## NAME

systemd.netdev - Virtual Network Device configuration

## SYNOPSIS

*netdev*.netdev

## DESCRIPTION

Network setup is performed by **systemd-networkd(8)**.

The main Virtual Network Device file must have the extension .netdev; other extensions are ignored. Virtual network devices are created as soon as networkd is started. If a netdev with the specified name already exists, networkd will use that as-is rather than create its own. Note that the settings of the pre-existing netdev will not be changed by networkd.

The .netdev files are read from the files located in the system network directory /lib/systemd/network, the volatile runtime network directory /run/systemd/network and the local administration network directory /etc/systemd/network. All configuration files are collectively sorted and processed in lexical order, regardless of the directories in which they live. However, files with identical filenames replace each other. Files in /etc have the highest priority, files in /run take precedence over files with the same name in /lib. This can be used to override a system-supplied configuration file with a local file if needed. As a special case, an empty file (file size 0) or symlink with the same name pointing to /dev/null disables the configuration file entirely (it is "masked").

Along with the netdev file foo.netdev, a "drop-in" directory foo.netdev.d/ may exist. All files with the suffix ".conf" from this directory will be parsed after the file itself is parsed. This is useful to alter or add configuration settings, without having to modify the main configuration file. Each drop-in file must have appropriate section headers.

In addition to /etc/systemd/network, drop-in ".d" directories can be placed in /lib/systemd/network or /run/systemd/network directories. Drop-in files in /etc take precedence over those in /run which in turn take precedence over those in /lib. Drop-in files under any of these directories take precedence over the main netdev file wherever located. (Of course, since /run is temporary and /usr/lib is for vendors, it is unlikely drop-ins should be used in either of those places.)

## SUPPORTED NETDEV KINDS

The following kinds of virtual network devices may be configured in .netdev files:

**Table 1. Supported kinds of virtual network devices**

## [MATCH] SECTION OPTIONS

A virtual network device is only created if the "[Match]" section matches the current environment, or if the section is empty. The following keys are accepted:

*Host=*

Matches against the hostname or machine ID of the host. See "ConditionHost=" in **systemd.unit(5)** for details.

*Virtualization=*

Checks whether the system is executed in a virtualized environment and optionally test whether it is a specific implementation. See "ConditionVirtualization=" in **systemd.unit(5)** for details.

*KernelCommandLine=*

Checks whether a specific kernel command line option is set (or if prefixed with the exclamation mark unset). See "ConditionKernelCommandLine=" in **systemd.unit(5)** for details.

*Architecture=*

Checks whether the system is running on a specific architecture. See "ConditionArchitecture=" in **systemd.unit(5)** for details.

## [NETDEV] SECTION OPTIONS

The "[NetDev]" section accepts the following keys:

*Description=*

A free-form description of the netdev.

*Name=*

The interface name used when creating the netdev. This option is compulsory.

*Kind=*

The netdev kind. This option is compulsory. See the "Supported netdev kinds" section for the valid keys.

*MTUBytes=*

The maximum transmission unit in bytes to set for the device. The usual suffixes K, M, G, are supported and are understood to the base of 1024. This key is not currently supported for "tun" or "tap" devices.

*MACAddress=*

The MAC address to use for the device. If none is given, one is generated based on the interface name and the **machine-id(5)**. This key is not currently supported for "tun" or "tap" devices.

## [BRIDGE] SECTION OPTIONS

The "[Bridge]" section only applies for netdevs of kind "bridge", and accepts the following keys:

*HelloTimeSec=*

HelloTimeSec specifies the number of seconds between two hello packets sent out by the root bridge and the designated bridges. Hello packets are used to communicate information about the topology throughout the entire bridged local area network.

*MaxAgeSec=*

MaxAgeSec specifies the number of seconds of maximum message age. If the last seen (received) hello packet is more than this number of seconds old, the bridge in question will start the takeover procedure in attempt to become the Root Bridge itself.

*ForwardDelaySec=*

ForwardDelaySec specifies the number of seconds spent in each of the Listening and Learning states before the Forwarding state is entered.

*AgeingTimeSec=*

This specifies the number of seconds a MAC Address will be kept in the forwarding database after having a packet received from this MAC Address.

