SteelHead Common Type Definitions v1.0

Created Jan 16, 2024 at 02:01 PM

Type: ipv4address

IPv4 address (x.y.z.k): A 32-bit IPv4 address in dotted-decimal format. It is described in IETF publication RFC791.

JSON

string

Property Name	Type	Description	Notes
ipv4address	*<string>*	IPv4 address (x.y.z.k): A 32-bit IPv4 address in dotted-decimal format. It is described in IETF publication RFC791.	Pattern: '^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])$';

Type: ipv4prefix

IPv4 prefix/subnet (x.y.z.k/<0-32>): A CIDR notion of IPv4 prefix/subnet representation that consists of an IPv4 address along with its routing prefix separated by a forward slash character("/").

JSON

string

Property Name	Type	Description	Notes
ipv4prefix	*<string>*	IPv4 prefix/subnet (x.y.z.k/<0-32>): A CIDR notion of IPv4 prefix/subnet representation that consists of an IPv4 address along with its routing prefix separated by a forward slash character("/").	Pattern: '^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(\d|[1-2]\d|3[0-2]))$';

Type: ipv6address

IPv6 address (x:y:z::k): A 128-bit IPv6 address described in IETF publication RFC2460, RFC4291, RFC5952, RFC6052, RFC6145.

JSON

string

Property Name	Type	Description	Notes
ipv6address	*<string>*	IPv6 address (x:y:z::k): A 128-bit IPv6 address described in IETF publication RFC2460, RFC4291, RFC5952, RFC6052, RFC6145.	Pattern: '^\s*((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){4}(((:[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:)))(%.+)?\s*$';

Type: ipv6prefix

IPv6 prefix/subnet (x:y:z::k/<0-128>): A CIDR notion of IPv6 prefix/subnet representation that consists of an IPv6 address along with its routing prefix separated by a forward slash character("/").

JSON

string

Property Name	Type	Description	Notes
ipv6prefix	*<string>*	IPv6 prefix/subnet (x:y:z::k/<0-128>): A CIDR notion of IPv6 prefix/subnet representation that consists of an IPv6 address along with its routing prefix separated by a forward slash character("/").	Pattern: '^\s*((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))(%.+)?\s*(\/(\d|\d\d|1[0-1]\d|12[0-8]))$';

Type: ipv4ipv6address

IPv4 or IPv6 address (x.y.z.k|x:y:z::k)

JSON

```
{
  "ipv4address": ipv4address,
  "ipv6address": ipv6address
}
```

Property Name	Type	Description	Notes
ipv4ipv6address	*<object>*	IPv4 or IPv6 address (x.y.z.k|x:y:z::k)	
ipv4ipv6address.ipv4address	*<ipv4address>*	IPv4 address (x.y.z.k): A 32-bit IPv4 address in dotted-decimal format. It is described in IETF publication RFC791.	Pattern: '^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])$';
ipv4ipv6address.ipv6address	*<ipv6address>*	IPv6 address (x:y:z::k): A 128-bit IPv6 address described in IETF publication RFC2460, RFC4291, RFC5952, RFC6052, RFC6145.	Pattern: '^\s*((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))(%.+)?\s*$';
ipv4ipv6address.oneOf[0]	*<object>*		Required properties: [ipv4address];
ipv4ipv6address.oneOf[0].<prop>	*<any>*		Optional;
ipv4ipv6address.oneOf[1]	*<object>*		Required properties: [ipv6address];
ipv4ipv6address.oneOf[1].<prop>	*<any>*		Optional;

Type: ipv4ipv6prefix

IPv4 or IPv6 prefix/subnet (x.y.z.k/<0-32>|x:y:z::k/<0-128>)

JSON

```
{
  "ipv4prefix": ipv4prefix,
  "ipv6prefix": ipv6prefix
}
```

Property Name	Type	Description	Notes
ipv4ipv6prefix	*<object>*	IPv4 or IPv6 prefix/subnet (x.y.z.k/<0-32>|x:y:z::k/<0-128>)	
ipv4ipv6prefix.ipv4prefix	*<ipv4prefix>*	IPv4 prefix/subnet (x.y.z.k/<0-32>): A CIDR notion of IPv4 prefix/subnet representation that consists of an IPv4 address along with its routing prefix separated by a forward slash character("/").	Pattern: '^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(\d|[1-2]\d|3[0-2]))$';
ipv4ipv6prefix.ipv6prefix	*<ipv6prefix>*	IPv6 prefix/subnet (x:y:z::k/<0-128>): A CIDR notion of IPv6 prefix/subnet representation that consists of an IPv6 address along with its routing prefix separated by a forward slash character("/").	Pattern: '^\s*((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(%.+)?\s*(\/(\d|\d\d|1[0-1]\d|12[0-8]))$';
ipv4ipv6prefix.oneOf[0]	*<object>*		Required properties: [ipv4prefix];
ipv4ipv6prefix.oneOf[0].<prop>	*<any>*		Optional;
ipv4ipv6prefix.oneOf[1]	*<object>*		Required properties: [ipv6prefix];
ipv4ipv6prefix.oneOf[1].<prop>	*<any>*		Optional;

Type: port

Valid port number

JSON

```
integer
```

Property Name	Type	Description	Notes
port	*<integer>*	Valid port number	Range: 0 to 65535;

Type: portlabel

A textual name assigned to a range of ports, for example the predefined port label "RBT-Proto" refers to ports in range of 7744, 7800-7801, 7810, 7820, 7850, 7860, 7870.

JSON

```
string
```

Property Name	Type	Description	Notes
portlabel	*<string>*	A textual name assigned to a range of ports, for example the predefined port label "RBT-Proto" refers to ports in range of 7744, 7800-7801, 7810, 7820, 7850, 7860, 7870.	

Type: port_portlabel

Port number or port label

JSON

```
{
  "port": port,
  "portlabel": portlabel
}
```

Property Name	Type	Description	Notes
port_portlabel	*<object>*	Port number or port label	
port_portlabel.port	*<port>*	Valid port number	
port_portlabel.portlabel	*<portlabel>*	A textual name assigned to a range of ports, for example the predefined port label "RBT-Proto" refers to ports in range of 7744, 7800-7801, 7810, 7820, 7850, 7860, 7870.	
port_portlabel.oneOf[0]	*<object>*		Required properties: [port];
port_portlabel.oneOf[0].<prop>	*<any>*		Optional;
port_portlabel.oneOf[1]	*<object>*		Required properties: [portlabel];
port_portlabel.oneOf[1].<prop>	*<any>*		Optional;

Type: vlan_tag

VLAN tag (all, untagged, tagged - [1-4094])

JSON

```
{
  "state": string,
  "tag": number
}
```

Property Name	Type	Description	Notes
vlan_tag	*<object>*	VLAN tag (all, untagged, tagged - [1-4094])	
vlan_tag.state	*<string>*	The state of a VLAN tag.	Optional; Values: all, untagged, tagged;
vlan_tag.tag	*<number>*	The "tag" is valid if "state" == "tagged".	Optional; Range: 1 to 4094;
vlan_tag.oneOf[0]	*<object>*		Required properties: [state];
vlan_tag.oneOf[0].state	*<string>*		Values: all, untagged;
vlan_tag.oneOf[1]	*<object>*		Required properties: [state, tag];
vlan_tag.oneOf[1].state	*<string>*		Values: tagged;
vlan_tag.oneOf[1].tag	*<number>*		