Anpassungen von DTLS zur sicheren Kommunikation in eingeschränkten Umgebungen

Lars Schmertmann

lars@tzi.org

TZI, Universität Bremen, Deutschland

Kolloquium zur Bachelorarbeit
12.09.2013

Gliederung

- Motivation
- Hardware & Umgebung
- DTLS
  - Handshake
  - Ausgewählte DTLS-Header
- Mögliche Lösungen
  - Handshake über CoAP
  - Stateless Header Compression
- Geeignete Ciphersuit
- Praktische Umsetzung
- Ergebnis & Fazit
- Zeitplan

Motivation

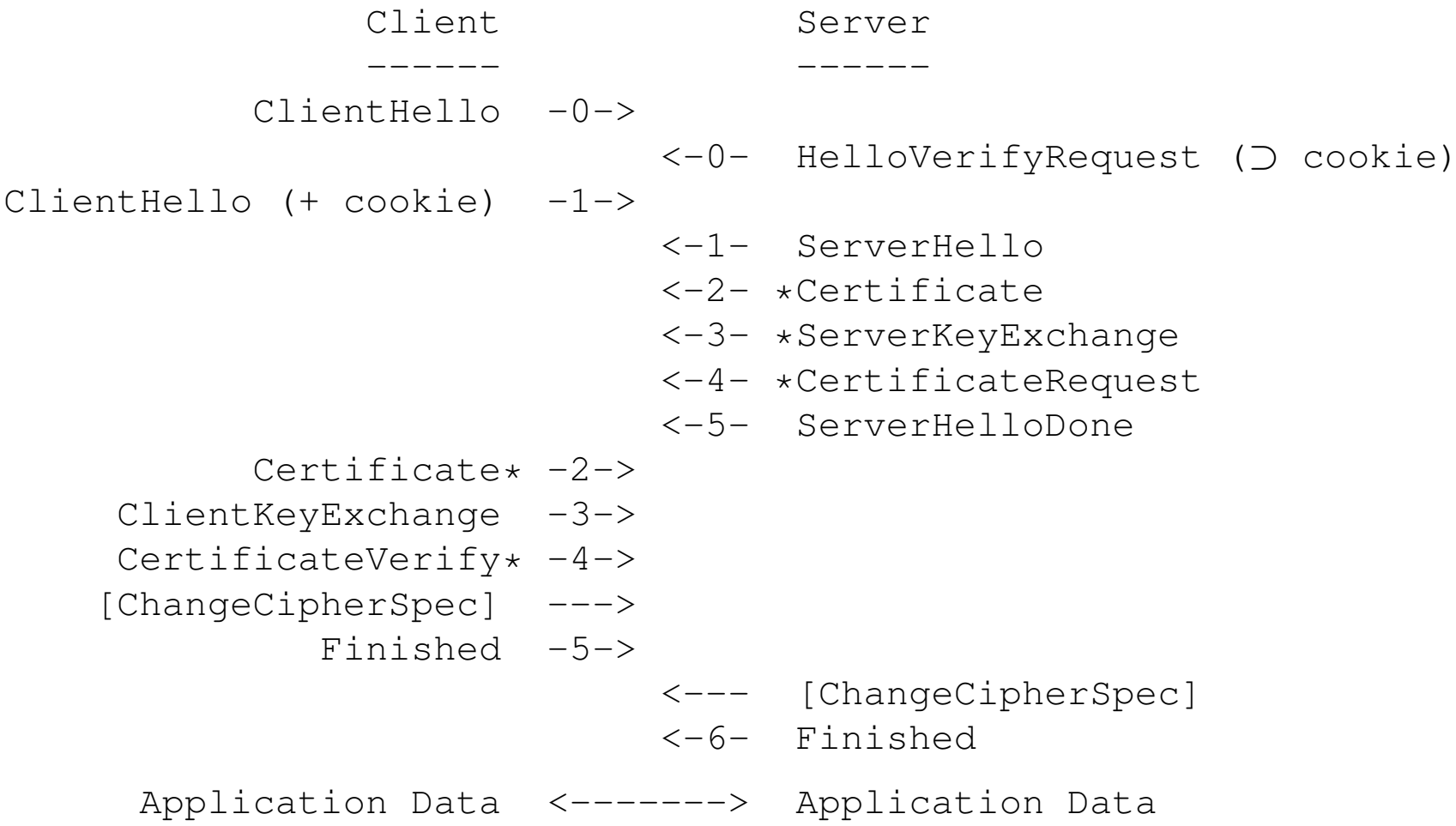
- Bachelorprojekt GOBI
  - Sicherheit ist auch in eingeschränkten Umgebungen notwendig
  - Mikro-DTLS (eher Mikro-Sicherheit)
- TLS und DTLS sind bewährte Standards
- DTLS komplexer als TLS

