# EdgeX Binary Encoder Analysis

This document uses the requirements identified over the last few weeks at the Device Working Group to compare binary encore implementations, to evaluate the two proposed implementations; CBOR and protobuf. Following is the list of agreed requirements and the result of applying then to CBOR and protobuf.

1. Preferably conforming to a standard (ideally one related to IoT):

   CBOR is an IETF IoT standard https://tools.ietf.org/html/rfc7049. Protobuf is not a standard and is more enterprise focused than IoT.

2. Should have mappings in C and Go and preferably Java:

   CBOR has all required bindings. Protobuf does not have an officially supported C binding.

3. Should support end to end binary data encryption and/or signing for sub contents:

   CBOR supports both data encryption and signing of contents https://tools.ietf.org/html/rfc8152. Protobuf does not.

4. Should preferably not rely on generated code (not a schema-less encoding):

   CBOR does not rely on code generation and carries schema. Protobuf does rely on code generation (it does not carry schema, instead relying on generated code for data encoding/decoding)

5. Should be as lightweight and efficient as possible:

   a) Encoding/Decoding speed:

   CBOR is a bit slower than protobuf as relies on generic rather than compiled encoding/decoding and also has to encode/decode schema. Protobuf is a bit faster than CBOR as uses compiled code to encode/decode and does not encode/decode schema.

   b) Library/code size:

   CBOR libraries are typically much smaller than the protobuf equivalent. Protobuf also has additional code over head because of generated/compiler encode/decode functions.

   c) Data encapsulation efficiency:

   For large data types CBOR encapsulation is effectively as good as protobuf. For small data samples with complex typing protobuf is better due to the overhead of encoding the schema in CBOR.

6. Can easily be converted to/from a JSON representation:

   CBOR provides a standard mapping to/from JSON https://tools.ietf.org/html/rfc7049#page-29. Protobuf can be converted to/from JSON

From this analysis it is clear that for most requirements CBOR is the best solution and even when it is not, it is still pretty close to protobuf in most scenarios. Finally the fact the protobuf lack a C binding, eliminates it from the running as this is an absolute requirement for support of the C device SDK.