

# Phone Firewall Reference Manual

v0.01

Generated by Doxygen 1.5.4

Wed Jul 9 11:00:11 2008



# Contents

<b>1</b>	<b>Phone Firewall Data Structure Index</b>	<b>1</b>
1.1	Phone Firewall Data Structures . . . . .	1
<b>2</b>	<b>Phone Firewall File Index</b>	<b>3</b>
2.1	Phone Firewall File List . . . . .	3
<b>3</b>	<b>Phone Firewall Data Structure Documentation</b>	<b>5</b>
3.1	Entry Struct Reference . . . . .	5
3.2	entry Struct Reference . . . . .	7
<b>4</b>	<b>Phone Firewall File Documentation</b>	<b>9</b>
4.1	libphonefirewall.h File Reference . . . . .	9
4.2	logfile.c File Reference . . . . .	17
4.3	logfile.h File Reference . . . . .	20
4.4	pf_daemon.c File Reference . . . . .	23
4.5	pf_daemon.h File Reference . . . . .	26
4.6	phonefirewall_administration.c File Reference . . . . .	27
4.7	phonefirewall_search.c File Reference . . . . .	32



# Chapter 1

## Phone Firewall Data Structure Index

### 1.1 Phone Firewall Data Structures

Here are the data structures with brief descriptions:

<a href="#">Entry</a> . . . . .	<a href="#">5</a>
<a href="#">entry</a> (Includes all informations for an <a href="#">entry</a> ) . . . . .	<a href="#">7</a>



# Chapter 2

## Phone Firewall File Index

### 2.1 Phone Firewall File List

Here is a list of all files with brief descriptions:

<a href="#">libphonefirewall.h</a> (API of the phone firewall ) . . . . .	9
<a href="#">logfile.c</a> . . . . .	17
<a href="#">logfile.h</a> . . . . .	20
<a href="#">pf_daemon.c</a> . . . . .	23
<a href="#">pf_daemon.h</a> . . . . .	26
<a href="#">phonefirewall_administration.c</a> . . . . .	27
<a href="#">phonefirewall_search.c</a> . . . . .	32





## Chapter 3

# Phone Firewall Data Structure Documentation

### 3.1 Entry Struct Reference

```
#include <libphonefirewall.h>
```

#### Data Fields

- int `country_code`
- int `area_code`
- unsigned long long `number`
- char \* `name`
- char \* `reason`
- int `priority`

#### 3.1.1 Detailed Description

Definition at line 52 of file libphonefirewall.h.

#### 3.1.2 Field Documentation

##### 3.1.2.1 int Entry::country\_code

Definition at line 53 of file libphonefirewall.h.

Referenced by `check_blacklist_entry()`, `check_whitelist_entry()`, `evaluate_stmt()`, `find_entry_by_name()`, and `get_blacklist_entry_by_name()`.

##### 3.1.2.2 int Entry::area\_code

Definition at line 54 of file libphonefirewall.h.

Referenced by `check_blacklist_entry()`, `check_whitelist_entry()`, `evaluate_stmt()`, `find_entry_by_name()`, and `get_blacklist_entry_by_name()`.

### 3.1.2.3 unsigned long long Entry::number

Definition at line 55 of file libphonefirewall.h.

Referenced by check\_blacklist\_entry(), check\_whitelist\_entry(), evaluate\_stmt(), find\_entry\_by\_name(), and get\_blacklist\_entry\_by\_name().

### 3.1.2.4 char\* Entry::name

Definition at line 56 of file libphonefirewall.h.

Referenced by find\_entry\_by\_name().

### 3.1.2.5 char\* Entry::reason

Definition at line 57 of file libphonefirewall.h.

Referenced by find\_entry\_by\_name(), and get\_blacklist\_entry\_by\_name().

### 3.1.2.6 int Entry::priority

Definition at line 58 of file libphonefirewall.h.

Referenced by check\_blacklist\_entry(), check\_whitelist\_entry(), evaluate\_stmt(), and find\_entry\_by\_name().

The documentation for this struct was generated from the following file:

- [libphonefirewall.h](#)

## 3.2 entry Struct Reference

Includes all informations for an [entry](#).

```
#include <libphonefirewall.h>
```

### 3.2.1 Detailed Description

Includes all informations for an [entry](#).

The struct which includes all information about entries (black- and whitelist).

The documentation for this struct was generated from the following file:

- [libphonefirewall.h](#)



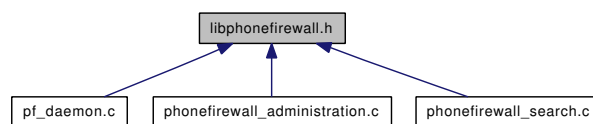
## Chapter 4

# Phone Firewall File Documentation

### 4.1 libphonefirewall.h File Reference

API of the phone firewall.

This graph shows which files directly or indirectly include this file:



### Data Structures

- struct [Entry](#)

### Defines

- #define [PRIO\\_ALL](#) -999
- #define [DB\\_FILE](#) "db/phone-firewall.db"
- #define [STMT\\_SIZE](#) 1024
- #define [MAX\\_LINE\\_LENGTH](#) 512
- #define [TB\\_COUNTRYCODE](#) "countrycode"
- #define [TB\\_AREACODE](#) "areacode"
- #define [TB\\_NUMBER](#) "number"
- #define [TB\\_NAME](#) "name"
- #define [TB\\_REASON](#) "reason"
- #define [TB\\_PRIORITY](#) "priority"

### Functions

- int [add\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, char \*name, char \*reason, int priority)
- int [rm\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number)

- int [check\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, int priority)
- int [add\\_whitelist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, char \*name, char \*reason, int priority)
- struct [Entry](#) \* [get\\_blacklist\\_entry\\_by\\_name](#) (char \*name)
- struct [Entry](#) \* [get\\_blacklist\\_entry\\_by\\_number](#) (int country\_code, int area\_code, unsigned long long number)
- int [rm\\_whitelist\\_entry](#) (int country\_code, int area\_code, unsigned long long number)
- int [check\\_whitelist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, int priority)
- struct [Entry](#) \* [get\\_whitelist\\_entry\\_by\\_name](#) (char \*name)
- struct [Entry](#) \* [get\\_whitelist\\_entry\\_by\\_number](#) (int country\_code, int area\_code, unsigned long long number)

## Variables

- struct [Entry](#) [entry](#)

### 4.1.1 Detailed Description

API of the phone firewall.

