NAME

sg_get_disk_io_stats, sg_get_disk_io_stats_diff − get disk io statistics

SYNOPSIS

```
#include <statgrab.h>

sg_disk_io_stats *sg_get_disk_io_stats(int *entries);

sg_disk_io_stats *sg_get_disk_io_stats_diff(int *entries);
```

DESCRIPTION

Both calls take a pointer to an int, entries, which is filled with the number of disks the machine has. This is needed to know how many sg_disk_io_stats structures have been returned. A pointer is returned to the first sg_disk_io_stats.

sg_get_disk_io_stats returns the disk IO stored in the kernel which holds the amount of data transferred since bootup. On some platforms, such as Solaris 7, this value is stored in a 32bit int, so wraps around when it reaches 4GB. Other platforms, such as Solaris 8, hold the value in a 64bit int, which wraps somewhere near 17 million terabytes.

sg_get_disk_io_stats_diff is the same as sg_get_disk_io_stats except it will return the difference since the last call. So, for instance a call to sg_get_disk_io_stats_diff is made, and called again 5 seconds later. Over that time, 2000 bytes of traffic were written and 10000 bytes read. write_bytes will store 2000 bytes, read_bytes will store 10000, and systime will store 5. This function copes with wrap arounds by the O/S so should be seemless to use.

On Solaris libstatgrab will attempt to get the cXtXdXsX representation for the disk_name string. If it fails it will use a name like sd0. On some systems programs calling libstatgrab will need elevated privileges to lookup some of the names. The mappings are built up when sg_init is called.

RETURN VALUES

All diskio statistics return a pointer to a structure of type sg_disk_io_stats.

```
typedef struct{
        char *disk_name;
        long long read_bytes;
        long long write_bytes;
        time_t systime;
}sg_disk_io_stats;
```

disk_name
    The name known to the operating system. (eg. on linux it might be hda)

read_bytes
    The number of bytes the disk has read.

write_bytes
    The number of bytes the disk has written.

systime The time period over which read_bytes and write_bytes were transferred.

BUGS

On the very first call sg_get_disk_io_stats_diff will return the same as sg_get_disk_io_stats. After the first call it will always return the difference.

On operating systems that hold only 32bits of data there is a problem if the values wrap twice. For example, on Solaris 7 if 9GB is transferred and the operating system wraps at 4GB, the sg_get_disk_io_stats_diff function will return 5GB.

SEE ALSO

statgrab(3)

WEBSITE

http://www.i−scream.org/libstatgrab/