Network Working Group
Internet-Draft
Category: Best Current Practice
Intended status: Standards Track
Expires: May 12, 2014

W. Hoehlhubmer

Nov 12, 2013

Informational Add-on for HTTP over the Secure Sockets Layer (SSL) Protocol and/or the Transport Layer Security (TLS) Protocol

draft-hoehlhubmer-https-upd-08

Abstract

This document describes an Add-on as a good practice for websites providing encrypted connectivity (HTTP over TLS).

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 12, 2014.

Copyright Notice

Table of Contents

1. Introduction

HTTP over TLS [HTTPTLS] is not limited to e.g. electronic banking sites. E-commerce is also using this technology on their websites for encrypted communication between clients (users) and them.

A list of a few encryption algorithms:

(1) Advanced Encryption Standard (AES)
(2) Data Encryption Standard (DES, 3DES)
(3) Ron's Code 4 (RC4)
(4) ...

As an example a list of some kinds of the Camellia encryption algorithm [CAMELLIA] (names taken from OpenSSL help [OPENSSL]):

(1) CAMELLIA-128-CBC: 128-bit Camellia encryption in CBC mode
(2) CAMELLIA-128-ECB: 128-bit Camellia encryption in ECB mode
(3) CAMELLIA-192-CBC: 192-bit Camellia encryption in CBC mode
(4) CAMELLIA-192-ECB: 192-bit Camellia encryption in ECB mode
(5) CAMELLIA-256-CBC: 256-bit Camellia encryption in CBC mode
(6) CAMELLIA-256-ECB: 256-bit Camellia encryption in ECB mode

A list of possible secure layer used:

(1) The Secure Sockets Layer (SSL) Protocol:
    (1a) Version 2.0 [SSLv2]
    (1b) Version 3.0 [SSLv3]

(2) The Transport Layer Security (TLS) Protocol:
    (2a) Version 1.0 [TLSv1.0]
    (2b) Version 1.1 [TLSv1.1]
    (2c) Version 1.2 [TLSv1.2]

A list of possible Ciphersuites for Transport Layer Security (TLS):

(1) Pre-Shared Key Cipher Suites [RFC4279]
(2) Elliptic Curve Cryptography (ECC) Cipher Suites [RFC4492]
(3) Pre-Shared Key Cipher Suites with NULL Encryption [RFC4785]
(4) AES Galois Counter Mode (GCM) Cipher Suites [RFC5288]
(5) DES and IDEA Cipher Suites [RFC5469]
(6) ECDHE_PSK Cipher Suites [RFC5489]
(7) Camellia Cipher Suites [RFC5932]
(8) ...

A list of possible Hashing Algorithms:

(1) the [MD2] Message-Digest Algorithm (historic see [RFC6149])
(2) the [MD4] Message-Digest Algorithm (historic see [RFC6150])
(3) the [MD5] Message-Digest Algorithm used commonly in past
(4) the US Secure Hash Algorithm 1 [SHA1]
(5) more US Secure Hash Algorithms [RFC6234]
(6) ...

Only the X.509 Certificates [PKIX] are static, all other
informations depend on the capabilities of the used web browser.

Not every browser allows you to view all these informations,
especially the Cipher Suite the browser has picked for use.

With most browsers let you view the used X.509 certificates of the
actual session, but you have no direct comparison if they are the
correct ones.

The X.509 certificates which are shown by the browser and those,
that are shown in this Add-on MUST match; with other words:
if they don not match, there is going on a man-in-the-middle attack.

It is a good practice to show these informations on the website.

2. Implementing this Add-on

This Add-on is just one page of the website. Its content MUST be completely generated on server side. The Common Gateway Interface [CGI1.1] is RECOMMENDED to be used. There MUST exist at least one relative reference to this page as defined in [RFC3986] Section 4.2.

For doing so see the sample scripts at Appendix A.
To see how this Add-on works, see [MYADDON].

