Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 25 September 2025

DM. Royer
RiverExplorer LLC

24 March 2025

Bits in ABNF
draft-royer-bits-in-abnf-00

Abstract

This is an extension to the ABNF specification to allow for bits definitions to be defined.

When interacting with hardware or bit oriented over the wire protocols ABNF is not currently the choice. This specification describes a method to add boolean and narrow bit values in order to fix that limitation.

And this note while in in draft status: A new Open Source XDR [RFC4506] and ABNF generation tool is being developed [xdrgen] which generates code from ABNF and XDR. It can also generate ABNF from XDR. And it can generate XDR from ABNF. Both also can produce C++ source and header files.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 25 September 2025.

Copyright Notice

Table of Contents

1. Introduction

This specification adds to [RFC5234], Section 2.3 Terminal values by adding an optional bit width to the binary, decimal, and hexadecimal terminals:

* The width is a positive integer expressed in decimal. This width is the number of bits in the terminal.

* The width must be equal to or greater than one (1).

* When the left side has a width then, the number of bits on the left of the equal sign must be the total of the widths on the right side of the equal sign.

* When the right side of the equal sign has widths and the left side does not. Then the right side can be a rule with a variable length.

* The left side only needs a width when it is significant and relevant to the protocol. A left side with no width may be present even when the right side is fixed with.

* The addition of a %p padding indicator with a required bit-width.

* An alteration of the "rulename" ABNF as shown in Figure 1.

A caution to the implementors of code generated from ABNF that specifies bit widths for signed integer values. Many computer languages will convert a narrower bit value into a wider bit value and move the sign bit to the most significant position. So when preparing a signed bit value, be sure to clamp the value and adjust the sign to the correct bit position before packing the bits. This would apply to signed integer values and not unsigned integer values.

The right most elements are placed into the least signification part of the value. The left most bits being more significant than the bits to the right of them as shown in examples Figure 2 and Figure 3.

The result are these existing definitions with the added optional width indicators.

ABNF

```
b           = binary

b:%d        = binary with a bit width of %d

d           = decimal

d:%d        = decimal with a bit width of %d

x           = hexadecimal

x:%d        = hexadecimal with a bit width of %d.

p:%d        = pad with %d zero bits.

one-or-more = %x31-39 *(0x30-39)

rulename    = ALPHA *(ALPHA / DIGIT / "_") *(":" one-or-more)
```

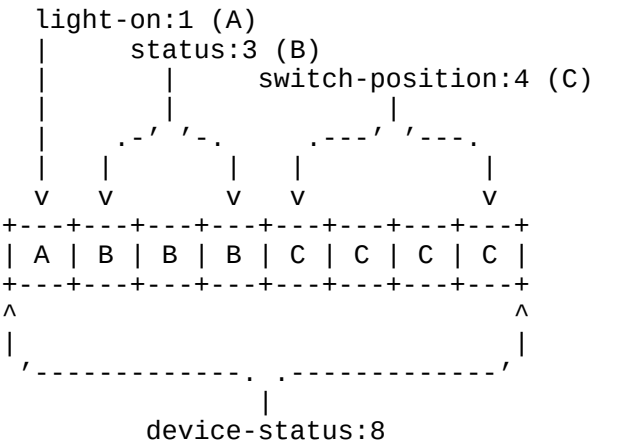
Figure 1: Extended ABNF Terminal Values

In Figure 2 example, "device-status" is 8-bits wide, and the three values to the right of the equal sign have a total width of 8-bits.

