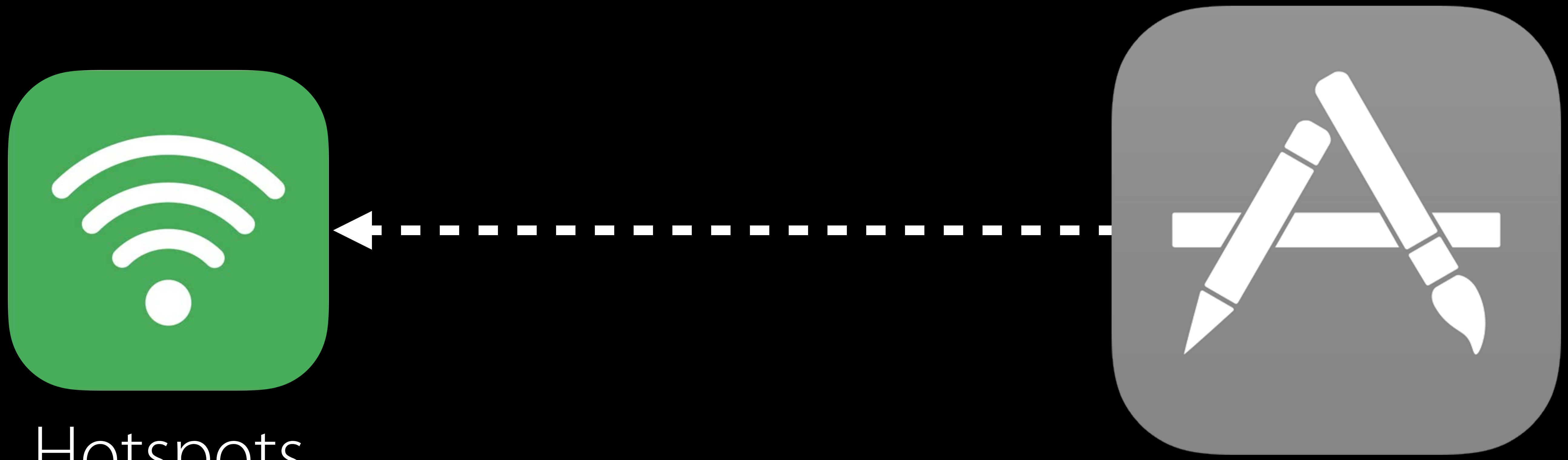
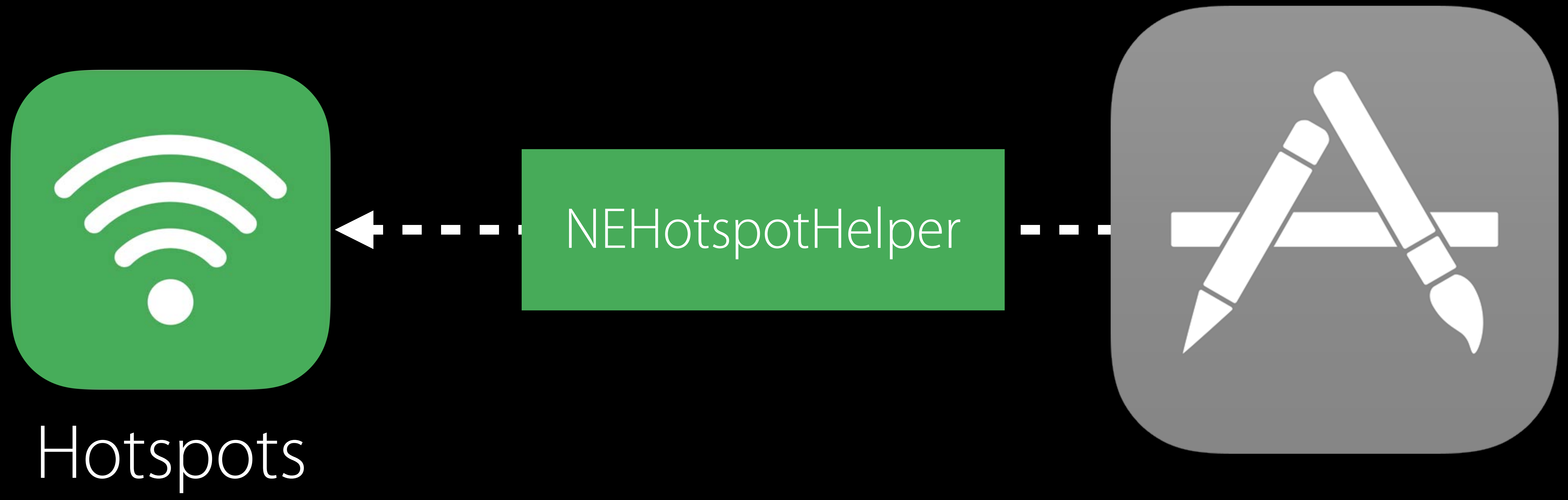
What's New in Network Extension and VPN

