Informational Add-on for HTTP over the Secure Sockets Layer (SSL) Protocol and/or the Transport Layer Security (TLS) Protocol draft-hoehlhubmer-https-addon-07

Abstract

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

The Add-on has two parts, one for the Domain Name System (DNS) — storing the X.509 certificate hashes — and one for the webserver itself — an additional webpage providing specific informations.

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 25, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction .......................... 3
1.1. Requirements Notation .................. 4
2. Implementing this Add-on .................. 5
2.1. Implementing the DNS part ................. 5
2.1.1. Calculating the Hashes ............... 5
2.1.1.1. Calculating the Packed form .......... 7
2.1.2. Formatting the Date and Time .......... 7
2.1.3. Arbitrary String Attribute Syntax .......... 7
2.1.4. DNS-entry Namespace ................ 9
2.2. Implementing the HTTP part .............. 9
2.2.1. Webpage Content ..................... 10
2.2.2. Formatting and Presenting the Webpage .......... 11
3. DNS part Details ........................ 11
3.1. Handling Certificate Rollover ............. 11
3.2. Verification Procedure ................ 12
4. IANA Considerations ..................... 12
5. Security Considerations ................ 12
6. Acknowledgements ....................... 12
7. Recommendations ....................... 12
8. References ............................ 13
8.1. Normative References ............................ 13
8.2. Informative References ................ 13
9. Discussions ........................... 16
A. Example certificates .................... 17
A.1. The DER-encoded CA certificate ............ 17
A.1.1. The CA’s public key .................... 17
A.2. The DER-encoded SSL certificate ........... 18
B. Script Examples for the Add-on webpage .......... 19
B.1. PHP-script ................................ 19
B.2. CGI-script: A BASH shell script ............ 20
B.3. CGI-script: A compiled C program .......... 20
C. Sample Content of the Add-on webpage .......... 23
C.1. A complete sample content ................ 23
C.1.1. ..., the client certificate part .......... 24
C.2. Picking another cipher suite .............. 24
C.2.1. ..., and one more .................... 24
Author’s Address .......................... 25
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
(7) ...

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 [RFC5280] 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 you can view the used X.509 certificates of the actual session, but you have no direct comparison if they are the correct ones.

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

The X.509 certificates which are shown by the browser and those, that are shown in this Add-on webpage MUST match; with other words:

If they do not match, there is going on a man-in-the-middle attack.

To give the browser, a plug-in, or just a stand-alone program, the ability to verify, that the X.509 certificate is correct, the Fingerprint and/or Hash of the X.509 certificates and also some additional informations for the community itself are stored in the Domain Name System (DNS) [DNS-1,DNS-2] as arbitrary string attributes as specified in [RFC1464].

This SHOULD be seen as an additional specification of the DNS-Based Authentication of Named Entities (DANE) [RFC6698] that increases the confidence and gives extended informations that can easily be read by the community.

1.1. Requirements Notation

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].
2. Implementing this Add-on

This Add-on has two parts.

Section 2.1. describes the implementation of the neccessary entries in the Domain Name System (DNS).

Section 2.2. describes the implementation of the additional webpage.

2.1. Implementing the DNS part

This part is implemented as an arbitrary string attribute, that has at least the following content:

1. The Hashes of all X.509 certificates of the whole certificate chain
2. The number of all X.509 certificates of the whole certificate chain
3. The used Hashing algorithm as specified in [SHA1] or [RFC6234]:
   3a) "sha1" for SHA-1
   3b) "sha224" for SHA-224
   3c) "sha256" for SHA-256
   3d) "sha384" for SHA-384
   3e) "sha512" for SHA-512
4. The way the Hashing values were calculated:
   4a) "0" for Non-packed Base64 encoded Hashes, see Section 2.1.1.
   4b) "1" for Packed Base64 encoded Hashes, see Section 2.1.1.1.
5. The Date and Time the X.509 certificate is valid, for format see Section 2.1.2.

For Syntax see Section 2.1.3.

2.1.1 Calculating the Hashes

For calculating the hashes use either [SHA1] or SHA-224, SHA-256, SHA-384, or SHA-512 as specified in [RFC6234].

Take each X.509 certificate of the whole chain and calculate the hash of the DER-encoded certificate.