ABNF

```
device-status:8 = light-on:1 status:3 switch-position:4
```

Figure 2: "device-status"



in Figure 3 "email-status" is defined to be 8-bits wide, and only three bits are used, so five zero bits have been defined to be at the most significant positions. The total to the right of the equal sign is 8-bits:

ABNF

```
email-status:8  = %p:5 seen:1 flagged:1 deleted:1
```

Figure 3: "email-status"

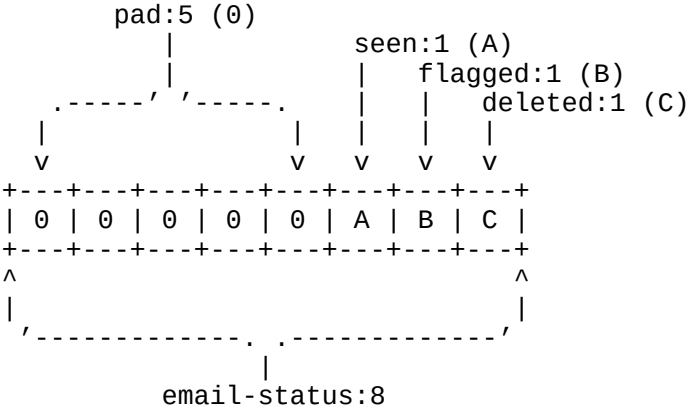


Figure Figure 4 shows an example where "c-string" on the left sides has no predefined width and the right side has a variable number of 8-bit wide values.

And "collection" has no predefined with and the right side is set of values with various and variable widths.

```
ABNF
----
c-string   = (ALPHA:8 / DIGIT:8 ) *(ALPHA:8 / DIGIT:8) %x00:8

collection = %p:4 header:12 c-string width:32 *(ALPHA:8)
```