Hardware & Umgebung

- Econotag: mc13224v Development-Board
  - Freescale MC13224v ARM7TDMI-S Microcontroller
  - IEEE 802.15.4 Funkstandard
  - AES Hardware-Engine
  - 128 KiB Flash-Speicher
  - 96 KiB RAM
- IEEE 802.15.4 MTU: 127 Byte
  - - 48 Byte 6LoWPAN-Header
  - - 8 Byte UDP-Header
  - = 71 Byte Nutzdaten
- SmartAppContiki = Contiki + Erbium + CoAP 13
  - 81 KiB (bei angepasster Konfiguration)

DTLS

Handshake



DTLS

Ausgewählte DTLS-Header

```
struct {
      ContentType type;
      ProtocolVersion version;
      uint16 epoch;
      uint48 sequence_number;
      uint16 length;
      uint8   payload[length];
} DTLS_Record;
```

= 13 Byte

```
struct {
      HandshakeType msg_type;
      uint24 length;
      uint16 message_seq;
      uint24 fragment_offset;
      uint24 fragment_length;
      uint8   payload[f._length];
} Handshake;
```

= 12 Byte

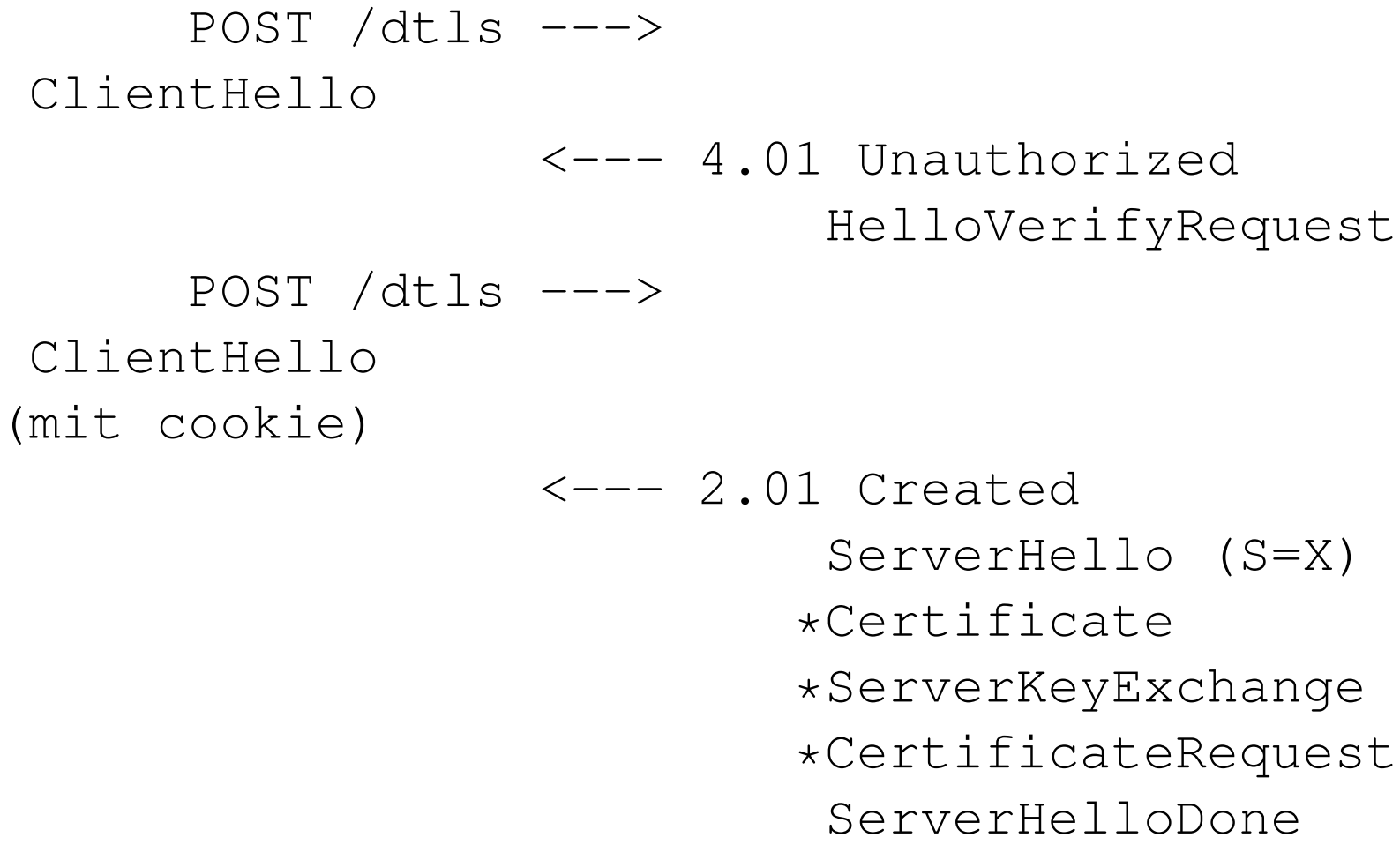
Mögliche Lösungen

Vorschläge im Entwurf von K. Hartke und O. Bergmann:
http://tools.ietf.org/html/draft-hartke-core-codtls-02

