TLS – Cached Information

Stefan Santesson
3xA Security
(http://AAA-sec.com)
Status

• Substantial discussions since last IETF
  – Problems related to use of hash algorithms
    • Agility complexity
    • Need to specify a must implement hash for interoperability
    • No strong security requirements
  – Need for client to query server for cached info support
  – Need to explicitly define substitution syntax for each object type which preserve syntax of re-used handshake messages – alternatively define new handshake messages for substitution

• Result: Major rewrite in draft 04
Major changes in draft 04

• All use of hash functions replaced with FNV-1 digest
  – FNV-1 specified in Annex 1
  – This specification uses the term “digest” instead of “hash” to avoid confusion and security concerns.

• Reconnaissance
  – Client may check server capability before caching
  – Server may provide supported digest values

• Defined substitution syntax for each object type
  – Preserving original handshake message syntax
FNV-1 digest

digest = FNV_offset_basis
for each octet_of_data to be digested
    digest = digest * FNV_prime
    digest = digest XOR octet_of_data
return digest
Replacing cached objects in the handshake protocol

OLD:

NEW:

Replacing cached objects in the handshake protocol
Extension syntax

Old

```c
enum {
    certificate_chain(1), trusted_cas(2),
    (255)
} CachedInformationType;

struct {
    HashAlgorithm hash;
    opaque hash_value<1..255>;
} CachedInformationHash;

struct {
    CachedInformationType type;
    CachedInformationHash hashes<1..2^16-1>;
} CachedObject;

struct {
    CachedObject cached_info<1..2^16-1>;
} CachedInformation;
```

New

```c
enum {
    certificate_chain(1), trusted_cas(2),
    (255)
} CachedInformationType;

struct {
    CachedInformationType type;
    opaque digest_value<0..8>;
} CachedObject;

struct {
    CachedObject cached_info<1..2^16-1>;
} CachedInformation;
```
Client Extension

• Reconnaissance
  – Client MAY send empty cached info extension to query server capabilities

• Cached information
  – Client provide one digest value for each cached information object
Server Extension response

- Empty CI extension
  - Server supports information caching
- Cached objects with absent digest
  - Server supports caching of specified object types
- Cached objects with digest
  - Server supports caching of specified object types with specified digest values
Message flow

Client Hello with Cached Information Extension → Server Hello with Cached Information Extension

Example substitution

Certificate Message


Len[1] | FNV-1 digest
Substitution Syntax – certificate_chain

Original handshake message syntax defined in RFC 5246 [RFC5246]:

```plaintext
opaque ASN.1Cert<1..2^24-1>;

struct {
    ASN.1Cert certificate_list<0..2^24-1>;
} Certificate;
```

Substitution syntax is defined by expanding the definition of the opaque ASN.1Cert structure:

```plaintext
struct {
    opaque digest_value<0..8>;
} ASN.1Cert
```
Original handshake message syntax defined in RFC 5246 [RFC5246]:

```c
opaque DistinguishedName<1..2^16-1>;

struct {
    ClientCertificateType certificate_types<1..2^8-1>;
    SignatureAndHashAlgorithm
        supported_signature_algorithms<2^16-1>;
    DistinguishedName certificateAuthorities<0..2^16-1>;
} CertificateRequest
```

The substitution syntax is defined by expanding the definition of the opaque DistinguishedName structure:

```c
struct {
    opaque digest_value<0..8>;
} DistinguishedName
```
Alternatives

1. Replace `opaque digest_value<0..8>;` with `opaque digest_value[8];`
   - Define a fixed 64 bit value to represent absent digest, e.g. `[0,0,0,0,0,0,0,0]`

2. Keep `opaque digest_value<0..8>;` but omit the one byte length in the substitution syntax
Way forward

• Evaluate the proposed updates
• Specify FNV in separate RFC?
• Obtain necessary permissions from FNV authors (copyright, not IPR)?
• If major changes are still required – Possible co-author
• WGLC
Questions / Comments

SOME QUESTIONS CAN'T BE ANSWERED BY GOOGLE
SUN WORSHIP 9 AM