Session 717

Jamie Wood Core OS Networking Engineer
Tommy Pauly Core OS Networking Engineer

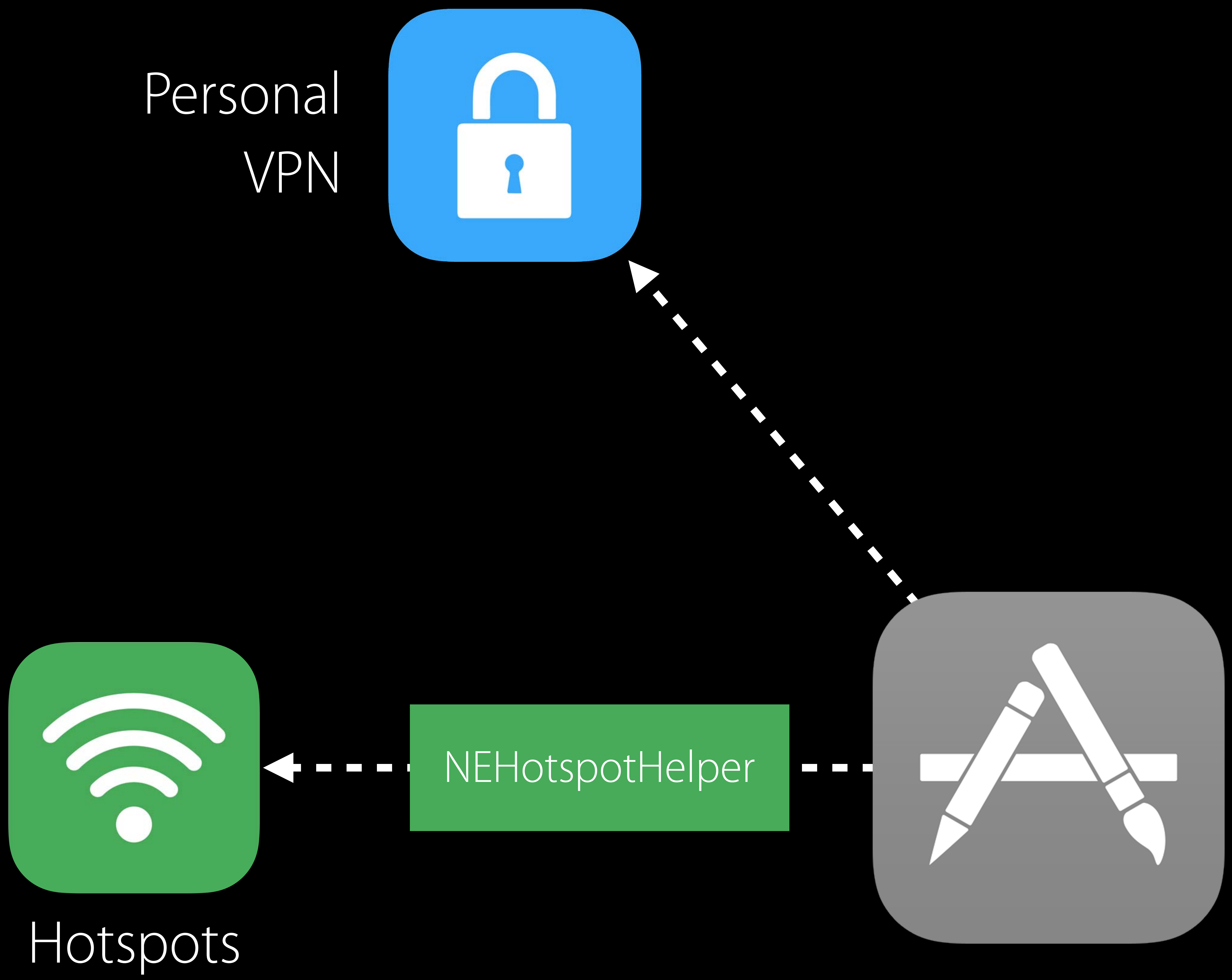


Hotspots



NEHotspotHelper

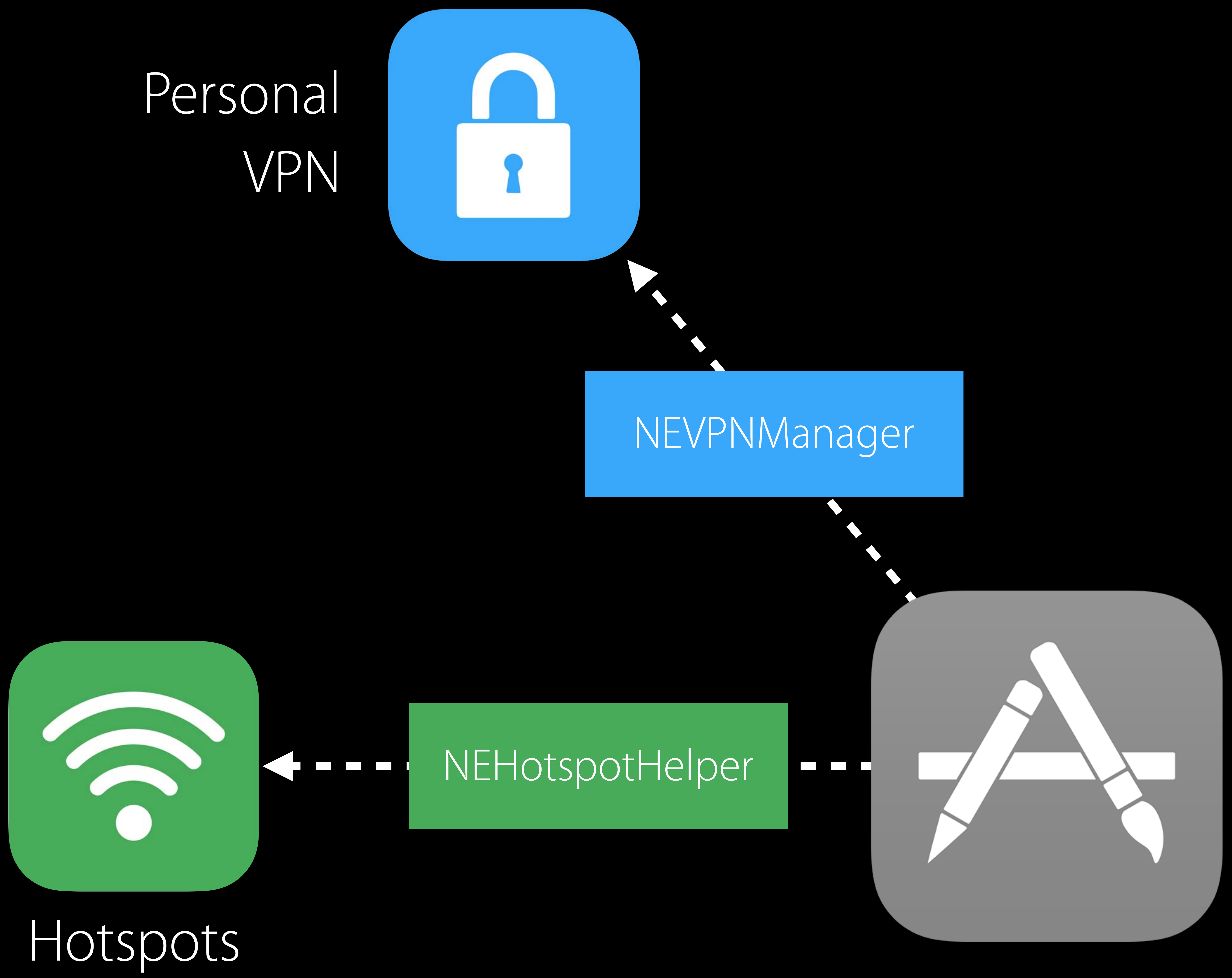
Hotspots



Personal VPN

NEHotspotHelper

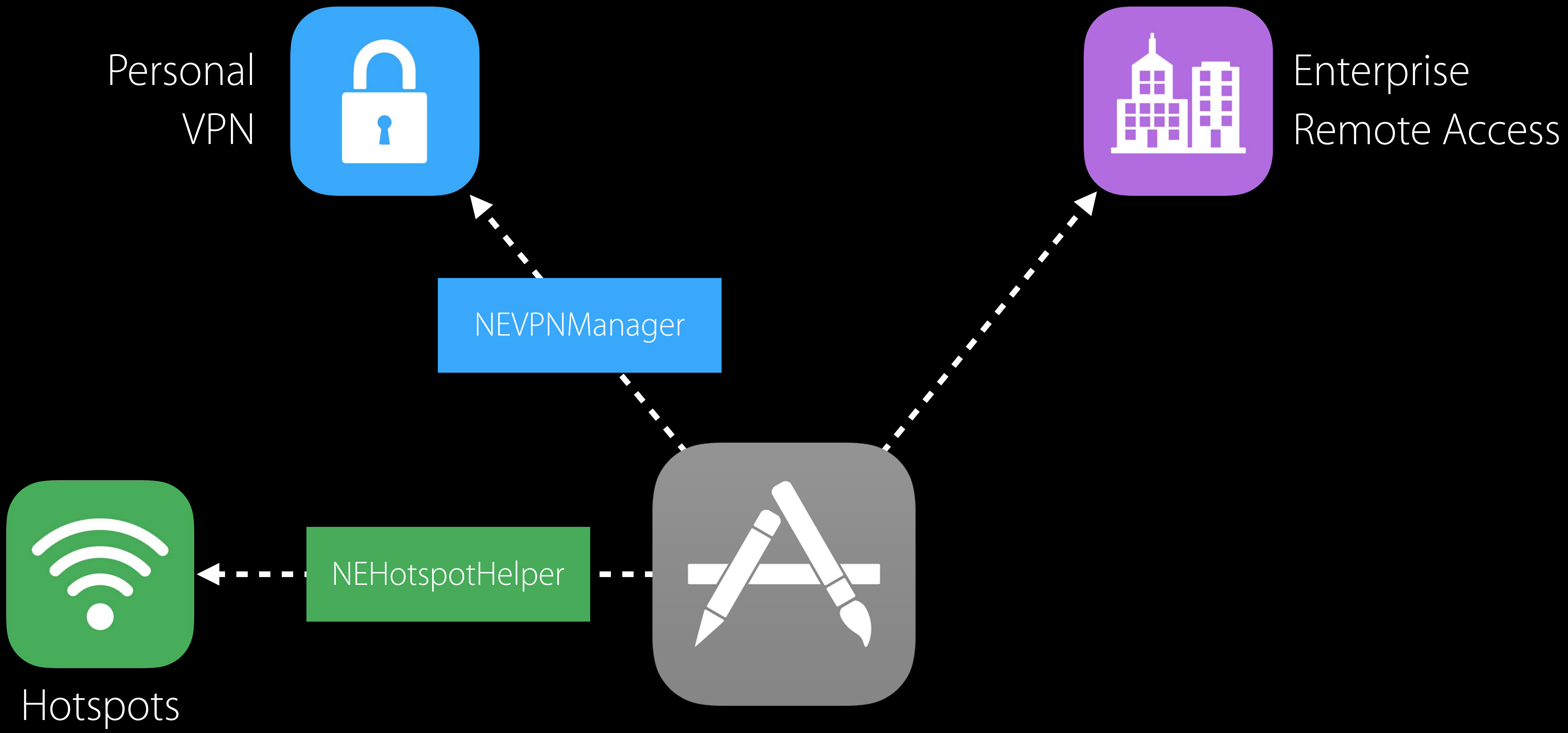
Hotspots


Personal VPN

NEVPNManager

NEHotspotHelper

Hotspots



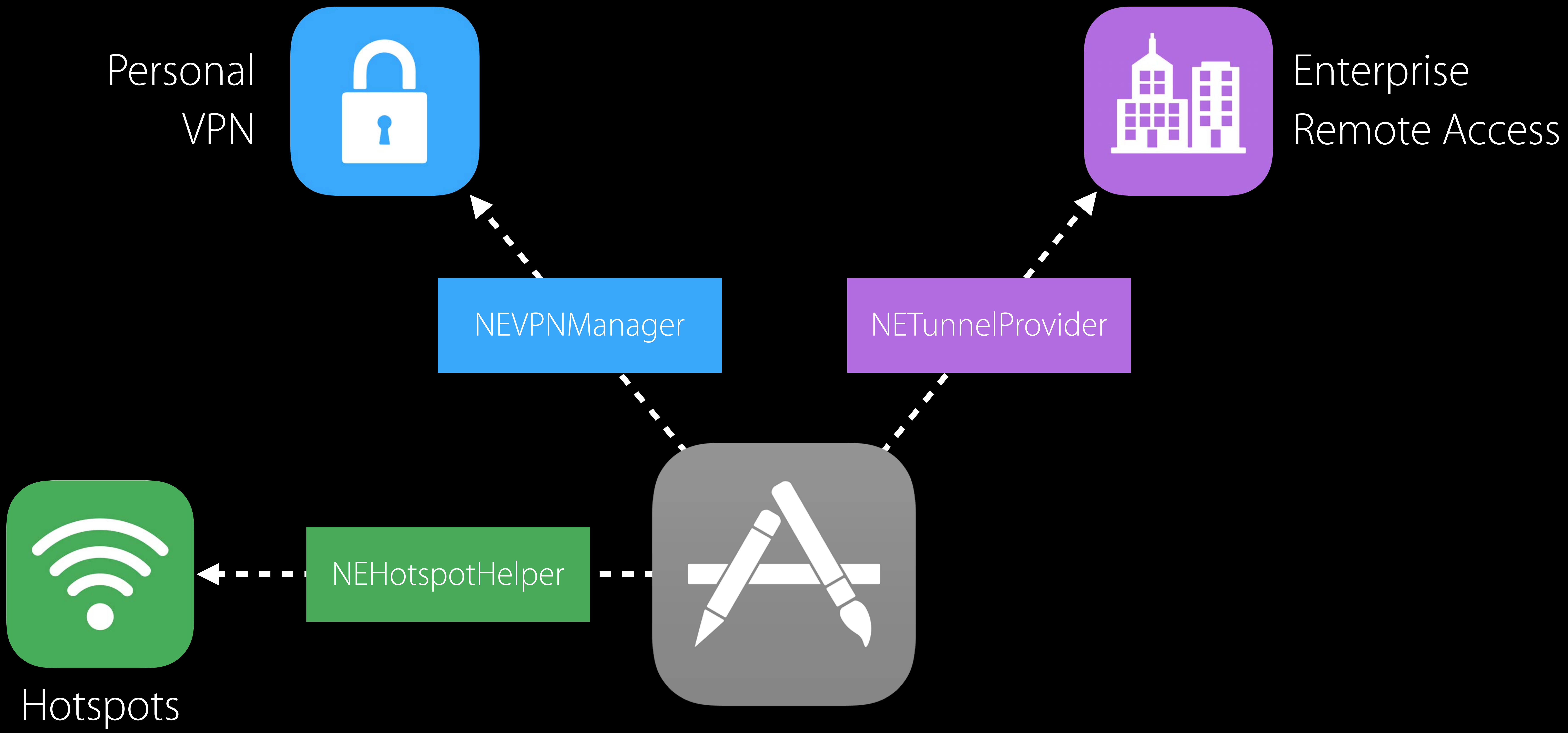
Personal VPN

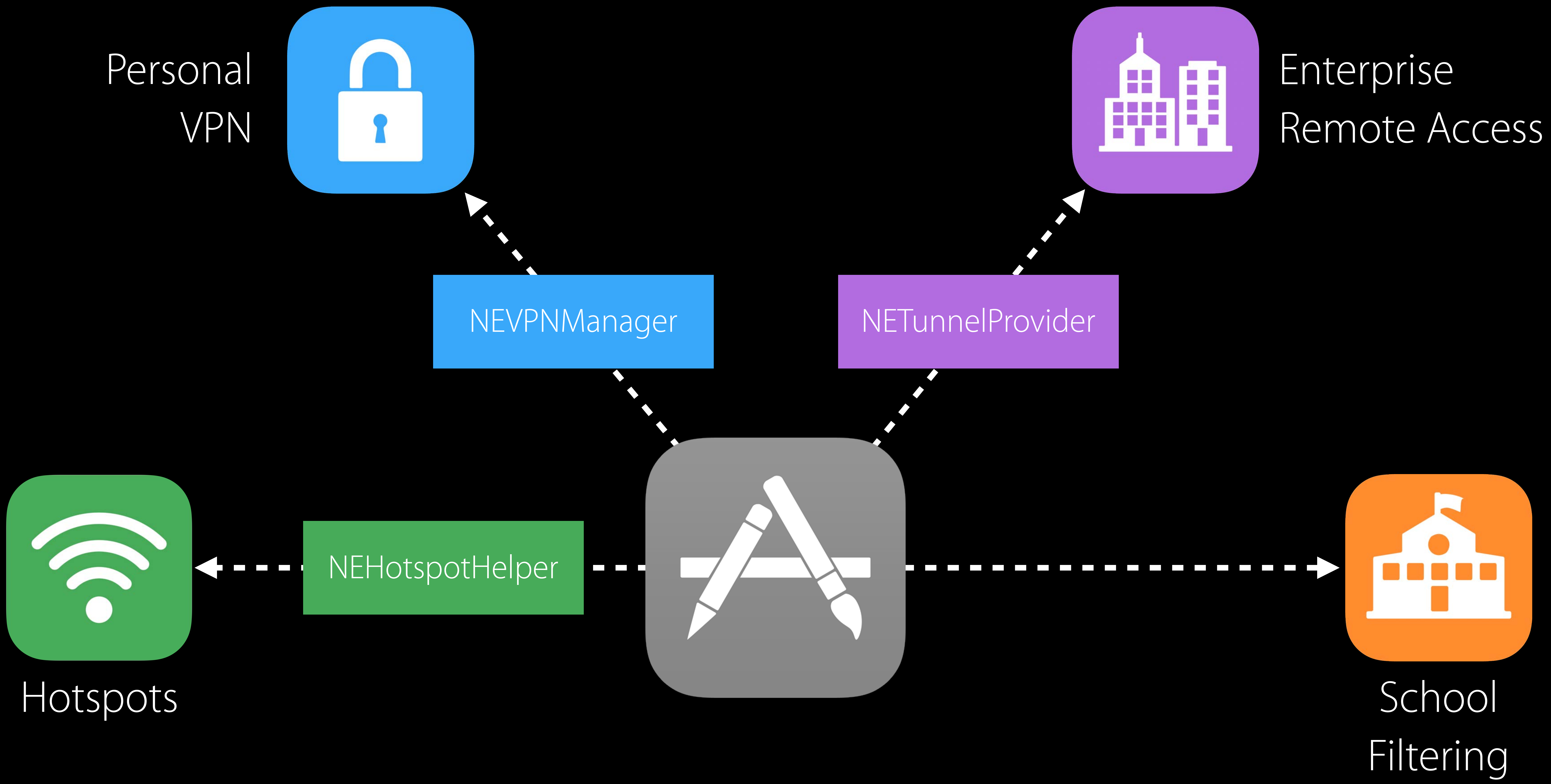
Enterprise Remote Access

NEVPNManager

NEHotspotHelper

Hotspots


Personal VPN
Enterprise Remote Access
NEVPNManager
NETunnelProvider
Hotspots
NEHotspotHelper



