Article

Routing your VPN network traffic

Configure your VPN to include and exclude some network traffic.

Overview

You can control some of the network traffic that the system routes to and from a personal VPN, packet tunnel provider, or app proxy provider. The Network Extension framework provides routing settings for all VPN types, and some settings specific to personal VPNs and packet tunnel providers. You can configure the routing of packets to per-app VPNs using rules, and make exceptions to always-on VPNs on MDM supervised devices.

For the complete APIs to create the different types of VPNs, see Personal VPN, Packet tunnel provider, and App proxy provider.

Configure traffic for a packet tunnel provider

You configure a packet tunnel provider's virtual interface by passing an `NEPacketTunnelNetworkSettings` instance to the `setTunnelNetworkSettings(_:completionHandler:)` method of an `NETunnelProvider` instance. The system uses the routing settings you provide in the `NEPacketTunnelNetworkSettings` instance by default.

To route traffic through the packet tunnel provider, create a homogenous array of `NEIPv4Route` or `NEIPv6Route` elements and assign it to the corresponding settings property, either:

- The `includedRoutes` property of the `ipv4Settings` property
- The `includedRoutes` property of the `ipv6Settings` property

The `NEIPv4Route` and `NEIPv6Route` objects that you pass in these arrays specify the types of traffic that the VPN includes. For example, the following code specifies the traffic for internet protocol version 4:

```
// Create an internet protocol settings object.
let ipv4Settings = NEIPv4Settings(addresses: [ "192.168.3.4" ],
                                  subnetMasks: [ "255.255.0.0" ])

// Specify the types of traffic to include and exclude.
ipv4Settings.includedRoutes = [ NEIPv4Route(destinationAddress: "192.168.0.0",
                                            subnetMask: "255.255.0.0") ]
```

If you include the default route (`0.0.0.0/0` or `::/0`) in the `includedRoutes` property, the system routes network traffic that doesn't match some other more specific rule in the system routing table through the VPN.

To exclude traffic from the VPN, create a homogenous array of `NEIPv4Route` or `NEIPv6Route` elements and assign it to the corresponding settings property, either:

- The `excludedRoutes` property of the `ipv4Settings` property
- The `excludedRoutes` property of the `ipv6Settings` property

The route instances specify the types of traffic that the VPN excludes.

```
// Specify the types of traffic to exclude.
ipv4Settings.excludedRoutes = [ NEIPv4Route(destinationAddress: "192.168.5.6",
                                            subnetMask: "255.255.255.255")]
```

Then set the packet tunnel provider's settings to an `NEPacketTunnelNetworkSettings` instance that contains the internet protocol settings using the `setTunnelNetworkSettings(_:completionHandler:)` method.

```
// Create a network settings object with the internet protocol settings.
let networkSettings = NEPacketTunnelNetworkSettings(tunnelRemoteAddress: "127.0.0.1"
networkSettings.ipv4Settings = ipv4Settings

// Set the packet tunnel provider's settings.
setTunnelNetworkSettings(networkSettings) { error in
    completionHandler(error)
}
```

Note that the system routing table supersedes the `includedRoutes` and `excludedRoutes` properties. For example, if a table routes traffic to hosts on the local network, those routes supersede these properties.

Also, if an app creates a network connection that routes traffic over a specific network interface, called *scoping*, then it supersedes the system routing table.

Route additional traffic through a personal VPN or packet tunnel provider

To route network traffic that's not related to designated system services necessary for maintaining expected device functionality through a personal VPN or packet tunnel provider, set the `includeAllNetworks` property to `true`.

```
// Create the tunnel provider configuration.
let protocolConfiguration = NETunnelProviderProtocol()
protocolConfiguration.serverAddress = "https://127.0.0.1"

// Include network traffic.
protocolConfiguration.includeAllNetworks = true
```

Then the system scopes network connections, with certain exceptions, to the VPN tunnel.

Also, after the VPN transitions from the disconnected (`NEVPNStatus.disconnected`) to the connecting (`NEVPNStatus.connecting`) state, the system doesn't close TCP connections, but drops subsequent packets that previously established network connections send or receive.

When the VPN transitions away from the connected state, the system drops network traffic. For example, the system drops the network traffic when a device transitions from a Wi-Fi network to a cellular network, and while the VPN is in the process of reconnecting to the VPN server over the cellular network.

The system always excludes the following traffic from the VPN, and the `includeAllNetworks` property has no impact on it:

- Network control plane traffic that maintains a device's connection to the local network, such as DHCP traffic, or that communicates directly with the VPN server.
- Captive portal negotiation network traffic that authorizes a device with a Wi-Fi hotspot. For example, a coffee shop Wi-Fi hotspot may require that the user accept legal terms and conditions before the hotspot grants the device access to the internet.
- Certain cellular services traffic that uses the cellular network only, such as VoLTE.
- Traffic that communicates with a companion device, such as an Apple Watch.

