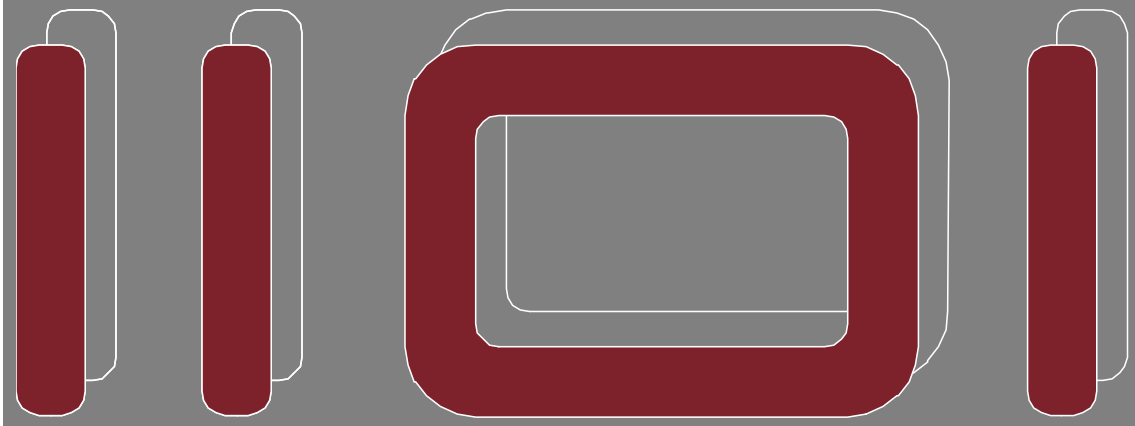

Painting by Numbers: Visualisation of structured IPv6-Adressing

helge.holz@dataport.de

dataport	**Painting by numbers** **Visualisation of structured IPv6-adressing**	**16.02.2011**

This document shows a simple guidance to structure a given ::/32 block of IPv6-Adresses.

An IPv6-adress has 128 Bit; the structure covers only the network part oft he first 64 bit, since the last 64 bit are reserved for hosts. If the network you want to structure is 2001:0DB8::/32, the first 32 bits are fix and you only have to cope with next 32 bits. For the adress-scheme you should plan in two steps: At first you split your network in 16 nets by the size of ::/36 and then in by the next iteration into 256 nets by the size of ::/40. That's sufficient to get a rough structure of the given adress-block; by iteration you get a finer plan (65536 nets of ::/48).

For your address-plan, you have to structure the adresses bitwise from front to end. Since IPv6-adresses are written hexadecimal, it is reasonable to split the adress-block in a first step into 16 blocks and number them consecutively (hexadecimal) to get structure of the first 4 bits (and a structure of the ::/36 nets).

2001:0DB8:0000::/36
2001:0DB8:1000::/36
2001:0DB8:2000::/36
2001:0DB8:3000::/36
2001:0DB8:4000::/36
2001:0DB8:5000::/36
2001:0DB8:6000::/36
2001:0DB8:7000::/36
2001:0DB8:8000::/36
2001:0DB8:9000::/36
2001:0DB8:A000::/36
2001:0DB8:B000::/36
2001:0DB8:C000::/36
2001:0DB8:D000::/36
2001:0DB8:E000::/36
2001:0DB8:F000::/36

This one-dimensional approach is not really scalable. For a two-dimensional presentation, you have to fill up squares by arranging the numbers (bitwise) in a matrix.

Binary the first two bits:

00	01
10	11

And by splitting the separate blocks **in the same way** for the next 2 bits (getting 4 bits and d 16 blocks):

dataport	Painting by numbers Visualisation of structured IPv6-adressing	16.02.2011

Binary :

0000	0001	0100	0101
0010	0011	0110	0111
1000	1001	1100	1101
1010	1011	1110	1111

equivalent Hexadecimal:

0	1	4	5
2	3	6	7
8	9	C	D
A	B	E	F

The red block shows the address-range of 2001:0DB8:8000::/36 or in other words all addresses beginning with 2001:0DB8:8 (not skipping leading zeros). There will be few customers, who will need such a large address-space, so you will need to split these block a little bit more. You split each of these blocks again into 16 fields and number them consecutively (hexadecimal), by keeping the prefix of the dissected field. By that you have dissected the given ::/32 block into ::/40 blocks, which was the goal.