#### Author:

Alex Oberhauser

The header file of the Phone Firewall. Blocks or accepts incoming phone calls, so it's possible to prevent disturbing phone calls. Provides a API which can used by other application to build nice programs.

Definition in file [libphonefirewall.h](#).

### 4.1.2 Define Documentation

#### 4.1.2.1 `#define DB_FILE "db/phone-firewall.db"`

Definition at line 34 of file [libphonefirewall.h](#).

Referenced by [add\\_blacklist\\_entry\(\)](#), [add\\_whitelist\\_entry\(\)](#), [check\\_blacklist\\_entry\(\)](#), [check\\_whitelist\\_entry\(\)](#), [get\\_blacklist\\_entry\\_by\\_name\(\)](#), [rm\\_blacklist\\_entry\(\)](#), and [rm\\_whitelist\\_entry\(\)](#).

#### 4.1.2.2 `#define MAX_LINE_LENGTH 512`

Definition at line 36 of file [libphonefirewall.h](#).

Referenced by [add\\_blacklist\\_entry\(\)](#), [add\\_whitelist\\_entry\(\)](#), [check\\_blacklist\\_entry\(\)](#), [check\\_whitelist\\_entry\(\)](#), [get\\_blacklist\\_entry\\_by\\_name\(\)](#), [rm\\_blacklist\\_entry\(\)](#), and [rm\\_whitelist\\_entry\(\)](#).

#### 4.1.2.3 `#define PRIO_ALL -999`

Definition at line 33 of file [libphonefirewall.h](#).

Referenced by [add\\_blacklist\\_entry\(\)](#), [add\\_whitelist\\_entry\(\)](#), and [evaluate\\_stmt\(\)](#).

#### 4.1.2.4 #define STMT\_SIZE 1024

Definition at line 35 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), check\_blacklist\_entry(), check\_whitelist\_entry(), get\_blacklist\_entry\_by\_name(), rm\_blacklist\_entry(), and rm\_whitelist\_entry().

#### 4.1.2.5 #define TB\_AREACODE "areacode"

Definition at line 39 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), check\_blacklist\_entry(), check\_whitelist\_entry(), evaluate\_stmt(), find\_entry\_by\_name(), get\_blacklist\_entry\_by\_name(), rm\_blacklist\_entry(), and rm\_whitelist\_entry().

#### 4.1.2.6 #define TB\_COUNTRYCODE "countrycode"

Definition at line 38 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), check\_blacklist\_entry(), check\_whitelist\_entry(), evaluate\_stmt(), find\_entry\_by\_name(), get\_blacklist\_entry\_by\_name(), rm\_blacklist\_entry(), and rm\_whitelist\_entry().

#### 4.1.2.7 #define TB\_NAME "name"

Definition at line 41 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), find\_entry\_by\_name(), and get\_blacklist\_entry\_by\_name().

#### 4.1.2.8 #define TB\_NUMBER "number"

Definition at line 40 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), check\_blacklist\_entry(), check\_whitelist\_entry(), evaluate\_stmt(), find\_entry\_by\_name(), get\_blacklist\_entry\_by\_name(), rm\_blacklist\_entry(), and rm\_whitelist\_entry().

#### 4.1.2.9 #define TB\_PRIORITY "priority"

Definition at line 43 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), check\_blacklist\_entry(), check\_whitelist\_entry(), evaluate\_stmt(), find\_entry\_by\_name(), and get\_blacklist\_entry\_by\_name().

#### 4.1.2.10 #define TB\_REASON "reason"

Definition at line 42 of file libphonefirewall.h.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), find\_entry\_by\_name(), and get\_blacklist\_entry\_by\_name().

### 4.1.3 Function Documentation

#### 4.1.3.1 `int add_blacklist_entry (int country_code, int area_code, unsigned long long number, char * name, char * reason, int priority)`

Add a number to the blacklist. The number will be blocked after that.

##### Parameters:

**country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)

**area\_code** The area code which indicates your mobile operator.

**number** The telephone number of the person (without country and area code.

**name** The name of the person.

**reason** Why you have blocked this person.

**priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.

The value "PRIO\_ALL" stands for all priorities.

##### Returns:

If all goes well 0 (zero) otherwise an errno code.

Definition at line 75 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, PRIO\_ALL, STMT\_SIZE, TB\_-AREACODE, TB\_COUNTRYCODE, TB\_NAME, TB\_NUMBER, TB\_PRIORITY, TB\_REASON, and write\_logentry().

Here is the call graph for this function:



#### 4.1.3.2 `int add_whitelist_entry (int country_code, int area_code, unsigned long long number, char * name, char * reason, int priority)`

Add a number to the whitelist. The number will be accepted after that.

##### Parameters:

**country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)

**area\_code** The area code which indicates your mobile operator.

**number** The telephone number of the person (without country and area code.

**name** The name of the person.

**reason** Why you have blocked this person.

**priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.

The value "PRIO\_ALL" stands for all priorities.

##### Returns:

If all goes well 0 (zero) otherwise an errno code.



Definition at line 117 of file `phonefirewall_administration.c`.

References `DB_FILE`, `ERR_FLAG`, `MAX_LINE_LENGTH`, `PRIO_ALL`, `STMT_SIZE`, `TB_AREACODE`, `TB_COUNTRYCODE`, `TB_NAME`, `TB_NUMBER`, `TB_PRIORITY`, `TB_REASON`, and `write_logentry()`.

Here is the call graph for this function:



#### 4.1.3.3 `int check_blacklist_entry (int country_code, int area_code, unsigned long long number, int priority)`

Checks if a number is on the blacklist.

##### Parameters:

**country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)

**area\_code** The area code which indicates your mobile operator.

**number** The telephone number of the person (without country and area code).

**priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is chosen.

The value "PRIO\_ALL" stands for all priorities.

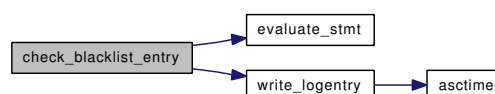
##### Returns:

If the number was found 1, otherwise 0.

Definition at line 234 of file `phonefirewall_administration.c`.