- Handshake über CoAP
- Stateless Header Compression

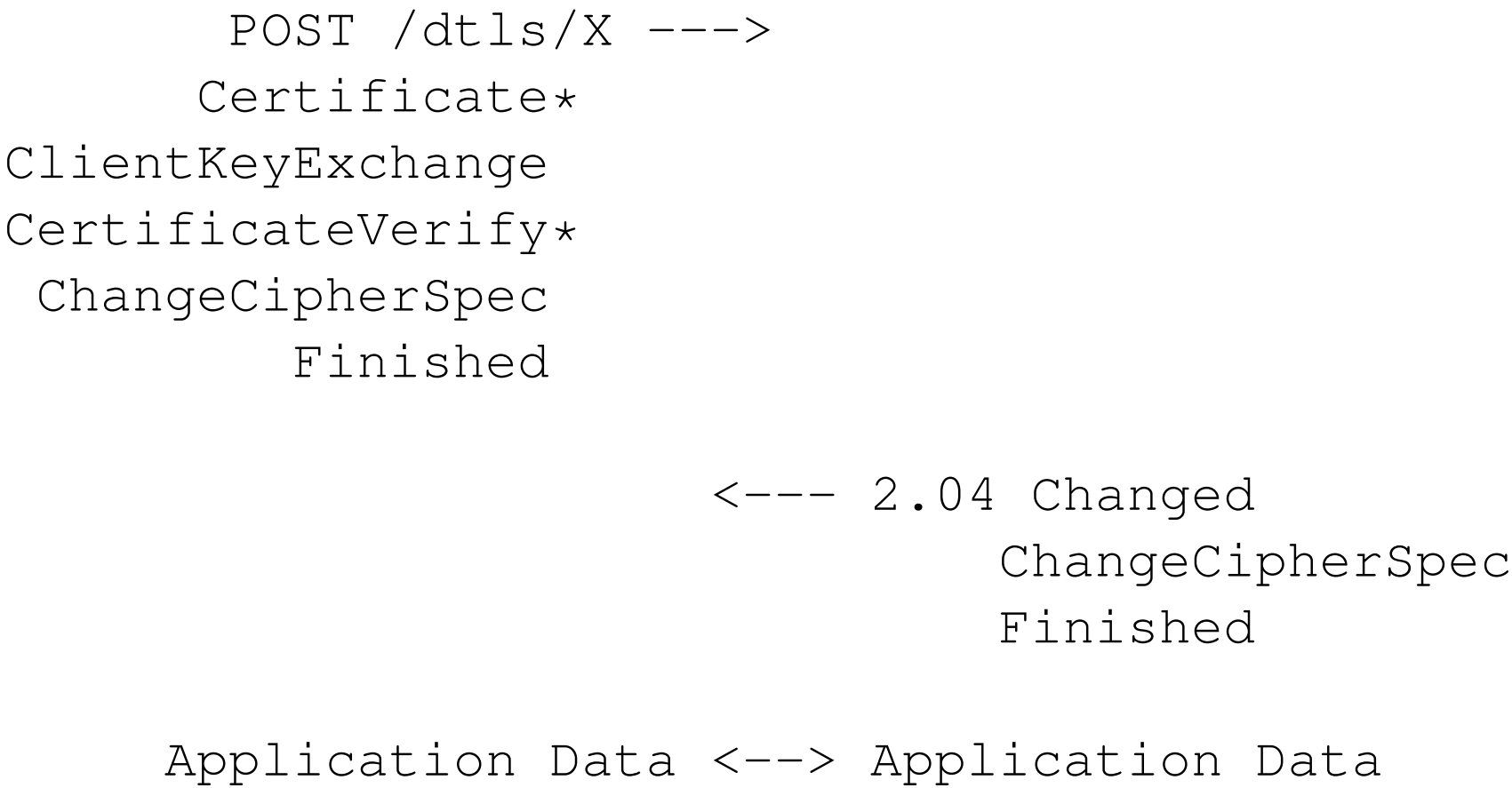
Mögliche Lösungen

Handshake über CoAP - Teil 1



Mögliche Lösungen

Handshake über CoAP - Teil 2



Mögliche Lösungen

Stateless Header Compression

```
struct {
        uint8   :1;
        RecordType type:2;
        Version version:2;
        Epoch epoch:3;
        uint8   :3;
        SequenceNumber snr:3;
        RecordLength length:2;
        uint8   payload[0];
} DTLSRecord_t;
```

= 2 - 15 Byte

```
struct {
        ContentType type:6;
        ContentLength len:2;
        uint8   payload[0];
} Content_t;
```

= 1 - 4 Byte

Geeignete Ciphersuite

- TLS_PSK_ECDH_WITH_AES_128_CCM_8

Eigenschaften:

- Durch CCM wird keine Hash-Funktion für den MAC benötigt
- Durch den Einsatz von CMAC mit PSK kann auf HMAC mit SHA-256 verzichtet werden

Auswirkungen auf die Programmgröße:

- Einsparung durch AES in Hardware und Nutzung von CCM + CMAC: ~3 KiB

Praktische Umsetzung

- Ablage von Read-Only-Daten im Flash-Speicher
- Ein Handshake zur Zeit
