TLS 1.3 Client Authentication

Andrei Popov, Microsoft Corp.

Issue 1: TLS Client Authentication After the Handshake

- The user navigates to the site’s landing page, no client authentication required.
- Eventually, the user requests a protected resource, the site triggers renegotiation with client authentication.
- The site validates client credential and authorizes the request.

TLS 1.3 does not allow renegotiation, therefore sites designed as shown above can’t use TLS1.3.

Proposal Discussed at the TLS WG Interim: Allow TLS Client Authentication After the Handshake

- Signal at the TLS layer, in line with current usage.
- No rekey on client authentication.
- No requirement to reconnect at the TCP layer (avoiding TCP round-trips).
- The resulting client-authenticated session must be resumable, i.e. the SessionTicket message is optionally sent by the server after the client authenticates.
- No interleaving of application data and client authentication-related messages.

The last point creates implementation difficulties: how can we prevent a client from sending application_data records after the server issues a CertificateRequest?

Issue 2: Client Certificate Selection

- Currently, CertificateRequest message allows the selection of client certificate by:
  - Signature algorithm,
  - Hash algorithm, and
  - Distinguished Names of acceptable (intermediate and/or root) certificate authorities.
- For some sites and EAP-TLS deployments, these selection criteria are insufficient, and result in poor UX where a confusing certificate picker dialog has to be displayed.
- Customers are requesting flexible certificate selection criteria using KU, EKU, Issuance Policy, other OIDs, logical expressions, etc...
- Over time, new extension OIDs are being added to certificates.

Proposal: Add CertificateExtensions field to CertificateRequest for TLS 1.3

certificate_extensions

- A list of certificate extension OIDs [RFC5280] with their allowed values, represented in DER-encoded format.
- If the server has included a non-empty certificate_extensions list, the client end-entity certificate MUST contain all of the specified extension OIDs that the client recognizes.
- For each extension OID recognized by the client, all of the specified values MUST be present in the client certificate.
- The client MUST ignore and skip any unrecognized certificate extension OIDs.
- If the client supplies a certificate that does not satisfy the request, the server MAY respond with a fatal unsupported_certificate alert.

Proposed CertificateRequest for TLS 1.3

```
struct {
    opaque certificate_extension_oid<0..2^8-1>;
    opaque certificate_extension_values<0..2^8-1>;
} CertificateExtension

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