References `Entry::area_code`, `Entry::country_code`, `DB_FILE`, `ERR_FLAG`, `evaluate_stmt()`, `INFO_FLAG`, `MAX_LINE_LENGTH`, `Entry::number`, `p_entry`, `Entry::priority`, `STMT_SIZE`, `TB_AREACODE`, `TB_COUNTRYCODE`, `TB_NUMBER`, `TB_PRIORITY`, and `write_logentry()`.

Here is the call graph for this function:



#### 4.1.3.4 `int check_whitelist_entry (int country_code, int area_code, unsigned long long number, int priority)`

Checks if a number is on the whitelist.

##### Parameters:

**country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)

**area\_code** The area code which indicates your mobile operator.

**number** The telephone number of the person (without country and area code).

**priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.

The value "PRIO\_ALL" stands for all priorities.

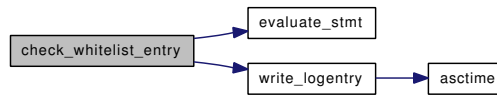
#### Returns:

If the number was found 1, otherwise 0.

Definition at line 293 of file phonefirewall\_administration.c.

References `Entry::area_code`, `Entry::country_code`, `DB_FILE`, `ERR_FLAG`, `evaluate_stmt()`, `INFO_FLAG`, `MAX_LINE_LENGTH`, `Entry::number`, `p_entry`, `Entry::priority`, `STMT_SIZE`, `TB_AREACODE`, `TB_COUNTRYCODE`, `TB_NUMBER`, `TB_PRIORITY`, and `write_logentry()`.

Here is the call graph for this function:



#### 4.1.3.5 struct Entry\* get\_blacklist\_entry\_by\_name (char \* name) [read]

Search a entrie by name.

#### Parameters:

**name** The name of the person which is blocked.

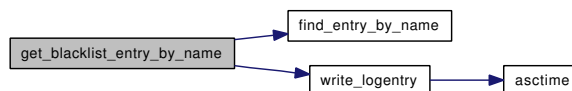
#### Returns:

[entry](#) Returns the found [entry](#).

Definition at line 66 of file phonefirewall\_search.c.

References `Entry::area_code`, `ASCII_PERCENT_CHAR`, `Entry::country_code`, `DB_FILE`, `entry_array`, `ERR_FLAG`, `find_entry_by_name()`, `MAX_LINE_LENGTH`, `Entry::number`, `Entry::reason`, `STMT_SIZE`, `TB_AREACODE`, `TB_COUNTRYCODE`, `TB_NAME`, `TB_NUMBER`, `TB_PRIORITY`, `TB_REASON`, and `write_logentry()`.

Here is the call graph for this function:



#### 4.1.3.6 struct Entry\* get\_blacklist\_entry\_by\_number (int country\_code, int area\_code, unsigned long long number) [read]

Search a entrie by number (country code + area code + number).

**Parameters:**

*country\_code* The country code (for example 39 for Italy, 43 for Austria, and so one)

*area\_code* The area code which indicates your mobile operator.

*number* The telephone number of the person (without country and area code).

**Returns:**

[entry](#) Returns the found [entry](#).

Definition at line 115 of file phonefirewall\_search.c.

**4.1.3.7 struct Entry\* get\_whitelist\_entry\_by\_name (char \* *name*)** [read]

Search a entrie by name.

**Parameters:**

*name* The name of the person which is accepted.

**Returns:**

[entry](#) Returns the found [entry](#).

Definition at line 122 of file phonefirewall\_search.c.

**4.1.3.8 struct Entry\* get\_whitelist\_entry\_by\_number (int *country\_code*, int *area\_code*, unsigned long long *number*)** [read]

Search a entrie by number (country code + area code + number).

**Parameters:**

*country\_code* The country code (for example 39 for Italy, 43 for Austria, and so one)

*area\_code* The area code which indicates your mobile operator.

*number* The telephone number of the person (without country and area code).

**Returns:**

[entry](#) Returns the found [entry](#).

Definition at line 127 of file phonefirewall\_search.c.

**4.1.3.9 int rm\_blacklist\_entry (int *country\_code*, int *area\_code*, unsigned long long *number*)**

Removes a blocked number from the blacklist.

**Parameters:**

*number* The number which will be deleted.

**Returns:**

If all goes right 0, otherwise an error code.

Definition at line 159 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NUMBER, and write\_logentry().

Here is the call graph for this function:



#### 4.1.3.10 int rm\_whitelist\_entry (int country\_code, int area\_code, unsigned long long number)

Removes a accepted number from the whitelist.

##### Parameters:

*number* The number which will be deleted.

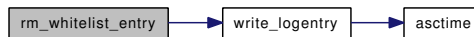
##### Returns:

If all goes right 0, otherwise an error code.

Definition at line 196 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NUMBER, and write\_logentry().

Here is the call graph for this function:



