TLS Extended Key Schedule

draft-jhoyla-tls-extended-key-schedule-01

Jonathan Hoyland & Chris Wood

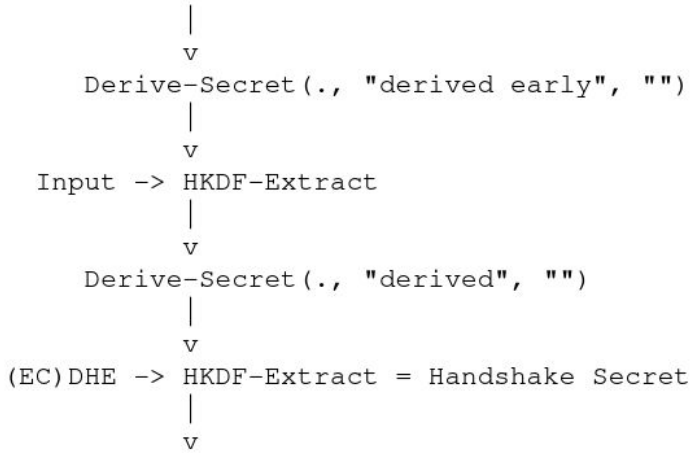
Importer Keys

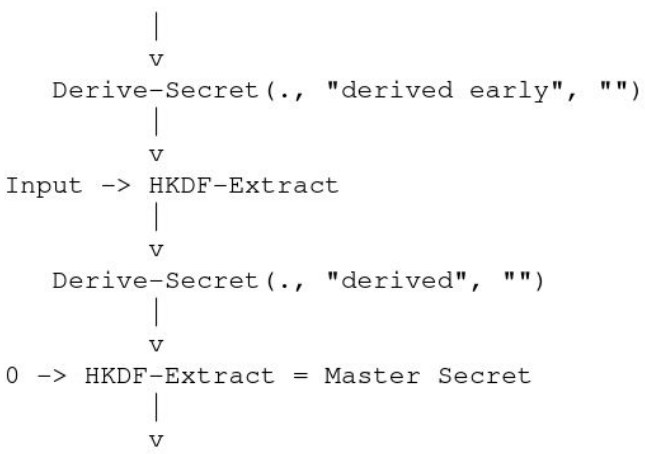
- TLS provides exporter keys to allow other protocols to build on top of TLS
- Importer keys would allow TLS to be layered onto other protocols in a generic way
- Potential Use Cases:
  - Bootstrapped TLS Authentication
  - Multiple cipher suites
    - Providing a generic interface means we can experiment with PQ cipher suites without any potential duplication of effort due to NIST competition.
  - Complex authentication properties
    - ECH, for example, could use importer keys to bind the inner and outer handshake together.

Why Importer Keys?

- Generic interface means security analysis only has to be done once.
  - The goal is that even an attacker that controls every injected input cannot weaken the security of the base handshake.
  - This would make it safer to experiment with PQ / new cipher suites.
- Multiple injections can happen in a single handshake
  - Other mutually exclusive suggestions have been made.
  - Importer keys allow for an effectively arbitrary number of injections (currently limited to 2^{16})

Two Injection Sites





Inputs structured

- Every user of the interface is given a type (an integer)
- Injections occur in ascending order
- A number of other structures could be used
  - Draft-stebila listed several
  - nKDF was suggested for MLS
    - Effectively XORs the inputs together in a secure way.
    - Removes ordering requirement on injection. Secrets can be added when available, as long as all are eventually available before the handshake progresses.

```
struct {
    KeyScheduleSecretType type;
    opaque secret_data<0..2^16-1>;
} KeyScheduleSecret;

enum {
    (65535)
} KeyScheduleSecretType;

struct {
    KeyScheduleSecret secrets<0..2^16-1>;
} KeyScheduleInput;
```

Questions?

- Is there interest in in making this a working group item?