Personal VPN

Enterprise Remote Access

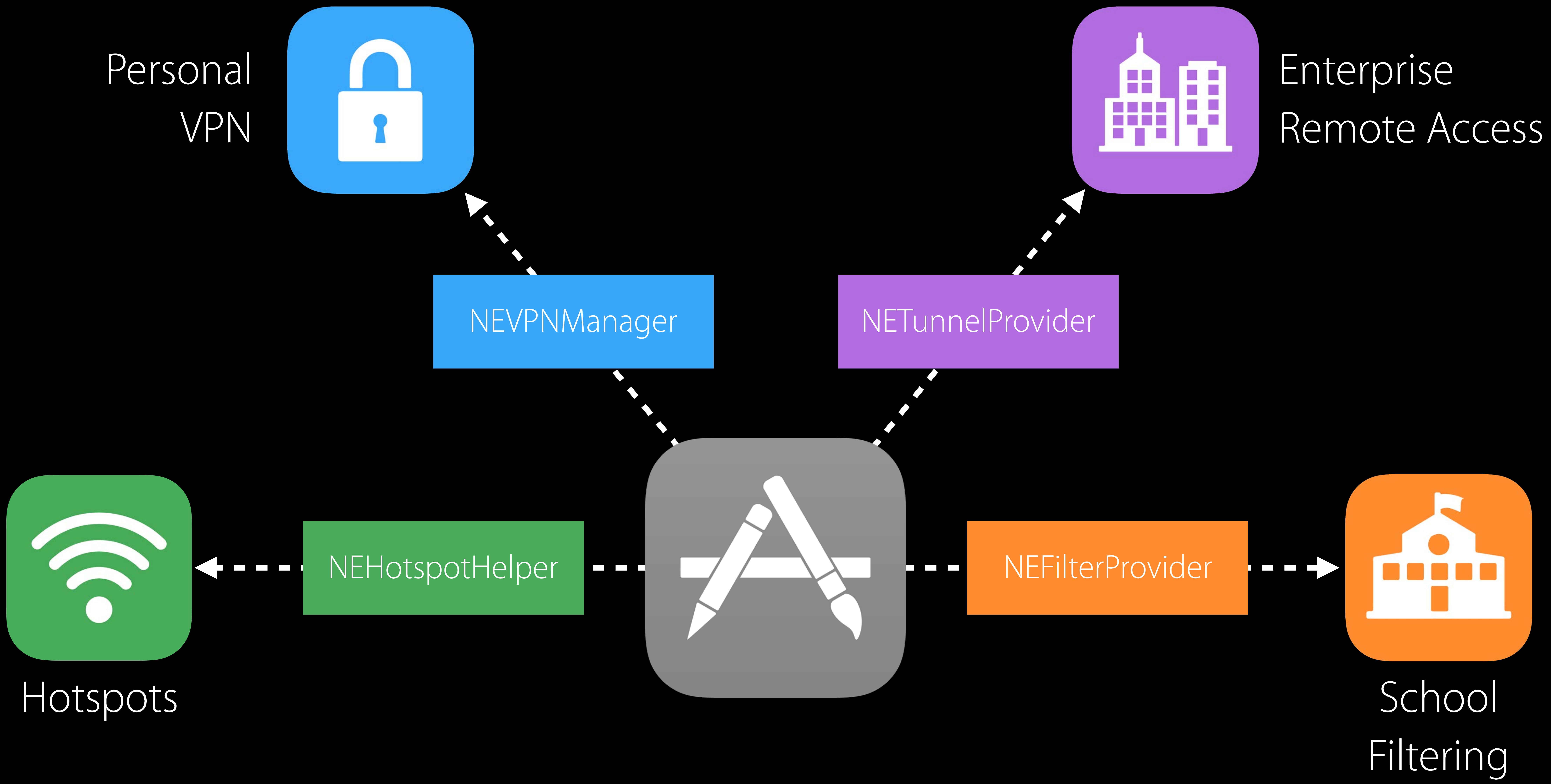
NEVPNManager

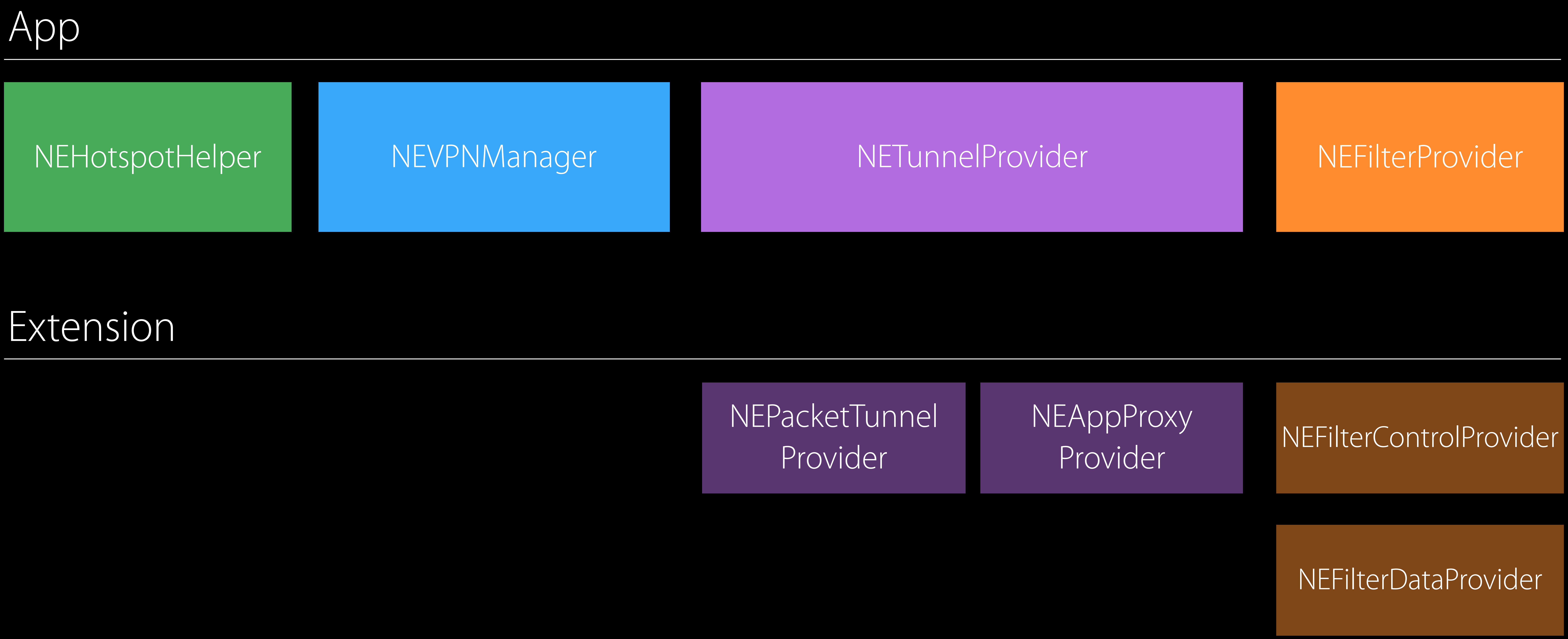
NETunnelProvider

NEHotspotHelper

Hotspots

School Filtering


Personal
VPN
Enterprise
Remote Access
NEVPNManager
NETunnelProvider
Hotspots
NEHotspotHelper
NEFilterProvider
School
Filtering



App

NEHotspotHelper

NEVPNManager

NETunnelProvider

NEFilterProvider

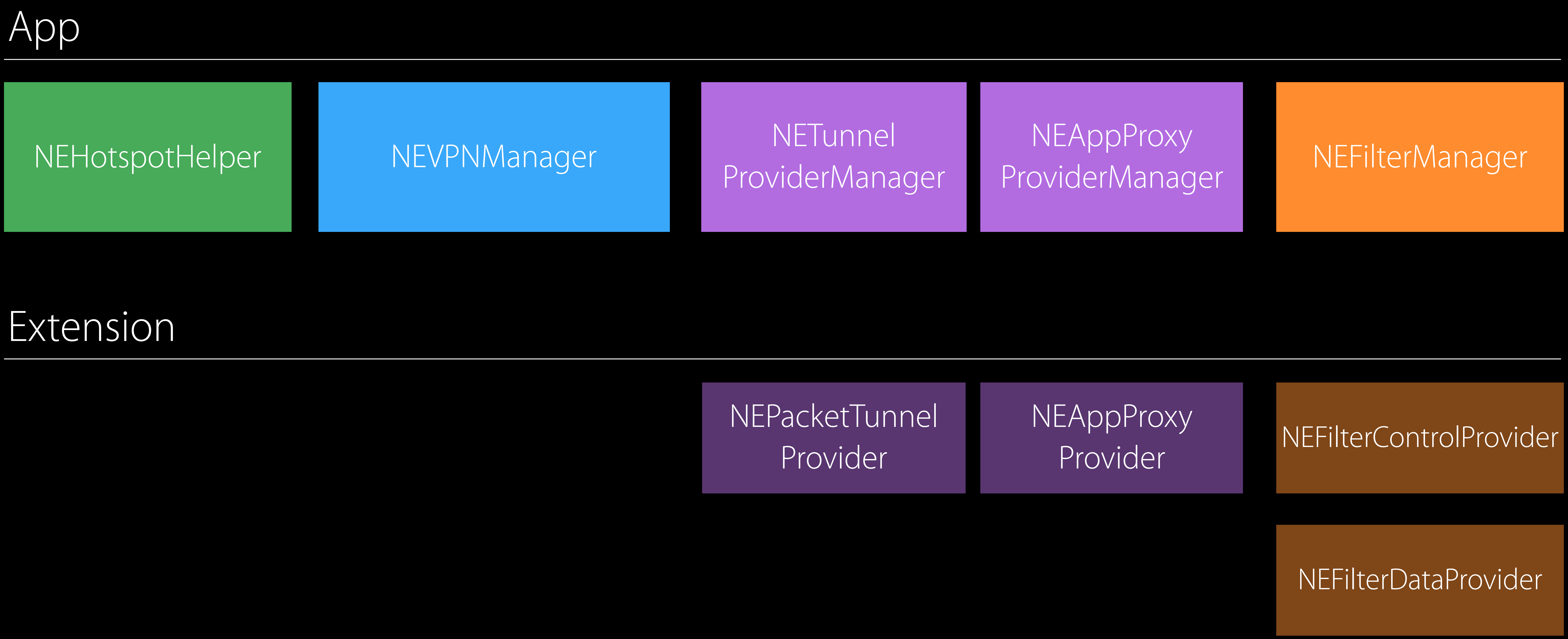
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider

App

- NEHotspotHelper
- NEVPNManager
- NETunnel ProviderManager
- NEAppProxy ProviderManager
- NEFilterManager

Extension

- NEPacketTunnel Provider
- NEAppProxy Provider
- NEFilterControlProvider
- NEFilterDataProvider

Wi-Fi Hotspot Authentication

Previous solutions

CNSetSupportedSSIDs and CNMarkPortalOnline

Wi-Fi Hotspot Authentication

Previous solutions

CNSetSupportedSSIDs and CNMarkPortalOnline

- Hotspot list has a size limitation

Wi-Fi Hotspot Authentication

Previous solutions

CNSetSupportedSSIDs and CNMarkPortalOnline

- Hotspot list has a size limitation
- Users have to manually run your app

Wi-Fi Hotspot Authentication

Previous solutions

CNSetSupportedSSIDs and CNMarkPortalOnline

- Hotspot list has a size limitation
- Users have to manually run your app
- No indication to the user that your app needs to run

NEHotspotHelper

Custom authentication for Wi-Fi Hotspots

Register with the system as a Hotspot Helper

NEHotspotHelper

Custom authentication for Wi-Fi Hotspots

Register with the system as a Hotspot Helper

From the background

- Claim Hotspots with a level of confidence

NEHotspotHelper

Custom authentication for Wi-Fi Hotspots

Register with the system as a Hotspot Helper

From the background

- Claim Hotspots with a level of confidence
- Perform the initial authentication

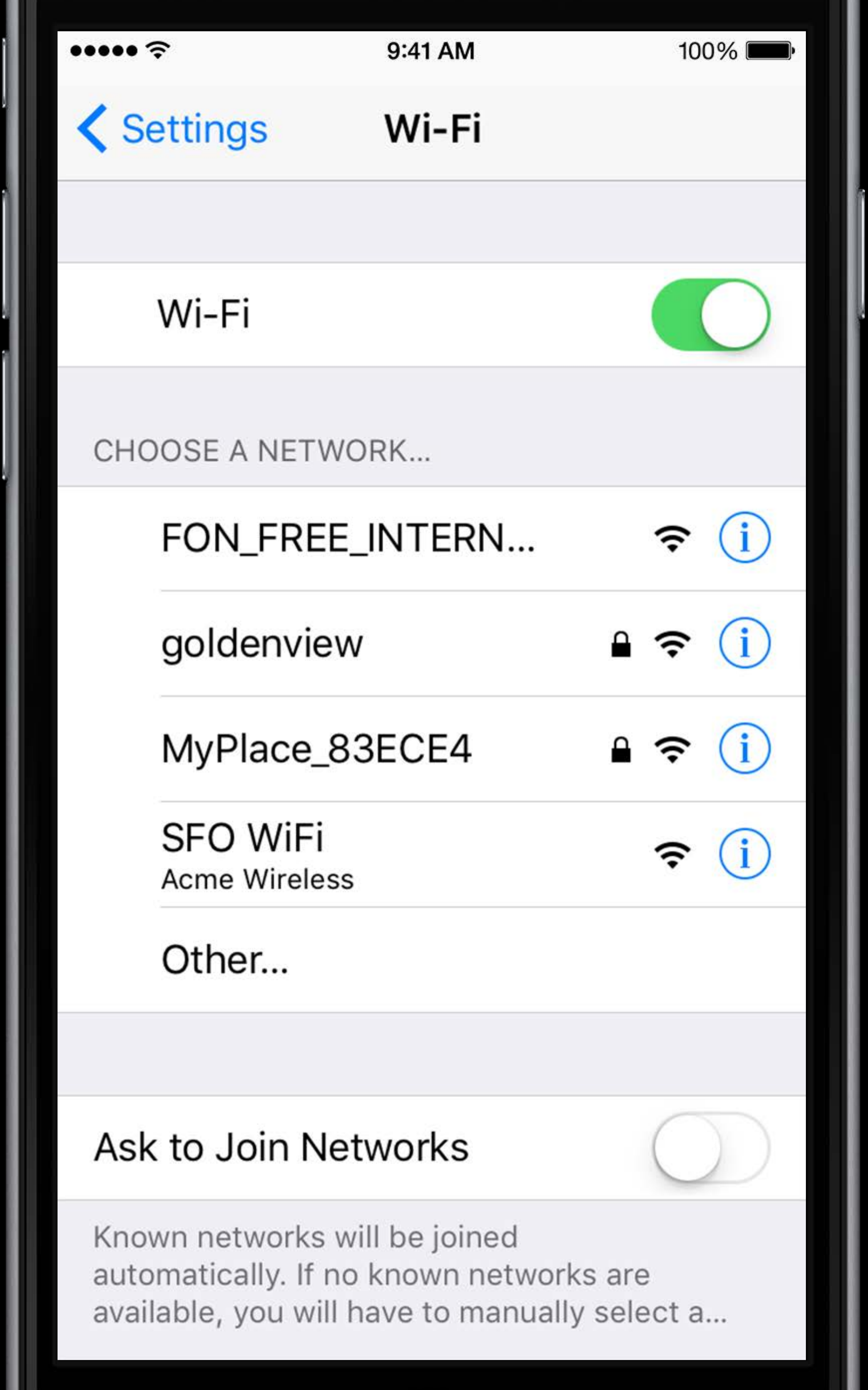
NEHotspotHelper

Custom authentication for Wi-Fi Hotspots

Register with the system as a Hotspot Helper

From the background

- Claim Hotspots with a level of confidence
- Perform the initial authentication
- Maintain the authentication session



Wi-Fi

Wi-Fi

CHOOSE A NETWORK...

FON_FREE_INTERN...

goldenview

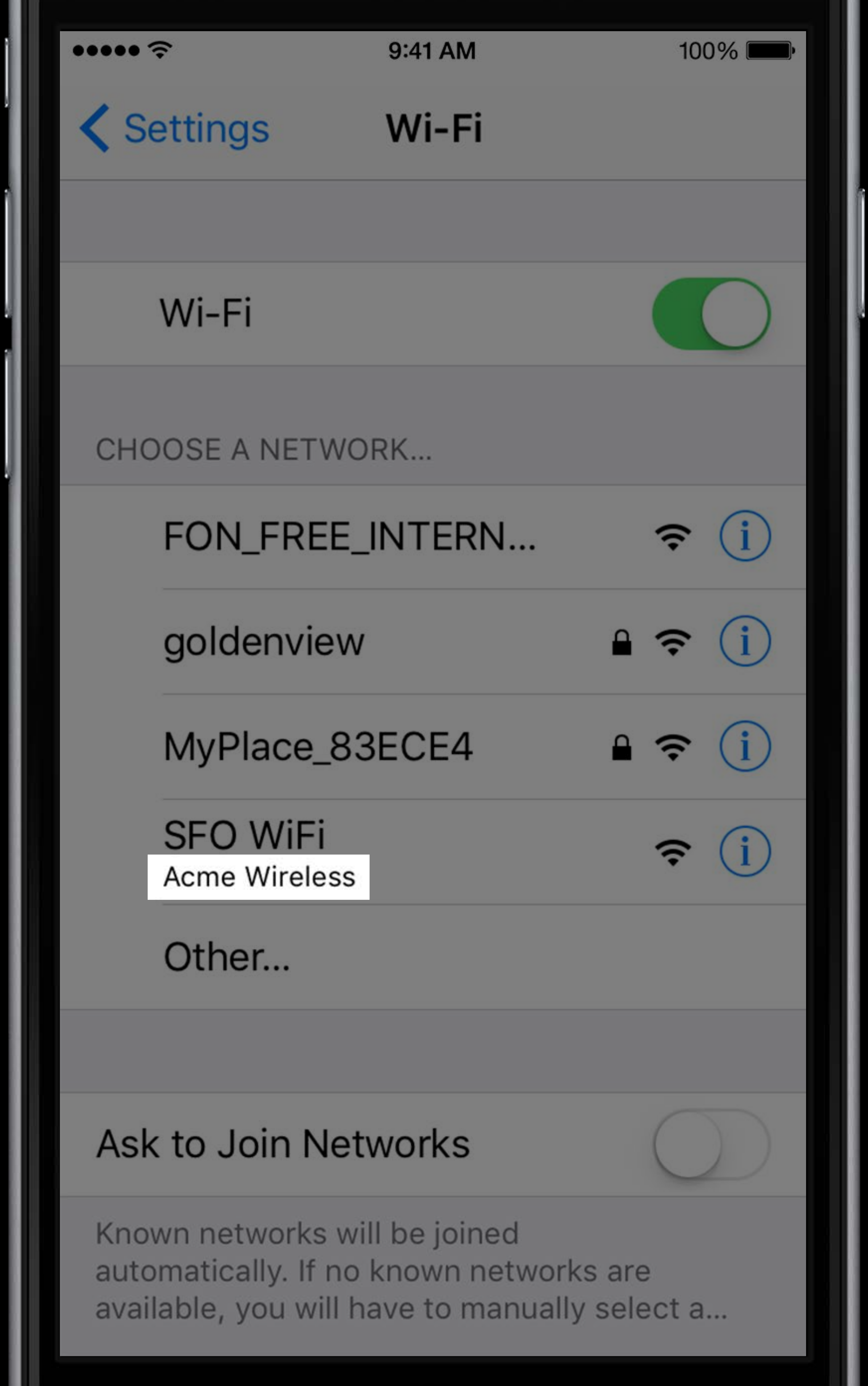
MyPlace_83ECE4

SFO WiFi
Acme Wireless

Other...

