NAME

systemd.netdev − Virtual Network Device configuration

SYNOPSIS

netdev.netdev

DESCRIPTION

A plain ini−style text file that encodes configuration about a virtual network device, used by **systemd-networkd**(8). See **systemd.syntax**(7) for a general description of the syntax.

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 merged in the alphanumeric order and parsed after the main file itself has been 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. When prefixed with an exclamation mark ("!"), the result is negated. If an empty string is assigned, then previously assigned value is cleared.

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. When prefixed with an exclamation mark ("!"), the result is negated. If an empty string is assigned, then previously assigned value is cleared.

KernelCommandLine=
: Checks whether a specific kernel command line option is set. See "ConditionKernelCommandLine=" in **systemd.unit**(5) for details. When prefixed with an exclamation mark ("!"), the result is negated. If an empty string is assigned, then previously assigned value is cleared.

KernelVersion=
: Checks whether the kernel version (as reported by **uname −r**) matches a certain expression. See "ConditionKernelVersion=" in **systemd.unit**(5) for details. When prefixed with an exclamation mark ("!"), the result is negated. If an empty string is assigned, then previously assigned value is cleared.

Architecture=
: Checks whether the system is running on a specific architecture. See "ConditionArchitecture=" in **systemd.unit**(5) for details. When prefixed with an exclamation mark ("!"), the result is negated. If an empty string is assigned, then previously assigned value is cleared.

Firmware=
: Checks whether the system is running on a machine with the specified firmware. See "ConditionFirmware=" in **systemd.unit**(5) for details. When prefixed with an exclamation mark ("!"), the result is negated. If an empty string is assigned, then previously assigned value is cleared.

[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 setting is compulsory.

Kind=
: The netdev kind. This setting 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. For "tun" or "tap" devices, *MTUBytes=* setting is not currently supported in [NetDev] section. Please specify it in [Link] section of corresponding **systemd.network**(5) files.

MACAddress=
: The MAC address to use for the device. For "tun" or "tap" devices, setting *MACAddress=* in the [NetDev] section is not supported. Please specify it in [Link] section of the corresponding **systemd.network**(5) file. If this option is not set, "vlan" devices inherit the MAC address of the physical interface. For other kind of netdevs, if this option is not set, then MAC address is generated based on the interface name and the **machine-id**(5).

[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.

GroupForwardMask=
: A 16−bit bitmask represented as an integer which allows forwarding of link local frames with 802.1D reserved addresses (01:80:C2:00:00:0X). A logical AND is performed between the specified bitmask and the exponentiation of 2^X, the lower nibble of the last octet of the MAC address. For example, a value of 8 would allow forwarding of frames addressed to 01:80:C2:00:00:03 (802.1X PAE).

DefaultPVID=
: This specifies the default port VLAN ID of a newly attached bridge port. Set this to an integer in the range 1...4094 or "none" to disable the PVID.

MulticastQuerier=
: Takes 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 will be used.

MulticastSnooping=
: Takes 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 will be used.

VLANFiltering=
: Takes 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 will be used.

VLANProtocol=
: Allows setting the protocol used for VLAN filtering. Takes **802.1q** or, **802.1ad**, and defaults to unset and kernel's default is used.

STP=
: Takes a boolean. This enables the bridge's Spanning Tree Protocol (STP). When unset, the kernel's default will be used.

MulticastIGMPVersion=
: Allows changing bridge's multicast Internet Group Management Protocol (IGMP) version. Takes an integer 2 or 3. When unset, the kernel's default will be used.

[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 setting is compulsory.

Protocol=
: Allows setting the protocol used for the VLAN interface. Takes "802.1q" or, "802.1ad", and defaults to unset and kernel's default is used.

GVRP=
: Takes a boolean. The Generic VLAN Registration Protocol (GVRP) is a protocol that allows automatic learning of VLANs on a network. When unset, the kernel's default will be used.

