Numbers in YANG

Scott Mansfield (Ericsson)

28 Jan 2024 (802.1 Interim session in Heidelberg Germany)

Purpose of this contribution is to discuss a recommendation for the use of Decimal64, rather than using a string-based “float”.

Recommendation:

Use Decimal64 if the number will fit.

Use of strings and a pattern statement should be discouraged.

Rationale and examples:

One issue with strings is that two values could be the same, but in different formats and YANG doesn’t have the ability to differentiate.

For example:

"Number32": "42420e-3"

And

“Number32”: “4242e-2”

Are the same number but would not be equal in a “must” statement.

Pointer to a test program: https://github.com/samans/testing-yang/tree/main/numbers

Some thoughts from Don Fedyk:

Float64 is IEEE format number that encompasses a huge range and is not easily humanly readable. One result of covering a vast range is it is not as accurate for certain number such as small fractions.

Decimal64 is a format that is both humanly readable and consistently precises across its dynamic range where you can select a large number or small number range. For example, to cover pico seconds decimal64 uses 12 digit exponents and can represent 1 pico second 0.000000000001 precisely. Whereas the closest float64 representation is 0x3D719799812DEA11 which is really 9.99999999999999979886647629256E-13 seconds.

An implementation that uses decimal64 may not have to ever use floating point arithmetic if they want to avoid it. An implementation that uses float64 must use float64 logic at the very least to display a number that no one can read. Decimal64 is 2 numbers a 64 bits signed number and a power of 10 exponent. Float64 has 52 bits of mantissa versus 63 for Decimal64. Recommend you use a decimal64 for these type of values. 0.00000000003 is obviously bigger than 0.00000000002. Is 0x3D919799812DEA11 smaller than 0x3D8A636641C4DF1A? You need a calculator to be sure.