NAME

ERR_raise, ERR_raise_data, ERR_put_error, ERR_add_error_data, ERR_add_error_vdata, ERR_add_error_txt, ERR_add_error_mem_bio - record an error

SYNOPSIS

```
#include <openssl/err.h>

void ERR_raise(int lib, int reason);
void ERR_raise_data(int lib, int reason, const char *fmt, ...);

void ERR_add_error_data(int num, ...);
void ERR_add_error_vdata(int num, va_list arg);
void ERR_add_error_txt(const char *sep, const char *txt);
void ERR_add_error_mem_bio(const char *sep, BIO *bio);
```

The following function has been deprecated since OpenSSL 3.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
void ERR_put_error(int lib, int func, int reason, const char *file, int line);
```

DESCRIPTION

ERR_raise() adds a new error to the thread's error queue. The error occurred in the library **lib** for the reason given by the **reason** code. Furthermore, the name of the file, the line, and name of the function where the error occurred is saved with the error record.

ERR_raise_data() does the same thing as **ERR_raise()**, but also lets the caller specify additional information as a format string **fmt** and an arbitrary number of values, which are processed with **BIO_snprintf**(3).

ERR_put_error() adds an error code to the thread's error queue. It signals that the error of reason code **reason** occurred in function **func** of library **lib**, in line number **line** of **file**. This function is usually called by a macro.

ERR_add_error_data() associates the concatenation of its **num** string arguments as additional data with the error code added last. **ERR_add_error_vdata()** is similar except the argument is a **va_list**. Multiple calls to these functions append to the current top of the error queue. The total length of the string data per error is limited to 4096 characters.

ERR_add_error_txt() appends the given text string as additional data to the last error queue entry, after inserting the optional separator string if it is not NULL and the top error entry does not yet have

additional data. In case the separator is at the end of the text it is not appended to the data. The **sep** argument may be for instance "\n" to insert a line break when needed. If the associated data would become more than 4096 characters long (which is the limit given above) it is split over sufficiently many new copies of the last error queue entry.

ERR_add_error_mem_bio() is the same as **ERR_add_error_txt()** except that the text string is taken from the given memory BIO. It appends '\0' to the BIO contents if not already NUL-terminated.

ERR_load_strings(3) can be used to register error strings so that the application can a generate human-readable error messages for the error code.

Reporting errors

OpenSSL library reports

Each OpenSSL sub-library has library code **ERR_LIB_XXX** and has its own set of reason codes **XXX_R_...**. These are both passed in combination to **ERR_raise()** and **ERR_raise_data()**, and the combination ultimately produces the correct error text for the reported error.

All these macros and the numbers they have as values are specific to OpenSSL's libraries. OpenSSL reason codes normally consist of textual error descriptions. For example, the function **ssl3_read_bytes()** reports a "handshake failure" as follows:

```
ERR_raise(ERR_LIB_SSL, SSL_R_SSL_HANDSHAKE_FAILURE);
```

There are two exceptions:

ERR_LIB_SYS

This "library code" indicates that a system error is being reported. In this case, the reason code given to **ERR_raise()** and **ERR_raise_data()** *must* be **errno**(3).

```
ERR_raise(ERR_LIB_SYS, errno);
```

ERR_R_XXX

This set of error codes is considered global, and may be used in combination with any sub-library code.

```
ERR_raise(ERR_LIB_RSA, ERR_R_PASSED_INVALID_ARGUMENT);
```

Other pieces of software

Other pieces of software that may want to use OpenSSL's error reporting system, such as engines or applications, must normally get their own numbers.

- To get a "library" code, call **ERR_get_next_error_library**(3); this gives the calling code a dynamic number, usable for the duration of the process.

- Reason codes for each such "library" are determined or generated by the authors of that code. They must be numbers in the range 1 to 524287 (in other words, they must be nonzero unsigned 18 bit integers).

The exceptions mentioned in "OpenSSL library reports" above are valid for other pieces of software, i.e. they may use **ERR_LIB_SYS** to report system errors:

```
ERR_raise(ERR_LIB_SYS, errno);
```

... and they may use **ERR_R_XXX** macros together with their own "library" code.

```
int app_lib_code = ERR_get_next_error_library();

/* ... */

ERR_raise(app_lib_code, ERR_R_PASSED_INVALID_ARGUMENT);
```

RETURN VALUES

ERR_raise(), **ERR_raise_data()**, **ERR_put_error()**, **ERR_add_error_data()**, **ERR_add_error_vdata()** **ERR_add_error_txt()**, and **ERR_add_error_mem_bio()** return no values.

NOTES

ERR_raise(), **ERR_raise()** and **ERR_put_error()** are implemented as macros.

SEE ALSO

ERR_load_strings(3), **ERR_get_next_error_library**(3)

HISTORY

ERR_raise, ERR_raise_data, **ERR_add_error_txt()** and **ERR_add_error_mem_bio()** were added in OpenSSL 3.0.

COPYRIGHT