MVRP=
: Takes a boolean. Multiple VLAN Registration Protocol (MVRP) formerly known as GARP VLAN Registration Protocol (GVRP) is a standards−based Layer 2 network protocol, for automatic configuration of VLAN information on switches. It was defined in the 802.1ak amendment to 802.1Q−2005. When unset, the kernel's default will be used.

LooseBinding=
: Takes a boolean. The VLAN loose binding mode, in which only the operational state is passed from the parent to the associated VLANs, but the VLAN device state is not changed. When unset, the kernel's default will be used.

ReorderHeader=
: Takes a boolean. When enabled, the VLAN reorder header is used and VLAN interfaces behave like physical interfaces. When unset, the kernel's default will be used.

EgressQOSMaps=
: Defines a mapping of Linux internal packet priority (**SO_PRIORITY**) to VLAN header PCP field for outgoing frames. Takes a whitespace−separated list of unsigned integer pairs in the format "from"−"to", e.g., "21−7 45−5" ranges 1–4294967294. Note that "from" must be greater than or equal to "to". When unset, the kernel's default will be used.

IngressQOSMaps=
: Defines a mapping of Linux internal packet priority (**SO_PRIORITY**) to VLAN header PCP field for incoming frames. Takes a whitespace−separated list of unsigned integer pairs in the format "from"−"to", e.g., "21−7 45−5" ranges 1–4294967294. Note that "from" must be greater than or equal to "to". When unset, the kernel's default will be used.

[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", "passthru", and "source".

SourceMACAddress=
: A whitespace−separated list of remote hardware addresses allowed on the MACVLAN. This option only has an effect in source mode. Use full colon−, hyphen− or dot−delimited hexadecimal. This option may appear more than once, in which case the lists are merged. If the empty string is assigned to this option, the list of hardware addresses defined prior to this is reset. Defaults to unset.

BroadcastMulticastQueueLength=
: Specifies the length of the receive queue for broadcast/multicast packets. An unsigned integer in the range 0...4294967294. Defaults to unset.

[MACVTAP] SECTION OPTIONS

The [MACVTAP] section applies for netdevs of kind "macvtap" and accepts the same keys 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","L3" and "L3S".

Flags=
: The IPVLAN flags to use. The supported options are "bridge","private" and "vepa".

[IPVTAP] SECTION OPTIONS

The [IPVTAP] section only applies for netdevs of kind "ipvtap" and accepts the same keys as [IPVLAN].

[VXLAN] SECTION OPTIONS

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

VNI=
: The VXLAN Network Identifier (or VXLAN Segment ID). Takes a number in the range 1...16777215.

Remote=
: Configures destination IP address.

Local=
: Configures local IP address.

Group=
: Configures VXLAN multicast group IP address. All members of a VXLAN must use the same multicast group 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. Takes "inherit" or a number in the range 0...255. 0 is a special value meaning inherit the inner protocol's TTL value. "inherit" means that it will inherit the outer protocol's TTL value.

MacLearning=
: Takes 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.

ReduceARPProxy=
: Takes a boolean. When true, bridge−connected VXLAN tunnel endpoint answers ARP requests from the local bridge on behalf of remote Distributed Overlay Virtual Ethernet **(DVOE)**[6] clients. Defaults to false.

L2MissNotification=
: Takes a boolean. When true, enables netlink LLADDR miss notifications.

L3MissNotification=
: Takes a boolean. When true, enables netlink IP address miss notifications.

RouteShortCircuit=
: Takes a boolean. When true, route short circuiting is turned on.

UDPChecksum=
: Takes a boolean. When true, transmitting UDP checksums when doing VXLAN/IPv4 is turned on.

UDP6ZeroChecksumTx=
: Takes a boolean. When true, sending zero checksums in VXLAN/IPv6 is turned on.

UDP6ZeroChecksumRx=

Takes a boolean. When true, receiving zero checksums in VXLAN/IPv6 is turned on.

RemoteChecksumTx=
: Takes a boolean. When true, remote transmit checksum offload of VXLAN is turned on.