Ask to Join Networks

Known networks will be joined automatically. If no known networks are available, you will have to manually select a...



9:41 AM 100%

Settings Wi-Fi

Wi-Fi

CHOOSE A NETWORK...

FON_FREE_INTERN...

goldenview

MyPlace_83ECE4

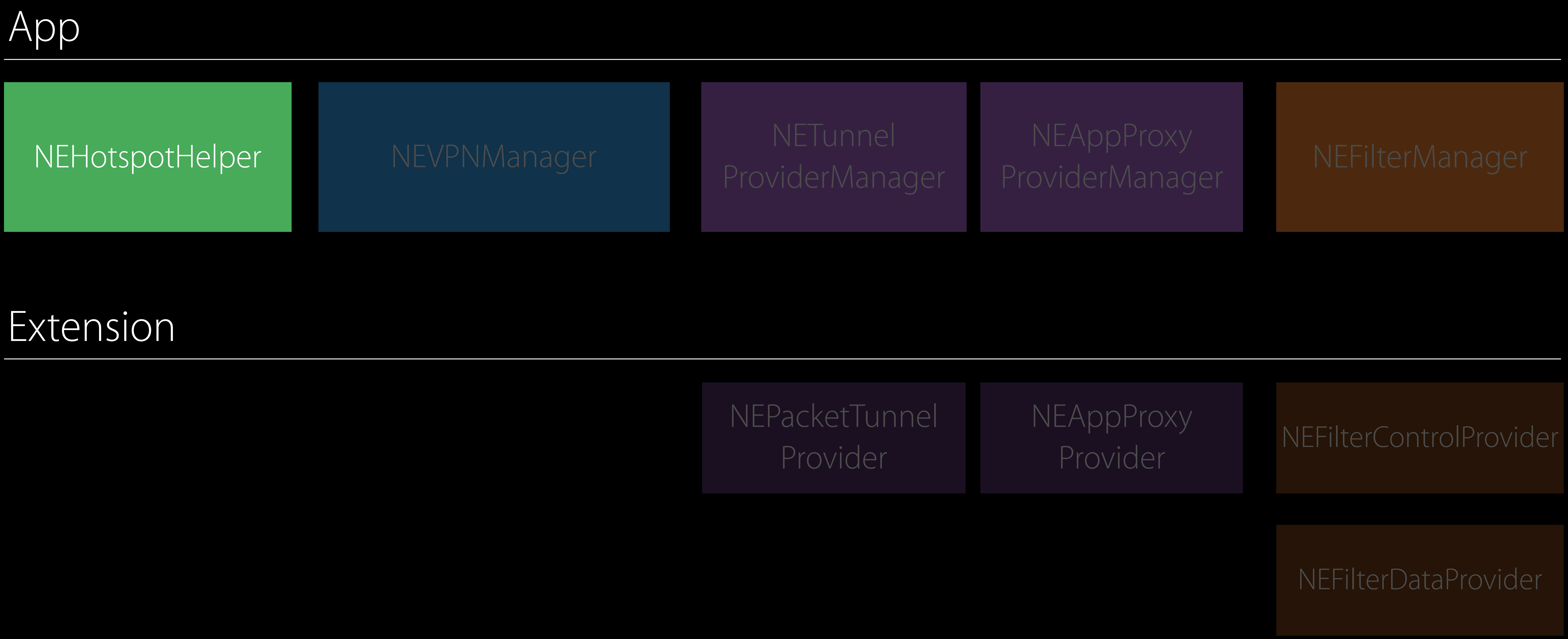
SFO WiFi

Acme Wireless

Other...

Ask to Join Networks

Known networks will be joined automatically. If no known networks are available, you will have to manually select a...



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

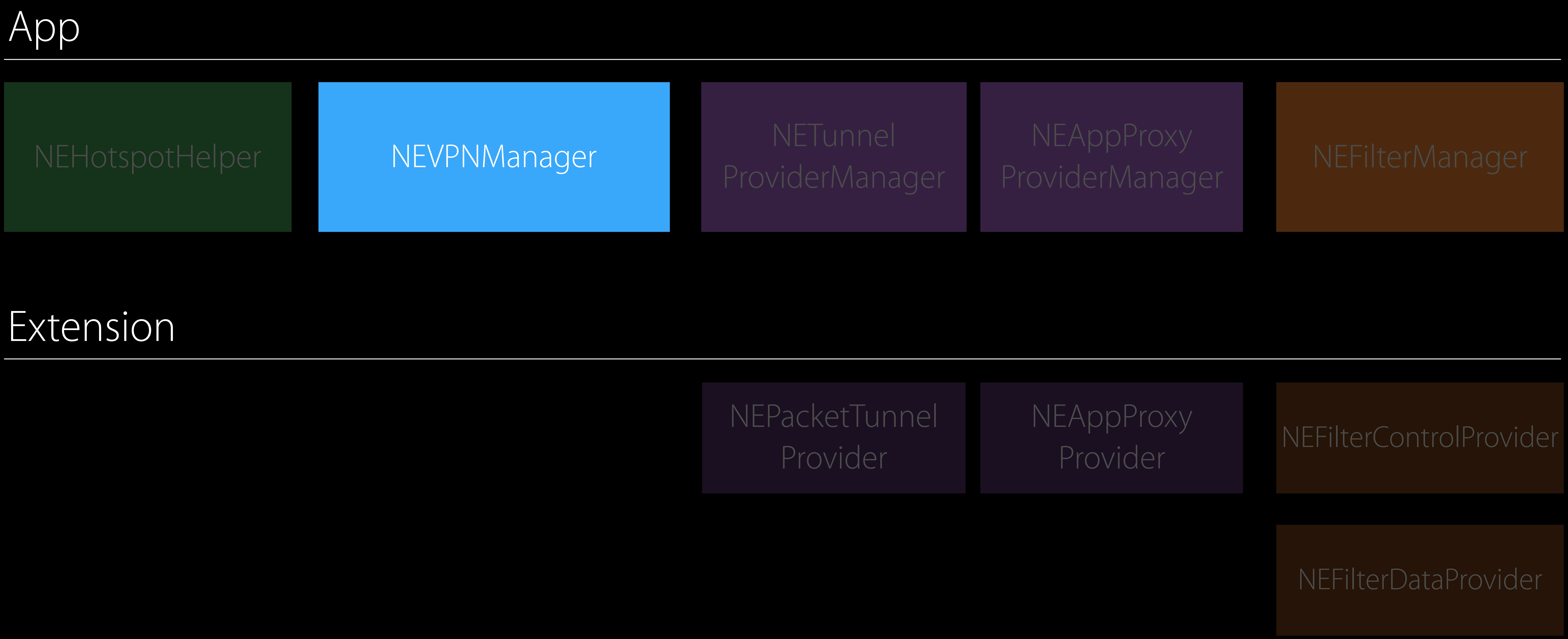
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider

NEVPNManager

Personal VPN

Configure and control the built-in IPsec client

NEVPNManager

Personal VPN

Configure and control the built-in IPSec client

- Create a Personal VPN configuration

NEVPNManager

Personal VPN

Configure and control the built-in IPSec client

- Create a Personal VPN configuration
- Supports IKEv1 and IKEv2

NEVPNManager

Personal VPN

Configure and control the built-in IPSec client

- Create a Personal VPN configuration
- Supports IKEv1 and IKEv2
- Configure Connect On Demand

NEVPNManager

Personal VPN

Configure and control the built-in IPSec client

- Create a Personal VPN configuration
- Supports IKEv1 and IKEv2
- Configure Connect On Demand
- Configure HTTP proxies

NEVPNManager

Personal VPN

Configure and control the built-in IPSec client

- Create a Personal VPN configuration
- Supports IKEv1 and IKEv2
- Configure Connect On Demand
- Configure HTTP proxies
- Cooperates with Enterprise VPN configurations

NEVPNManager

IKEv2 protocol enhancements

NEVPNManager

IKEv2 protocol enhancements

Mobility across interfaces using MOBIKE

NEVPNManager

IKEv2 protocol enhancements

Mobility across interfaces using MOBIKE

Full IPv6 support for inside and outside of tunnel

NEVPNManager

IKEv2 protocol enhancements

Mobility across interfaces using MOBIKE

Full IPv6 support for inside and outside of tunnel

IKE fragmentation support

NEVPNManager

IKEv2 protocol enhancements

Mobility across interfaces using MOBIKE

Full IPv6 support for inside and outside of tunnel

IKE fragmentation support

IKE redirect support

NEVPNManager

Create the configuration

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
```

```
}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in

}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
    // Protocol is a keyword, so it needs to be quoted.
    if manager.`protocol` == nil {
        let newIPSec = NEVPNProtocolIKEv2()
        newIPSec.serverAddress = "my.personal.vpn"
        // Set all of the properties on newIPSec.

        manager.`protocol` = newIPSec
        manager.enabled = true

    }
}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
    // Protocol is a keyword, so it needs to be quoted.
    if manager.`protocol` == nil {
        let newIPSec = NEVPNProtocolIKEv2()
        newIPSec.serverAddress = "my.personal.vpn"
        // Set all of the properties on newIPSec.

        manager.`protocol` = newIPSec
        manager.enabled = true
    }
}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
    // Protocol is a keyword, so it needs to be quoted.
    if manager.`protocol` == nil {
        let newIPSec = NEVPNProtocolIKEv2()
        newIPSec.serverAddress = "my.personal.vpn"
        // Set all of the properties on newIPSec.

        manager.`protocol` = newIPSec
        manager.enabled = true
    }
}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
    // Protocol is a keyword, so it needs to be quoted.
    if manager.`protocol` == nil {
        let newIPSec = NEVPNProtocolIKEv2()
        newIPSec.serverAddress = "my.personal.vpn"
        // Set all of the properties on newIPSec.

        manager.`protocol` = newIPSec
        manager.enabled = true
    }
}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
    // Protocol is a keyword, so it needs to be quoted.
    if manager.`protocol` == nil {
        let newIPSec = NEVPNProtocolIKEv2()
        newIPSec.serverAddress = "my.personal.vpn"
        // Set all of the properties on newIPSec.

        manager.`protocol` = newIPSec
        manager.enabled = true
    }
}
```

NEVPNManager

Create the configuration

```
import NetworkExtension

// Each app gets a single configuration.
let manager = NEVPNManager.sharedManager()
manager.loadFromPreferencesWithCompletionHandler { error in
    // Protocol is a keyword, so it needs to be quoted.
    if manager.`protocol` == nil {
        let newIPSec = NEVPNProtocolIKEv2()
        newIPSec.serverAddress = "my.personal.vpn"
        // Set all of the properties on newIPSec.

        manager.`protocol` = newIPSec
        manager.enabled = true

        manager.saveToPreferencesWithCompletionHandler { error in
            // Handle error or success.
        }
    }
}
```

NEVPNManager

Configure Connect On Demand

NEVPNManager

Configure Connect On Demand

```
let manager = NEVPNManager.sharedManager()

// Connect automatically on a matching network.
let connectRule = NEOnDemandRuleConnect()
```

NEVPNManager

Configure Connect On Demand

```
let manager = NEVPNManager.sharedManager()

// Connect automatically on a matching network.
let connectRule = NEOnDemandRuleConnect()

// Match all Wi-Fi networks.
connectRule.interfaceTypeMatch = .WiFi
```

NEVPNManager

Configure Connect On Demand

```
let manager = NEVPNManager.sharedManager()

// Connect automatically on a matching network.
let connectRule = NEOnDemandRuleConnect()

// Match all Wi-Fi networks.
connectRule.interfaceTypeMatch = .WiFi

// Set rule in array.
manager.onDemandRules = [ connectRule ]
```

NEVPNManager

Configure Connect On Demand

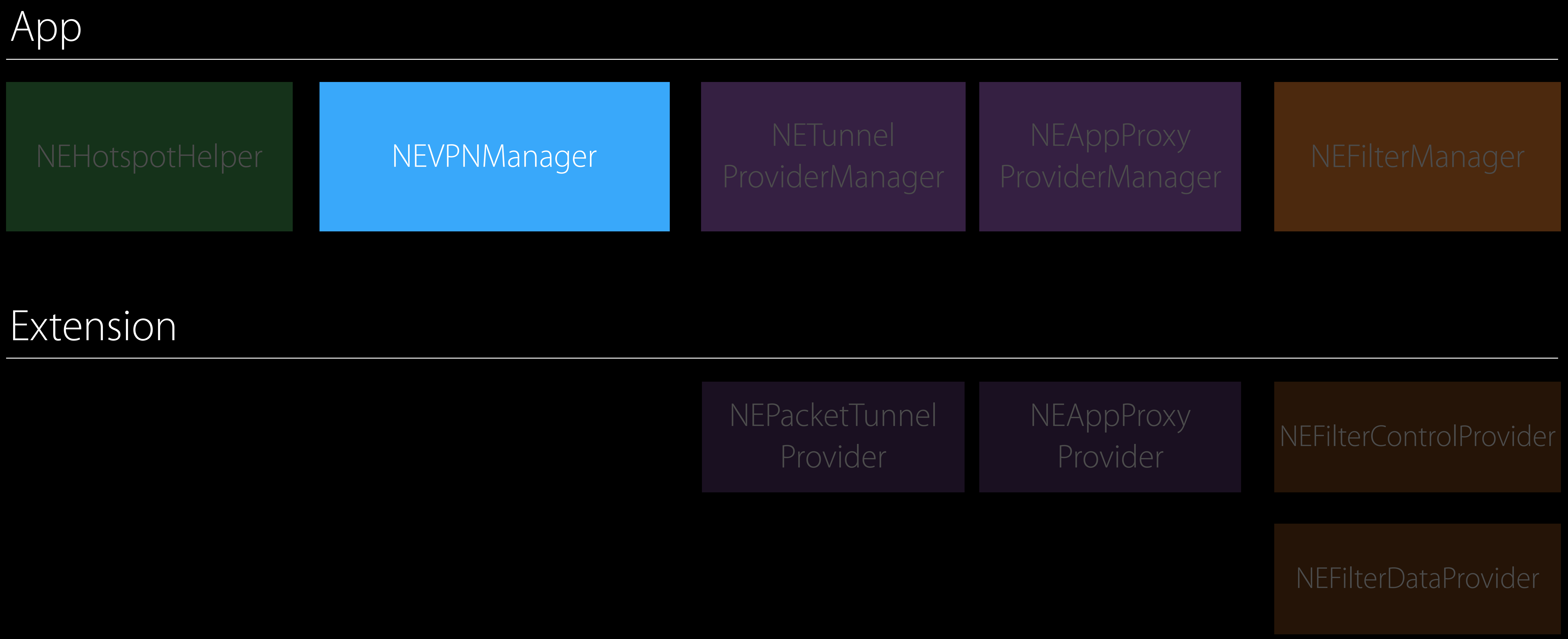
```
let manager = NEVPNManager.sharedManager()