The example certificates of Appendix A give these SHA-224 hashes in hex:

CA: 00fccc1bb4d09a392f5729a0c4e1ed4247db6b21da1fa9c6d218d4
SSL: eacbd6c276b34c887b4e953b56c6987d8730b682b1f13031b04de
and these SHA-512 hashes in hex:

```
CA:  6744023893a9a046e713b5615b9f1a267a41da3712f4eb964e496754bd9
     43105a5a3a8b9b071dea25f90fa7aa9c877dcc2ec81a7c97b640b30dd350
     83252078
SSL:  df0dee228b19aaleac6d2227d11cb243562058db5a4041b208ed7702869
     98747ed7ba08026791961d338cb2063f3485ec9fe07d5631a8a1b1da340
     25cb8962f5
```

Concatenate the binary form of the calculated hashes in the correct order beginning at the root.

Generate the Base64 encoding [RFC4648] from the concatenated hashes.

This example gives the following Base64 from the concatenated SHA-224 hashes:

```
APzBu00Jo5L1cpoMHh7UJH22sh2h/Km/bSGNtOrL3Gwny6TsyHtO1TtWxph9h0ML
aCsfEwMbBN4=
```

and from the SHA-512 hashes:

```
Z0QCOJopoEbne7Vhw88aJnpB2hnxL065ZOSWdUvZQxBawjgLMwcd6iX5D6eqn1d9
zC71GnyXtkCzdNqgyUgeN8N7iKLGaeerG0iJ9EcskNWIFjbbkBBsgjtdwKgmHd
7XuggCz5GWHTDmsgY/NIXsn+B9VjGobHAANAJcuJYv0=
```

Due to size limitations as specified in [DNS-2] Section 2.3.4. and the Syntax as specified in Section 2.1.2. below this Base64 encoded hash MUST NOT be longer than 196 octets.

This table shows when to use the packed form of calculation explained in next Section 2.1.1.1.

```
+-----------+--------------+
| Hashing   | X.509        |
| algorithm | certificates |
+-----------+--------------+
| SHA-1     | 8 or more    |
+-----------+--------------+
| SHA-224   | 6 or more    |
+-----------+--------------+
| SHA-256   | 5 or more    |
+-----------+--------------+
| SHA-384   | 4 or more    |
+-----------+--------------+
| SHA-512   | 3 or more    |
+-----------+--------------+
```

Using of the non packed form SHOULD be preferred.
2.1.1.1. Calculating the Packed form

The calculation is the same except, that the binary form of the concatenated hashes is hashed again using the SHA-512 algorithm.

Generate the Base64 encoding from this SHA-512 hash.

The example from the previous Section 2.1.1. has only two X.509 certificates. There would not be any need of packing this by hashing again.

The Base64 encoding of this packed SHA-512 hash is the following:

4iBTHcxpK4GG0thWbLaq9gQx2UmFDPI2DJDyWeKYk3RmUwS+nuCXYYXR6ED4iGy4Ft15nFcsa9rwMvsQx/wg==

NOTE: The algorithm attribute refers to the calculation method of the X.509 certificate hashes. For calculation of the packed form there is always used the SHA-512 hashing algorithm.

2.1.2. Formatting the Date and Time

The date and time is formatted in GeneralizedTime as specified in [RFC5280] Section 4.1.2.5.2.

For the purposes here, GeneralizedTime values MUST be expressed in Greenwich Mean Time (Zulu) and MUST include seconds (i.e., times are YYYYMMDDHHMMSSZ), even where the number of seconds is zero. GeneralizedTime values MUST NOT include fractional seconds.

2.1.3. Arbitrary String Attribute Syntax

The syntax for a complete arbitrary string attribute, using the ABNF notation and core rules of [RFC5234], is:

```plaintext
attribute = DQUOTE
    attr-algo 1*SP ; (3)
    attr-count 1*SP ; (2)
    attr-packed 1*SP ; (4)
    attr-time 1*SP ; (5)
    attr-hashes 1*SP ; (1)
DQUOTE

attr-algo = "a=" hash-algo ";"
attr-count = "c=" cert-count ";"
attr-packed = "f=" packed-form ";"
attr-time = "v=" valid-time-from "-" valid-time-to ";"
attr-hashes = "x=" cert-hashes ";"
```
cert-count = DIGIT ; number of X.509 certificates of
; the whole certificate chain