*Priority=*

The priority of the bridge. An integer between 0 and 65535. A lower value means higher priority. The bridge having the lowest priority will be elected as root bridge.

*DefaultPVID=*
> This specifies the default port VLAN ID of a newly attached bridge port.

*MulticastQuerier=*
> A boolean. This setting controls the IFLA_BR_MCAST_QUERIER option in the kernel. If enabled, the kernel will send general ICMP queries from a zero source address. This feature should allow faster convergence on startup, but it causes some multicast-aware switches to misbehave and disrupt forwarding of multicast packets. When unset, the kernel's default setting applies.

*MulticastSnooping=*
> A boolean. This setting controls the IFLA_BR_MCAST_SNOOPING option in the kernel. If enabled, IGMP snooping monitors the Internet Group Management Protocol (IGMP) traffic between hosts and multicast routers. When unset, the kernel's default setting applies.

*VLANFiltering=*
> A boolean. This setting controls the IFLA_BR_VLAN_FILTERING option in the kernel. If enabled, the bridge will be started in VLAN-filtering mode. When unset, the kernel's default setting applies.

*STP=*
> A boolean. This enables the bridge's Spanning Tree Protocol (STP). When unset, the kernel's default setting applies.

## [VLAN] SECTION OPTIONS

The "[VLAN]" section only applies for netdevs of kind "vlan", and accepts the following key:

*Id=*
> The VLAN ID to use. An integer in the range 0–4094. This option is compulsory.

## [MACVLAN] SECTION OPTIONS

The "[MACVLAN]" section only applies for netdevs of kind "macvlan", and accepts the following key:

*Mode=*
> The MACVLAN mode to use. The supported options are "private", "vepa", "bridge", and "passthru".

## [MACVTAP] SECTION OPTIONS

The "[MACVTAP]" section applies for netdevs of kind "macvtap" and accepts the same key as "[MACVLAN]".

## [IPVLAN] SECTION OPTIONS

The "[IPVLAN]" section only applies for netdevs of kind "ipvlan", and accepts the following key:

*Mode=*
> The IPVLAN mode to use. The supported options are "L2" and "L3".

## [VXLAN] SECTION OPTIONS

The "[VXLAN]" section only applies for netdevs of kind "vxlan", and accepts the following keys:

*Id=*
> The VXLAN ID to use.

*Group=*
> An assigned multicast group IP address.

*TOS=*
> The Type Of Service byte value for a vxlan interface.

*TTL=*
> A fixed Time To Live N on Virtual eXtensible Local Area Network packets. N is a number in the range 1–255. 0 is a special value meaning that packets inherit the TTL value.

*MacLearning=*
> A boolean. When true, enables dynamic MAC learning to discover remote MAC addresses.

*FDBAgeingSec=*
    The lifetime of Forwarding Database entry learnt by the kernel, in seconds.

*MaximumFDBEntries=*
    Configures maximum number of FDB entries.

*ARPProxy=*
    A boolean. When true, enables ARP proxying.

*L2MissNotification=*
    A boolean. When true, enables netlink LLADDR miss notifications.

*L3MissNotification=*
    A boolean. When true, enables netlink IP address miss notifications.

*RouteShortCircuit=*
    A boolean. When true, route short circuiting is turned on.

*UDPChecksum=*
    A boolean. When true, transmitting UDP checksums when doing VXLAN/IPv4 is turned on.

*UDP6ZeroChecksumTx=*
    A boolean. When true, sending zero checksums in VXLAN/IPv6 is turned on.

*UDP6ZeroChecksumRx=*
    A boolean. When true, receiving zero checksums in VXLAN/IPv6 is turned on.

*RemoteChecksumTx=*
    A boolean. When true, remote transmit checksum offload of VXLAN is turned on.

*RemoteChecksumRx=*
    A boolean. When true, remote receive checksum offload in VXLAN is turned on.

*GroupPolicyExtension=*
    A boolean. When true, it enables Group Policy VXLAN extension security label mechanism across network peers based on VXLAN. For details about the Group Policy VXLAN, see the **VXLAN Group Policy**[5] document. Defaults to false.

*DestinationPort=*
    Configures the default destination UDP port on a per-device basis. If destination port is not specified then Linux kernel default will be used. Set destination port 4789 to get the IANA assigned value, and destination port 0 to get default values.

