

NAME

curl_version_info – returns runtime libcurl version info

SYNOPSIS

```
#include <curl/curl.h>
```

```
curl_version_info_data *curl_version_info(CURLversion age);
```

DESCRIPTION

Returns a pointer to a filled in static struct with information about various features in the running version of libcurl. The input argument *age* has no use and we recommend you set it to *CURLVERSION_NOW*.

This function may alter the returned static data as long as *curl_global_init(3)* has not been called. It is therefore not thread-safe before libcurl initialization occurs.

The curl_version_info_data struct looks like this

```
typedef struct {
    CURLversion age;      /* see description below */

    const char *version;   /* human readable string */
    unsigned int version_num; /* numeric representation */
    const char *host;      /* human readable string */
    int features;          /* bitmask, see below */
    char *ssl_version;     /* human readable string */
    long ssl_version_num;  /* not used, always zero */
    const char *libz_version; /* human readable string */
    const char *const *protocols; /* protocols */

    /* when 'age' is CURLVERSION_SECOND or higher, the members below exist */
    const char *ares;      /* human readable string */
    int ares_num;          /* number */

    /* when 'age' is CURLVERSION_THIRD or higher, the members below exist */
    const char *libidn;    /* human readable string */

    /* when 'age' is CURLVERSION_FOURTH or higher (>= 7.16.1), the members
       below exist */
    int iconv_ver_num;     /* '_libiconv_version' if iconv support enabled */

    const char *libssh_version; /* human readable string */

    /* when 'age' is CURLVERSION_FIFTH or higher (>= 7.57.0), the members
       below exist */
    unsigned int brotli_ver_num; /* Numeric Brotli version
                                  (MAJOR << 24) | (MINOR << 12) | PATCH */
    const char *brotli_version; /* human readable string. */

    /* when 'age' is CURLVERSION_SIXTH or higher (>= 7.66.0), the members
       below exist */
    unsigned int nghttp2_ver_num; /* Numeric nghttp2 version
                                   (MAJOR << 16) | (MINOR << 8) | PATCH */
    const char *nghttp2_version; /* human readable string. */

    const char *quic_version; /* human readable quic (+ HTTP/3) library +
                               version or NULL */
}
```

```

/* when 'age' is CURLVERSION_SEVENTH or higher (>= 7.70.0), the members
   below exist */
const char *cainfo; /* the built-in default CURLOPT_CAINFO, might
                     be NULL */
const char *capath; /* the built-in default CURLOPT_CAPATH, might
                     be NULL */
/* when 'age' is CURLVERSION_EIGHTH or higher (>= 7.71.0), the members
   below exist */
unsigned int zstd_ver_num; /* Numeric Zstd version
                           (MAJOR << 24) | (MINOR << 12) | PATCH */
const char *zstd_version; /* human readable string. */
/* when 'age' is CURLVERSION_NINTH or higher (>= 7.75.0), the members
   below exist */
const char *hyper_version; /* human readable string. */
/* when 'age' is CURLVERSION_TENTH or higher (>= 7.77.0), the members
   below exist */
const char *gsasl_version; /* human readable string. */
/* when 'age' is CURLVERSION_ELEVENTH or higher (>= 7.87.0), the members
   below exist */
const char *const *feature_names; /* Feature names. */
/* when 'age' is CURLVERSION_TWELFTH or higher (>= 8.8.0), the members
   below exist */
const char *const *rtmp_version; /* human readable string */
} curl_version_info_data;

```

age describes what the age of this struct is. That number is different depending on how recent your libcurl is. The documentation above describes which struct fields that were added at which age. Trying to access a struct field that is newer than the age of your struct may cause undefined behavior and possibly crashes.

version is an ASCII string for the libcurl version.

version_num is a 24-bit number created like this: <8 bits major number> | <8 bits minor number> | <8 bits patch number>. Version 7.9.8 is therefore returned as 0x070908.

host is an ASCII string showing what host information that this libcurl was built for. As discovered by a configure script or set by the build environment.

features is a bit mask representing available features. It can have none, one or more bits set. The use of this field is deprecated: use *feature_names* instead. The feature names description below lists the associated bits.

feature_names is a pointer to an array of string pointers, containing the names of the features that libcurl supports. The array is terminated by a NULL entry. See the list of features names below.

ssl_version is an ASCII string for the TLS library name + version used. If libcurl has no SSL support, this is NULL. For example "Schannel", "Secure Transport" or "OpenSSL/1.1.0g". For MultiSSL builds the string contains all SSL backend names and the inactive backend names are in parentheses. For example "(OpenSSL/3.0.8) Schannel" or "OpenSSL/3.0.8 (Schannel)".

ssl_version_num is always 0.

libz_version is an ASCII string (there is no numerical version). If libcurl has no libz support, this is NULL.

protocols is a pointer to an array of char * pointers, containing the names protocols that libcurl supports

(using lowercase letters). The protocol names are the same as would be used in URLs. The array is terminated by a NULL entry.

FEATURES

alt-svc *features* mask bit: CURL_VERSION_ALTSVC

HTTP Alt-Svc parsing and the associated options (Added in 7.64.1)

AppleSecTrust

features mask bit: non-existent

libcurl was built with support for Apple's SecTrust service to verify server certificates (Added in 8.17.0).