2.1. Formating and Presenting of this Add-on

You SHALL present this information simple, plain Text is enough. When using HTML, only relative references as defined in [RFC3986] Section 4.2. MAY be used. It is RECOMMENDED to use only a subset of [HTML2.0].

This content MAY be presented AS IS without doing any translation.

Presenting this content in sorted order is OPTIONAL.

2.2. Content of this Add-on

The informations MUST be the following:

(1) The actual date and time formatted as specified in [RFC5322] Section 3.3. This MUST NOT differ more than 5 seconds from the real date/time

(2) The cipher specification name

(3) Number of cipher bits (actually used)
(4) Number of cipher bits (possible)

(5) The SSL Protocol version: SSLv2, SSLv3, TLSv1.0, TLSv1.1, TLSv1.2, ...

(6) If cipher is an export cipher: false, true
(7) If secure renegotiation is supported: false, true

(8) Algorithm used for the public key of server's certificate
(9) Algorithm used for the signature of server's certificate
(10) Issuer DN of server's certificate
(11) Subject DN in server's certificate
(12) The serial of the server certificate
(13) The version of the server certificate
(14) Validity of server's certificate (start time)
(15) Validity of server's certificate (end time)

(16) Client certificate verification:
NONE, SUCCESS, GENEROUS or FAILED:reason

(17) SSL compression method negotiated: NULL when disabled

For connections where X.509 certificates are used for authentication these informations are RECOMMENDED:

(18) Algorithm used for the public key of client's certificate
(19) Algorithm used for the signature of client's certificate
(20) Issuer DN of client's certificate
(21) Subject DN in client's certificate
(22) The serial of the client certificate
(23) The version of the client certificate
(24) Validity of client's certificate (start time)
(25) Validity of client's certificate (end time)
(26) Number of days until client's certificate expires

This information MAY be given:

(27) The hex-encoded SSL session id
(28) Contents of the SNI TLS extension (if supplied with ClientHello)

These OPTIONAL informations depend on the used software:

(29) The SSL-module program version: e.g. Apache mod_ssl version
(30) The SSL program version: e.g. OpenSSL version

See Appendix B for a sample content.

3. IANA Considerations

There are no requests for IANA actions in this document.

4. Security Considerations

When implementing this information as a popup window in the browser, this information MUST also be available with enabled popup-blocker.

The Implementation MUST NOT use any scripts, that run on client side: e.g. Javascript, ...

There SHOULD also be no references to other websites inside this Add-on page.

5. Acknowledgements

6. Recommendations

Using a standardized URL is RECOMMENDED, for more see Section 8.

7. References

7.1. Normative References

[HTTPTLS] Rescorla, E., "HTTP over TLS", RFC 2818, May 2000.

7.2. Informative References

[CAMELLIA] Matsui, M., Nakajima, J., and S. Moriai, "A Description of the Camellia Encryption Algorithm", RFC 3713, April 2004.

[PKIX] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[CGI1.1] Robinson, D. and K. Coar, "The Common Gateway Interface (CGI) Version 1.1", RFC 3875, October 2004.

[HTML2.0] Berners-Lee, T. and D. Connolly, "Hypertext Markup Language - 2.0", RFC 1866, November 1995.

[MD2] Kaliski, B., "The MD2 Message-Digest Algorithm", RFC 1319, April 1992.

[MD4] Rivest, R., "The MD4 Message-Digest Algorithm", RFC 1320, April 1992.

[MD5] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[SHA1] Eastlake 3rd, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, September 2001.

[SSLv2] Hickman, Kipp, "The SSL Protocol", Netscape Communications Corp., Feb 9, 1995.

[SSLv3] Freier, A., Karlton, P., and P. Kocher, "The Secure Sockets Layer (SSL) Protocol Version 3.0", RFC 6101, August 2011.

[TLS1.0] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[TLS1.1] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[TLS1.2] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[OPENSSL] OpenSSL Cryptography and SSL/TLS Toolkit at http://www.openssl.org/

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4279] Eronen, P., Ed., and H. Tschofenig, Ed., "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.

[RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.

[RFC4785] Blumenthal, U. and P. Goel, "Pre-Shared Key (PSK) Ciphersuites with NULL Encryption for Transport Layer Security (TLS)", RFC 4785, January 2007.

[RFC5288] Salowey, J., Choudhury, A., and D. McGrew, "AES Galois Counter Mode (GCM) Cipher Suites for TLS", RFC 5288, August 2008.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.

[RFC5469] Eronen, P., Ed., "DES and IDEA Cipher Suites for Transport Layer Security (TLS)", RFC 5469, February 2009.

[RFC5489] Badra, M. and I. Hajjeh, "ECDHE_PSK Cipher Suites for Transport Layer Security (TLS)", RFC 5489, March 2009.

[RFC5932] Kato, A., Kanda, M., and S. Kanno, "Camellia Cipher Suites for TLS", RFC 5932, June 2010.

[RFC6149] Turner, S. and L. Chen, "MD2 to Historic Status", RFC 6149, March 2011.

[RFC6150] Turner, S. and L. Chen, "MD4 to Historic Status", RFC 6150, March 2011.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, May 2011.

[MYADDON] A working implementation of this Add-on on my website at https://ssl.mathemainzel.info/sslinfo/

8. Discussions

It would be good to have a standardized URL for this Add-on;
e.g. https://www.example.com/sslinfo/

To place an Absolute URI as defined in [RFC3986] Section 4.3.
outside the encrypted website part is RECOMMENDED.

Appendix A. Script Examples

Use the following script examples as a template for your implementation of this Add-on.

The first two examples generate identical content in plain ASCII-text, the third example makes use of HTML and is a compiled C program.

Script Examples:

(1) Example 1. PHP-script
(2) Example 2. CGI-script: can be used on most Linux systems
(3) Example 3. CGI-script: can be used on any system

Example 1. PHP-script

```
<CODE BEGINS>
<?php

header( "Content-type: text/plain" );

print "SSL informations: " . date( "r" ) . "\r\n";
print "================\r\n\r\n";

if ( isset( $_SERVER['HTTPS'] ) && ( $_SERVER['HTTPS'] == "on" ) ) {
  $list = array( );
  $nmbrOfValues = 0;

  foreach ( $_SERVER as $key => $value ) {
    if ( substr( $key, 0, 4 ) == "SSL_" ) {
      $list[ $nmbrOfValues++ ] = $key . "=" . $value;
    }
  }

  sort( $list );   // sort content before printing ...

  for ( $iter = 0; $iter < $nmbrOfValues; $iter++ ) {
    print $list[ $iter ] . "\r\n";
  }
}
else {
  echo "No SSL information available.\r\n";
}
?>
<CODE ENDS>
```

Example 2. CGI-script: can be used on most Linux systems

```
<CODE BEGINS>
#!/bin/sh

printf "Content-type: text/plain\n\n"

printf "SSL informations: $(date --rfc-2822)\n"
printf "================\n\n"

if [ "$HTTPS" == "on" ]; then
  env | grep --regexp="^SSL_" | sort
else
  printf "No SSL information available.\n"
fi
<CODE ENDS>
```

Example 3. CGI-script: can be used on any system

This CGI-script is a compiled C program, and in comparison to the other 2 examples, it makes use of HTML.