*PortRange=*
    Configures VXLAN port range. VXLAN bases source UDP port based on flow to help the receiver to be able to load balance based on outer header flow. It restricts the port range to the normal UDP local ports, and allows overriding via configuration.

## [TUNNEL] SECTION OPTIONS

The "[Tunnel]" section only applies for netdevs of kind "ipip", "sit", "gre", "gretap", "ip6gre", "ip6gretap", "vti", "vti6", and "ip6tnl" and accepts the following keys:

*Local=*
    A static local address for tunneled packets. It must be an address on another interface of this host.

*Remote=*
    The remote endpoint of the tunnel.

*TOS=*
    The Type Of Service byte value for a tunnel interface. For details about the TOS, see the **Type of Service in the Internet Protocol Suite**[6] document.

*TTL=*
    A fixed Time To Live N on tunneled packets. N is a number in the range 1–255. 0 is a special value meaning that packets inherit the TTL value. The default value for IPv4 tunnels is: inherit. The default

value for IPv6 tunnels is 64.

*DiscoverPathMTU=*
A boolean. When true, enables Path MTU Discovery on the tunnel.

*IPv6FlowLabel=*
Configures the 20-bit flow label (see **RFC 6437**[7]) field in the IPv6 header (see **RFC 2460**[8]), which is used by a node to label packets of a flow. It is only used for IPv6 tunnels. A flow label of zero is used to indicate packets that have not been labeled. It can be configured to a value in the range 0–0xFFFFF, or be set to "inherit", in which case the original flowlabel is used.

*CopyDSCP=*
A boolean. When true, the Differentiated Service Code Point (DSCP) field will be copied to the inner header from outer header during the decapsulation of an IPv6 tunnel packet. DSCP is a field in an IP packet that enables different levels of service to be assigned to network traffic. Defaults to "no".

*EncapsulationLimit=*
The Tunnel Encapsulation Limit option specifies how many additional levels of encapsulation are permitted to be prepended to the packet. For example, a Tunnel Encapsulation Limit option containing a limit value of zero means that a packet carrying that option may not enter another tunnel before exiting the current tunnel. (see **RFC 2473**[9]). The valid range is 0–255 and "none". Defaults to 4.

*Key=*
The *Key=* parameter specifies the same key to use in both directions (*InputKey=* and *OutputKey=*). The *Key=* is either a number or an IPv4 address-like dotted quad. It is used as mark-configured SAD/SPD entry as part of the lookup key (both in data and control path) in ip xfrm (framework used to implement IPsec protocol). See **ip-xfrm — transform configuration**[10] for details. It is only used for VTI/VTI6 tunnels.

*InputKey=*
The *InputKey=* parameter specifies the key to use for input. The format is same as *Key=*. It is only used for VTI/VTI6 tunnels.

*OutputKey=*
The *OutputKey=* parameter specifies the key to use for output. The format is same as *Key=*. It is only used for VTI/VTI6 tunnels.

*Mode=*
An "ip6tnl" tunnel can be in one of three modes "ip6ip6" for IPv6 over IPv6, "ipip6" for IPv4 over IPv6 or "any" for either.

## [PEER] SECTION OPTIONS
The "[Peer]" section only applies for netdevs of kind "veth" and accepts the following keys:

*Name=*
The interface name used when creating the netdev. This option is compulsory.

*MACAddress=*
The peer MACAddress, if not set, it is generated in the same way as the MAC address of the main interface.

## [TUN] SECTION OPTIONS
The "[Tun]" section only applies for netdevs of kind "tun", and accepts the following keys:

*OneQueue=*
Takes a boolean argument. Configures whether all packets are queued at the device (enabled), or a fixed number of packets are queued at the device and the rest at the "qdisc". Defaults to "no".

*MultiQueue=*
Takes a boolean argument. Configures whether to use multiple file descriptors (queues) to parallelize packets sending and receiving. Defaults to "no".

*PacketInfo=*

Takes a boolean argument. Configures whether packets should be prepended with four extra bytes (two flag bytes and two protocol bytes). If disabled, it indicates that the packets will be pure IP packets. Defaults to "no".

*VNetHeader=*

Takes a boolean argument. Configures IFF_VNET_HDR flag for a tap device. It allows sending and receiving larger Generic Segmentation Offload (GSO) packets. This may increase throughput significantly. Defaults to "no".

