Aalto University
School of Science

HUAWEI

Angel Lomeli
Aalto University, Finland
Security and Cloud Computing (SECCLO)
Arto Niemi
Huawei, Finland
Helsinki System Security Laboratory (HSSL)

eGMT: Deep Fuzzing of Cryptographic Protocols Using Syntax Tree Mutation

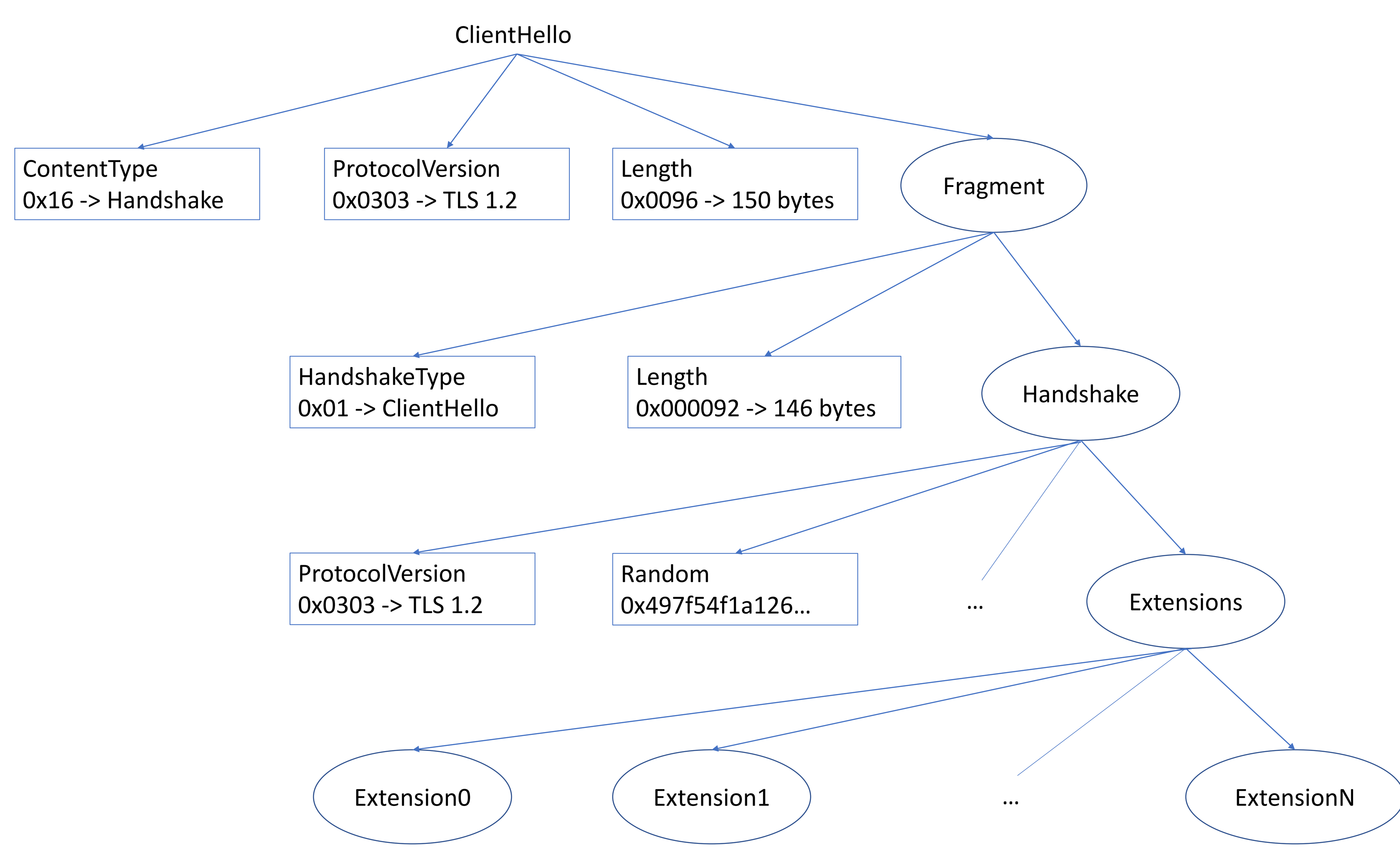
Introduction

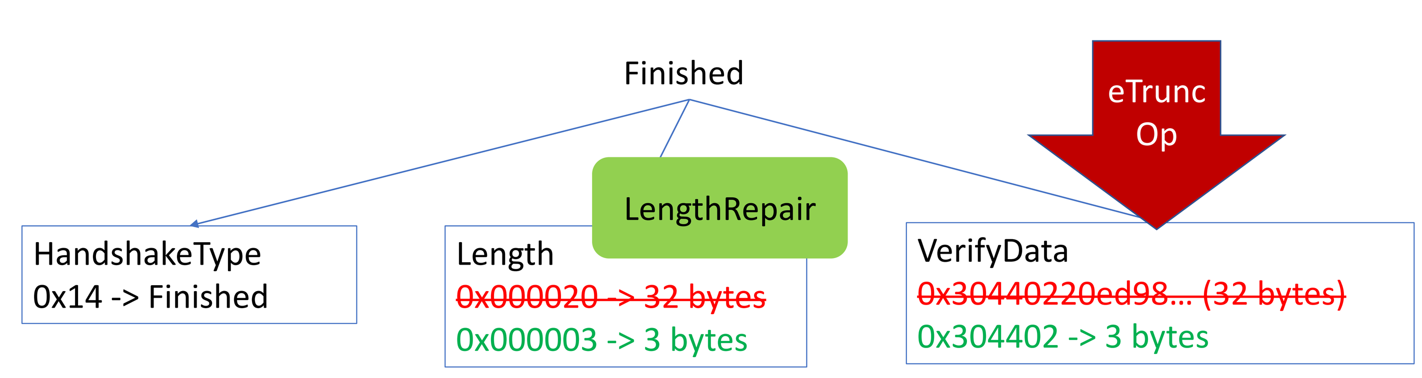
- *Fuzz testing*
  - Allows detecting hard-to-reach vulnerabilities by feeding applications mutated data and monitoring their behavior
- Common tools like AFL and Honggfuzz: good for *file-based* fuzzing
- *Interactive protocols*
  - Messages depend on earlier ones
  - Require specialized fuzzers such as AFLNet
- *Cryptographic protocols* (like TLS)
  - Messages must pass cryptographic checks (e.g. signatures, MACs)
  - Fuzzing still a major challenge
- This work:
  - New syntax tree mutation based fuzzer for cryptographic protocols
  - Test target: htls (HSSL's experimental small-footprint, TEE-compatible, dependency-free TLS 1.3 implementation)