// Connect automatically on a matching network.
let connectRule = NEOnDemandRuleConnect()

// Match all Wi-Fi networks.
connectRule.interfaceTypeMatch = .WiFi

// Set rule in array.
manager.onDemandRules = [ connectRule ]

// Call saveToPreferencesWithCompletionHandler()
manager.saveToPreferencesWithCompletionHandler { error in
    // Handle error or success.
}
```



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

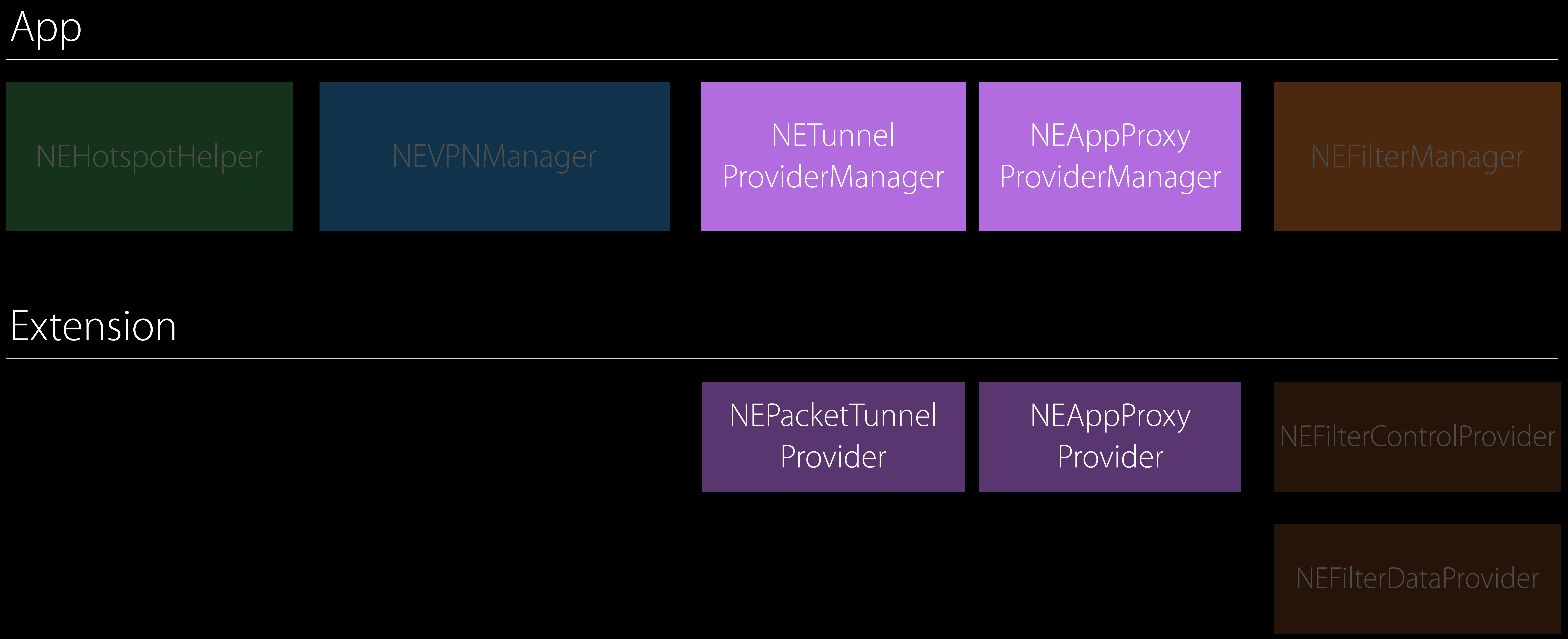
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider

NETunnelProvider

Custom Enterprise VPN solutions

Create a custom VPN protocol provider

NETunnelProvider

Custom Enterprise VPN solutions

Create a custom VPN protocol provider

- Runs as an app extension

NETunnelProvider

Custom Enterprise VPN solutions

Create a custom VPN protocol provider

- Runs as an app extension
- Packet Tunnel Provider for IP layer tunneling

NETunnelProvider

Custom Enterprise VPN solutions

Create a custom VPN protocol provider

- Runs as an app extension
- Packet Tunnel Provider for IP layer tunneling
- App Proxy Provider for app layer tunneling

NETunnelProvider

Custom Enterprise VPN solutions