*User=*

User to grant access to the /dev/net/tun device.

*Group=*

Group to grant access to the /dev/net/tun device.

## [TAP] SECTION OPTIONS

The "[Tap]" section only applies for netdevs of kind "tap", and accepts the same keys as the "[Tun]" section.

## [BOND] SECTION OPTIONS

The "[Bond]" section accepts the following key:

*Mode=*

Specifies one of the bonding policies. The default is "balance-rr" (round robin). Possible values are "balance-rr", "active-backup", "balance-xor", "broadcast", "802.3ad", "balance-tlb", and "balance-alb".

*TransmitHashPolicy=*

Selects the transmit hash policy to use for slave selection in balance-xor, 802.3ad, and tlb modes. Possible values are "layer2", "layer3+4", "layer2+3", "encap2+3", and "encap3+4".

*LACPTransmitRate=*

Specifies the rate with which link partner transmits Link Aggregation Control Protocol Data Unit packets in 802.3ad mode. Possible values are "slow", which requests partner to transmit LACPDUs every 30 seconds, and "fast", which requests partner to transmit LACPDUs every second. The default value is "slow".

*MIIMonitorSec=*

Specifies the frequency that Media Independent Interface link monitoring will occur. A value of zero disables MII link monitoring. This value is rounded down to the nearest millisecond. The default value is 0.

*UpDelaySec=*

Specifies the delay before a link is enabled after a link up status has been detected. This value is rounded down to a multiple of MIIMonitorSec. The default value is 0.

*DownDelaySec=*

Specifies the delay before a link is disabled after a link down status has been detected. This value is rounded down to a multiple of MIIMonitorSec. The default value is 0.

*LearnPacketIntervalSec=*

Specifies the number of seconds between instances where the bonding driver sends learning packets to each slave peer switch. The valid range is 1–0x7fffffff; the default value is 1. This option has an effect only for the balance-tlb and balance-alb modes.

*AdSelect=*

Specifies the 802.3ad aggregation selection logic to use. Possible values are "stable", "bandwidth" and "count".

*FailOverMACPolicy=*

Specifies whether the active-backup mode should set all slaves to the same MAC address at the time of enslavement or, when enabled, to perform special handling of the bond's MAC address in accordance with the selected policy. The default policy is none. Possible values are "none", "active" and "follow".

*ARPValidate=*
　　Specifies whether or not ARP probes and replies should be validated in any mode that supports ARP monitoring, or whether non-ARP traffic should be filtered (disregarded) for link monitoring purposes. Possible values are "none", "active", "backup" and "all".

*ARPIntervalSec=*
　　Specifies the ARP link monitoring frequency in milliseconds. A value of 0 disables ARP monitoring. The default value is 0.

*ARPIPTargets=*
　　Specifies the IP addresses to use as ARP monitoring peers when ARPIntervalSec is greater than 0. These are the targets of the ARP request sent to determine the health of the link to the targets. Specify these values in IPv4 dotted decimal format. At least one IP address must be given for ARP monitoring to function. The maximum number of targets that can be specified is 16. The default value is no IP addresses.

*ARPAllTargets=*
　　Specifies the quantity of ARPIPTargets that must be reachable in order for the ARP monitor to consider a slave as being up. This option affects only active-backup mode for slaves with ARPValidate enabled. Possible values are "any" and "all".

*PrimaryReselectPolicy=*
　　Specifies the reselection policy for the primary slave. This affects how the primary slave is chosen to become the active slave when failure of the active slave or recovery of the primary slave occurs. This option is designed to prevent flip-flopping between the primary slave and other slaves. Possible values are "always", "better" and "failure".

*ResendIGMP=*
　　Specifies the number of IGMP membership reports to be issued after a failover event. One membership report is issued immediately after the failover, subsequent packets are sent in each 200ms interval. The valid range is 0–255. Defaults to 1. A value of 0 prevents the IGMP membership report from being issued in response to the failover event.

*PacketsPerSlave=*
　　Specify the number of packets to transmit through a slave before moving to the next one. When set to 0, then a slave is chosen at random. The valid range is 0–65535. Defaults to 1. This option only has effect when in balance-rr mode.