For compiling this program any compiler SHOULD be suitable. Be sure your runtime supports the function strftime with standard format specifiers.

```
<CODE BEGINS>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

#ifdef __linux__
#include <unistd.h>
#endif

const char* pszHtmlEndPart [ ] = { "<HR>",
  "<ADDRESS>https at www.example.com Port 443</ADDRESS>",
  "</BODY>",
  "</HTML>" };

const char* pszHtmlBeginPart[ ] = {
  "<!DOCTYPE HTML PUBLIC \"-//IETF//DTD HTML 2.0//EN\">",
  "<HTML>",
  "<HEAD>",
  "<TITLE>SSL informations</TITLE>",
  "</HEAD>",
```

```
    "<BODY>",
    "<H3>SSL informations</H3>" };

  /* function prototype used for sorting */
  int compareFunc( const void* pvd1, const void* pvd2 );

  int main( int argc, char* argv[ ], char** envp )
  {                               /* char* envp[ ] */
    char* * ppszContent;
    char* * ppsz;
    char* psz;
    char szDateTime[ 80 ];
    int i, nCount;

    time_t tnow = time( NULL );
    struct tm* tmnow = localtime( &tnow );

    strftime( szDateTime, sizeof( szDateTime ) - 4,
      "%a, %d %b %Y %H:%M:%S %z", tmnow );

    printf( "Content-type: text/html; charset=ISO-8859-1\r\n\r\n" );

    nCount = sizeof( pszHtmlBeginPart ) / sizeof( char* );
    for ( i = 0; i < nCount; i++ )
      printf( "%s\r\n", pszHtmlBeginPart[ i ] );

    printf( "<B>SSL informations</B>: %s\r\n", szDateTime );
    printf( "<P>\r\n" );

    if ( ( psz = getenv( "HTTPS" ) ) && ( strcmp( psz, "on" ) == 0 ) )
    {
      /* count relevant values ... */
      ppsz = envp;
      nCount = 0;
      while ( ppsz && *ppsz )
      {
        if ( strncmp( *ppsz, "SSL_", 4 ) == 0 )
          nCount++;
        ppsz++;
      }

      /* allocate memory */
      ppszContent = (char* *) calloc( nCount, sizeof( char* ) );
```

```
      if ( ppszContent )
      {
        /* extract relevant values from environment ... */
        i = 0;
        ppsz = envp;
        while ( ppsz && *ppsz )
        {
          if ( strncmp( *ppsz, "SSL_", 4 ) == 0 )
            *( ppszContent + i++ ) = *ppsz;
          ppsz++;
        }

        /* sort content */
        qsort( (void*) ppszContent, nCount, sizeof( char* ),
          compareFunc );

        printf( "<CODE>\r\n" );

        /* output sorted content */
        for ( i = 0; i < nCount; i++ )
          printf( "%s<BR>\r\n", *( ppszContent + i ) );

        printf( "</CODE>\r\n" );

        /* free up memory */
        free( (void*) ppszContent );
      }
      else
        printf( "Internal error (unable to allocate memory).\r\n" );
    }
    else
      printf( "No SSL information available.\r\n" );

    nCount = sizeof( pszHtmlEndPart ) / sizeof( char* );
    for ( i = 0; i < nCount; i++ )
      printf( "%s\r\n", pszHtmlEndPart[ i ] );

    return 0;
  }

  /* comparison function for sorting */
  int compareFunc( const void* pvd1, const void* pvd2 )
  {
    return strcmp( *( (char* *) pvd1 ), *( (char* *) pvd2 ) );
  }
  <CODE ENDS>
```

Appendix B. Add-on Sample Content

The first example shows a complete Add-on sample content in sorted order. The second example shows the client certificate part, in case client certificate authentication is used.
The other two examples show only the part that may differ when the browser picks another cipher suite.

For meaning of the numbers in brackets of the examples see Section 2.1.

(1) Example 1. A complete sample content
(1a) Example 1a. ..., the client certificate part
(2) Example 2.
(3) Example 3.

