the server can fulfill the client's request.

The current version of IBP supports two levels of reliable storage, with the client choosing the level of reliability of an allocated storage area.

- **Stable storage**: Allocating a storage area with this reliability level guarantees the permanent presence of the storage area until the read reference counter to it drops to zero, at which time the data therein is removedand the allocated storage is reclaimed by the IBP server.
- **Volatile storage**: This is total storage that can be used by depot. Some part of the volatile storage (Volatile Size - Stable Size), could be reclaimed by the IBP server at any time. The decision to reclaim part of volatile storage area could be dictated by site management policies or changes in storage requirements of local jobs.

In addition to the aforementioned reliability levels, IBP supports indefinite storage, where storage areas are only reclaimed through client requests or reliability-induced server actions, and time-limited storage, in which the storage area is reclaimed by the IBP server at a certain instance of time specified by a managing IBP client (a managing IBP client is a client that allocates a storage area, or acquires the management capability from the creating client). See description of the client **IBP_allocate()** and **IBP_manage()** calls for ways of controlling properties of storage areas.

2 The IBP Server v 1.2.0 Configuration

The IBP depot can be configured through the options set while calling the server, or through the configuration file **"ibp.cfg"** which should be located in the **/etc** directory (unless otherwise specified in the line options). The following table illustrates the call-time options:

[1] From version 1.2.0

option	description
-s *stable storage*	This option sets the amount of the stable storage to allocate
-v *vol storage*	This option sets the amount of the volatile storage to allocate
-ds *stable dir*	This option sets the path of stable storage directory
-dv *volatile dir*	This option sets the path of volatile storage directory
-df *FIFO dir*	This option sets the path of FIFO storage directory
-l *lifetime*	This option sets the max duration allowed
-p *port*	This option sets the port used
-pw *password*	This option sets the IBP depot password
-cf *cfg path*	This option sets the path to find the cfg file and to store log file
-c *cfg name*	This option sets the name of the config file
-hn *hostname*	the hostname used in the depots which have multiple host names
-tn *threads number*	the number of threads, only used in the depots that support thread
-tl	enable log files for every thread
-rt *recover time*	This option sets the time for server grace recovery
-nr	skip the recovery of the old capabilities

Note: the size of the volatile amount is expressed in megabytes unless otherwise specified. To specify another size base, a letter must be added to the size:**k** for kilobytes, **m** for megabytes and **g** for gigabytes. For instance, **-v 10g -s 60m** sets a volatile storage area of 10 gigabytes and a stable storage area of 60 megabytes. The same notation is used for the configuration file.

The **"ibp.cfg"** file contains a list of the form
Configuration parameter *space* Parameter value
with one entry per line, except with the **CL_IPv4** and **CL_IPv6** fields, which have the following format:
Configuration parameter *space* IP address *space* IP mask

The following table lists the currently supported configuration parameters, their names, types, and default values.

Parameter name (case sensitive)	Description	Type	Default value	Type	Default value
SOFT/VOLSIZE	Total Size of available storage storage areas	ulong_t	0		
VOLDIR	Directory where volatile storage areas are to be stored (absolute path)	String	/tmp/		
HARD/STABLESIZE	Size of available storage for stable storage areas	ulong_t	0		
STABLEDIR	Directory where stable storage areas are to be stored (absolute path)	String	/tmp/		
MINFREESIZE	The minimal size of free storage for the disk	ulong_t	150M		
FIFODIR	Directory where storage areas of type **IBP_FIFO** are stored.	String	/tmp/		
CFGPORT	Port that must be used by clients to connect to the IBP server	Int	6714 (IANA approved)		
MAXDURATION	The maximum possible duration (in days) for a newly allocated storage area.	Float	-1 (No limit on storage area lifetime)		
PASSWORD	The IBP server password	String	ibp		
THREADS	The number of the threads	Int	16		
CL_IPv4	The Client with this IPv4 address has allocation permission	String IP address	NULL	String IP mask	NULL
CL_IPv6	The Client with this IPv6 address has allocation permission	String IP address	NULL	String IP mask	NULL

It is important to notice that if any **CL_IPv4** or **CL_IPv6** client are specified, they are the only clients allowed to allocate space on the IBP depot. If no client is specified, then all clients have allocate permission. Default values are used if the corresponding parameter is not specified in the **"ibp.cfg"** configuration file (or if the file does not exist).

3 IBP server blocking rules

The IBP server processes requests on a First Come First Serve basis. Due to the fact that all write operations to storage areas have append semantics, a maximum of one write operation can be actively accessing a storage area at any given time. There is no limit on the number of read processes (for storage areas of type *IBP_BYTEARRAY*), and an upper limit of one read process for storage areas of type *IBP_FIFO* and *IBP_CIRQ*. If a write request is received while another one is active to the same storage area, the new request is queued pending completion of the existing request (the same applies to multiple read requests to storage areas of type *IBP_FIFO* and *IBP_CIRQ*).