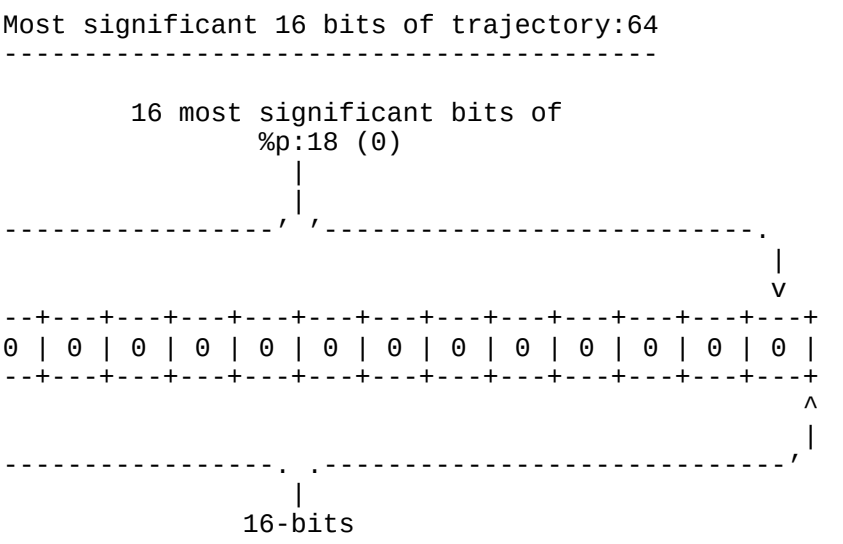
Figure 4: Padding in the Middle

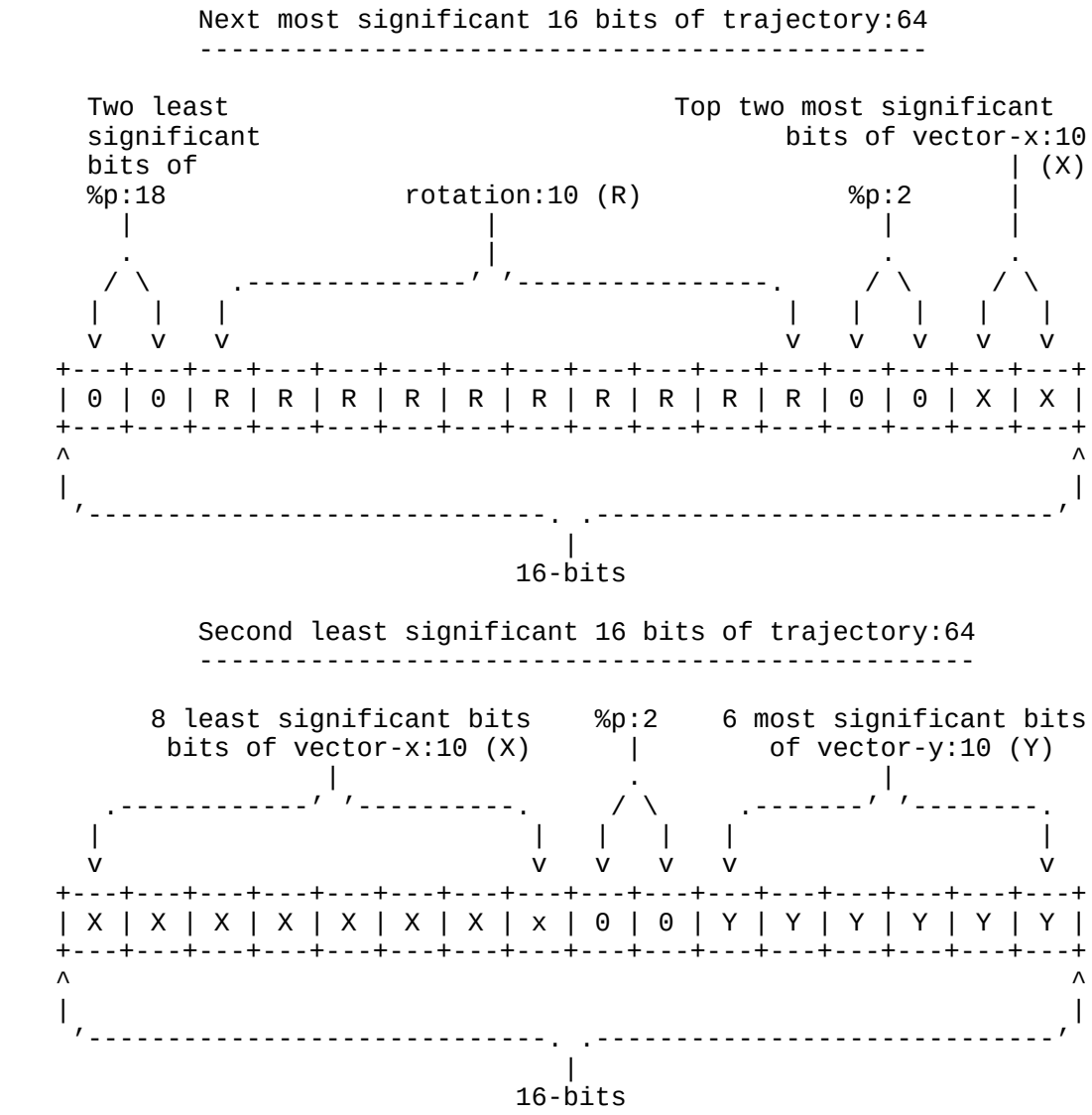
The padding does not need to be at the left. it can be in the middle and could have many zero bits as shown in Figure 5.

