Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'OSSL_CMP_LOG_DEBUG.3ossl' command

$ man OSSL_CMP_LOG_DEBUG.3ossl

OSSL_CMP_LOG_OPEN(3ossl) OpenSSL OSSL_CMP_LOG_OPEN(3ossl)

NAME

OSSL_CMP_log_open, OSSL_CMP_log_close, OSSL_CMP_severity,

OSSL_CMP_LOG_EMERG, OSSL_CMP_LOG_ALERT, OSSL_CMP_LOG_CRIT,

OSSL_CMP_LOG_ERR, OSSL_CMP_LOG_WARNING, OSSL_CMP_LOG_NOTICE,

OSSL_CMP_LOG_INFO, OSSL_CMP_LOG_DEBUG, OSSL_CMP_LOG_TRACE,

OSSL_CMP_log_cb_t, OSSL_CMP_print_to_bio, OSSL_CMP_print_errors_cb -

functions for logging and error reporting

SYNOPSIS

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

int  OSSL_CMP_log_open(void);
void OSSL_CMP_log_close(void);

/* severity level declarations resemble those from syslog.h */
typedef int OSSL_CMP_severity;
#define OSSL_CMP_LOG_EMERG   0
#define OSSL_CMP_LOG_ALERT   1
#define OSSL_CMP_LOG_CRIT    2
#define OSSL_CMP_LOG_ERR     3
```

```
#define OSSL_CMP_LOG_WARNING 4
#define OSSL_CMP_LOG_NOTICE  5
#define OSSL_CMP_LOG_INFO    6
#define OSSL_CMP_LOG_DEBUG   7
#define OSSL_CMP_LOG_TRACE   8

typedef int (*OSSL_CMP_log_cb_t)(const char *component,
                                 const char *file, int line,
                                 OSSL_CMP_severity level, const char *msg);
int OSSL_CMP_print_to_bio(BIO *bio, const char *component, const char *file,
                          int line, OSSL_CMP_severity level, const char *msg);
void OSSL_CMP_print_errors_cb(OSSL_CMP_log_cb_t log_fn);
```

DESCRIPTION

The logging and error reporting facility described here contains convenience functions for CMP-specific logging, including a string prefix mirroring the severity levels of syslog.h, and enhancements of the error queue mechanism needed for large diagnostic messages produced by the CMP library in case of certificate validation failures.

When an interesting activity is performed or an error occurs, some detail should be provided for user information, debugging, and auditing purposes. A CMP application can obtain this information by providing a callback function with the following type:

```
typedef int (*OSSL_CMP_log_cb_t)(const char *component,
                                 const char *file, int line,
                                 OSSL_CMP_severity level, const char *msg);
```

The parameters may provide some component info (which may be a module name and/or function name) or NULL, a file pathname or NULL, a line number or 0 indicating the source code location, a severity level, and a message string describing the nature of the event, terminated by

'\n'.

Even when an activity is successful some warnings may be useful and some degree of auditing may be required. Therefore, the logging facility supports a severity level and the callback function has a level parameter indicating such a level, such that error, warning, info, debug, etc. can be treated differently. The callback is activated only when the severity level is sufficient according to the current level of verbosity, which by default is OSSL_CMP_LOG_INFO.

The callback function may itself do non-trivial tasks like writing to a log file or remote stream, which in turn may fail. Therefore, the function should return 1 on success and 0 on failure.

OSSL_CMP_log_open() initializes the CMP-specific logging facility to output everything to STDOUT. It fails if the integrated tracing is disabled or STDIO is not available. It may be called during application startup. Alternatively, OSSL_CMP_CTX_set_log_cb(3) can be used for more flexibility. As long as neither if the two is used any logging output is ignored.

OSSL_CMP_log_close() may be called when all activities are finished to flush any pending CMP-specific log output and deallocate related resources. It may be called multiple times. It does get called at OpenSSL stutdown.

OSSL_CMP_print_to_bio() prints the given component info, filename, line number, severity level, and log message or error queue message to the given bio. component usually is a function or module name. If it is NULL, empty, or "(unknown function)" then "CMP" is used as fallback.

OSSL_CMP_print_errors_cb() outputs any entries in the OpenSSL error queue. It is similar to ERR_print_errors_cb(3) but uses the CMP log

callback function log_fn for uniformity with CMP logging if not NULL. Otherwise it prints to STDERR using OSSL_CMP_print_to_bio(3) (unless OPENSSL_NO_STDIO is defined).

RETURN VALUES

OSSL_CMP_log_close() and OSSL_CMP_print_errors_cb() do not return anything.

All other functions return 1 on success, 0 on error.

HISTORY

The OpenSSL CMP support was added in OpenSSL 3.0.

COPYRIGHT