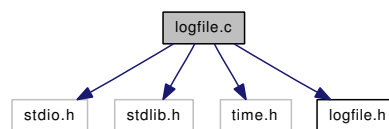
### 4.1.4 Variable Documentation

#### 4.1.4.1 struct Entry entry

## 4.2 logfile.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "logfile.h"
```

Include dependency graph for logfile.c:



### Functions

- char \* [asctime](#) (const struct tm \*timeptr)
- int [write\\_logentry](#) (char \*msg, char \*component, int flag)

### 4.2.1 Function Documentation

#### 4.2.1.1 char\* asctime (const struct tm \* timeptr)

Compounds a humand readable date and time string.

##### Parameters:

*timeptr* A pointer to the actual time.

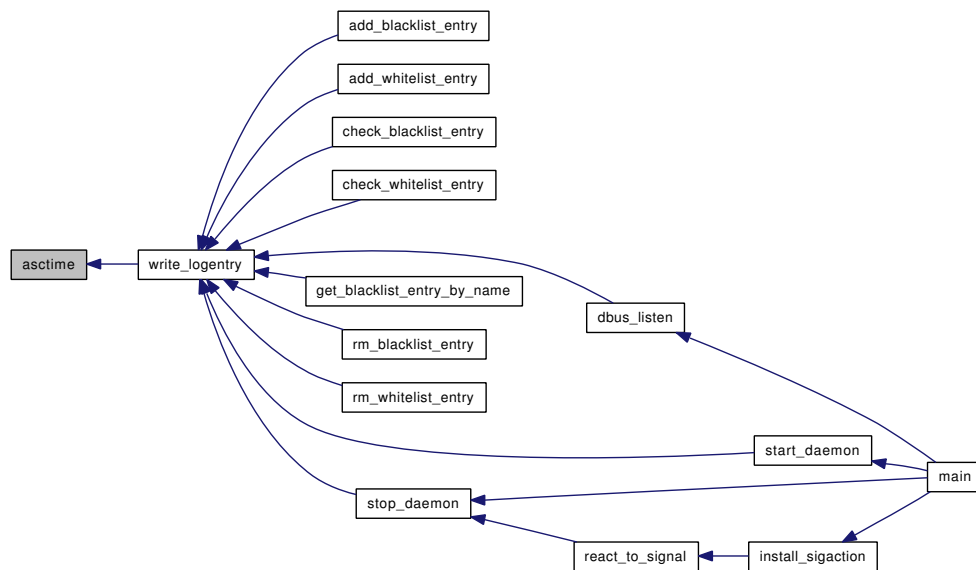
##### Returns:

The date and time as a string.

Definition at line 33 of file `logfile.c`.

Referenced by `write_logentry()`.

Here is the caller graph for this function:



#### 4.2.1.2 int write\_logentry (char \* msg, char \* component, int flag)

Writes a logfile entry.

##### Parameters:

**msg** The message which should be written in the logfile.

**component** The program which calls the write\_logentry function, for example "phonefirewall"

**flag** What message should be written. Use the defined flags.

##### Returns:

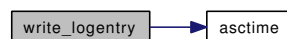
-1 if something fails, otherwise 0

Definition at line 56 of file logfile.c.

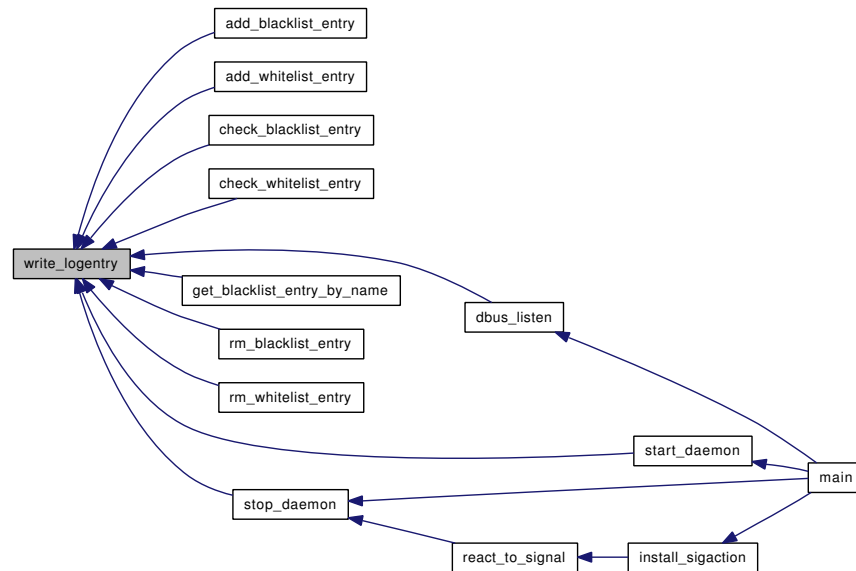
References `asctime()`, `ERR_FLAG`, `INFO_FLAG`, `LOGFILE`, `MAX_ENTRY_LENGTH`, `UNKNOWN`, and `WARN_FLAG`.

Referenced by `add_blacklist_entry()`, `add_whitelist_entry()`, `check_blacklist_entry()`, `check_whitelist_entry()`, `dbus_listen()`, `get_blacklist_entry_by_name()`, `rm_blacklist_entry()`, `rm_whitelist_entry()`, `start_daemon()`, and `stop_daemon()`.

Here is the call graph for this function:

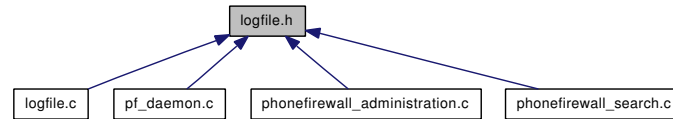


Here is the caller graph for this function:



## 4.3 logfile.h File Reference

This graph shows which files directly or indirectly include this file:



### Defines

- #define [LOGFILE](#) "log/moksec.log"
- #define [MAX\\_ENTRY\\_LENGTH](#) 128
- #define [UNKNOWN](#) 0
- #define [ERR\\_FLAG](#) 1
- #define [WARN\\_FLAG](#) 2
- #define [INFO\\_FLAG](#) 3

### Functions

- int [write\\_logentry](#) (char \*msg, char \*component, int flag)

#### 4.3.1 Define Documentation

##### 4.3.1.1 #define ERR\_FLAG 1

Definition at line 25 of file logfile.h.

Referenced by [add\\_blacklist\\_entry\(\)](#), [add\\_whitelist\\_entry\(\)](#), [check\\_blacklist\\_entry\(\)](#), [check\\_whitelist\\_entry\(\)](#), [dbus\\_listen\(\)](#), [get\\_blacklist\\_entry\\_by\\_name\(\)](#), [rm\\_blacklist\\_entry\(\)](#), [rm\\_whitelist\\_entry\(\)](#), and [write\\_logentry\(\)](#).

##### 4.3.1.2 #define INFO\_FLAG 3

Definition at line 27 of file logfile.h.

Referenced by [check\\_blacklist\\_entry\(\)](#), [check\\_whitelist\\_entry\(\)](#), [dbus\\_listen\(\)](#), [start\\_daemon\(\)](#), [stop\\_daemon\(\)](#), and [write\\_logentry\(\)](#).

##### 4.3.1.3 #define LOGFILE "log/moksec.log"

Definition at line 21 of file logfile.h.

Referenced by [write\\_logentry\(\)](#).

##### 4.3.1.4 #define MAX\_ENTRY\_LENGTH 128

Definition at line 22 of file logfile.h.

Referenced by [write\\_logentry\(\)](#).



#### 4.3.1.5 #define UNKNOWN 0

Definition at line 24 of file logfile.h.

Referenced by write\_logentry().

#### 4.3.1.6 #define WARN\_FLAG 2

Definition at line 26 of file logfile.h.

Referenced by write\_logentry().

### 4.3.2 Function Documentation

#### 4.3.2.1 int write\_logentry (char \* *msg*, char \* *component*, int *flag*)

Writes a logfile entry.

##### Parameters:

*msg* The message which should be written in the logfile.

*component* The program which calls the write\_logentry function, for example "phonefirewall"

*flag* What message should be written. Use the defined flags.

##### Returns:

-1 if something fails, otherwise 0

Definition at line 56 of file logfile.c.

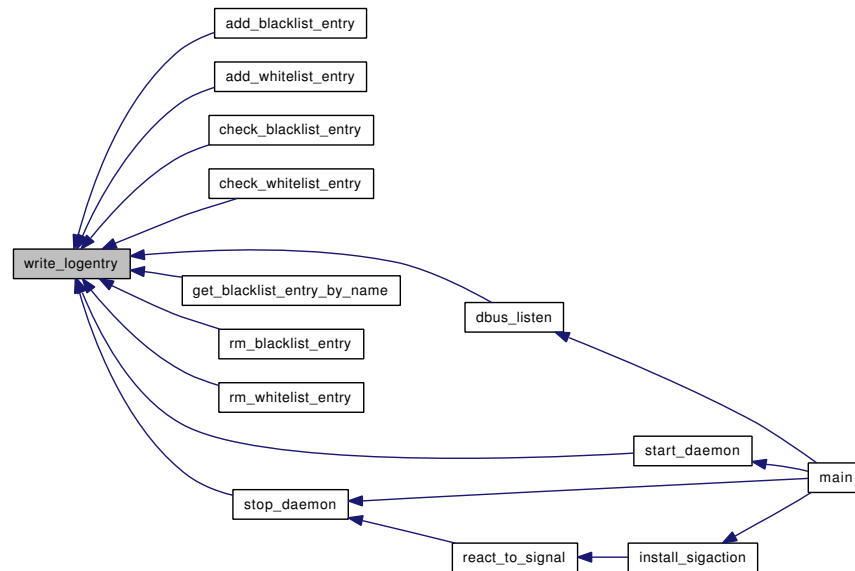
References asctime(), ERR\_FLAG, INFO\_FLAG, LOGFILE, MAX\_ENTRY\_LENGTH, UNKNOWN, and WARN\_FLAG.

Referenced by add\_blacklist\_entry(), add\_whitelist\_entry(), check\_blacklist\_entry(), check\_whitelist\_entry(), dbus\_listen(), get\_blacklist\_entry\_by\_name(), rm\_blacklist\_entry(), rm\_whitelist\_entry(), start\_daemon(), and stop\_daemon().

Here is the call graph for this function:



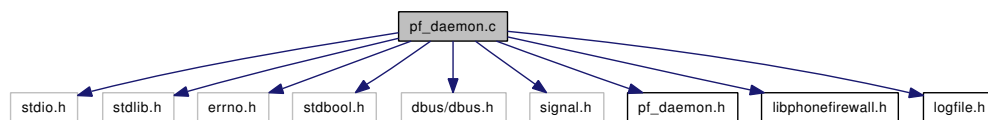
Here is the caller graph for this function:



## 4.4 pf\_daemon.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <stdbool.h>
#include <dbus/dbus.h>
#include <signal.h>
#include "pf_daemon.h"
#include "libphonefirewall.h"
#include "logfile.h"
```