RemoteChecksumRx=
: Takes a boolean. When true, remote receive checksum offload in VXLAN is turned on.

GroupPolicyExtension=
: Takes 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**[7] document. Defaults to false.

GenericProtocolExtension=
: Takes a boolean. When true, Generic Protocol Extension extends the existing VXLAN protocol to provide protocol typing, OAM, and versioning capabilities. For details about the VXLAN GPE Header, see the **Generic Protocol Extension for VXLAN**[8] document. If destination port is not specified and Generic Protocol Extension is set then default port of 4790 is used. Defaults to false.

DestinationPort=
: Configures the default destination UDP port. If the destination port is not specified then Linux kernel default will be used. Set to 4789 to get the IANA assigned value.

PortRange=
: Configures the source port range for the VXLAN. The kernel assigns the source UDP port based on the flow to help the receiver to do load balancing. When this option is not set, the normal range of local UDP ports is used.

FlowLabel=
: Specifies the flow label to use in outgoing packets. The valid range is 0−1048575.

IPDoNotFragment=
: Allows setting the IPv4 Do not Fragment (DF) bit in outgoing packets, or to inherit its value from the IPv4 inner header. Takes a boolean value, or "inherit". Set to "inherit" if the encapsulated protocol is IPv6. When unset, the kernel's default will be used.

[GENEVE] SECTION OPTIONS

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

Id=
: Specifies the Virtual Network Identifier (VNI) to use, a number between 0 and 16777215. This field is mandatory.

Remote=
: Specifies the unicast destination IP address to use in outgoing packets.

TOS=
: Specifies the TOS value to use in outgoing packets. Takes a number between 1 and 255.

TTL=
: Accepts the same values as in the [VXLAN] section, except that when unset or set to 0, the kernel's default will be used, meaning that packet TTL will be set from /proc/sys/net/ipv4/ip_default_ttl.

UDPChecksum=
: Takes a boolean. When true, specifies that UDP checksum is calculated for transmitted packets over IPv4.

UDP6ZeroChecksumTx=
: Takes a boolean. When true, skip UDP checksum calculation for transmitted packets over IPv6.

UDP6ZeroChecksumRx=
: Takes a boolean. When true, allows incoming UDP packets over IPv6 with zero checksum field.

DestinationPort=
: Specifies destination port. Defaults to 6081. If not set or assigned the empty string, the default port of

6081 is used.

FlowLabel=
Specifies the flow label to use in outgoing packets.

IPDoNotFragment=
Accepts the same key as in [VXLAN] section.

Independent=
Takes a boolean. When true, the vxlan interface is created without any underlying network interface. Defaults to false, which means that a .network file that requests this tunnel using *Tunnel=* is required for the tunnel to be created.

[BAREUDP] SECTION OPTIONS

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

DestinationPort=
Specifies the destination UDP port (in range 1...65535). This is mandatory.

EtherType=
Specifies the L3 protocol. Takes one of "ipv4", "ipv6", "mpls−uc" or "mpls−mc". This is mandatory.

[L2TP] SECTION OPTIONS

The [L2TP] section only applies for netdevs of kind "l2tp", and accepts the following keys:

TunnelId=
Specifies the tunnel identifier. Takes an number in the range 1...4294967295. The value used must match the "PeerTunnelId=" value being used at the peer. This setting is compulsory.

PeerTunnelId=
Specifies the peer tunnel id. Takes a number in the range 1...4294967295. The value used must match the "TunnelId=" value being used at the peer. This setting is compulsory.

Remote=
Specifies the IP address of the remote peer. This setting is compulsory.

Local=
Specifies the IP address of the local interface. Takes an IP address, or the special values "auto", "static", or "dynamic". When an address is set, then the local interface must have the address. If "auto", then one of the addresses on the local interface is used. Similarly, if "static" or "dynamic" is set, then one of the static or dynamic addresses on the local interface is used. Defaults to "auto".

EncapsulationType=
Specifies the encapsulation type of the tunnel. Takes one of "udp" or "ip".