AsynchDNS

features mask bit: CURL_VERSION_ASYNCDNS

libcurl was built with support for asynchronous name lookups, which allows more exact timeouts (even on Windows) and less blocking when using the multi interface.

brotil *features* mask bit: CURL_VERSION_BROTLI

supports HTTP Brotli content encoding using libbrotldec

asyn-rr *features* mask bit: non-existent

libcurl was built to use c-ares for EXPERIMENTAL HTTPS resource record resolves, but uses the threaded resolver for "normal" resolves (Added in 8.12.0)

Debug *features* mask bit: CURL_VERSION_DEBUG

libcurl was built with debug capabilities

ECH *features* mask bit: non-existent

libcurl was built with ECH support (experimental, added in 8.8.0)

gsasl *features* mask bit: CURL_VERSION_GSASL

libcurl was built with libgsasl and thus with some extra SCRAM-SHA authentication methods. (added in 7.76.0)

GSS-API

features mask bit: CURL_VERSION_GSSAPI

libcurl was built with support for GSS-API. This makes libcurl use provided functions for Kerberos and SPNEGO authentication. It also allows libcurl to use the current user credentials without the app having to pass them on.

HSTS *features* mask bit: CURL_VERSION_HSTS

libcurl was built with support for HSTS (HTTP Strict Transport Security) (Added in 7.74.0)

HTTP2 *features* mask bit: CURL_VERSION_HTTP2

libcurl was built with support for HTTP2.

HTTP3 *features* mask bit: CURL_VERSION_HTTP3

HTTP/3 and QUIC support are built-in (Added in 7.66.0)

HTTPS-proxy

features mask bit: CURL_VERSION_HTTPS_PROXY

libcurl was built with support for HTTPS-proxy.

HTTPSRR

features mask bit: non-existent

libcurl was built with EXPERIMENTAL support for HTTPS resource records (Added in 8.12.0)

IDN

features mask bit: CURL_VERSION_IDN

libcurl was built with support for IDNA, domain names with international letters.

IPv6

features mask bit: CURL_VERSION_IPV6

supports IPv6

Kerberos

features mask bit: CURL_VERSION_KERBEROS5

supports Kerberos V5 authentication for FTP, IMAP, LDAP, POP3, SMTP and SOCKSv5 proxy.

Largefile

features mask bit: CURL_VERSION_LARGEFILE

libcurl was built with support for large files.

libz

features mask bit: CURL_VERSION_LIBZ

supports HTTP deflate using libz

MultiSSL

features mask bit: CURL_VERSION_MULTI_SSL

libcurl was built with multiple SSL backends. For details, see *curl_global_sslset(3)*.

NativeCA

features mask bit: non-existent

libcurl was built to enable native CA store, to verify server certificates (Added in 8.19.0).

NTLM

features mask bit: CURL_VERSION_NTLM

supports HTTP NTLM

NTLM_WB

features mask bit: CURL_VERSION_NTLM_WB

libcurl was built with support for NTLM delegation to a winbind helper. This feature was removed from curl in 8.8.0.

PSL

features mask bit: CURL_VERSION_PSL

libcurl was built with support for Mozilla's Public Suffix List. This makes libcurl ignore cookies with a domain that is on the list.

SPNEGO

features mask bit: CURL_VERSION_SPNEGO

libcurl was built with support for SPNEGO authentication (Simple and Protected GSS-API Negotiation Mechanism, defined in RFC 2478.)

SSL *features* mask bit: CURL_VERSION_SSL

supports SSL (HTTPS/FTPS)

SSLS-EXPORT

features mask bit: non-existent

libcurl was built with SSL session import/export support (experimental, added in 8.12.0)

SSPI *features* mask bit: CURL_VERSION_SSPI

libcurl was built with support for SSPI. This is only available on Windows and makes libcurl use Windows-provided functions for Kerberos, NTLM, SPNEGO and Digest authentication. It also allows libcurl to use the current user credentials without the app having to pass them on.

threadsafe

features mask bit: CURL_VERSION_THREADSAFE

libcurl was built with thread-safety support (Atomic or SRWLOCK) to protect curl initialization. (Added in 7.84.0) See *libcurl-thread(3)*

TLS-SRP

features mask bit: CURL_VERSION_TLSAUTH_SRP

libcurl was built with support for TLS-SRP (in one or more of the built-in TLS backends).

Unicode

features mask bit: CURL_VERSION_UNICODE

libcurl was built with Unicode support on Windows. This makes non-ASCII characters work in filenames and options passed to libcurl. (Added in 7.72.0)

UnixSockets

features mask bit: CURL_VERSION_UNIX_SOCKETS

libcurl was built with support for Unix domain sockets.

zstd *features* mask bit: CURL_VERSION_ZSTD

supports HTTP zstd content encoding using zstd library (Added in 7.72.0)

no name

features mask bit: CURL_VERSION_CONV

libcurl was built with support for character conversions provided by callbacks. Always 0 since 7.82.0. Deprecated.

no name

features mask bit: CURL_VERSION_CURLDEBUG

libcurl was built with memory tracking debug capabilities. This is mainly of interest for libcurl hackers. Always the same as CURL_VERSION_DEBUG since 8.19.0. Deprecated.

no name

features mask bit: CURL_VERSION_GSSNEGOTIATE

supports HTTP GSS-Negotiate. Deprecated.

no name

features mask bit: CURL_VERSION_KERBEROS4

supports Kerberos V4 (when using FTP). Legacy bit. Deprecated since 7.33.0.

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
int main(void)
{
    curl_version_info_data *ver = curl_version_info(CURLVERSION_NOW);
    printf("libcurl version %u.%u.%u\n",
        (ver->version_num >> 16) & 0xff,
        (ver->version_num >> 8) & 0xff,
        ver->version_num & 0xff);
}
```

AVAILABILITY

Added in curl 7.10

RETURN VALUE

A pointer to a curl_version_info_data struct.

SEE ALSO

curl_version(3)