cert-hashes = base64-string
 ; base64 encoding of the certificate hashes

hash-algo = 1*("sha1" / "sha224" / "sha256" / "sha384" / "sha512")

packed-form = 1BIT ; "0" non-packed base64 encoded hashes,
 ; "1" packed base64 encoded hashes

valid-time-from = time-str
valid-time-to = time-str

base64-string = 1*(ALPHA / DIGIT / "+" / "/") [ "=" [ "=" ] ]

time-str = time-year time-month time-day time-hours time-minutes
time-seconds "Z"

time-year = 4DIGIT
time-month = 2DIGIT
time-day = 2DIGIT
time-hours = 2DIGIT
time-minutes = 2DIGIT
time-seconds = 2DIGIT

The example from Section 2.1.1. gives these:

"a=SHA224; c=2; f=0; v=19700101000000Z-19701231235959Z; x=APzBu0
0Jo5I1cpoMHh7UJH22sh2h/Rm/bSGNtOrL3Gwry6TsyHt01TtWxph9h0MLaCsfEw
MbBN4=;"

"a=SHA512; c=2; f=0; v=19700101000000Z-19701231235959Z; x=Z0QCOJ
OpoEbW7VhW88aJnpB2hxNzL65ZOSbewZQxBaWjqlmwd6I5D6eqnI9zC71Gn
yXtkCzdDNQjyUgeN8N7iKLGaeerG0J9EcshkNWFjfbWKBBgJtMdwKGMYH7XugCC
Z5GWHTQ sągY/NIXsn+BBVgGoobHAtA מקyJvU=
"

"a=SHA512; c=2; f=1; v=19700101000000Z-19701231235959Z; x=4iBTHc
xpK4G0thWbLaq9gQx2UmFDPI2DJDWyEYk3RmUwS+nkuCXyXR6ED4iGy4Ft.15nF
csta9rwMvsQx/wg==;"

All three are valid.
2.1.4. DNS-entry Namespace

For this Add-on a subdomain named ",sslinfo" is used.

INFORMATIVE OPERATIONAL NOTE: Wildcard DNS records (e.g., *.sslinfo.example.com) are only used in context with Wildcard X.509 certificates. Note also that wildcards within domains (e.g., s.sslinfo.*.example.com) are not supported by the DNS.

The DNS entries in the Zone file for this example look like these:

; IPv4 address
www.example.com. IN A 192.0.2.1

; IPv6 address
www.example.com. IN AAAA 2001:db8::1

; X.509 certificates hashes, SHA-224
www.sslinfo.example.com. IN TXT "a=SHA224; c=2; f=0; v=1970010100000002-19701231235959Z; x=APzBu00Jo51cpoMHh7UJH22sh2h/Km/bSGNtOrL3Gwny6TsyHt0lTtWxph9h0MLaCsfEwMbBN4;"

; X.509 certificates hashes, SHA-512
www.sslinfo.example.com. IN TXT "a=SHA512; c=2; f=0; v=1970010100000002-19701231235959Z; x=Z0QCOJopoEbnE7VhW88aJnpB2hNxL065Z0SWdUv2QxBaWjqLmcd6iX5D6egn1d9zC71GnyXtkCzDdNQgyUgeN8N7iKLGaoerG0iJ9EcSkKNWIFjbWkBsgjttdwKGmYdH7XuggCZ5GWHTOMsgI/NIXsn+B9VjGoobHaNAJcuJYvU;"

; X.509 certificates hashes, SHA-512, packed
www.sslinfo.example.com. IN TXT "a=SHA512; c=2; f=1; v=1970010100000002-19701231235959Z; x=4iBTHcxpK4GG0thWbLaqg9qX2UmFDPiZ2DJDWyeKYk3rmUwS+nkuCXYXR6ED4iGy4Ft15nFcsta9rwMvsaQx/wg;"

2.2. Implementing the HTTP part

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.

See Section 2.2.1. for the necessary content of this webpage.

For doing so see the sample scripts from Appendix B. To see how this Add-on works, see [MYADDON].
2.2.1.  Webpage Content

The informations MUST be the following:

(1) The actual date and 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 C for a sample content.