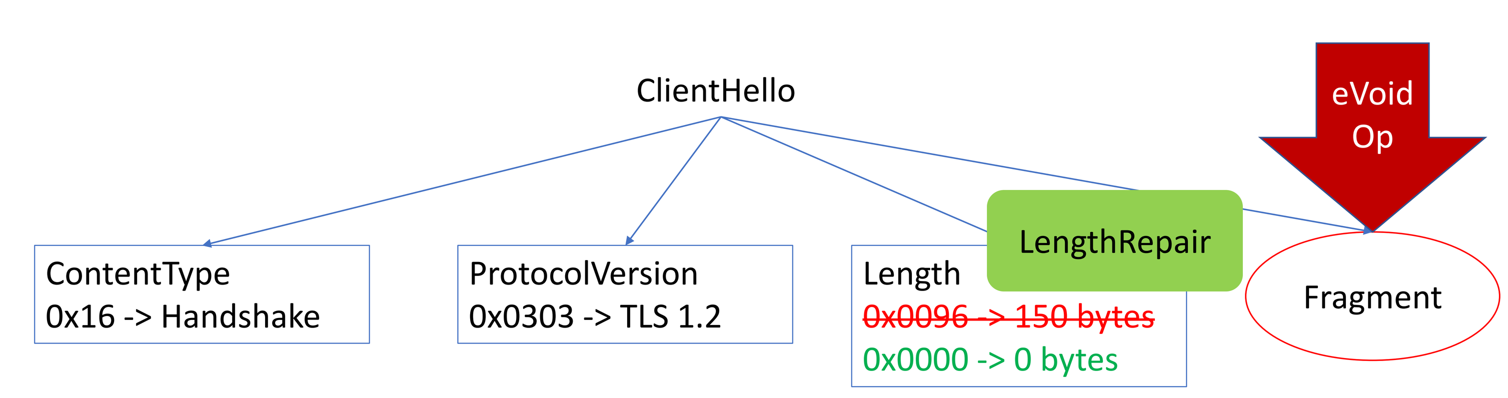
```
16 03 01 01 2C 01 00 01 28 03 03 40 C7 0E 24 30 01 B9 6D 8C 63 68 77 38 69 64 32 D3 E6 F9 49 10 7A
--------------------------------------------------------------------------------------------------
{0:TLSRecord}                     TLSRecord
|--[0:type]                       RecordType                  | 16
|--[1:version]                    ProtocolVersion             | 03 01
|--[2:length]                     Integer                     | 01 2C
 --{3:msg}                        HandshakeMessage
   |--[0:type]                    HandshakeType               | 01
   |--[1:length]                  Integer                     | 00 01 28
    --{2:msg}                     ClientHello
      |--[0:version]              ProtocolVersion             | 03 03
      |--[1:random]               OpaqueBlob                  | 40 C7 0E 24 30 01 B9 6D
      |                                                       | 8C 63 68 77 38 69 64 32
      |                                                       | D3 E6 F9 49 10 7A AB AD
      |                                                       | 84 50 CD FF D6 A2 66 E4
      |--[2:session_id_length]    Integer                     | 00
      |--[3:session_id]           OpaqueBlob                  |
      |--{4:cipher_suites}        ClientHello_cipher_suites
      |  |--[0:_N]                Integer                     | 00 92
      |   --{1:_V}                DynamicVector
      |     |--[0:CipherSuite]    CipherSuite                 | C0 30
      ...   ...                   ...                           ...
```

eGMT proposal

- Walz et al[1] : Generic Message Trees (GMTs)
  - Syntax trees + fuzz operators for TLS 1.2
  - Applicable only to the ClientHello message
- Our proposal: Enhanced Generic Message Tree (eGMT)
  - Improved fuzz operators
  - New operators such as ZeroOperator, BitFlipOperator
  - Applicable to all handshake messages, including encrypted ones
  - Focus on TLS 1.3, but also works for e.g. ASN.1/ECDSA signatures