UDPSourcePort=
Specifies the UDP source port to be used for the tunnel. When UDP encapsulation is selected it's mandatory. Ignored when IP encapsulation is selected.

UDPDestinationPort=
Specifies destination port. When UDP encapsulation is selected it's mandatory. Ignored when IP encapsulation is selected.

UDPChecksum=
Takes a boolean. When true, specifies that UDP checksum is calculated for transmitted packets over IPv4.

UDP6ZeroChecksumTx=
Takes a boolean. When true, skip UDP checksum calculation for transmitted packets over IPv6.

UDP6ZeroChecksumRx=
Takes a boolean. When true, allows incoming UDP packets over IPv6 with zero checksum field.

[L2TPSESSION] SECTION OPTIONS

The [L2TPSession] section only applies for netdevs of kind "l2tp", and accepts the following keys:

Name=
: Specifies the name of the session. This setting is compulsory.

SessionId=
: Specifies the session identifier. Takes an number in the range 1...4294967295. The value used must match the "SessionId=" value being used at the peer. This setting is compulsory.

PeerSessionId=
: Specifies the peer session identifier. Takes an number in the range 1...4294967295. The value used must match the "PeerSessionId=" value being used at the peer. This setting is compulsory.

Layer2SpecificHeader=
: Specifies layer2specific header type of the session. One of "none" or "default". Defaults to "default".

[MACSEC] SECTION OPTIONS

The [MACsec] section only applies for network devices of kind "macsec", and accepts the following keys:

Port=
: Specifies the port to be used for the MACsec transmit channel. The port is used to make secure channel identifier (SCI). Takes a value between 1 and 65535. Defaults to unset.

Encrypt=
: Takes a boolean. When true, enable encryption. Defaults to unset.

[MACSECRECEIVECHANNEL] SECTION OPTIONS

The [MACsecReceiveChannel] section only applies for network devices of kind "macsec", and accepts the following keys:

Port=
: Specifies the port to be used for the MACsec receive channel. The port is used to make secure channel identifier (SCI). Takes a value between 1 and 65535. This option is compulsory, and is not set by default.

MACAddress=
: Specifies the MAC address to be used for the MACsec receive channel. The MAC address used to make secure channel identifier (SCI). This setting is compulsory, and is not set by default.

[MACSECTRANSMITASSOCIATION] SECTION OPTIONS

The [MACsecTransmitAssociation] section only applies for network devices of kind "macsec", and accepts the following keys:

PacketNumber=
: Specifies the packet number to be used for replay protection and the construction of the initialization vector (along with the secure channel identifier [SCI]). Takes a value between 1−4,294,967,295. Defaults to unset.

KeyId=
: Specifies the identification for the key. Takes a number between 0−255. This option is compulsory, and is not set by default.

Key=
: Specifies the encryption key used in the transmission channel. The same key must be configured on the peer's matching receive channel. This setting is compulsory, and is not set by default. Takes a 128−bit key encoded in a hexadecimal string, for example "dffafc8d7b9a43d5b9a3dfbbf6a30c16".

KeyFile=
: Takes an absolute path to a file which contains a 128−bit key encoded in a hexadecimal string, which will be used in the transmission channel. When this option is specified, *Key=* is ignored. Note that the file must be readable by the user "systemd−network", so it should be, e.g., owned by "root:systemd−network" with a "0640" file mode. If the path refers to an **AF_UNIX** stream socket in

the file system a connection is made to it and the key read from it.

Activate=
: Takes a boolean. If enabled, then the security association is activated. Defaults to unset.

UseForEncoding=
: Takes a boolean. If enabled, then the security association is used for encoding. Only one [MACsecTransmitAssociation] section can enable this option. When enabled, *Activate=yes* is implied. Defaults to unset.

[MACSECRECEIVEASSOCIATION] SECTION OPTIONS

The [MACsecReceiveAssociation] section only applies for network devices of kind "macsec", and accepts the following keys:

Port=
: Accepts the same key as in [MACsecReceiveChannel] section.