2.2.2. Formatting and Presenting the Webpage

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].

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

Any translation or sorting the order of this content is OPTIONAL.

Consider using either one of the character sets as specified in [ISO8859] or the UTF-8 character set as specified in [ISO10646].

3. DNS part Details

3.1. Handling Certificate Rollover

This is analogous to [RFC6698] Section A.4.

Suppose www.example.com has a single TXT record:

www._sslinfo.example.com. IN TXT "a=SHA224; ...; x=Z0QCO..."

To start the rollover process, obtain or generate the new certificate or SubjectPublicKeyInfo to be used after the rollover and generate the new TXT record. Add that record alongside the old one:

www._sslinfo.example.com. IN TXT "a=SHA224; ...; x=Z0QCO..."
www._sslinfo.example.com. IN TXT "a=SHA384; ...; x=A078x..."

After the new records have propagated to the authoritative nameservers and the TTL of the old record has expired, switch to the new certificate on the Web server. Once this has occurred, the old TXT record can be removed:

www._sslinfo.example.com. IN TXT "a=SHA384; ...; x=A078x..."

This completes the certificate rollover.
3.2. Verification Procedure

When the webbrowser or a plug-in honors the additional DNS entries, it SHOULD give a warning to the user:

(1) when it doesn’t find the entry
(2) when the entry doesn’t match

In case the DNS entries were retrieved by [DNSSEC] instead of simple DNS, then the user MUST give a permission to go further, when one of the two scenarios occurs.

The user MAY be warned, but MUST NOT be prevented to use the website.

4. IANA Considerations

There are no requests for IANA actions in this document.

5. Security Considerations

When implementing the HTTP part 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.

6. Acknowledgements

7. Recommendations

[DNSSEC] SHOULD be used for the DNS part.

Using a standardized URL for the HTTP part is RECOMMENDED, for more see Discussions at Section 9.
8. References

8.1. Normative References


8.2. Informative References


9. Discussions

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

Placing an Absolute URI as defined in [RFC3986] Section 4.3. outside the encrypted website part is RECOMMENDED.
A. Example certificates

A.1. The DER-encoded CA certificate

This section contains the full, DER-encoded certificate, in hex.

30820267308201D0A00302010202010030D06092A864886F70D01010500030
47310B3009606355040613022D23110300E06035504A130536F6D654F7267556E6974
31143012060355040B130B536F6D654F7267556E69743110300E060355040313
07526F6F7420431301E170D37303013031303030303030303A170D37313233
31323335393595A3047310B300960355040613022D23110300E060355040A
13075536F6D654F726731143012060355040B130B536F6D654F7267556E697431
10300E06035504031307526F6F742043130819F300D06092A864886F70D0101
010500038D0030189208218100CE72969A5433226FBC26310BBEDA8EA0DC2E
0532CB99CB314A1451D3A57CB2ADCAF463449B2D1C6A2BC7722825DF17FB12
067613CF3328459D7D7C4D847CF51C0F9562F296E8E8399C03CBE888FFBE4C11
57E03D208C8E87BF9A507F3D5DD00DE2A5FF19B4D2B89DF732DA7CBEA034C9
A4F1FEF58240943FDF25793794E7772020301001A3633061300F060355D130101
FF04050030101FF300E0603551DF0101FF040F020300106300DF0603551D0E04
16041473311472F1D56473C8DE0DDE39CCC2792B71EDDE301F0603551D230418
3016801473311472F1D56473C8DE0DDE39CCC2792B71EDDE300D06092A864886
F7D010105000038181000C446885FF2B8451B00E668530493474524EDE1E
3B1AC325E677D9BE92204A13369AFC48445AF3E01359B6C054D1049028DBC7A
E2F8A68BF5DC89010C488B41896EB34C7B1DA195B2B7C26579CC2F7A705C4AE
9C4F72D80E83D7AEC7B315492B7FF8410712C9330E3FA98A5949A283CD599
FCBSD97E330E05086844C

A.1.1. The CA’s public key

This section contains the DER-encoded public RSA key of the CA who signed the example SSL certificate. It is included with the purpose of simplifying verifications of the example certificate.