Include dependency graph for pf\_daemon.c:



## Functions

- void [start\\_daemon](#) ()
- void [stop\\_daemon](#) ()
- void [dbus\\_listen](#) ()
- void [install\\_sigaction](#) ()
- void [react\\_to\\_signal](#) (int sig)
- int [main](#) (int argc, char \*\*argv)

### 4.4.1 Function Documentation

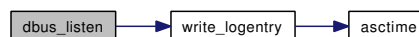
#### 4.4.1.1 void dbus\_listen ()

Definition at line 87 of file pf\_daemon.c.

References `ERR_FLAG`, `INFO_FLAG`, and `write_logentry()`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



#### 4.4.1.2 void install\_sigaction ()

Definition at line 143 of file pf\_daemon.c.

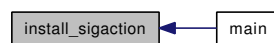
References react\_to\_signal().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:

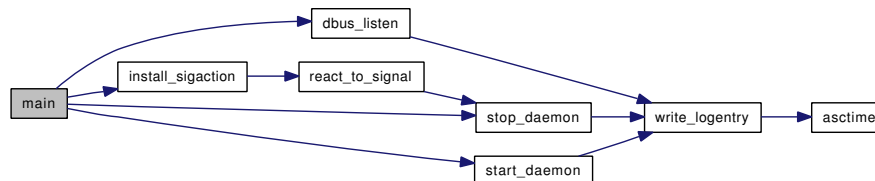


#### 4.4.1.3 int main (int argc, char \*\* argv)

Definition at line 38 of file pf\_daemon.c.

References dbus\_listen(), install\_sigaction(), start\_daemon(), and stop\_daemon().

Here is the call graph for this function:



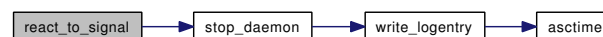
#### 4.4.1.4 void react\_to\_signal (int sig)

Definition at line 157 of file pf\_daemon.c.

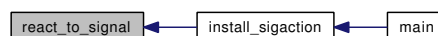
References stop\_daemon().

Referenced by install\_sigaction().

Here is the call graph for this function:



Here is the caller graph for this function:



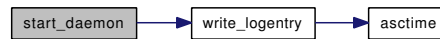
#### 4.4.1.5 void start\_daemon ()

Definition at line 53 of file pf\_daemon.c.

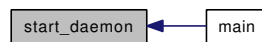
References INFO\_FLAG, LOCKFILE, and write\_logentry().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



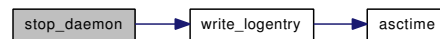
#### 4.4.1.6 void stop\_daemon ()

Definition at line 79 of file pf\_daemon.c.

References INFO\_FLAG, LOCKFILE, and write\_logentry().

Referenced by main(), and react\_to\_signal().

Here is the call graph for this function:

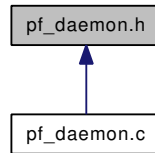


Here is the caller graph for this function:



## 4.5 pf\_daemon.h File Reference

This graph shows which files directly or indirectly include this file:



### Defines

- #define `LOCKFILE` `".lock"`

### Functions

- void `start_daemon` ()
- void `stop_daemon` ()
- void `dbus_listen` ()

#### 4.5.1 Define Documentation

##### 4.5.1.1 #define `LOCKFILE` `".lock"`

Definition at line 21 of file pf\_daemon.h.

