POHMELFS network protocol.

Basic structure used in network communication is following command:

```
struct netfs_cmd
{
	__u16			cmd;	/* Command number */
	__u16			csize;	/* Attached crypto information size */
	__u16			cpad;	/* Attached padding size */
	__u16			ext;	/* External flags */
	__u32			size;	/* Size of the attached data */
	__u32			trans;	/* Transaction id */
	__u64			id;	/* Object ID to operate on. Used for
feedback.*/
	__u64			start;	/* Start of the object. */
	__u64			iv;	/* IV sequence */
	__u8			data[0];
};
```

Commands can be embedded into transaction command (which in turn has own command),
so one can extend protocol as needed without breaking backward compatibility as long
as old commands are supported. All string lengths include tail 0 byte.

All commans are transfered over the network in big-endian. CPU endianess is used at the end peers.

@cmd - command number, which specifies command to be processed. Following
	commands are used currently:

```
	NETFS_READDIR	= 1,	/* Read directory for given inode number */
	NETFS_READ_PAGE,	/* Read data page from the server */
	NETFS_WRITE_PAGE,	/* Write data page to the server */
	NETFS_CREATE,		/* Create directory entry */
	NETFS_REMOVE,		/* Remove directory entry */
	NETFS_LOOKUP,		/* Lookup single object */
	NETFS_LINK,		/* Create a link */
	NETFS_TRANS,		/* Transaction */
	NETFS_OPEN,		/* Open intent */
	NETFS_INODE_INFO,	/* Metadata cache coherency synchronization
message */
	NETFS_PAGE_CACHE,	/* Page cache invalidation message */
	NETFS_READ_PAGES,	/* Read multiple contiguous pages in one go */
	NETFS_RENAME,		/* Rename object */
	NETFS_CAPABILITIES,	/* Capabilities of the client, for example
supported crypto */
	NETFS_LOCK,		/* Distributed lock message */
	NETFS_XATTR_SET,	/* Set extended attribute */
	NETFS_XATTR_GET,	/* Get extended attribute */
```

@ext - external flags. Used by different commands to specify some extra arguments
	like partial size of the embedded objects or creation flags.

@size - size of the attached data. For NETFS_READ_PAGE and NETFS_READ_PAGES no

```
data is attached,
        but size of the requested data is incorporated here. It does not include
size of the command
        header (struct netfs_cmd) itself.
```

@id - id of the object this command operates on. Each command can use it for own purpose.

@start - start of the object this command operates on. Each command can use it for own purpose.

@csize, @cpad - size and padding size of the (attached if needed) crypto information.

Command specifications.

@NETFS_READDIR
This command is used to sync content of the remote dir to the client.

@ext - length of the path to object.
@size - the same.
@id - local inode number of the directory to read.
@start - zero.

@NETFS_READ_PAGE
This command is used to read data from remote server.
Data size does not exceed local page cache size.

@id - inode number.
@start - first byte offset.
@size - number of bytes to read plus length of the path to object.
@ext - object path length.

@NETFS_CREATE
Used to create object.
It does not require that all directories on top of the object were
already created, it will create them automatically. Each object has
associated @netfs_path_entry data structure, which contains creation
mode (permissions and type) and length of the name as long as name itself.

@start - 0
@size - size of the all data structures needed to create a path
@id - local inode number
@ext - 0

@NETFS_REMOVE
Used to remove object.

@ext - length of the path to object.
@size - the same.
@id - local inode number.
@start - zero.

```
@NETFS_LOOKUP
Lookup information about object on server.

@ext - length of the path to object.
@size - the same.
@id - local inode number of the directory to look object in.
@start - local inode number of the object to look at.


@NETFS_LINK
Create hard of symlink.
Command is sent as "object_path|target_path".

@size - size of the above string.
@id - parent local inode number.
@start - 1 for symlink, 0 for hardlink.
@ext - size of the "object_path" above.


@NETFS_TRANS
Transaction header.

@size - incorporates all embedded command sizes including theirs header sizes.
@start - transaction generation number - unique id used to find transaction.
@ext - transaction flags. Unused at the moment.
@id - 0.


@NETFS_OPEN
Open intent for given transaction.

@id - local inode number.
@start - 0.
@size - path length to the object.
@ext - open flags (O_RDWR and so on).


@NETFS_INODE_INFO
Metadata update command.
It is sent to servers when attributes of the object are changed and received
when data or metadata were updated. It operates with the following structure:

struct netfs_inode_info
{
        unsigned int            mode;
        unsigned int            nlink;
        unsigned int            uid;
        unsigned int            gid;
        unsigned int            blocksize;
        unsigned int            padding;
        __u64                   ino;
        __u64                   blocks;
        __u64                   rdev;
        __u64                   size;
        __u64                   version;
```

};

It effectively mirrors stat(2) returned data.

@ext - path length to the object.
@size - the same plus size of the netfs_inode_info structure.
@id - local inode number.
@start - 0.

@NETFS_PAGE_CACHE
Command is only received by clients. It contains information about
page to be marked as not up-to-date.

@id - client's inode number.
@start - last byte of the page to be invalidated. If it is not equal to
        current inode size, it will be vmtruncated().
@size - 0
@ext - 0

@NETFS_READ_PAGES
Used to read multiple contiguous pages in one go.

@start - first byte of the contiguous region to read.
@size - contains of two fields: lower 8 bits are used to represent page cache
shift
        used by client, another 3 bytes are used to get number of pages.
@id - local inode number.
@ext - path length to the object.

@NETFS_RENAME
Used to rename object.
Attached data is formed into following string: "old_path|new_path".

@id - local inode number.
@start - parent inode number.
@size - length of the above string.
@ext - length of the old path part.

@NETFS_CAPABILITIES
Used to exchange crypto capabilities with server.
If crypto capabilities are not supported by server, then client will disable it
or fail (if 'crypto_fail_unsupported' mount options was specified).

@id - superblock index. Used to specify crypto information for group of servers.
@size - size of the attached capabilities structure.
@start - 0.
@size - 0.
@scsize - 0.

@NETFS_LOCK
Used to send lock request/release messages. Although it sends byte range request

```
and is capable of flushing pages based on that, it is not used, since all Linux
filesystems lock the whole inode.

@id - lock generation number.
@start - start of the locked range.
@size - size of the locked range.
@ext - lock type: read/write. Not used actually. 15'th bit is used to determine,
        if it is lock request (1) or release (0).

@NETFS_XATTR_SET
@NETFS_XATTR_GET
Used to set/get extended attributes for given inode.
@id - attribute generation number or xattr setting type
@start - size of the attribute (request or attached)
@size - name length, path len and data size for given attribute
@ext - path length for given object
```