

Traces of EDHOC

draft-ietf-lake-traces-03

Göran Selander, Ericsson
John P. Mattsson, Ericsson
Marek Serafin, ASSA ABLOY
Marco Tiloca, RISE

... and overview of EDHOC testing

IETF 115 Meeting – London – November 8th, 2022

Status of draft-ietf-lake-traces

› Submitted version -03 before the cut-off

- Both traces are aligned with the latest EDHOC, i.e., version -17
- Output based on the RISE Java implementation (*Eclipse Californium*) [1]

› What has remained the same, in both traces

- CRED_I, CRED_R, ID_CRED_I, ID_CRED_R
- Diffie-Hellman ephemeral keys
- EDHOC Connection Identifiers and OSCORE Sender/Recipient IDs
- EDHOC message_1 and the intermediate steps before it
- Until and including TH_2 and G_XY, when preparing EDHOC message_2

› Anything else did change in both traces, due to:

- PRK_2e = HMAC-SHA-256(salt, G_XY), with salt = TH_2
- TH_3 = H(TH_2, PLAINTEXT_2, CRED_R)
- TH_4 = H(TH_3, PLAINTEXT_3, CRED_I)

Status of draft-ietf-lake-traces

› The second trace has been confirmed to be correct

- Cipher suite 2 (curve P-256) ; Method 3 (static-static)
- CCS as authentication credentials ; ‘kid’ as credential identifiers
- Validated by Mališa Vučinić with his Rust implementation [2][3]

› Additional test vectors from Marek Serafin

- Also aligned with the latest EDHOC version -17
- Cipher suite 2 (curve P-256); Method 0 (sign-sign) or 3 (static-static)
- Test vectors available at [4]

[2] Denis Merigoux, Franziskus Kiefer, Karthikeyan Bhargavan. *Hacspec: succinct, executable, verifiable specification for high-assurance cryptography embedded in Rust*, <https://hal.inria.fr/hal-03176482/document>

[3] <https://github.com/hacspec/hacspec>

[4] <https://github.com/stoprocent/edhoc/tree/master/test-vectors-16>

Testing before/during IETF 115

› Two implementations aligned with the latest version -17 of EDHOC

- Mališa Vučinić (INRIA): Rust/hacspec
- Marco Tiloca (RISE): Java (*Eclipse Californium*)

› Mališa: Initiator ; Marco: Responder

- Cipher suite: 2 (curve P-256)
- Method 3: Static-Static
- Credential Type: CCS for both peers
- ID_CRED Type: 'kid' for both peers (int on the wire)
- **Minimum possible size for message_2 (45 bytes)**
- **Correctly completed EDHOC execution**

```
Constrained Application Protocol, Acknowledgement, 2.04 Changed, MID:0
01.. .... = Version: 1
..10 .... = Type: Acknowledgement (2)
.... 0000 = Token Length: 0
Code: 2.04 Changed (68)
Message ID: 0
> Opt Name: #1: Content-Format: Unknown Type 65000
End of options marker: 255
Payload: Payload Content-Format: Unknown Type 65000, Length: 45
Payload Desc: Unknown Type 65000
[Payload Length: 45]

EDHOC Message 2 (45 bytes):
58 2a 69 ca 5b b2 54 2e cf a5
65 47 ed 98 93 6e e9 8d db 27
a6 de 0c 7f c4 8c a4 30 86 3e
be c3 ff af 79 d5 05 cc 93 84
e8 d4 89 d1 01
```

```
OSCORE Master Secret (16 bytes):
9e 59 41 04 90 6a 88 dd 28 f7
95 6f fc 0f 0f b3

OSCORE Master Salt (8 bytes):
58 99 36 ab 25 b9 1b 57

oscore_salt: [58, 99, 36, ab, 25, b9, 1b, 57,
oscore_secret: [9e, 59, 41, 04, 90, 6a, 88, dd, 28, f7, 95, 6f, fc, 0f, 0f, b3,
```

[2] Denis Merigoux, Franziskus Kiefer, Karthikeyan Bhargavan. *Hacspec: succinct, executable, verifiable specification for high-assurance cryptography embedded in Rust*, <https://hal.inria.fr/hal-03176482/document>

[3] <https://github.com/hacspec/hacspec>

Testing before/during IETF 115

› Two implementations aligned with the latest version -17 of EDHOC

- Mališa Vučinić (INRIA): Rust/hacspec
- Marco Tiloca (RISE): Java (*Eclipse Californium*)

› Mališa: Responder (NEW); Marco: Initiator

- Cipher suite: 2 (curve P-256)
- Method 3: Static-Static
- Credential Type: CCS for both peers
- ID_CRED Type: 'kid' for both peers (int on the wire)
- **Minimum possible size for message_2 (45 bytes)**
- **Correctly completed EDHOC execution**

```
Constrained Application Protocol, Acknowledgement, 2.05 Content, MID:199
01.. .... = Version: 1
..10 .... = Type: Acknowledgement (2)
.... 1000 = Token Length: 8
Code: 2.05 Content (69)
Message ID: 199
Token: ec83bca87e5e74ae
End of options marker: 255
Payload: Payload Content-Format: application/octet-stream (no Content-Format), Length: 4
Payload Descr: application/octet-stream
[Payload Length: 45]
[Uri-Path: /.well-known/edhoc]
[Request_In: 552]
[Response Time: 3.249170000 seconds]
Data (45 bytes)
[Length: 45]
0000 ac ed 5c 90 b7 17 38 f9 d3 90 be 59 08 00 45 00 ..\..8 ...Y..E..
0010 00 56 a3 a5 00 00 40 11 94 fd 1f 85 82 70 1f 85 ..V...@ ...p..
0020 80 7a 16 33 ba ff 00 42 07 84 68 45 00 c7 ec 03 ..z...B..hE...
0030 bc a0 7e 5e 74 ae ff 58 28 a1 97 81 07 f0 09 26 ...t..X..A...@
0040 c2 dc 58 7a 36 dd 75 25 49 f3 37 63 c8 93 42 2c ...X26-u^ I-7c..B,
0050 8e a0 f9 55 a1 3a 4f f5 d5 c1 c2 ff 3c 93 03 8e ...U:0:.....
0060 a1 cb f6 00
```

```
OSCORE Master Secret (16 bytes):
e2 68 01 a9 34 ca df c1 6f 26
85 e1 b4 41 b1 e8

OSCORE Master Salt (8 bytes):
70 33 02 6d b2 5b 3f 31

oscore_salt: [70, 33, 02, 6d, b2, 5b, 3f, 31]
oscore_secret: [e2, 68, 01, a9, 34, ca, df, c1, 6f, 26, 85, e1, b4, 41, b1, e8]
```

[2] Denis Merigoux, Franziskus Kiefer, Karthikeyan Bhargavan. *Hacspec: succinct, executable, verifiable specification for high-assurance cryptography embedded in Rust*, <https://hal.inria.fr/hal-03176482/document>

[3] <https://github.com/hacspec/hacspec>

Next steps

- › **More implementations are under ongoing updates**
 - Further tests will happen soon

- › **On *draft-ietf-lake-traces***
 - Wait for the WGLC comments of *draft-ietf-lake-edhoc* to be processed
 - If anything changes for EDHOC on the wire, prepare a new *draft-ietf-lake-traces-04*
 - Otherwise and hopefully, *draft-ietf-lake-traces-03* should be ready for WGLC

Thank you!