Create a custom VPN protocol provider

- Runs as an app extension
- Packet Tunnel Provider for IP layer tunneling
- App Proxy Provider for app layer tunneling

Configure and control your protocol provider from your app

App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider

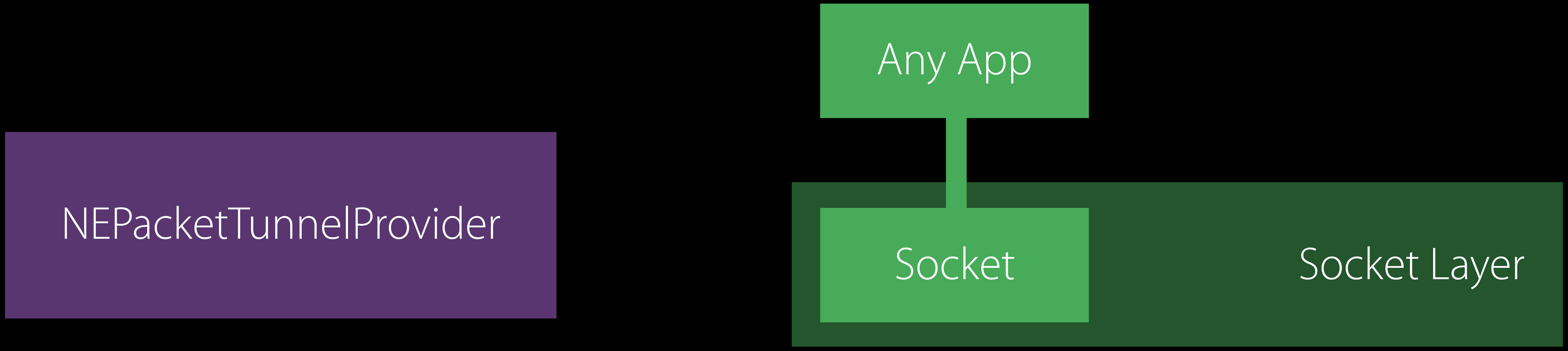
NEPacketTunnelProvider

VPN data path

NEPacketTunnelProvider

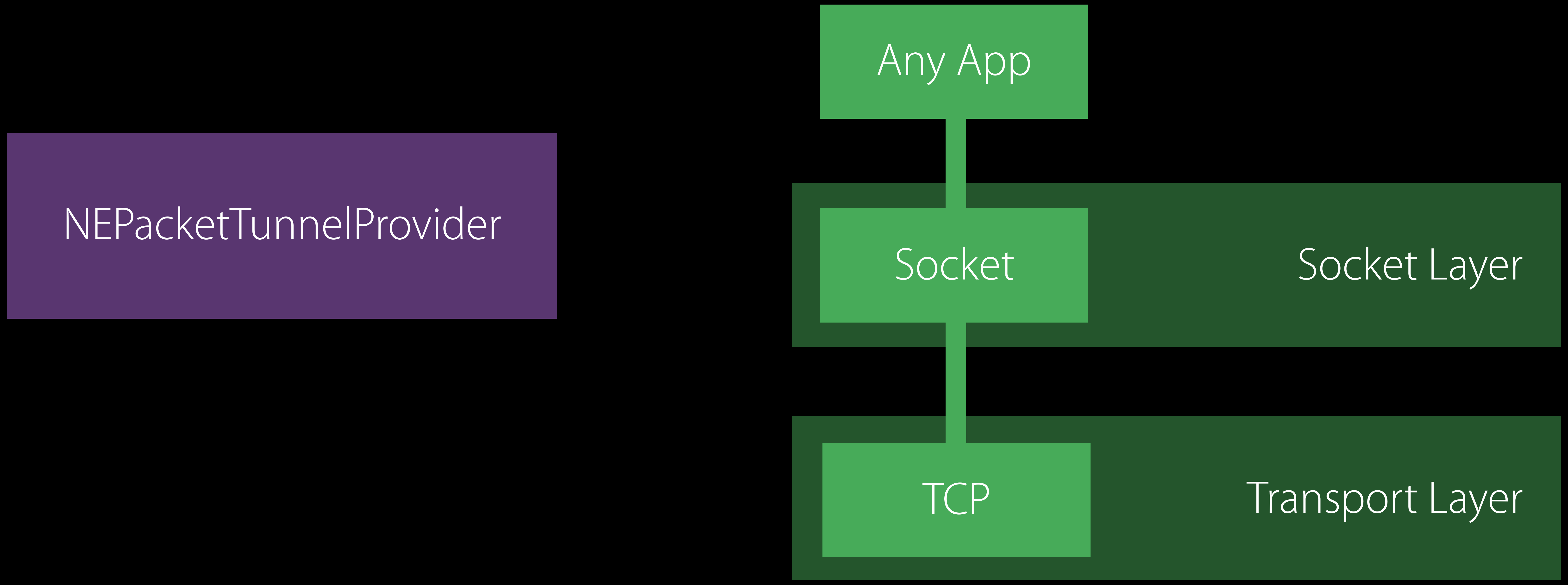
NEPacketTunnelProvider

VPN data path



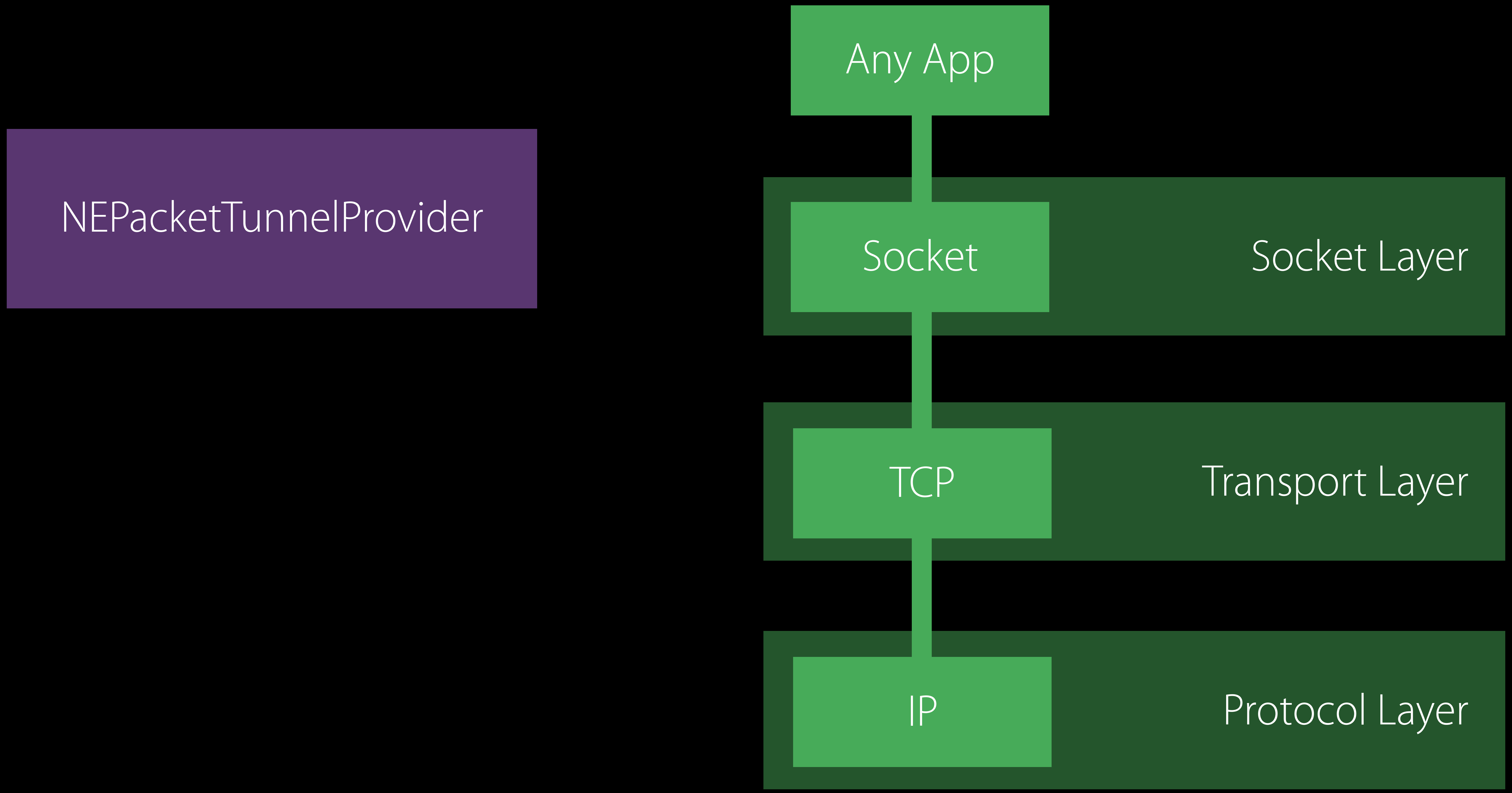
NEPacketTunnelProvider

VPN data path



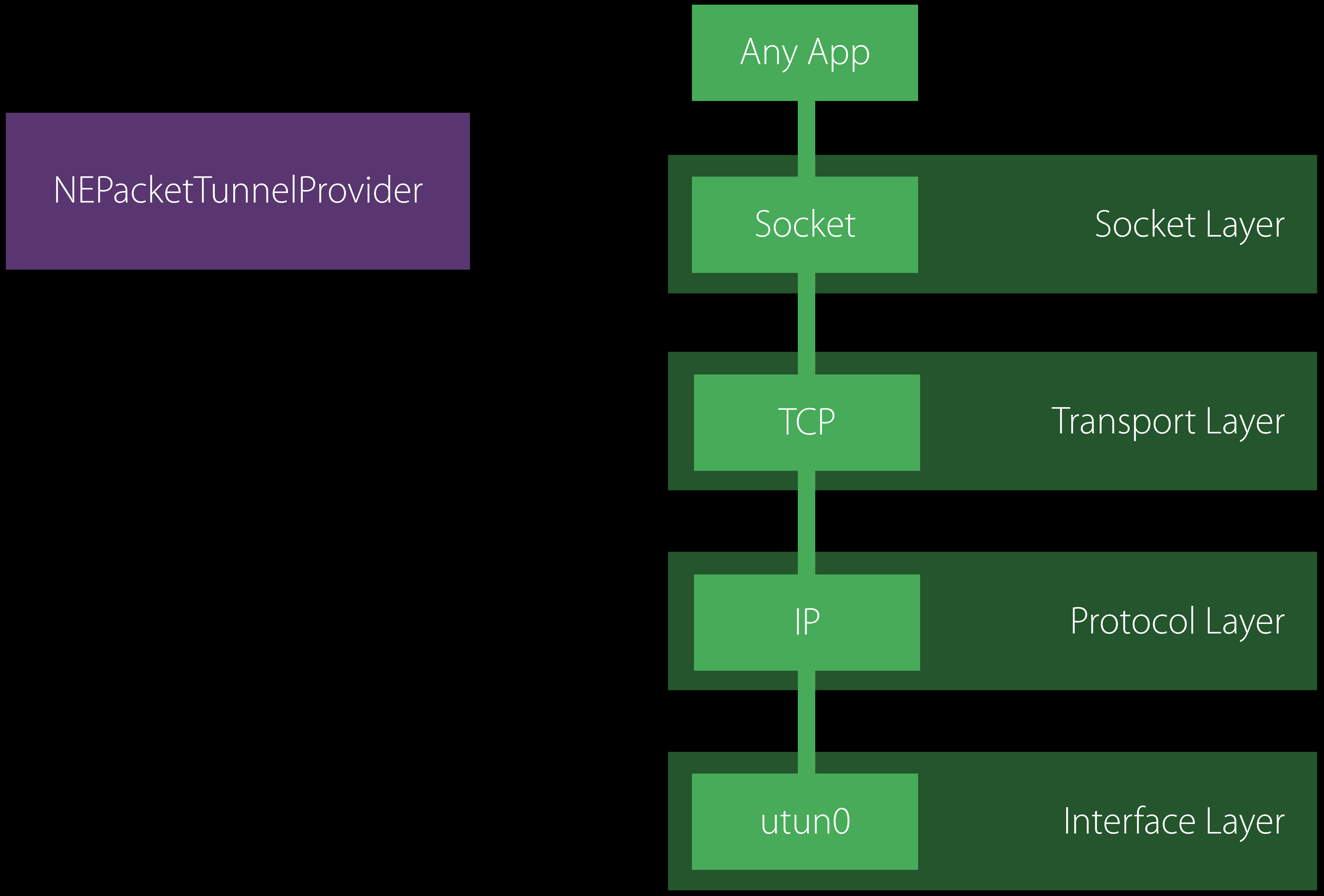
NEPacketTunnelProvider

VPN data path



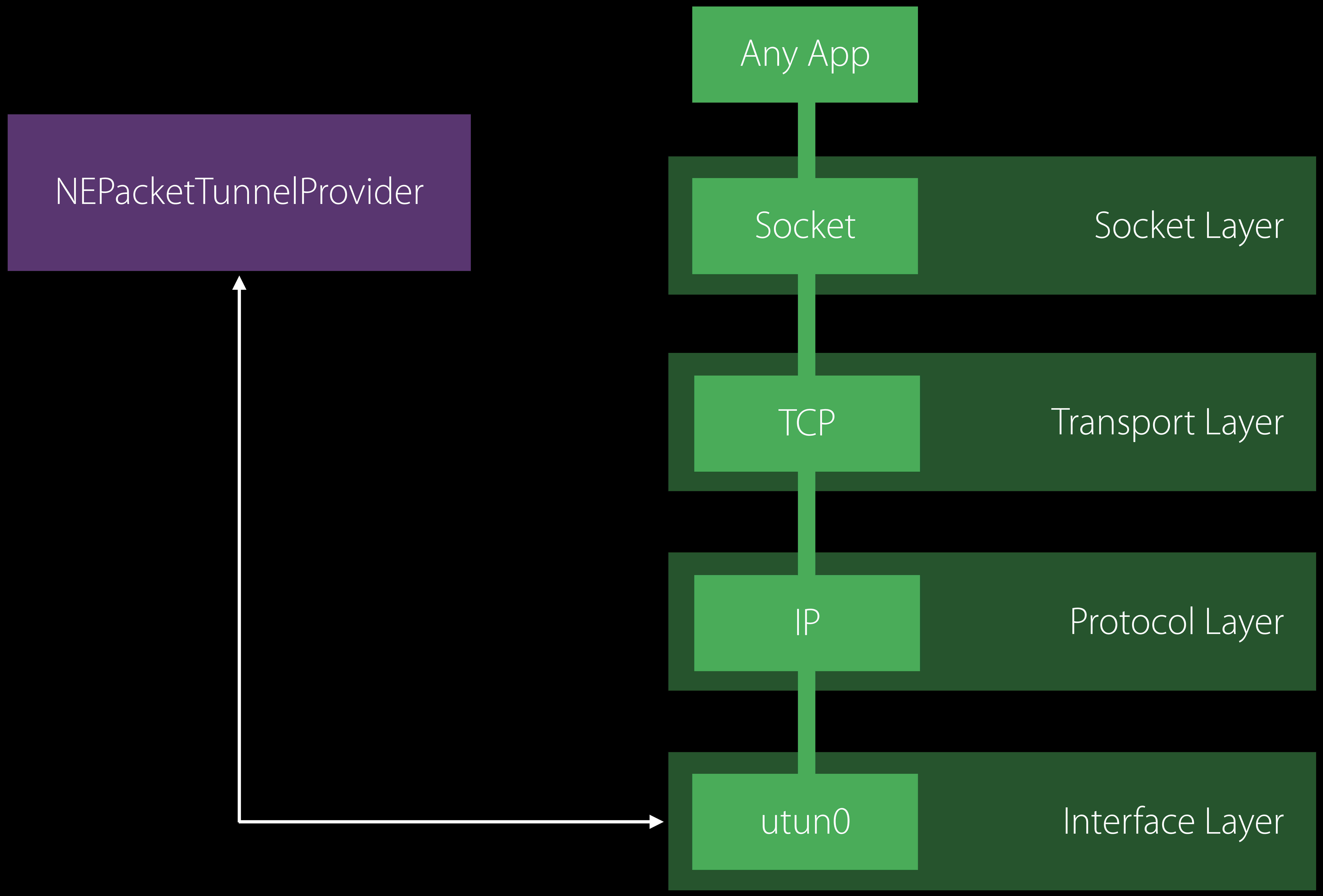
NEPacketTunnelProvider

VPN data path



NEPacketTunnelProvider

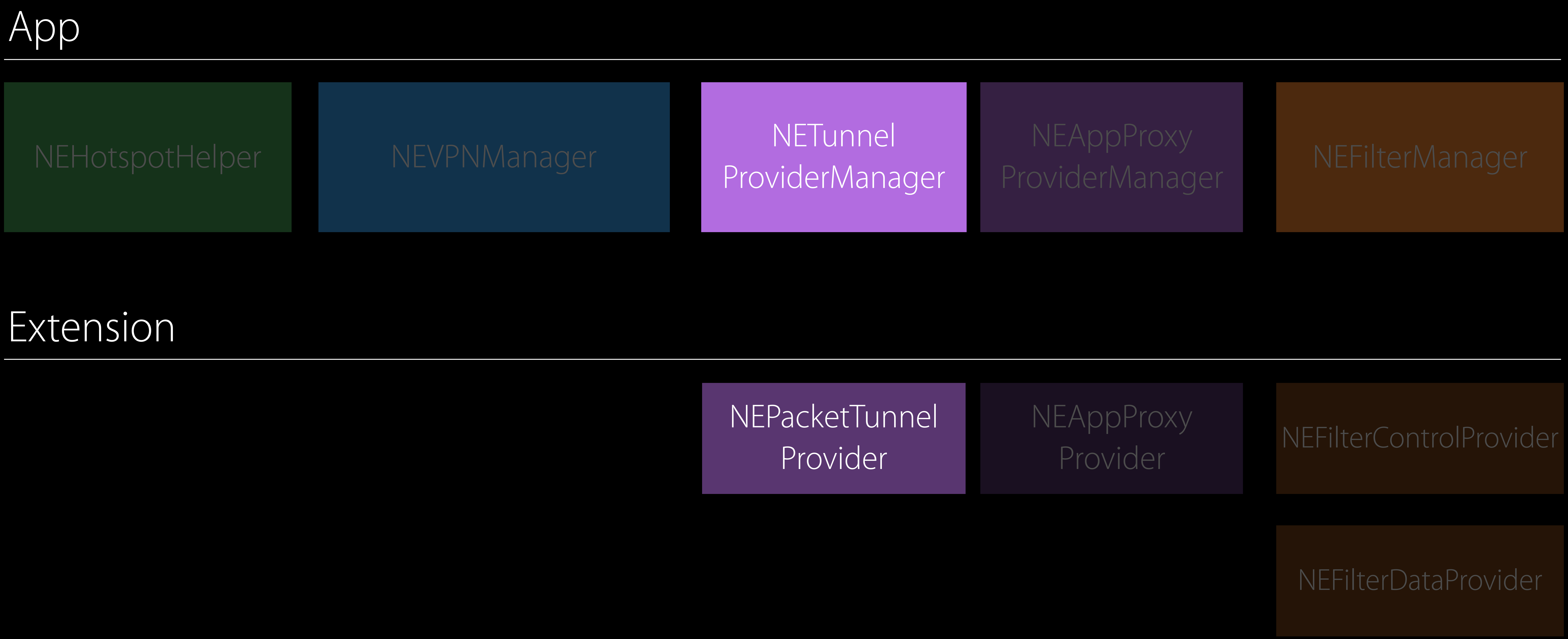
VPN data path



Demo

NEPacketTunnelProvider

Tommy Pauly



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider

Per-App VPN for Managed Apps

VPN for BYOD

Per-App VPN for Managed Apps

VPN for BYOD

Configure Per-App VPN using Mobile Device Management (MDM)

Per-App VPN for Managed Apps

VPN for BYOD

Configure Per-App VPN using Mobile Device Management (MDM)