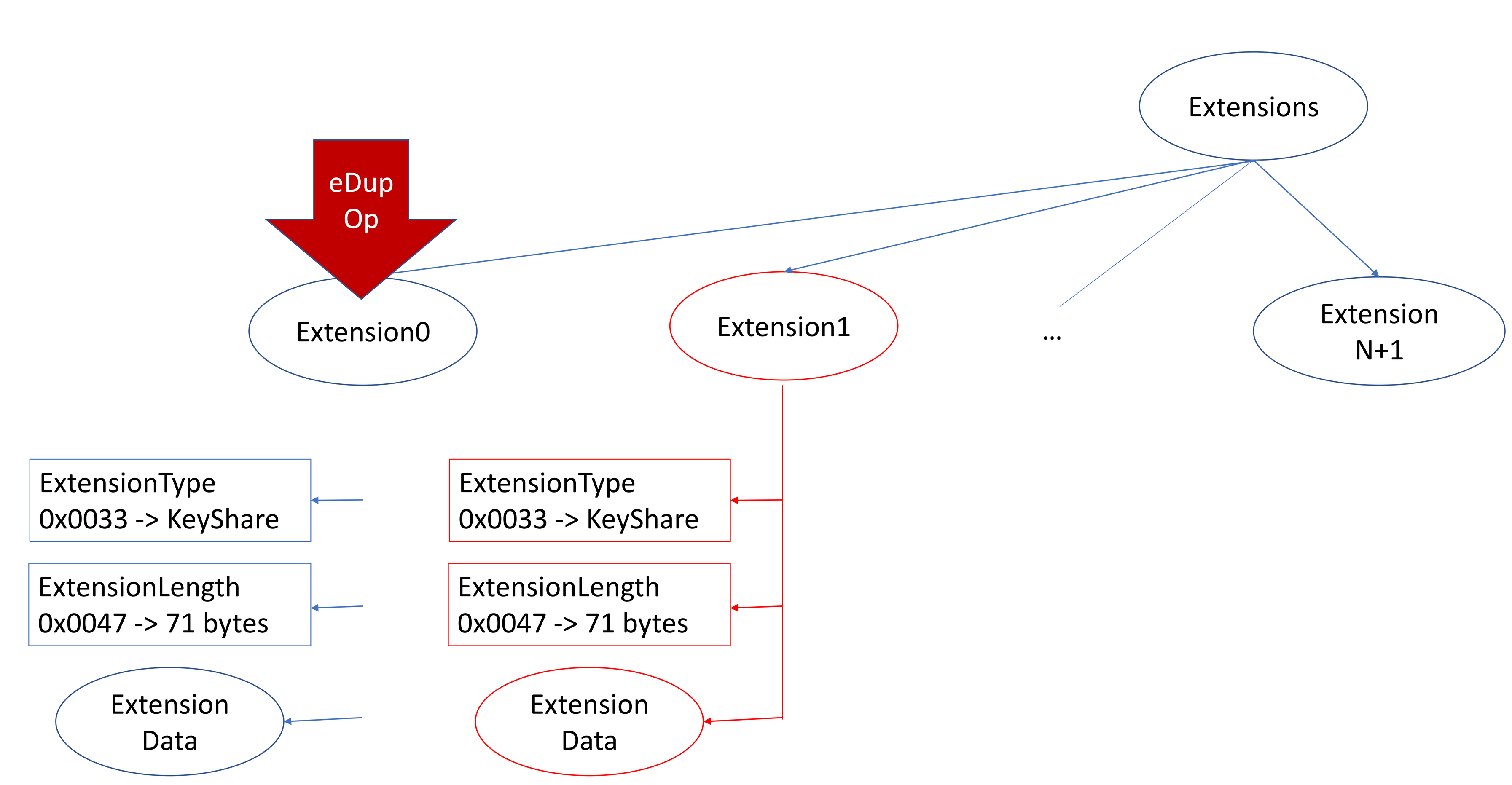






Vulnerabilities found

- **Missing ECDH public key validation.** Any arbitrary (e.g. attacker-injected) value is accepted as an ECDH key share.
- **Segmentation fault in log print.** An error causes an (almost) infinite loop that makes the application read from a restricted memory address.
- **Null pointer dereference in Finished message.** A short signature triggers an attempt to memcpy from a NULL address.
- **Segmentation fault in certificate validation.** Invalid memory read when parsing invalid X.509 certificates.
- **Wrong length in TLV objects.** Incorrect lengths in ASN.1 structures crash the application.
- **Garbage bytes after signature.** Signatures with garbage bytes are incorrectly accepted.
- **Non-zero compression methods.** Messages with non-zero compression field are accepted, violating the specification.
- **Too many same-type extensions.** Two or more extensions of the same type are accepted when they should not.
- **Invalid session ID.** Invalid session IDs are incorrectly accepted.
- **Missing required extensions.** The app ignores when a message misses a required extension.



References

1. A. Walz and A. Sikora, "Exploiting Dissent: Towards Fuzzing-Based Differential Black-Box Testing of TLS Implementations," IEEE Transactions on Dependable and Secure Computing, vol. 17, pp. 278–291, 2020.