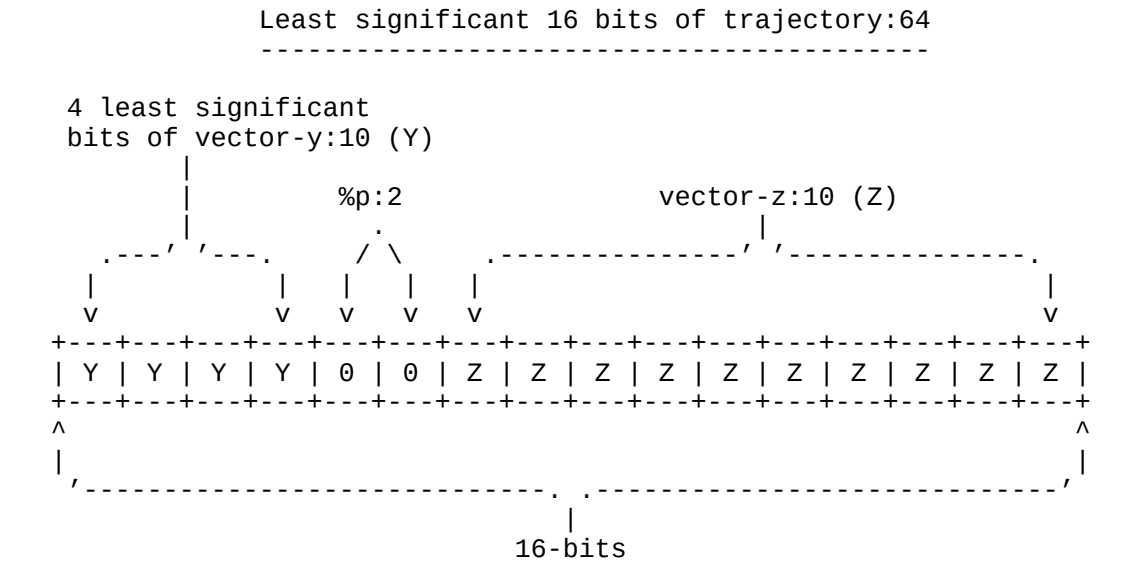
```
ABNF
----

trajectory:64 = %p:18 rotation:10 %p:2 vector-x:10 %p:2
                vector-y:10 %p:2 vector-z:10
```

Figure 5: Padding in the Middle




Next most significant 16 bits of trajectory:64
Two least significant bits of %p:18
rotation:10 (R)
Top two most significant bits of vector-x:10 (X)
%p:2
0 0 R R R R R R R R R R 0 0 X X
16-bits
Second least significant 16 bits of trajectory:64
8 least significant bits of vector-x:10 (X)
%p:2
6 most significant bits of vector-y:10 (Y)
X X X X X X X x 0 0 Y Y Y Y Y Y
16-bits



2. Additional Examples

The following sets of rules are identical and valid:

```
ABNF
----

; Original CRLF
CRLF          = %d13.10

; Declaring CRLF is 16-bits wide.
CRLF:16       = %d13.10

; Declaring CRLF is 16-bits wide.
CRLF:16       = %d13:8.10:8

; CRLF still is 16-bits wide, but declaring it is optional.
CRLF          = %d13:8.10:8

; Original rulename
rulename      = %d97 %d98 %d99

; Declaring rulename is 24-bits wide
rulename:24   = %d97:8 %d98:8 %d99:8

; rulename still is 24-bits wide, but declaring it is optional.
rulename      = %d97:8 %d98:8 %d99:8

; Orignal DIGIT
DIGIT         = %x30-39

; Declaring DIGIT is 8-bits wide.
DIGIT:8       = %x30-39

; Declaring DIGIT is 8-bits wide.
DIGIT:8       = %x30:8-39:8

; DIGIT is still 8-bits wide, but declaring it is optional.
DIGIT         = %x30:8-39:8
```

Figure 6: Comparing Rulesets.

3. IANA Considerations

This memo includes no request to IANA.

4. Security Considerations

This document should not affect the security of the Internet.

5. Normative References

[RFC4506] Eisler, M., Ed., "XDR: External Data Representation Standard", STD 67, RFC 4506, DOI 10.17487/RFC4506, May 2006, <https://www.rfc-editor.org/info/rfc4506>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

6. Informative References

[xdrgen] Royer, DM., "XDR Code Generator, Open Source", 2025, <https://github.com/RiverExplorer/Phoenix>.

Acknowledgments

Contributors

Author’s Address

Doug Royer
RiverExplorer LLC
848 N. Rainbow Blvd, Ste-1120
Las Vegas, Nevada 89107
United States of America
Phone: +1-208-806-1358
Email: DouglasRoyer@gmail.com
URI: https://DougRoyer.US