MACAddress=
: Accepts the same key as in [MACsecReceiveChannel] section.

PacketNumber=
: Accepts the same key as in [MACsecTransmitAssociation] section.

KeyId=
: Accepts the same key as in [MACsecTransmitAssociation] section.

Key=
: Accepts the same key as in [MACsecTransmitAssociation] section.

KeyFile=
: Accepts the same key as in [MACsecTransmitAssociation] section.

Activate=
: Accepts the same key as in [MACsecTransmitAssociation] section.

[TUNNEL] SECTION OPTIONS

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

Local=
: A static local address for tunneled packets. It must be an address on another interface of this host, or the special value "any".

Remote=
: The remote endpoint of the tunnel. Takes an IP address or the special value "any".

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**[9] 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 0 (inherit). The default value for IPv6 tunnels is 64.

DiscoverPathMTU=
: Takes a boolean. When true, enables Path MTU Discovery on the tunnel.

IPv6FlowLabel=
: Configures the 20−bit flow label (see **RFC 6437**[10]) field in the IPv6 header (see **RFC 2460**[11]), 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=

Takes 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**[12]). 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**[13] for details. It is only used for VTI/VTI6, GRE, GRETAP, and ERSPAN 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, GRE, GRETAP, and ERSPAN 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, GRE, GRETAP, and ERSPAN 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.

Independent=
: Takes a boolean. When false (the default), the tunnel is always created over some network device, and a .network file that requests this tunnel using *Tunnel=* is required for the tunnel to be created. When true, the tunnel is created independently of any network as "tunnel@NONE".

AssignToLoopback=
: Takes a boolean. If set to "yes", the loopback interface "lo" is used as the underlying device of the tunnel interface. Defaults to "no".

AllowLocalRemote=
: Takes a boolean. When true allows tunnel traffic on *ip6tnl* devices where the remote endpoint is a local host address. When unset, the kernel's default will be used.

FooOverUDP=
: Takes a boolean. Specifies whether *FooOverUDP=* tunnel is to be configured. Defaults to false. This takes effects only for IPIP, SIT, GRE, and GRETAP tunnels. For more detail information see **Foo over UDP**[14]

FOUDestinationPort=
: This setting specifies the UDP destination port for encapsulation. This field is mandatory when *FooOverUDP=yes*, and is not set by default.

FOUSourcePort=
: This setting specifies the UDP source port for encapsulation. Defaults to **0** — that is, the source port for packets is left to the network stack to decide.

Encapsulation=
: Accepts the same key as in the [FooOverUDP] section.

IPv6RapidDeploymentPrefix=
: Reconfigure the tunnel for **IPv6 Rapid Deployment**[15], also known as 6rd. The value is an ISP−specific IPv6 prefix with a non−zero length. Only applicable to SIT tunnels.

ISATAP=

Takes a boolean. If set, configures the tunnel as Intra−Site Automatic Tunnel Addressing Protocol (ISATAP) tunnel. Only applicable to SIT tunnels. When unset, the kernel's default will be used.

SerializeTunneledPackets=
Takes a boolean. If set to yes, then packets are serialized. Only applies for GRE, GRETAP, and ERSPAN tunnels. When unset, the kernel's default will be used.

ERSPANIndex=
Specifies the ERSPAN index field for the interface, an integer in the range 1...1048575 associated with the ERSPAN traffic's source port and direction. This field is mandatory.

[FOOOVERUDP] SECTION OPTIONS

The [FooOverUDP] section only applies for netdevs of kind "fou" and accepts the following keys:

Encapsulation=
Specifies the encapsulation mechanism used to store networking packets of various protocols inside the UDP packets. Supports the following values: "FooOverUDP" provides the simplest no−frills model of UDP encapsulation, it simply encapsulates packets directly in the UDP payload. "GenericUDPEncapsulation" is a generic and extensible encapsulation, it allows encapsulation of packets for any IP protocol and optional data as part of the encapsulation. For more detailed information see **Generic UDP Encapsulation**[16]. Defaults to "FooOverUDP".