30819F300D06092A864886F70D010105000301810003081892081000CE7296
9A54332263FBC26310BBEDA8EA0DC2E0532CB99CB314A1451D3A57CB2ADCAF
463449B2D1C6A2BC7722825DF17FB1267613CF3328459D7D7C4D847CF51C0F95
62F296E8E8399C03CBE888FFBE4C115E03D208C8E87BF9A507F3D5DD00DE2A
5FF19B4D2B89DF732DA7CBEA034C904F1FEF58240943FDF25793794E7720301
0001
A.2. The DER-encoded SSL certificate

This section contains the full, DER-encoded certificate, in hex.

```
30820289308201F2A00302010202010300D06092A864886F70D0105000300
47310B3009060355040613022D23110300E060355040A1307536F6D654F7267
31143012060355040B130B536F6D654F7267556E69743110300E060355040313
07526F6F742043101E170D3730303130313030303030305A170D3730313233
313233353935395A3027310B3009060355040613022D2311830160603550403
130F777772E6578616D706C652E636F6D30819F300D06092A864886F70D0101
0105003818D00308189028181009D311D25BEDCC2765D1BF6BE9AB43C2ED41B
A9AF953154186940E28AA5C80B460FED2EE1ABD5BE2BD6E351EAF9F0DCE4388
27B42E166FAE594C83F40B72175EE875342E3450FAF1407A12267E85041C949F
A6DBBDCF6593958D0204199AE457EAB87D7E85487123C73398156F2AF1B87C49
0EF27B20F93A81C01656F6CBEE1E10203010001A381A43081A13090603551D13
40403000300E0603551D0F0101FF0404030205A0301D0603551D0E0416041454
6A242E922027FB19CFF91E7ECF846A0D050F2DC301F0603551D23041830168014
73311472F1D56473C8DE0D0E39CC2792B71EDDE301A0603551D11041301182
0F777772E6578616D706C652E636F6D300301603551D25040C300A06082B0601
050507030130130603551D20040C300A3008060667810C010201300D06092A86
4886F70D0101050003811008A7F7627D2390ED474D591F2F4C94FCCFCAEFA
DB04C8FD0619678A6001B1BC19CFCF29AE96D48949DA81D1BCFEE8F5E764BA7F91
C52BC50C28A472C2A6B22FE4EB27BEE6B0C989AF17CFCF9E3A52F641B7C34E2A
7FB994B55F3843C55280E9C46DD9EABACBC915EE6D92E3C542C93799A6DBFE
637EB2B82566FBC46A3A60D46
```
B. Script Examples for the Add-on webpage

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

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:

B.1. PHP-script
B.2. CGI-script: A BASH shell script, for most Linux systems
B.3. CGI-script: A compiled C program, for any other system

B.1. PHP-script

<CODE BEGINS>
<?php

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

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

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>
B.2. CGI-script: A BASH shell script, for most Linux systems

```bash
#!/bin/bash

echo -e -n "Content-type: text/plain\n\n"

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

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

B.3. CGI-script: A compiled C program, for any other 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 C compiler SHOULD be suitable. Be sure your runtime supports the function strftime with standard format specifiers.

```c
#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>"; 
```
int main(int argc, char* argv[], 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>\n" );

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

    printf( "</CODE>\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>
C. Sample Content of the Add-on webpage

The first example shows a complete 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 2.2.1.

C.1. A complete sample content
C.1a. ..., the client certificate part
C.2. Picking another cipher suite
C.2a. ..., and one more

C.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)
C.1.1. ..., 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_I_DN_SERIAL=02                                       (22)
SSL_CLIENT_I_DN_VERSION=3                                       (23)
SSL_CLIENT_I_DN_S_DN=/CN=Name/emailAddress=name@example.com    (21)
SSL_CLIENT_I_DN_S_DN_CN=Name                                    (21)
SSL_CLIENT_I_DN_S_DN_Email=name@example.com                    (21)
SSL_CLIENT_I_DN_VERIFY=SUCCESS                                   (16)
SSL_CLIENT_I_DN_V_REMAIN=365                                    (26)
SSL_CLIENT_I_DN_V_START=Jan 01 00:00:00 1970 GMT                 (24)

C.2. Picking another cipher suite

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

C.2.1. ... , and one more

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

Hoehlhubmer                Expires May 25, 2014                  [Page 24]
Author’s Address

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

EMail: walter.h@mathemainzel.info