Referenced by `start_daemon()`, and `stop_daemon()`.

#### 4.5.2 Function Documentation

##### 4.5.2.1 void `dbus_listen` ()

Function that provides method calls.

Bus name: to.networld.moksec.phonefirewall Object name: to.networld.moksec.phonefirewall.Object interface: to.networld.moksec.phonefirewall.Checking methods: checkblacklist(...) checkwhitelist(...)

##### 4.5.2.2 void `start_daemon` ()

Starts the program as a daemon and creates a lockfile (specified in the `LOCKFILE` constant). So it's impossible to start the daemon twice.

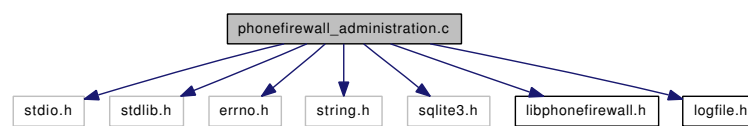
##### 4.5.2.3 void `stop_daemon` ()

Stops the daemon and deletes the lockfile, so a new instance can be started.

## 4.6 phonefirewall\_administration.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <sqlite3.h>
#include "libphonefirewall.h"
#include "logfile.h"
```

Include dependency graph for phonefirewall\_administration.c:



### Functions

- int [evaluate\\_stmt](#) (sqlite3\_stmt \*pp\_stmt, struct [Entry](#) \*p\_entry)
- int [add\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, char \*name, char \*reason, int priority)
- int [add\\_whitelist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, char \*name, char \*reason, int priority)
- int [rm\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number)
- int [rm\\_whitelist\\_entry](#) (int country\_code, int area\_code, unsigned long long number)
- int [check\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, int priority)
- int [check\\_whitelist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, int priority)

### 4.6.1 Function Documentation

#### 4.6.1.1 int [add\\_blacklist\\_entry](#) (int country\_code, int area\_code, unsigned long long number, char \*name, char \*reason, int priority)

Add a number to the blacklist. The number will be blocked after that.

##### Parameters:

**country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)

**area\_code** The area code which indicates your mobile operator.

**number** The telephone number of the person (without country and area code).

**name** The name of the person.

**reason** Why you have blocked this person.

**priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.

The value "PRIO\_ALL" stands for all priorities.

**Returns:**

If all goes well 0 (zero) otherwise an errno code.

Definition at line 75 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, PRIO\_ALL, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NAME, TB\_NUMBER, TB\_PRIORITY, TB\_REASON, and write\_logentry().

Here is the call graph for this function:



#### 4.6.1.2 int add\_whitelist\_entry (int country\_code, int area\_code, unsigned long long number, char \* name, char \* reason, int priority)

Add a number to the whitelist. The number will be accepted after that.

**Parameters:**

**country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)

**area\_code** The area code which indicates your mobile operator.

**number** The telephone number of the person (without country and area code).

**name** The name of the person.

**reason** Why you have blocked this person.

**priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.

The value "PRIO\_ALL" stands for all priorities.

**Returns:**

If all goes well 0 (zero) otherwise an errno code.

Definition at line 117 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, PRIO\_ALL, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NAME, TB\_NUMBER, TB\_PRIORITY, TB\_REASON, and write\_logentry().

Here is the call graph for this function:



#### 4.6.1.3 int check\_blacklist\_entry (int country\_code, int area\_code, unsigned long long number, int priority)

Checks if a number is on the blacklist.



**Parameters:**

- country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)
- area\_code** The area code which indicates your mobile operator.
- number** The telephone number of the person (without country and area code).
- priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.  
The value "PRIO\_ALL" stands for all priorities.

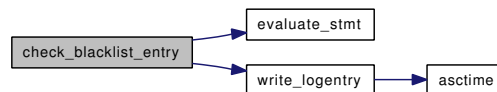
**Returns:**

If the number was found 1, otherwise 0.

Definition at line 234 of file phonefirewall\_administration.c.

References `Entry::area_code`, `Entry::country_code`, `DB_FILE`, `ERR_FLAG`, `evaluate_stmt()`, `INFO_FLAG`, `MAX_LINE_LENGTH`, `Entry::number`, `p_entry`, `Entry::priority`, `STMT_SIZE`, `TB_AREACODE`, `TB_COUNTRYCODE`, `TB_NUMBER`, `TB_PRIORITY`, and `write_logentry()`.

Here is the call graph for this function:



#### 4.6.1.4 int check\_whitelist\_entry (int country\_code, int area\_code, unsigned long long number, int priority)

Checks if a number is on the whitelist.

**Parameters:**

- country\_code** The country code (for example 39 for Italy, 43 for Austria, and so one)
- area\_code** The area code which indicates your mobile operator.
- number** The telephone number of the person (without country and area code).
- priority** Gives the [entry](#) a priority. 0 is standard. If the priority is higher the value will be also blocked/accepted if a higher priority is choosen.  
The value "PRIO\_ALL" stands for all priorities.

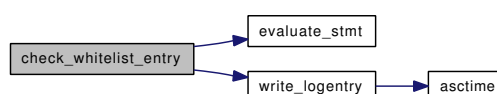
**Returns:**

If the number was found 1, otherwise 0.

Definition at line 293 of file phonefirewall\_administration.c.

References `Entry::area_code`, `Entry::country_code`, `DB_FILE`, `ERR_FLAG`, `evaluate_stmt()`, `INFO_FLAG`, `MAX_LINE_LENGTH`, `Entry::number`, `p_entry`, `Entry::priority`, `STMT_SIZE`, `TB_AREACODE`, `TB_COUNTRYCODE`, `TB_NUMBER`, `TB_PRIORITY`, and `write_logentry()`.

Here is the call graph for this function:



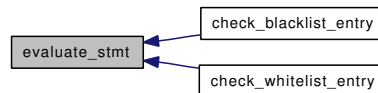
#### 4.6.1.5 int evaluate\_stmt (sqlite3\_stmt \* pp\_stmt, struct Entry \* p\_entry)

Definition at line 28 of file phonefirewall\_administration.c.

References Entry::area\_code, Entry::country\_code, Entry::number, PRIO\_ALL, Entry::priority, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NUMBER, and TB\_PRIORITY.

Referenced by check\_blacklist\_entry(), and check\_whitelist\_entry().

Here is the caller graph for this function:



#### 4.6.1.6 int rm\_blacklist\_entry (int country\_code, int area\_code, unsigned long long number)

Removes a blocked number from the blacklist.

##### Parameters:

**number** The number which will be deleted.

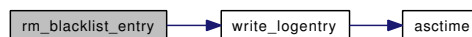
##### Returns:

If all goes right 0, otherwise an error code.

Definition at line 159 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NUMBER, and write\_logentry().

Here is the call graph for this function:



#### 4.6.1.7 int rm\_whitelist\_entry (int country\_code, int area\_code, unsigned long long number)

Removes a accepted number from the whitelist.

##### Parameters:

**number** The number which will be deleted.

##### Returns:

If all goes right 0, otherwise an error code.

Definition at line 196 of file phonefirewall\_administration.c.

References DB\_FILE, ERR\_FLAG, MAX\_LINE\_LENGTH, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NUMBER, and write\_logentry().

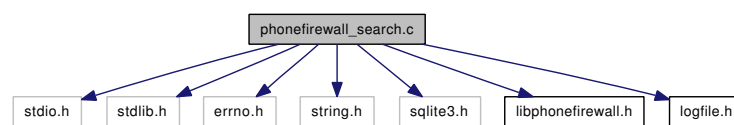
Here is the call graph for this function:



## 4.7 phonefirewall\_search.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <sqlite3.h>
#include "libphonefirewall.h"
#include "logfile.h"
```

