

**NAME**

**blacklist\_open**, **blacklist\_close**, **blacklist\_r**, **blacklist**, **blacklist\_sa**, **blacklist\_sa\_r** - Blacklistd notification library

**LIBRARY**

library "libblacklist"

**SYNOPSIS**

```
#include <blacklist.h>
```

```
struct blacklist *
```

```
blacklist_open(void);
```

```
void
```

```
blacklist_close(struct blacklist *cookie);
```

```
int
```

```
blacklist(int action, int fd, const char *msg);
```

```
int
```

```
blacklist_r(struct blacklist *cookie, int action, int fd, const char *msg);
```

```
int
```

```
blacklist_sa(int action, int fd, const struct sockaddr *sa, socklen_t salen, const char *msg);
```

```
int
```

```
blacklist_sa_r(struct blacklist *cookie, int action, int fd, const struct sockaddr *sa, socklen_t salen, const char *msg);
```

**DESCRIPTION**

These functions can be used by daemons to notify blacklistd(8) about successful and failed remote connections so that blacklistd can block or release port access to prevent Denial of Service attacks.

The function **blacklist\_open**() creates the necessary state to communicate with blacklistd(8) and returns a pointer to it, or NULL on failure.

The **blacklist\_close**() function frees all memory and resources used.

The **blacklist**() function sends a message to blacklistd(8), with an integer *action* argument specifying the type of notification, a file descriptor *fd* specifying the accepted file descriptor connected to the client,

and an optional message in the *msg* argument.

The *action* parameter can take these values:

<i>BLACKLIST_AUTH_FAIL</i>	There was an unsuccessful authentication attempt.
<i>BLACKLIST_AUTH_OK</i>	A user successfully authenticated.
<i>BLACKLIST_ABUSIVE_BEHAVIOR</i>	The sending daemon has detected abusive behavior from the remote system. The remote address should be blocked as soon as possible.
<i>BLACKLIST_BAD_USER</i>	The sending daemon has determined the username presented for authentication is invalid. The <code>blacklistd(8)</code> daemon compares the username to a configured list of forbidden usernames and blocks the address immediately if a forbidden username matches. (The <i>BLACKLIST_BAD_USER</i> support is not currently available.)

The `blacklist_r()` function is more efficient because it keeps the blacklist state around.

The `blacklist_sa()` and `blacklist_sa_r()` functions can be used with unconnected sockets, where `getpeername(2)` will not work, the server will pass the peer name in the message.

In all cases the file descriptor passed in the *fd* argument must be pointing to a valid socket so that `blacklistd(8)` can establish ownership of the local endpoint using `getsockname(2)`.

By default, `syslogd(8)` is used for message logging. The internal `bl_create()` function can be used to create the required internal state and specify a custom logging function.

## RETURN VALUES

The function `blacklist_open()` returns a cookie on success and `NULL` on failure setting `errno` to an appropriate value.

The functions `blacklist()`, `blacklist_sa()`, and `blacklist_sa_r()` return 0 on success and -1 on failure setting `errno` to an appropriate value.

## SEE ALSO

`blacklistd.conf(5)`, `blacklistd(8)`

**AUTHORS**

Christos Zoulas