  - Stack mit Push und Clear im Flash-Speicher für „Finished“
- Wechsel des Pre-shared Key nach Handshake
  - Abrufbar über CoAP-URI
- Ablage der Session-Daten im Flash-Speicher
- Implementierung einzelner Funktionen in Assembler
- Software-Update über CoAP
  - Session-Daten bleiben erhalten
  - Sequenznummer geht verloren

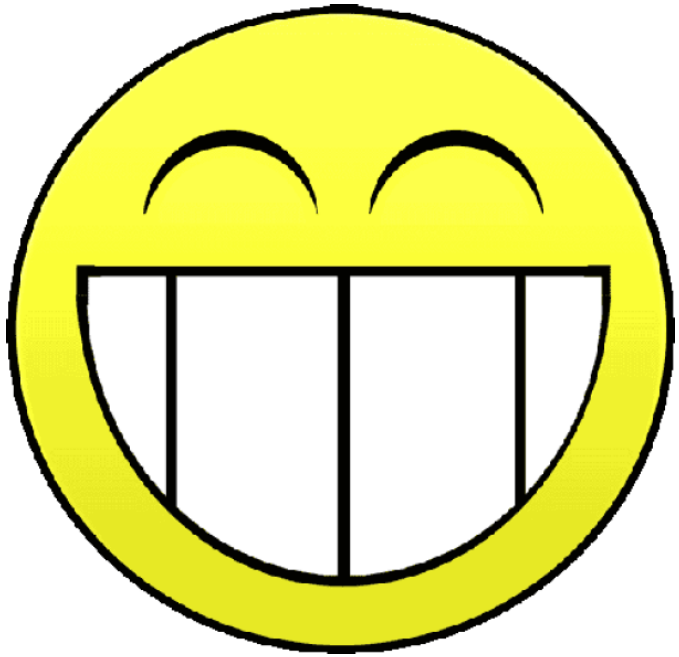
Ergebnis & Fazit

- Implementierung ist derzeit 10 KiB groß
- 7,2 KiB setzen sich zusammen:
  - 2,41 KiB ECC-Funktionen
  - 0,95 KiB AES-Funktionen (CCM + CMAC)
  - 0,80 KiB Flash-Speicher-Funktionen
  - 0,79 KiB Session-Verwaltung
  - 0,15 KiB Pseudo-Random-Funktion
  - 1,78 KiB Handshake-Ressource
  - 0,32 KiB Parse & Send
- Stack-Größe von 2 KiB reicht aus
- ECC-Multiplikation: ø6,0 s
- Handshake: ø14 s

Zeitplan

- Es verbleiben 4 Wochen bis zum Vorlesungsbeginn
  - 2 Wochen für Kapitel „Praktische Umsetzung“
  - 1 Woche für Kapitel „Vergleich“ und „Fazit“
  - 1 Woche Feintuning
- 14.10.2013 Abgabe der Arbeit

Vielen Dank für eure Aufmerksamkeit



Fragen