Include dependency graph for phonefirewall\_search.c:



### Defines

- `#define ASCII_PERCENT_CHAR 37`
- `#define MAX_ENTRY_ARRAY 1024`

### Functions

- `struct Entry * find_entry_by_name (sqlite3_stmt *pp_stmt, char *name)`
- `struct Entry * get_blacklist_entry_by_name (char *name)`
- `struct Entry * get_blacklist_entry_by_number (int country_code, int area_code, unsigned long long number)`
- `struct Entry * get_whitelist_entry_by_name (char *name)`
- `struct Entry * get_whitelist_entry_by_number (int country_code, int area_code, unsigned long long number)`

### Variables

- `struct Entry * p_entry = &entry`
- `struct Entry entry_array [MAX_ENTRY_ARRAY]`

#### 4.7.1 Define Documentation

##### 4.7.1.1 `#define ASCII_PERCENT_CHAR 37`

Definition at line 28 of file phonefirewall\_search.c.

Referenced by `get_blacklist_entry_by_name()`.

#### 4.7.1.2 #define MAX\_ENTRY\_ARRAY 1024

Definition at line 29 of file phonefirewall\_search.c.

### 4.7.2 Function Documentation

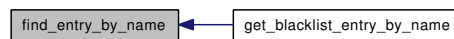
#### 4.7.2.1 struct Entry\* find\_entry\_by\_name (sqlite3\_stmt \* *pp\_stmt*, char \* *name*) [read]

Definition at line 34 of file phonefirewall\_search.c.

References Entry::area\_code, Entry::country\_code, Entry::name, Entry::number, Entry::priority, Entry::reason, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NAME, TB\_NUMBER, TB\_PRIORITY, and TB\_REASON.

Referenced by get\_blacklist\_entry\_by\_name().

Here is the caller graph for this function:



#### 4.7.2.2 struct Entry\* get\_blacklist\_entry\_by\_name (char \* *name*) [read]

Search a entrie by name.

##### Parameters:

***name*** The name of the person which is blocked.

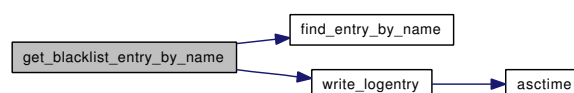
##### Returns:

[entry](#) Returns the found [entry](#).

Definition at line 66 of file phonefirewall\_search.c.

References Entry::area\_code, ASCII\_PERCENT\_CHAR, Entry::country\_code, DB\_FILE, entry\_array, ERR\_FLAG, find\_entry\_by\_name(), MAX\_LINE\_LENGTH, Entry::number, Entry::reason, STMT\_SIZE, TB\_AREACODE, TB\_COUNTRYCODE, TB\_NAME, TB\_NUMBER, TB\_PRIORITY, TB\_REASON, and write\_logentry().

Here is the call graph for this function:



#### 4.7.2.3 struct Entry\* get\_blacklist\_entry\_by\_number (int *country\_code*, int *area\_code*, unsigned long long *number*) [read]

Search a entrie by number (country code + area code + number).

**Parameters:**

*country\_code* The country code (for example 39 for Italy, 43 for Austria, and so one)

*area\_code* The area code which indicates your mobile operator.

*number* The telephone number of the person (without country and area code).

**Returns:**

[entry](#) Returns the found [entry](#).

Definition at line 115 of file phonefirewall\_search.c.

**4.7.2.4 struct Entry\* get\_whitelist\_entry\_by\_name (char \* *name*) [read]**

Search a entrie by name.

**Parameters:**

*name* The name of the person which is accepted.

**Returns:**

[entry](#) Returns the found [entry](#).

Definition at line 122 of file phonefirewall\_search.c.

**4.7.2.5 struct Entry\* get\_whitelist\_entry\_by\_number (int *country\_code*, int *area\_code*, unsigned long long *number*) [read]**

Search a entrie by number (country code + area code + number).

**Parameters:**

*country\_code* The country code (for example 39 for Italy, 43 for Austria, and so one)

*area\_code* The area code which indicates your mobile operator.

*number* The telephone number of the person (without country and area code).

**Returns:**

[entry](#) Returns the found [entry](#).

Definition at line 127 of file phonefirewall\_search.c.

**4.7.3 Variable Documentation****4.7.3.1 struct Entry entry\_array[MAX\_ENTRY\_ARRAY]**

Definition at line 32 of file phonefirewall\_search.c.

Referenced by get\_blacklist\_entry\_by\_name().

**4.7.3.2 struct Entry\* p\_entry = &entry**

Definition at line 31 of file phonefirewall\_search.c.

Referenced by check\_blacklist\_entry(), and check\_whitelist\_entry().