Port=
Specifies the port number where the encapsulated packets will arrive. Those packets will be removed and manually fed back into the network stack with the encapsulation removed to be sent to the real destination. This option is mandatory.

PeerPort=
Specifies the peer port number. Defaults to unset. Note that when peer port is set "Peer=" address is mandatory.

Protocol=
The *Protocol=* specifies the protocol number of the packets arriving at the UDP port. When *Encapsulation=FooOverUDP*, this field is mandatory and is not set by default. Takes an IP protocol name such as "gre" or "ipip", or an integer within the range 1...255. When *Encapsulation=GenericUDPEncapsulation*, this must not be specified.

Peer=
Configures peer IP address. Note that when peer address is set "PeerPort=" is mandatory.

Local=
Configures local IP address.

[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 setting is compulsory.

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

[VXCAN] SECTION OPTIONS

The [VXCAN] section only applies for netdevs of kind "vxcan" and accepts the following key:

Peer=
The peer interface name used when creating the netdev. This setting is compulsory.

[TUN] SECTION OPTIONS

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

MultiQueue=
Takes a boolean. Configures whether to use multiple file descriptors (queues) to parallelize packets

sending and receiving. Defaults to "no".

PacketInfo=
Takes a boolean. 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. Configures IFF_VNET_HDR flag for a tun or 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.

[WIREGUARD] SECTION OPTIONS

The [WireGuard] section accepts the following keys:

PrivateKey=
The Base64 encoded private key for the interface. It can be generated using the **wg genkey** command (see **wg**(8)). This option or *PrivateKeyFile=* is mandatory to use WireGuard. Note that because this information is secret, you may want to set the permissions of the .netdev file to be owned by "root:systemd−network" with a "0640" file mode.

PrivateKeyFile=
Takes an absolute path to a file which contains the Base64 encoded private key for the interface. When this option is specified, then *PrivateKey=* is ignored. Note that the file must be readable by the user "systemd−network", so it should be, e.g., owned by "root:systemd−network" with a "0640" file mode. If the path refers to an **AF_UNIX** stream socket in the file system a connection is made to it and the key read from it.

ListenPort=
Sets UDP port for listening. Takes either value between 1 and 65535 or "auto". If "auto" is specified, the port is automatically generated based on interface name. Defaults to "auto".

FirewallMark=
Sets a firewall mark on outgoing WireGuard packets from this interface. Takes a number between 1 and 4294967295.

[WIREGUARDPEER] SECTION OPTIONS

The [WireGuardPeer] section accepts the following keys:

PublicKey=
Sets a Base64 encoded public key calculated by **wg pubkey** (see **wg**(8)) from a private key, and usually transmitted out of band to the author of the configuration file. This option is mandatory for this section.

PresharedKey=
Optional preshared key for the interface. It can be generated by the **wg genpsk** command. This option adds an additional layer of symmetric−key cryptography to be mixed into the already existing public−key cryptography, for post−quantum resistance. Note that because this information is secret, you may want to set the permissions of the .netdev file to be owned by "root:systemd−network" with a "0640" file mode.

PresharedKeyFile=
Takes an absolute path to a file which contains the Base64 encoded preshared key for the peer. When this option is specified, then *PresharedKey=* is ignored. Note that the file must be readable by the user

"systemd−network", so it should be, e.g., owned by "root:systemd−network" with a "0640" file mode. If the path refers to an **AF_UNIX** stream socket in the file system a connection is made to it and the key read from it.

AllowedIPs=

Sets a comma−separated list of IP (v4 or v6) addresses with CIDR masks from which this peer is allowed to send incoming traffic and to which outgoing traffic for this peer is directed.

The catch−all 0.0.0.0/0 may be specified for matching all IPv4 addresses, and ::/0 may be specified for matching all IPv6 addresses.