Example fort he red marked block:

80	81	84	85
82	83	86	87
88	89	8C	8D
8A	8B	8E	8F

2001:0DB8:8000::/40
2001:0DB8:8100::/40
2001:0DB8:8200::/40
2001:0DB8:8300::/40
2001:0DB8:8400::/40
2001:0DB8:8500::/40
2001:0DB8:8600::/40
2001:0DB8:8700::/40
2001:0DB8:8800::/40
2001:0DB8:8900::/40
2001:0DB8:8A00::/40
2001:0DB8:8B00::/40
2001:0DB8:8C00::/40
2001:0DB8:8D00::/40
2001:0DB8:8E00::/40
2001:0DB8:8F00::/40

The blue marked block shows the address-space 2001:0DB8:8600::/40 or all addresses beginning with 2001:0DB8:86 (not skipping leading zeros). By this method you get a structured split of the given prefix, in an easy and readable form. The complete splitting into ::/40 blocks can be seen in the next matrix.

dataport	Painting by numbers Visualisation of structured IPv6-adressing	16.02.2011

Visualization-Matrix of aggregatable IPv6-address-ranges (8 Bit)

00	01	04	05	10	11	14	15	40	41	44	45	50	51	54	55
02	03	06	07	12	13	16	17	42	43	46	47	52	53	56	57
08	09	0C	0D	18	19	1C	1D	48	49	4C	4D	58	59	5C	5D
0A	0B	0E	0F	1A	1B	1E	1F	4A	4B	4E	4F	5A	5B	5E	5F
20	21	24	25	30	31	34	35	60	61	64	65	70	71	74	75
22	23	26	27	32	33	36	37	62	63	66	67	72	73	76	77
28	29	2C	2D	38	39	3C	3D	68	69	6C	6D	78	79	7C	7D
2A	2B	2E	2F	3A	3B	3E	3F	6A	6B	6E	6F	7A	7B	7E	7F
80	81	84	85	90	91	94	95	C0	C1	C4	C5	D0	D1	D4	D5
82	83	86	87	92	93	96	97	C2	C3	C6	C7	D2	D3	D6	D7
88	89	8C	8D	98	99	9C	9D	C8	C9	CC	CD	D8	D9	DC	DD
8A	8B	8E	8F	9A	9B	9E	9F	CA	CB	CE	CF	DA	DB	DE	DF
A0	A1	A4	A5	B0	B1	B4	B5	E0	E1	E4	E5	F0	F1	F4	F5
A2	A3	A6	A7	B2	B3	B6	B7	E2	E3	E6	E7	F2	F3	F6	F7
A8	A9	AC	AD	B8	B9	BC	BD	E8	E9	EC	ED	F8	F9	FC	FD
AA	AB	AE	AF	BA	BB	BE	BF	EA	EB	EE	EF	FA	FB	FE	FF

For each netwok-size there are marked smples in the above square:

2001:0DB8:1600::/40 violet
2001:0DB8:0600::/39 orange
2001:0DB8:2000::/38 dark-green
2001:0DB8:3800::/37 light-green
2001:0DB8:8000::/36 red
2001:0DB8:4000::/35 yellow
2001:0DB8:C000::/34 blue
2001:0DB8:8000::/33 the lower half
2001:0DB8:0000::/32 the whole address-block

By aggregation of aquares you get all sizes of address-spaces (4-times dark-green, 16-times red, 256-times blue). You also can aggregate two touching squares, as long as they are not belonging to seperate aquares, i.e. the division-line isn't wider as the side-lines (2-times orange, 8-times light-green, 32-fach yellow).
This Matrix can be uses for any given ::/32 prefix: prepending the prefix and you get a clearly structured net. Depending on the desired network-size you can allocate aggregatable networks from the above matrix.

The shown method can be iterated for splitting up the ::/40 blocks into ::/48, now getting a complete structured network of ::/48 blocks. The next iterations will get you to ::/56 and ::/64. (Or you are starting with a ::/48, then you are ready, because you've splitted your network already into 65536 of ::/64 network in a very structured way.