# Index

- add\_blacklist\_entry
  - libphonefirewall.h, [12](#)
  - phonefirewall\_administration.c, [27](#)
- add\_whitelist\_entry
  - libphonefirewall.h, [12](#)
  - phonefirewall\_administration.c, [28](#)
- area\_code
  - Entry, [5](#)
- ASCII\_PERCENT\_CHAR
  - phonefirewall\_search.c, [32](#)
- asctime
  - logfile.c, [17](#)
- check\_blacklist\_entry
  - libphonefirewall.h, [13](#)
  - phonefirewall\_administration.c, [28](#)
- check\_whitelist\_entry
  - libphonefirewall.h, [13](#)
  - phonefirewall\_administration.c, [29](#)
- country\_code
  - Entry, [5](#)
- DB\_FILE
  - libphonefirewall.h, [10](#)
- dbus\_listen
  - pf\_daemon.c, [23](#)
  - pf\_daemon.h, [26](#)
- Entry, [5](#)
  - area\_code, [5](#)
  - country\_code, [5](#)
  - name, [6](#)
  - number, [5](#)
  - priority, [6](#)
  - reason, [6](#)
- entry, [7](#)
  - libphonefirewall.h, [16](#)
- entry\_array
  - phonefirewall\_search.c, [34](#)
- ERR\_FLAG
  - logfile.h, [20](#)
- evaluate\_stmt
  - phonefirewall\_administration.c, [29](#)
- find\_entry\_by\_name
  - phonefirewall\_search.c, [33](#)
- get\_blacklist\_entry\_by\_name
  - libphonefirewall.h, [14](#)
  - phonefirewall\_search.c, [33](#)
- get\_blacklist\_entry\_by\_number
  - libphonefirewall.h, [14](#)
  - phonefirewall\_search.c, [33](#)
- get\_whitelist\_entry\_by\_name
  - libphonefirewall.h, [15](#)
  - phonefirewall\_search.c, [34](#)
- get\_whitelist\_entry\_by\_number
  - libphonefirewall.h, [15](#)
  - phonefirewall\_search.c, [34](#)
- INFO\_FLAG
  - logfile.h, [20](#)
- install\_sigaction
  - pf\_daemon.c, [23](#)
- libphonefirewall.h, [9](#)
  - add\_blacklist\_entry, [12](#)
  - add\_whitelist\_entry, [12](#)
  - check\_blacklist\_entry, [13](#)
  - check\_whitelist\_entry, [13](#)
  - DB\_FILE, [10](#)
  - entry, [16](#)
  - get\_blacklist\_entry\_by\_name, [14](#)
  - get\_blacklist\_entry\_by\_number, [14](#)
  - get\_whitelist\_entry\_by\_name, [15](#)
  - get\_whitelist\_entry\_by\_number, [15](#)
  - MAX\_LINE\_LENGTH, [10](#)
  - PRIO\_ALL, [10](#)
  - rm\_blacklist\_entry, [15](#)
  - rm\_whitelist\_entry, [16](#)
  - STMT\_SIZE, [10](#)
  - TB\_AREACODE, [11](#)
  - TB\_COUNTRYCODE, [11](#)
  - TB\_NAME, [11](#)
  - TB\_NUMBER, [11](#)
  - TB\_PRIORITY, [11](#)
  - TB\_REASON, [11](#)
- LOCKFILE
  - pf\_daemon.h, [26](#)
- LOGFILE
  - logfile.h, [20](#)
- logfile.c, [17](#)

- asctime, 17
- write\_logentry, 18
- logfile.h, 20
  - ERR\_FLAG, 20
  - INFO\_FLAG, 20
  - LOGFILE, 20
  - MAX\_ENTRY\_LENGTH, 20
  - UNKNOWN, 20
  - WARN\_FLAG, 21
  - write\_logentry, 21
- main
  - pf\_daemon.c, 24
- MAX\_ENTRY\_ARRAY
  - phonefirewall\_search.c, 32
- MAX\_ENTRY\_LENGTH
  - logfile.h, 20
- MAX\_LINE\_LENGTH
  - libphonefirewall.h, 10
- name
  - Entry, 6
- number
  - Entry, 5
- p\_entry
  - phonefirewall\_search.c, 34
- pf\_daemon.c, 23
  - dbus\_listen, 23
  - install\_sigaction, 23
  - main, 24
  - react\_to\_signal, 24
  - start\_daemon, 24
  - stop\_daemon, 25
- pf\_daemon.h, 26
  - dbus\_listen, 26
  - LOCKFILE, 26
  - start\_daemon, 26
  - stop\_daemon, 26
- phonefirewall\_administration.c, 27
  - add\_blacklist\_entry, 27
  - add\_whitelist\_entry, 28
  - check\_blacklist\_entry, 28
  - check\_whitelist\_entry, 29
  - evaluate\_stmt, 29
  - rm\_blacklist\_entry, 30
  - rm\_whitelist\_entry, 30
- phonefirewall\_search.c, 32
  - ASCII\_PERCENT\_CHAR, 32
  - entry\_array, 34
  - find\_entry\_by\_name, 33
  - get\_blacklist\_entry\_by\_name, 33
  - get\_blacklist\_entry\_by\_number, 33
  - get\_whitelist\_entry\_by\_name, 34
  - get\_whitelist\_entry\_by\_number, 34
  - MAX\_ENTRY\_ARRAY, 32
  - p\_entry, 34
  - PRIO\_ALL
    - libphonefirewall.h, 10
  - priority
    - Entry, 6
  - react\_to\_signal
    - pf\_daemon.c, 24
  - reason
    - Entry, 6
  - rm\_blacklist\_entry
    - libphonefirewall.h, 15
    - phonefirewall\_administration.c, 30
  - rm\_whitelist\_entry
    - libphonefirewall.h, 16
    - phonefirewall\_administration.c, 30
  - start\_daemon
    - pf\_daemon.c, 24
    - pf\_daemon.h, 26
  - STMT\_SIZE
    - libphonefirewall.h, 10
  - stop\_daemon
    - pf\_daemon.c, 25
    - pf\_daemon.h, 26
  - TB\_AREACODE
    - libphonefirewall.h, 11
  - TB\_COUNTRYCODE
    - libphonefirewall.h, 11
  - TB\_NAME
    - libphonefirewall.h, 11
  - TB\_NUMBER
    - libphonefirewall.h, 11
  - TB\_PRIORITY
    - libphonefirewall.h, 11
  - TB\_REASON
    - libphonefirewall.h, 11
  - UNKNOWN
    - logfile.h, 20
  - WARN\_FLAG
    - logfile.h, 21
  - write\_logentry
    - logfile.c, 18
    - logfile.h, 21