Example 1. A complete sample content

```
SSL informations: Thu, 01 Jan 1970 00:00:00 +0000              (1)
================

SSL_CIPHER=AES256-SHA                                          (2)
SSL_CIPHER_ALGKEYSIZE=256                                      (4)
SSL_CIPHER_EXPORT=false                                        (6)
SSL_CIPHER_USEKEYSIZE=256                                      (3)
SSL_CLIENT_VERIFY=NONE                                         (16)
SSL_COMPRESS_METHOD=NULL                                       (17)
SSL_PROTOCOL=TLSv1                                             (5)
SSL_SECURE_RENEG=true                                          (7)
SSL_SERVER_A_KEY=rsaEncryption                                 (8)
SSL_SERVER_A_SIG=sha1WithRSAEncryption                         (9)
SSL_SERVER_I_DN=/C=--/O=SomeOrg/OU=SomeOrgUnit/CN=Root CA      (10)
SSL_SERVER_I_DN_C=--                                           (10)
SSL_SERVER_I_DN_CN=Root CA                                     (10)
SSL_SERVER_I_DN_O=SomeOrg                                      (10)
SSL_SERVER_I_DN_OU=SomeOrgUnit                                 (10)
SSL_SERVER_M_SERIAL=01                                         (12)
SSL_SERVER_M_VERSION=3                                         (13)
SSL_SERVER_S_DN=/C=--/CN=www.example.com                       (11)
SSL_SERVER_S_DN_C=--                                           (11)
SSL_SERVER_S_DN_CN=www.example.com                             (11)
SSL_SERVER_V_END=Dec 31 23:59:59 1970 GMT                      (15)
SSL_SERVER_V_START=Jan 01 00:00:00 1970 GMT                    (14)
SSL_SESSION_ID=0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF (27)
SSL_TLS_SNI=www.example.com                                    (28)
SSL_VERSION_INTERFACE=mod_ssl/2.2.15                           (29)
SSL_VERSION_LIBRARY=OpenSSL/1.0.0-fips                         (30)
```

Example 1a. ..., the client certificate part

```
   ...
   SSL_CLIENT_A_KEY=rsaEncryption                                    (18)
   SSL_CLIENT_A_SIG=sha1WithRSAEncryption                            (19)
   SSL_CLIENT_I_DN=/C=--/O=SomeOrg/OU=SomeOrgUnit/CN=Root CA         (20)
   SSL_CLIENT_I_DN_C=--                                              (20)
   SSL_CLIENT_I_DN_CN=Root CA                                        (20)
   SSL_CLIENT_I_DN_O=SomeOrg                                         (20)
   SSL_CLIENT_I_DN_OU=SomeOrgUnit                                    (20)
   SSL_CLIENT_M_SERIAL=02                                            (22)
   SSL_CLIENT_M_VERSION=3                                            (23)
   SSL_CLIENT_S_DN=/CN=Name/emailAddress=name@example.com            (21)
   SSL_CLIENT_S_DN_CN=Name                                           (21)
   SSL_CLIENT_S_DN_Email=name@example.com                            (21)
   SSL_CLIENT_VERIFY=SUCCESS                                         (16)
   SSL_CLIENT_V_END=Dec 31 23:59:59 1970 GMT                         (25)
   SSL_CLIENT_V_REMAIN=365                                           (26)
   SSL_CLIENT_V_START=Jan 01 00:00:00 1970 GMT                       (24)
   ...
```

Example 2.

```
   ...
   SSL_CIPHER=RC4-MD5
   SSL_CIPHER_ALGKEYSIZE=128
   SSL_CIPHER_EXPORT=false
   SSL_CIPHER_USEKEYSIZE=128
   ...
   SSL_PROTOCOL=SSLv3
   SSL_SECURE_RENEG=false
   ...
```

Example 3.

```
   ...
   SSL_CIPHER=AES128-SHA256
   SSL_CIPHER_ALGKEYSIZE=128
   SSL_CIPHER_EXPORT=false
   SSL_CIPHER_USEKEYSIZE=128
   ...
   SSL_PROTOCOL=TLSv1.2
   SSL_SECURE_RENEG=true
   ...
```

Author's Address

Walter Hoehlhubmer
Lederergasse 47a
A-4020 Linz
Austria, EUROPE

EMail: walter.h@mathemainzel.info