Note that this only affects *routing inside the network interface itself*, i.e. the packets that pass through the tunnel itself. To cause packets to be sent via the tunnel in the first place, an appropriate route needs to be added as well — either in the "[Routes]" section on the ".network" matching the wireguard interface, or externally to systemd−networkd.

Endpoint=

Sets an endpoint IP address or hostname, followed by a colon, and then a port number. This endpoint will be updated automatically once to the most recent source IP address and port of correctly authenticated packets from the peer at configuration time.

PersistentKeepalive=

Sets a seconds interval, between 1 and 65535 inclusive, of how often to send an authenticated empty packet to the peer for the purpose of keeping a stateful firewall or NAT mapping valid persistently. For example, if the interface very rarely sends traffic, but it might at anytime receive traffic from a peer, and it is behind NAT, the interface might benefit from having a persistent keepalive interval of 25 seconds. If set to 0 or "off", this option is disabled. By default or when unspecified, this option is off. Most users will not need this.

[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".

AdActorSystemPriority=
: Specifies the 802.3ad actor system priority. Takes a number in the range 1...65535.

AdUserPortKey=
: Specifies the 802.3ad user defined portion of the port key. Takes a number in the range 0...1023.

AdActorSystem=
: Specifies the 802.3ad system MAC address. This cannot be a null or multicast address.

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. A value of 0 disables ARP monitoring. The default value is 0, and the default unit seconds.

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=
: Takes 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).

DynamicTransmitLoadBalancing=
: Takes a boolean. Specifies if dynamic shuffling of flows is enabled. Applies only for balance−tlb mode. Defaults to unset.

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]

[XFRM] SECTION OPTIONS

The [Xfrm] section accepts the following keys:

InterfaceId=
: Sets the ID/key of the xfrm interface which needs to be associated with a SA/policy. Can be decimal or hexadecimal, valid range is 1−0xffffffff. This is mandatory.

Independent=
: Takes a boolean. If false (the default), the xfrm interface must have an underlying device which can be used for hardware offloading.

For more detail information see **Virtual XFRM Interfaces**[17].

[VRF] SECTION OPTIONS

The [VRF] section only applies for netdevs of kind "vrf" and accepts the following key:

Table=
: The numeric routing table identifier. This setting is compulsory.

[BATMANADVANCED] SECTION OPTIONS

The [BatmanAdvanced] section only applies for netdevs of kind "batadv" and accepts the following keys:

GatewayMode=
: Takes one of "off", "server", or "client". A batman−adv node can either run in server mode (sharing its internet connection with the mesh) or in client mode (searching for the most suitable internet connection in the mesh) or having the gateway support turned off entirely (which is the default setting).

Aggregation=
: Takes a boolean value. Enables or disables aggregation of originator messages. Defaults to true.

BridgeLoopAvoidance=
: Takes a boolean value. Enables or disables avoidance of loops on bridges. Defaults to true.

DistributedArpTable=
: Takes a boolean value. Enables or disables the distributed ARP table. Defaults to true.

Fragmentation=
: Takes a boolean value. Enables or disables fragmentation. Defaults to true.

HopPenalty=
: The hop penalty setting allows to modify **batctl**(8) preference for multihop routes vs. short routes. This integer value is applied to the TQ (Transmit Quality) of each forwarded OGM (Originator

Message), thereby propagating the cost of an extra hop (the packet has to be received and retransmitted which costs airtime). A higher hop penalty will make it more unlikely that other nodes will choose this node as intermediate hop towards any given destination. The default hop penalty of '15' is a reasonable value for most setups and probably does not need to be changed. However, mobile nodes could choose a value of 255 (maximum value) to avoid being chosen as a router by other nodes. The minimum value is 0.

OriginatorIntervalSec=
: The value specifies the interval in seconds, unless another time unit is specified in which batman−adv floods the network with its protocol information. See **systemd.time**(7) for more information.

