Rocky Enterprise Linux 9.2 Manual Pages on command 'fips_module_indicators.7ossl'

$ man fips_module_indicators.7ossl

FIPS_MODULE_INDICATORS(7ossl) OpenSSL FIPS_MODULE_INDICATORS(7ossl)

NAME

fips_module_indicators - Red Hat OpenSSL FIPS module indicators guide

DESCRIPTION

This guide documents how the Red Hat Enterprise Linux 9 OpenSSL FIPS provider implements Approved Security Service Indicators according to the FIPS 140-3 Implementation Guidelines, section 2.4.C. See

<https://csrc.nist.gov/CSRC/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS%20140-3%20IG.pdf>

for the FIPS 140-3 Implementation Guidelines.

For all approved services except signatures, the Red Hat OpenSSL FIPS provider uses the return code as the indicator as understood by FIPS 140-3. That means that every operation that succeeds denotes use of an approved security service. Operations that do not succeed may not have

been approved security services, or may have been used incorrectly.

For signatures, an explicit indicator API is available to determine whether a selected operation is an approved security service, in combination with the return code of the operation. For a signature operation to be approved, the explicit indicator must claim it as approved, and it must succeed.

Querying the explicit indicator

The Red Hat OpenSSL FIPS provider exports a symbol named redhat_ossl_query_fipsindicator that provides information on which signature operations are approved security functions. To use this function, either link against fips.so directly, or load it at runtime using dlopen(3) and dlsym(3).

```
#include <openssl/core_dispatch.h>
#include "providers/fips/indicator.h"

void *provider = dlopen("/usr/lib64/ossl-modules/fips.so", RTLD_LAZY);
if (provider == NULL) {
    fprintf(stderr, "%s\n", dlerror());
    // handle error
}

const OSSL_RH_FIPSINDICATOR_ALORITHM *(*redhat_ossl_query_fipsindicator)(int) \
    = dlsym(provider, "redhat_ossl_query_fipsindicator");
if (redhat_ossl_query_fipsindicator == NULL) {
    fprintf(stderr, "%s\n", dlerror());
    fprintf(stderr, "Does your copy of fips.so have the required Red Hat"
                    " patches?\n");
    // handle error
}
```

Note that this uses the providers/fips/indicator.h header, which is not public. Install the openssl-debugsource package from the BaseOS-debuginfo repository using dnf debuginfo-install openssl and include /usr/src/debug/openssl-3.*/ in the compiler's include path.

redhat_ossl_query_fipsindicator expects an operation ID as its only argument. Currently, the only supported operation ID is OSSL_OP_SIGNATURE to obtain the indicators for signature operations. On success, the return value is a pointer to an array of OSSL_RH_FIPSINDICATOR_STRUCTs. On failure, NULL is returned. The last entry in the array is indicated by algorithm_names being NULL.

```
typedef struct ossl_rh_fipsindicator_algorithm_st {
    const char *algorithm_names;     /* key */
    const char *property_definition; /* key */
    const OSSL_RH_FIPSINDICATOR_DISPATCH *indicators;
} OSSL_RH_FIPSINDICATOR_ALGORITHM;

typedef struct ossl_rh_fipsindicator_dispatch_st {
    int function_id;
    int approved;
} OSSL_RH_FIPSINDICATOR_DISPATCH;
```

The algorithm_names field is a colon-separated list of algorithm names from one of the PROV_NAMES_... constants, e.g., PROV_NAMES_RSA. strtok(3) can be used to locate the appropriate entry. See the example below, where algorithm contains the algorithm name to search for:

```
const OSSL_RH_FIPSINDICATOR_DISPATCH *indicator_dispatch = NULL;
const OSSL_RH_FIPSINDICATOR_ALGORITHM *indicator =
    redhat_ossl_query_fipsindicator(operation_id);
if (indicator == NULL) {
    fprintf(stderr, "No indicator for operation, probably using implicit"
```

```
                    " indicators.\n");
        // handle error
    }

    for (; indicator->algorithm_names != NULL; ++indicator) {
        char *algorithm_names = strdup(indicator->algorithm_names);
        if (algorithm_names == NULL) {
            perror("strdup(3)");
            // handle error
        }

        const char *algorithm_name = strtok(algorithm_names, ":");
        for (; algorithm_name != NULL; algorithm_name = strtok(NULL, ":")) {
            if (strcasecmp(algorithm_name, algorithm) == 0) {
                indicator_dispatch = indicator->indicators;
                free(algorithm_names);
                algorithm_names = NULL;
                break;
            }
        }
        free(algorithm_names);
    }
    if (indicator_dispatch == NULL) {
        fprintf(stderr, "No indicator for algorithm %s.\n", algorithm);
        // handle error
    }
```

If an appropriate OSSL_RH_FIPSINDICATOR_DISPATCH array is available for the given algorithm name, it maps function IDs to their approval status. The last entry is indicated by a zero function_id. approved is OSSL_RH_FIPSINDICATOR_APPROVED if the operation is an approved security service, or part of an approved security service, or OSSL_RH_FIPSINDICATOR_UNAPPROVED otherwise. Any other value is invalid.

Function IDs are OSSL_FUNC_* constants from openssl/core_dispatch.h, e.g., OSSL_FUNC_SIGNATURE_DIGEST_SIGN_UPDATE or OSSL_FUNC_SIGNATURE_SIGN.

Assuming function_id is the function in question, the following code can be used to query the approval status:

```
for (; indicator_dispatch->function_id != 0; ++indicator_dispatch) {
    if (indicator_dispatch->function_id == function_id) {
        switch (indicator_dispatch->approved) {
            case OSSL_RH_FIPSINDICATOR_APPROVED:
                // approved security service
                break;
            case OSSL_RH_FIPSINDICATOR_UNAPPROVED:
                // unapproved security service
                break;
            default:
                // invalid result
                break;
        }
        break;
    }
}
```

SEE ALSO

fips_module(7), provider(7)

COPYRIGHT

<https://www.openssl.org/source/license.html>.