- Enroll devices in an MDM service

Per-App VPN for Managed Apps

VPN for BYOD

Configure Per-App VPN using Mobile Device Management (MDM)

- Enroll devices in an MDM service
- Link managed apps with Per-App VPN configurations

Per-App VPN for Managed Apps

VPN for BYOD

Configure Per-App VPN using Mobile Device Management (MDM)

- Enroll devices in an MDM service
- Link managed apps with Per-App VPN configurations

Supported protocols

- Custom App Proxy Providers

Per-App VPN for Managed Apps

VPN for BYOD

Configure Per-App VPN using Mobile Device Management (MDM)

- Enroll devices in an MDM service
- Link managed apps with Per-App VPN configurations

Supported protocols

- Custom App Proxy Providers
- Custom Packet Tunnel Providers

Per-App VPN for Managed Apps

VPN for BYOD

Configure Per-App VPN using Mobile Device Management (MDM)

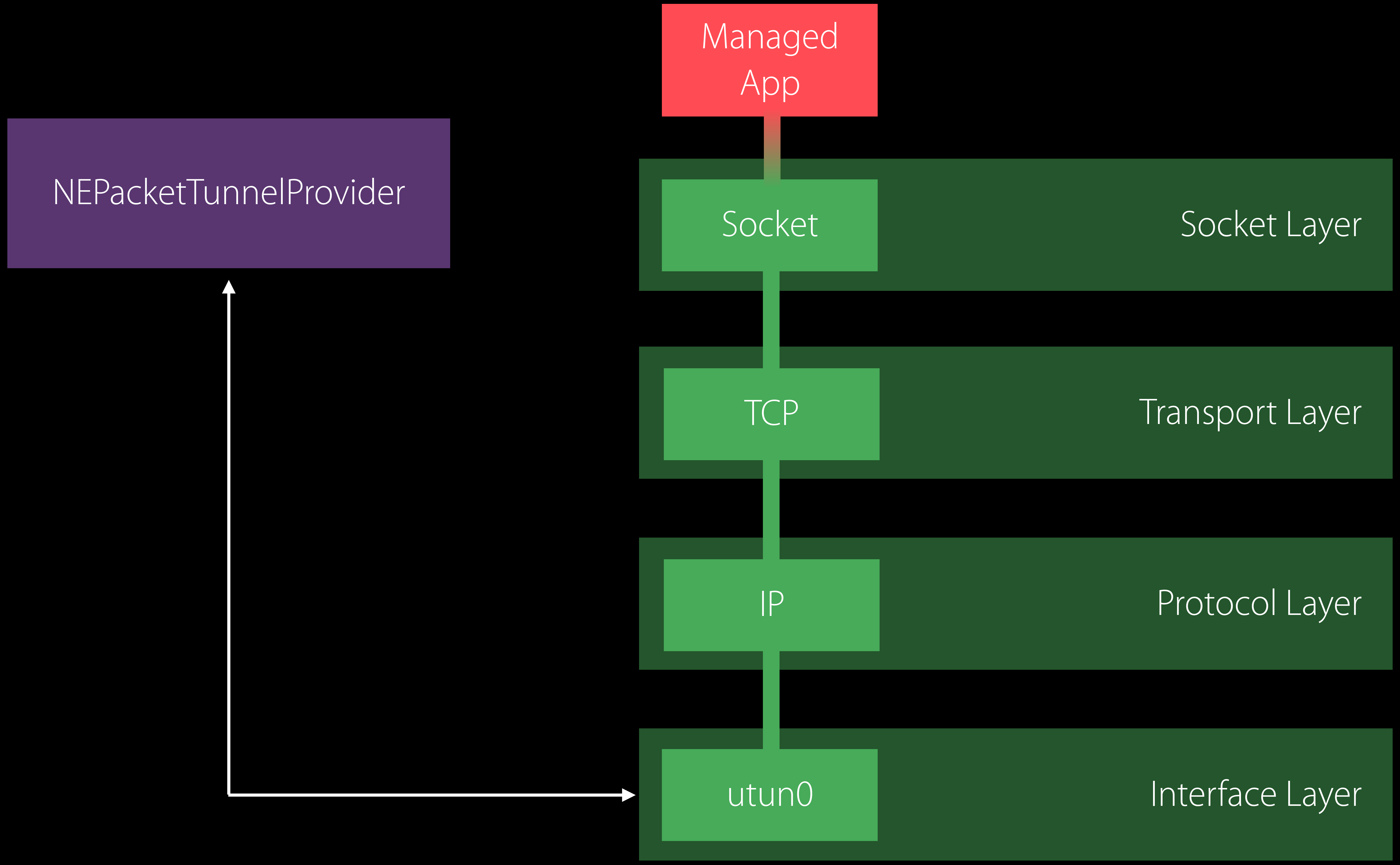
- Enroll devices in an MDM service
- Link managed apps with Per-App VPN configurations

Supported protocols

- Custom App Proxy Providers
- Custom Packet Tunnel Providers
- Built-in IPsec clients (IKEv1 and IKEv2)

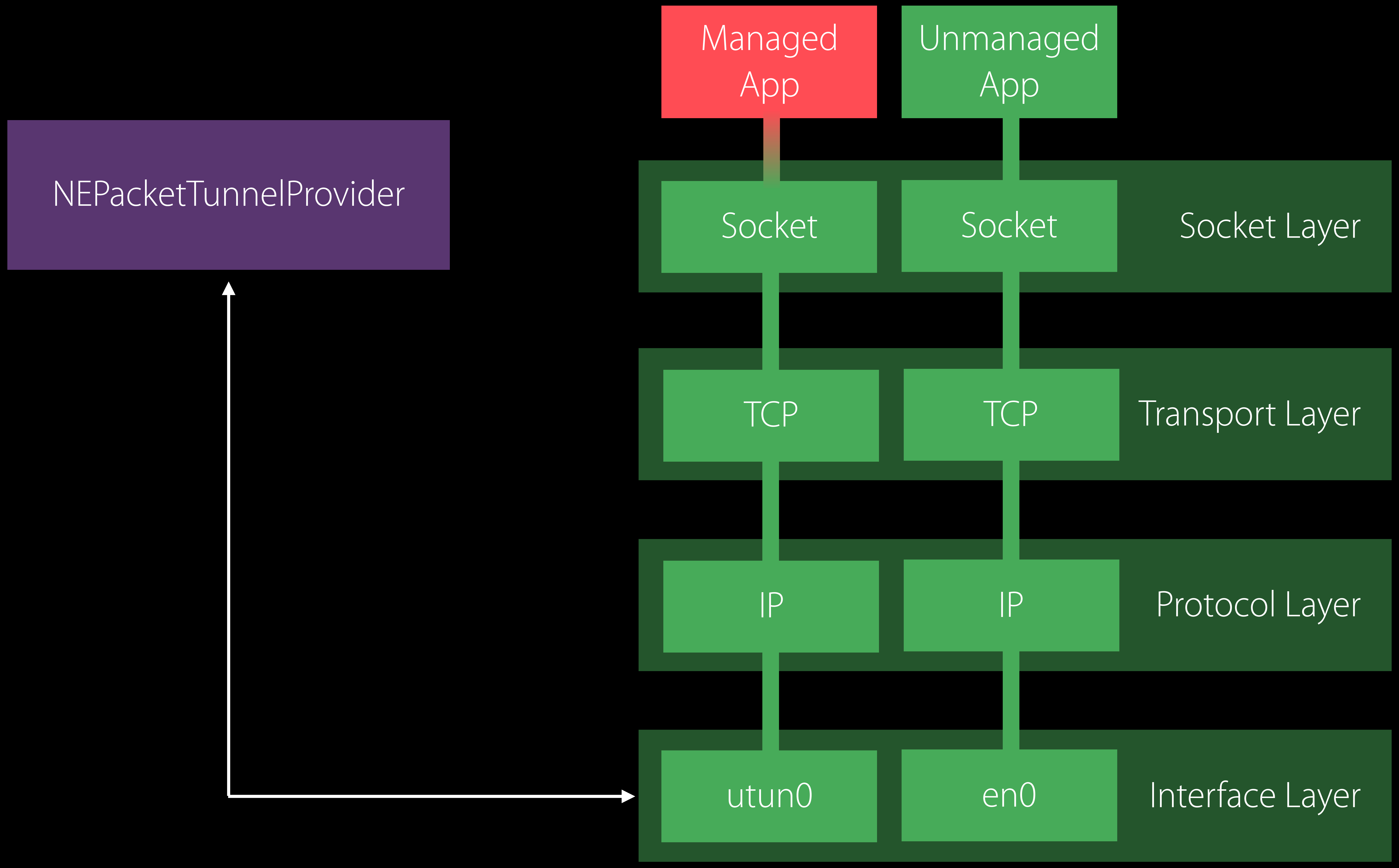