GatewayBandwidthDown=
: If the node is a server, this parameter is used to inform other nodes in the network about this node's internet connection download bandwidth in bits per second. Just enter any number suffixed with K, M, G or T (base 1000) and the batman−adv module will propagate the entered value in the mesh.

GatewayBandwidthUp=
: If the node is a server, this parameter is used to inform other nodes in the network about this node's internet connection upload bandwidth in bits per second. Just enter any number suffixed with K, M, G or T (base 1000) and the batman−adv module will propagate the entered value in the mesh.

RoutingAlgorithm=
: This can be either "batman−v" or "batman−iv" and describes which routing_algo of **batctl**(8) to use. The algorithm cannot be changed after interface creation. Defaults to "batman−v".

EXAMPLES

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/1−fou−tunnel.netdev

```
[NetDev]
```

```
Name=fou−tun
Kind=fou

[FooOverUDP]
Port=5555
Protocol=4
```

Example 5. /etc/systemd/network/25−fou−ipip.netdev

```
[NetDev]
Name=ipip−tun
Kind=ipip

[Tunnel]
Independent=yes
Local=10.65.208.212
Remote=10.65.208.211
FooOverUDP=yes
FOUDestinationPort=5555
```

Example 6. /etc/systemd/network/25−tap.netdev

```
[NetDev]
Name=tap−test
Kind=tap

[Tap]
MultiQueue=yes
PacketInfo=yes
```

Example 7. /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 8. /etc/systemd/network/25−6rd.netdev

```
[NetDev]
Name=6rd−tun
Kind=sit
MTUBytes=1480

[Tunnel]
Local=10.65.223.238
IPv6RapidDeploymentPrefix=2602::/24
```

Example 9. /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 10. /etc/systemd/network/25−ip6gre.netdev

```
[NetDev]
Name=ip6gre−tun
Kind=ip6gre

[Tunnel]
Key=123
```

Example 11. /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 12. /etc/systemd/network/25−veth.netdev

```
[NetDev]
Name=veth−test
Kind=veth

[Peer]
Name=veth−peer
```

Example 13. /etc/systemd/network/25−bond.netdev

```
[NetDev]
Name=bond1
Kind=bond

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

Example 14. /etc/systemd/network/25−dummy.netdev

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

Example 15. /etc/systemd/network/25−vrf.netdev

Create a VRF interface with table 42.

```
[NetDev]
Name=vrf−test
Kind=vrf

[VRF]
Table=42
```

Example 16. /etc/systemd/network/25−macvtap.netdev

Create a MacVTap device.

```
[NetDev]
Name=macvtap−test
Kind=macvtap
```

Example 17. /etc/systemd/network/25−wireguard.netdev

```
[NetDev]
Name=wg0
Kind=wireguard

[WireGuard]
PrivateKey=EEGlnEPYJV//kbvvIqxKkQwOiS+UENyPncC4bF46ong=
ListenPort=51820

[WireGuardPeer]
PublicKey=RDf+LSpeEre7YEIKaxg+wbpsNV7du+ktR99uBEtIiCA=
AllowedIPs=fd31:bf08:57cb::/48,192.168.26.0/24
Endpoint=wireguard.example.com:51820
```

Example 18. /etc/systemd/network/27−xfrm.netdev

```
[NetDev]
Name=xfrm0
Kind=xfrm

[Xfrm]
Independent=yes
```

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. B.A.T.M.A.N. Advanced
   https://www.open-mesh.org/projects/open-mesh/wiki
6. (DVOE)
   https://en.wikipedia.org/wiki/Distributed_Overlay_Virtual_Ethernet

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

8. Generic Protocol Extension for VXLAN
https://tools.ietf.org/html/draft-ietf-nvo3-vxlan-gpe-07

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

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

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

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

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

14. Foo over UDP
https://lwn.net/Articles/614348

15. IPv6 Rapid Deployment
https://tools.ietf.org/html/rfc5569

16. Generic UDP Encapsulation
https://lwn.net/Articles/615044

17. Virtual XFRM Interfaces
https://lwn.net/Articles/757391