Exclude some traffic from a personal VPN or packet tunnel provider

You can make other exceptions when routing network traffic through a VPN using other `NEVPNProtocol` properties. If you set the `includeAllNetworks` property to `true`, you can exclude specific types of traffic from the VPN.

- To exclude network connections to hosts on the local network — such as AirPlay, AirDrop, and CarPlay — set the `excludeLocalNetworks` property to `true`.
- To exclude cellular services network traffic — such as Wi-Fi Calling, MMS, SMS, and Visual Voicemail — set the `excludeCellularServices` property to `true`. This property doesn't impact services that use the cellular network only — such as VoLTE — which the system automatically excludes.
- To exclude Apple Push Notification services (APNs) traffic, set the `excludeAPNs` property to `true`. Many system and app features use APNs, which may not work reliably when you route APNs through the VPN, resulting in missed or delayed push notifications and iMessages.

```
// Include network traffic.
protocolConfiguration.includeAllNetworks = true

// Except for local network, APNs, and cellular traffic.
protocolConfiguration.excludeLocalNetworks = true
protocolConfiguration.excludeAPNs = true
protocolConfiguration.excludeCellularServices = true
```

Enforce the inclusions and exclusions for a packet tunnel provider

To enforce the `includedRoutes` and `excludedRoutes` properties, set the `enforceRoutes` property to `true`.

```
protocolConfiguration.enforceRoutes = true
```

The system scopes the included routes to the VPN and the excluded routes to the current primary network interface, such as a Wi-Fi or a cellular network. This property supersedes the system routing table and scoping operations by apps. It's also mutually exclusive from the `includeAllNetworks` property. If you set `includeAllNetworks` to `true`, the system ignores the `enforceRoutes` property.

If you set both the `enforceRoutes` and `excludeLocalNetworks` properties to `true`, the system excludes network connections to hosts on the local network from the enforce routes behavior.

```
protocolConfiguration.excludeLocalNetworks = true
protocolConfiguration.enforceRoutes = true
```

For example, if the `includedRoutes` property contains the `10.0.0.0/8` address and the local network subnet is `10.10.0.0/16`, the system scopes network connections to hosts in the `10.0.0.0/8` network to the VPN, but not those in the `10.10.0.0/16` network.

Route network traffic to and from specific apps

You can use two types of per-app VPN features in the Network Extension framework to control network traffic to and from specific apps. You can use a packet tunnel provider to scope network connections from an app to the VPN, or use an app proxy to divert network connections from an app to a transparent proxy. To programmatically configure a per-app VPN, use the `NEPacketTunnelProvider` class for packet tunnels, and the `NEAppProxyProviderManager` class for app proxies.

For packet tunnel providers, use the per-app VPN rules to override any scoping operations that apps perform.

- For macOS apps, use the `appRules` property to associate MDM-managed apps with the per-app VPN configuration. Then configure exclusions using the `excludedDomains` property. Set this property to a list of domain names that you want to exclude from the per-app VPN.
- For iOS apps, create rules using the MDM system and access the rules in your code using the `copyAppRules()` method.

The system honors your disconnect on-demand rules. If you enable the on-demand feature (set the `isOnDemandEnabled` property to `true`) and provide a disconnect app rule, the system bypasses the per-app VPN for devices on the network that match the on-demand rule criteria.

Note that the apps generally can't communicate over the network at all unless the VPN is in the connected state.

For more information on per-app VPN features, see the `NETunnelProviderManager` class overview.

Exclude traffic from an always-on VPN

In iOS, you can exclude some network traffic from an always-on VPN device configuration. An always-on VPN acts as the only connection to the internet for an iOS device. You enable this

feature by installing an always-on VPN profile on an MDM-supervised device.

Then the system routes network traffic on the device through the VPN with some exclusions. When the VPN isn't in a connected state, and while the device starts up before the VPN starts, the system drops network traffic. For example, the VPN disconnects when the system transitions from a Wi-Fi network or the user disables the VPN.

The system always excludes network control plane traffic — such as DHCP — but only excludes captive portal negotiation traffic, if you enable the `AllowAllCaptiveNetworkPlugins`, `AllowCaptiveWebSheet`, or `AllowedCaptiveNetworkPlugins` payload keys in the profile.

You can exclude other traffic using the following keys in the always-on VPN profile:

- `ApplicationExceptions` (VPN.AlwaysOn.ApplicationExceptionElement) — specifies apps to exclude from an always-on VPN.
- `ServiceExceptions` (VPN.AlwaysOn.ServiceExceptionElement) — specifies services to exclude from an always-on VPN. Possible values are `VoiceMail`, `AirPrint`, and `CellularServices`.

For more information on MDM and configuring always-on VPNs, see Configuring Multiple Devices Using Profiles and Profile-Specific Payload Keys.

See Also

Virtual private networks

- Personal VPN

  Create and manage a VPN configuration that uses one of the built-in VPN protocols (IPsec or IKEv2).

- Packet tunnel provider

  Implement a VPN client for a packet-oriented, custom VPN protocol.

- App proxy provider

  Implement a VPN client for a flow-oriented, custom VPN protocol.