*GratuitousARP=*
　　Specify the number of peer notifications (gratuitous ARPs and unsolicited IPv6 Neighbor Advertisements) to be issued after a failover event. As soon as the link is up on the new slave, a peer notification is sent on the bonding device and each VLAN sub-device. This is repeated at each link monitor interval (ARPIntervalSec or MIIMonitorSec, whichever is active) if the number is greater than 1. The valid range is 0–255. The default value is 1. These options affect only the active-backup mode.

*AllSlavesActive=*
　　A boolean. Specifies that duplicate frames (received on inactive ports) should be dropped when false, or delivered when true. Normally, bonding will drop duplicate frames (received on inactive ports), which is desirable for most users. But there are some times it is nice to allow duplicate frames to be delivered. The default value is false (drop duplicate frames received on inactive ports).

*MinLinks=*
　　Specifies the minimum number of links that must be active before asserting carrier. The default value is 0.

For more detail information see **Linux Ethernet Bonding Driver HOWTO**[1]

## EXAMPLE
**Example 1. /etc/systemd/network/25-bridge.netdev**

[NetDev]
Name=bridge0

Kind=bridge

**Example 2. /etc/systemd/network/25-vlan1.netdev**

[Match]
Virtualization=no

[NetDev]
Name=vlan1
Kind=vlan

[VLAN]
Id=1

**Example 3. /etc/systemd/network/25-ipip.netdev**

[NetDev]
Name=ipip-tun
Kind=ipip
MTUBytes=1480

[Tunnel]
Local=192.168.223.238
Remote=192.169.224.239
TTL=64

**Example 4. /etc/systemd/network/25-tap.netdev**

[NetDev]
Name=tap-test
Kind=tap

[Tap]
MultiQueue=true
PacketInfo=true

**Example 5. /etc/systemd/network/25-sit.netdev**

[NetDev]
Name=sit-tun
Kind=sit
MTUBytes=1480

[Tunnel]
Local=10.65.223.238
Remote=10.65.223.239

**Example 6. /etc/systemd/network/25-gre.netdev**

[NetDev]
Name=gre-tun
Kind=gre
MTUBytes=1480

[Tunnel]
Local=10.65.223.238
Remote=10.65.223.239

**Example 7. /etc/systemd/network/25-vti.netdev**

[NetDev]
Name=vti-tun
Kind=vti
MTUBytes=1480

[Tunnel]
Local=10.65.223.238
Remote=10.65.223.239

**Example 8. /etc/systemd/network/25-veth.netdev**

[NetDev]
Name=veth-test
Kind=veth

[Peer]
Name=veth-peer

**Example 9. /etc/systemd/network/25-bond.netdev**

[NetDev]
Name=bond1
Kind=bond

[Bond]
Mode=802.3ad
TransmitHashPolicy=layer3+4
MIIMonitorSec=1s
LACPTransmitRate=fast

**Example 10. /etc/systemd/network/25-dummy.netdev**

[NetDev]
Name=dummy-test
Kind=dummy
MACAddress=12:34:56:78:9a:bc

**Example 11. /etc/systemd/network/25-vrf.netdev**

Create a VRF interface with table 42.

[NetDev]
Name=vrf-test
Kind=vrf

[VRF]
TableId=42

## SEE ALSO

systemd(1), systemd-networkd(8), systemd.link(5), systemd.network(5)

## NOTES

1. Linux Ethernet Bonding Driver HOWTO
   https://www.kernel.org/doc/Documentation/networking/bonding.txt

2. RFC 2784
   https://tools.ietf.org/html/rfc2784

3. IEEE 802.1Q
   http://www.ieee802.org/1/pages/802.1Q.html

4. VRF
   https://www.kernel.org/doc/Documentation/networking/vrf.txt

5. VXLAN Group Policy
   https://tools.ietf.org/html/draft-smith-vxlan-group-policy

6. Type of Service in the Internet Protocol Suite
   http://tools.ietf.org/html/rfc1349

7.  RFC 6437
    https://tools.ietf.org/html/rfc6437

8.  RFC 2460
    https://tools.ietf.org/html/rfc2460

9.  RFC 2473
    https://tools.ietf.org/html/rfc2473#section-4.1.1

10. ip-xfrm — transform configuration
    http://man7.org/linux/man-pages/man8/ip-xfrm.8.html