NEPacketTunnelProvider

Per-App VPN data path



NEPacketTunnelProvider

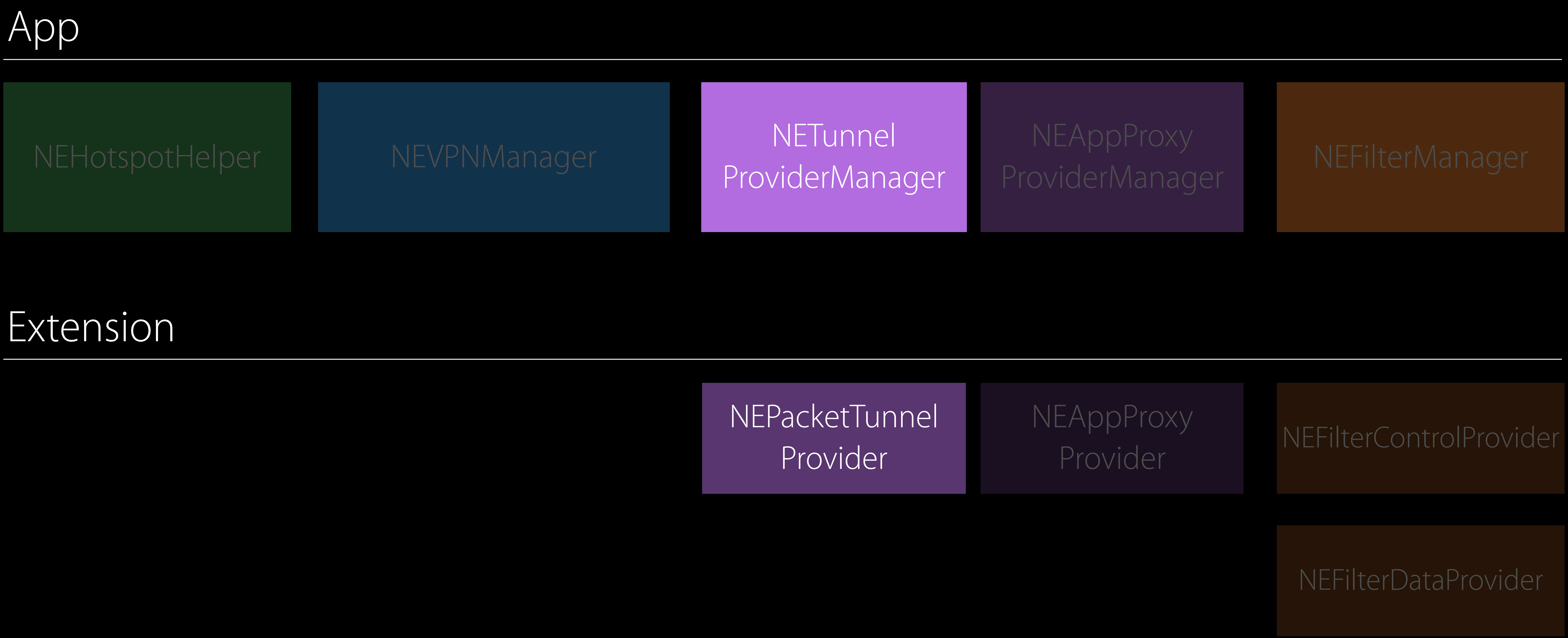
Per-App VPN data path



Demo

Per-App VPN with NEPacketTunnelProvider

Tommy Pauly



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

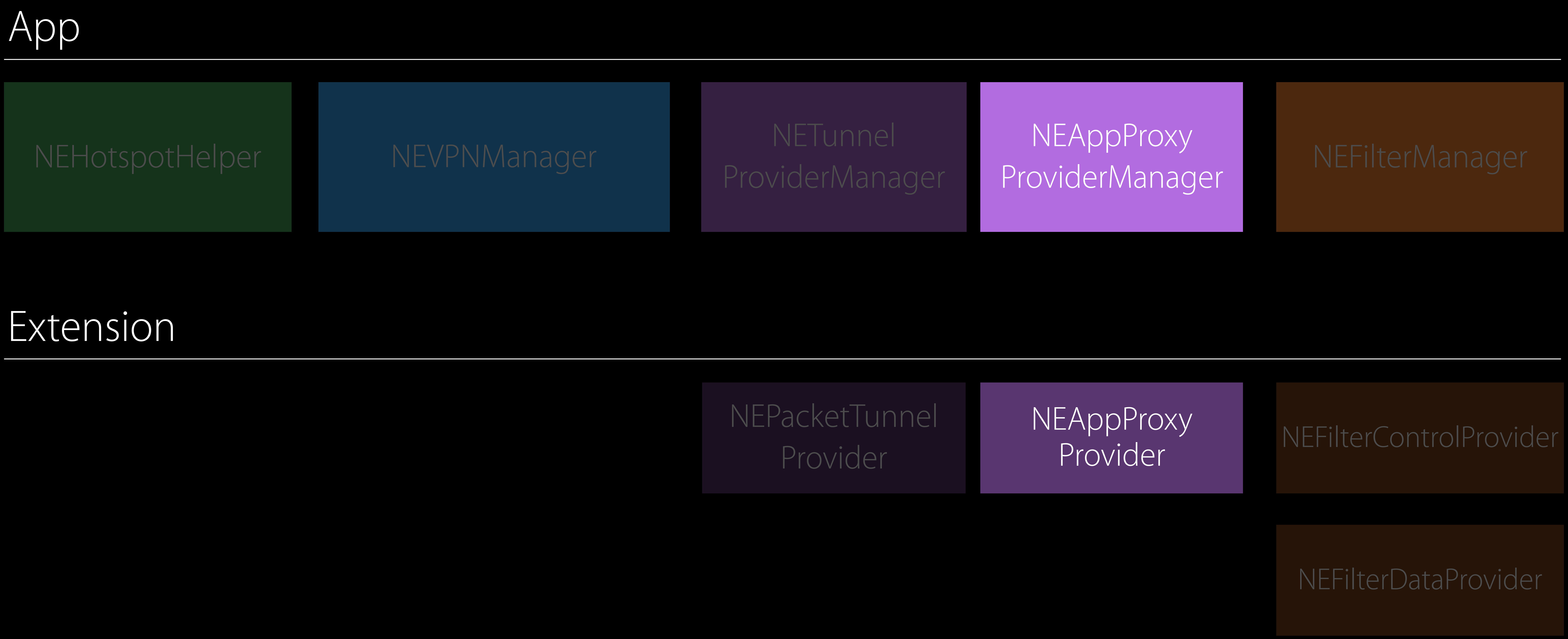
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

Extension

NEPacketTunnel Provider

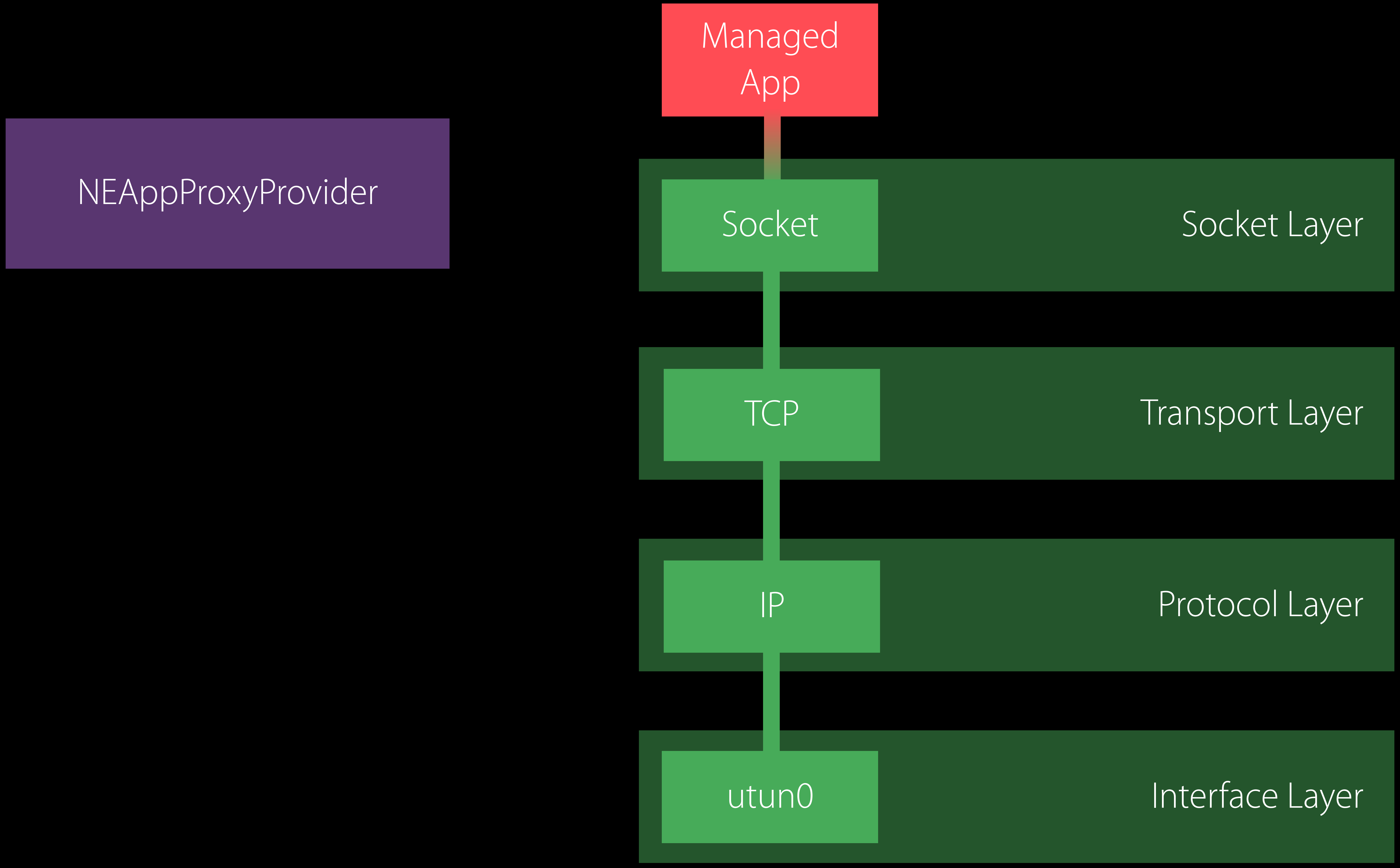
NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider

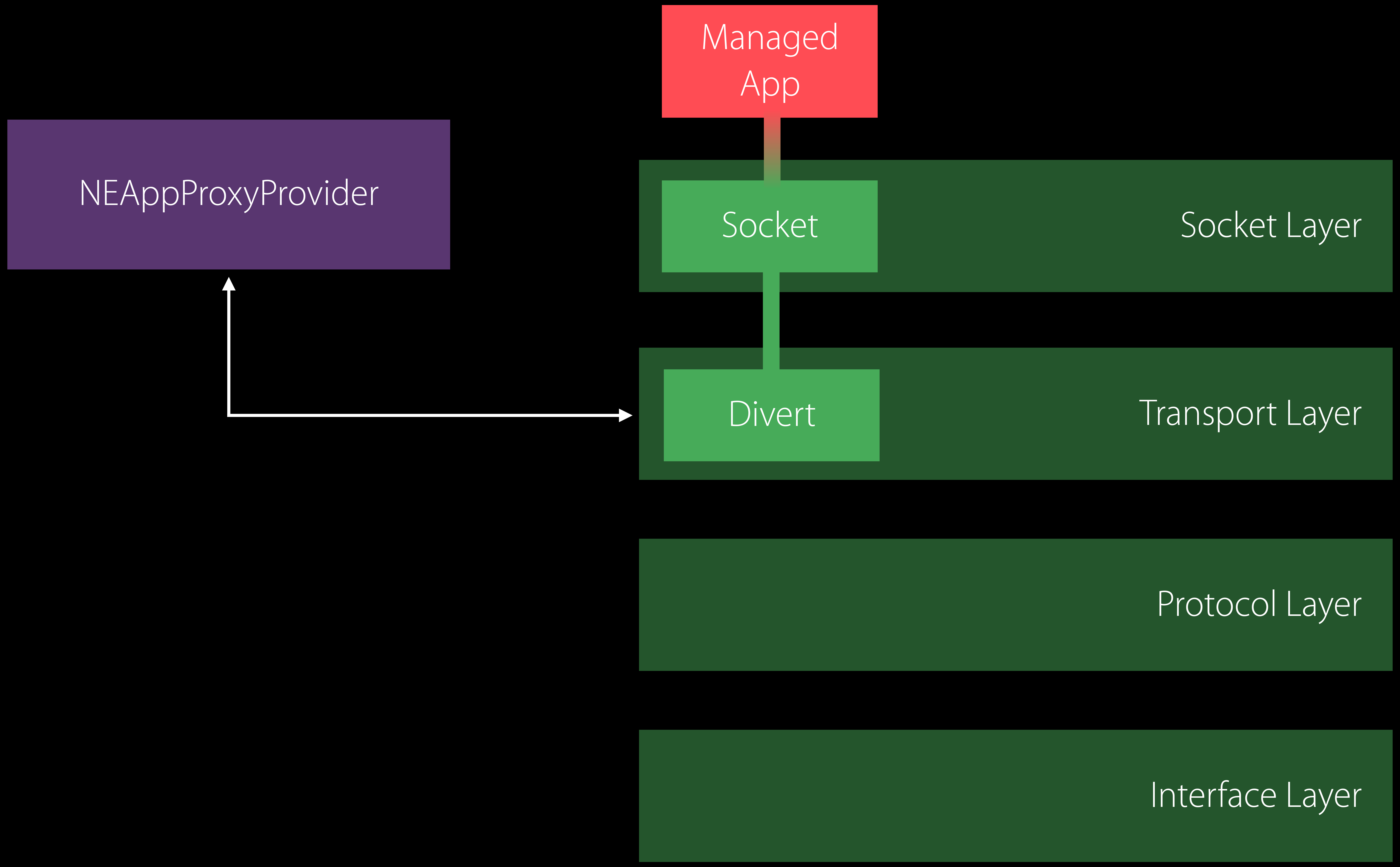
NEAppProxyProvider

Per-App VPN data path



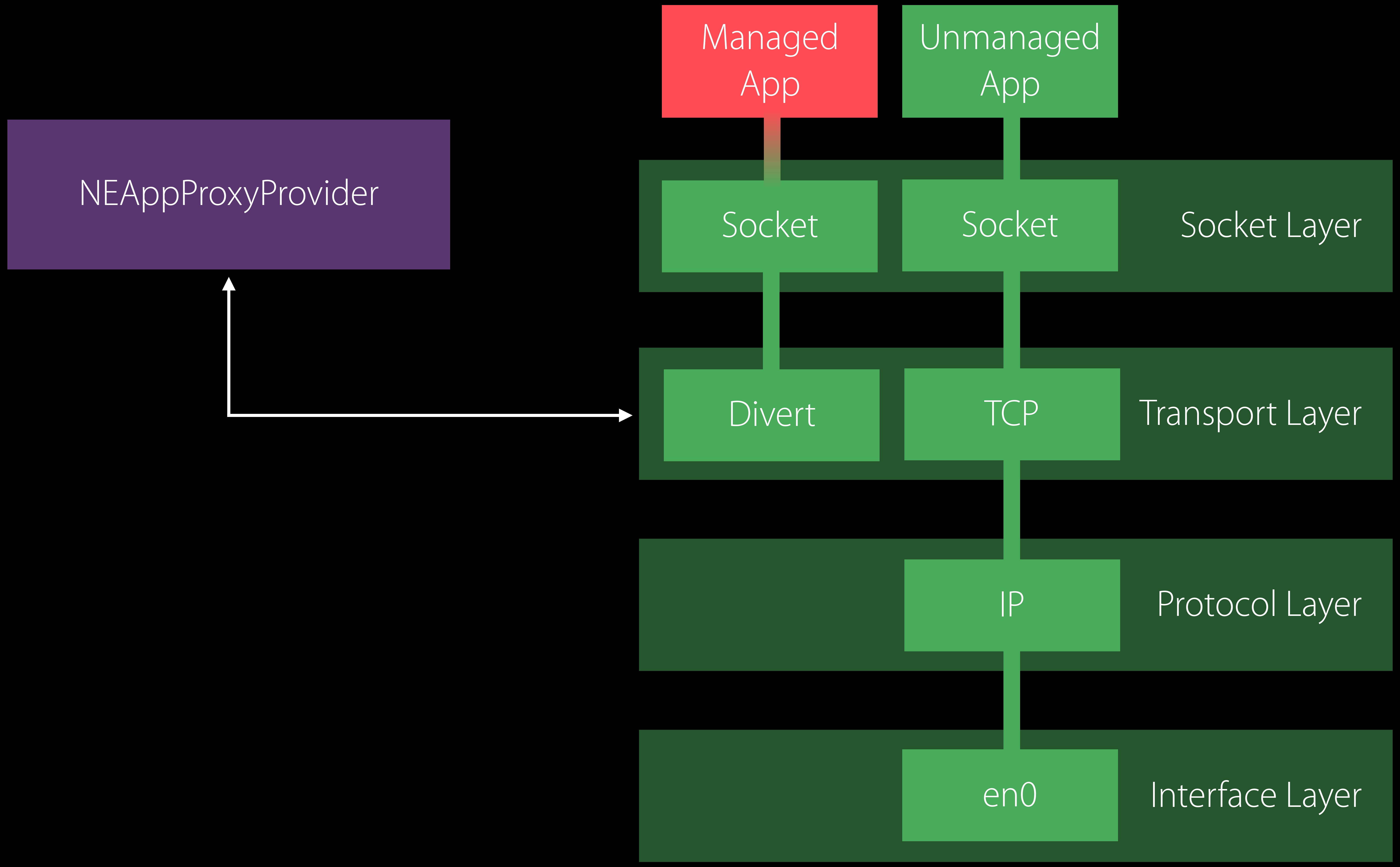
NEAppProxyProvider

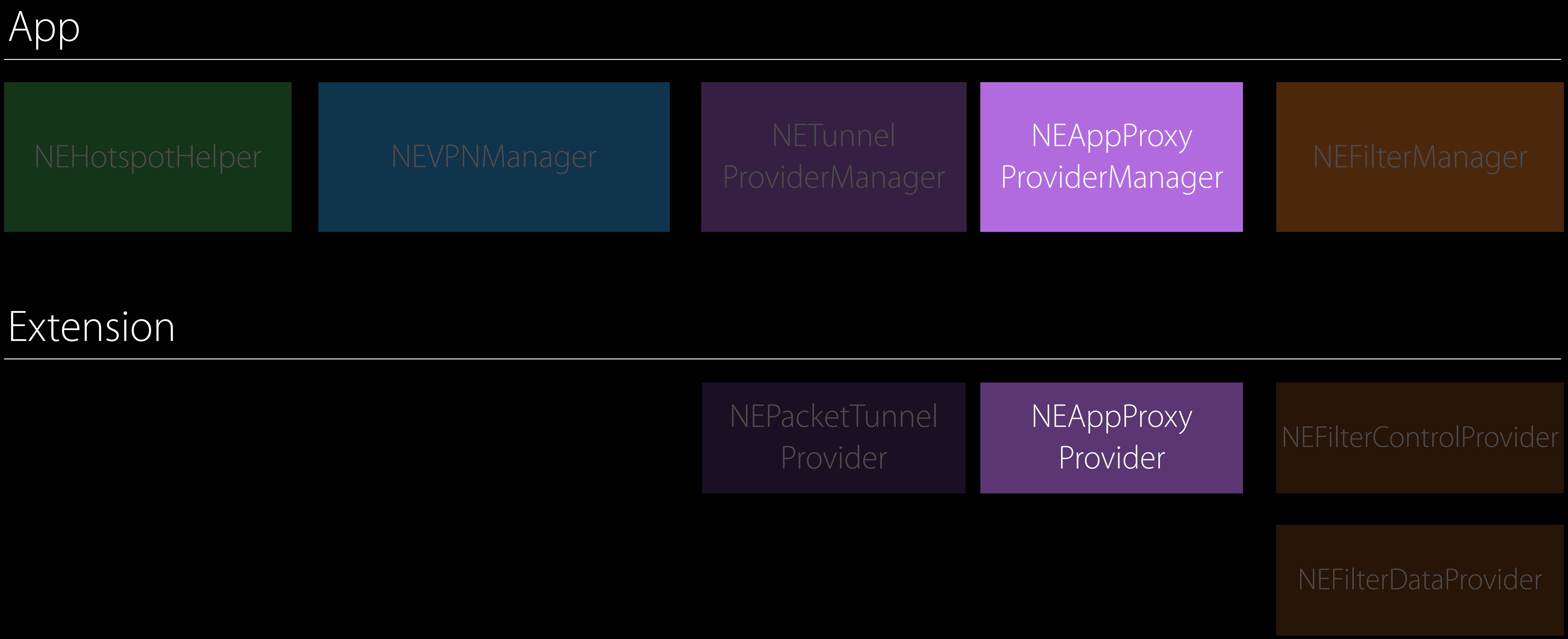
Per-App VPN data path



NEAppProxyProvider

Per-App VPN data path





App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

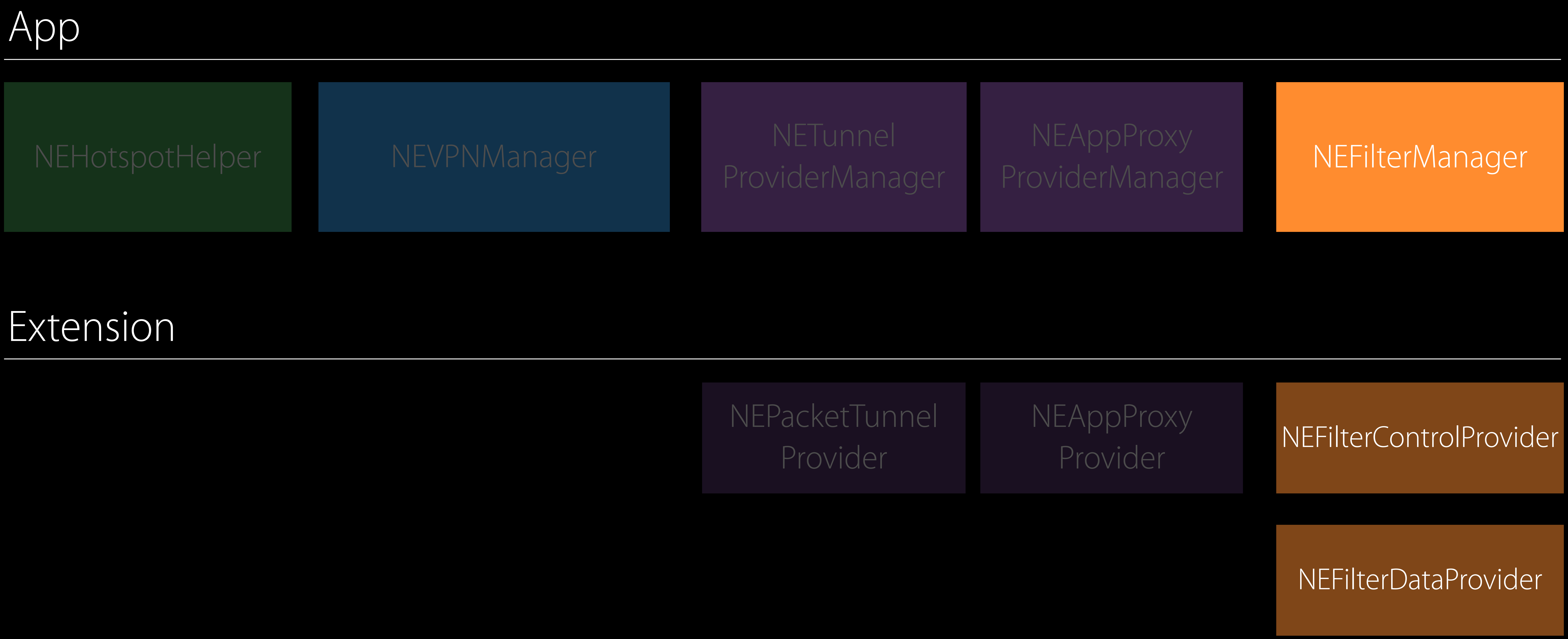
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App	NEHotspotHelper	NEVPNManager	NETunnel ProviderManager	NEAppProxy ProviderManager	NEFilterManager
Extension			NEPacketTunnel Provider	NEAppProxy Provider	NEFilterControlProvider NEFilterDataProvider

NEFilterProvider

Previous solutions for school content filtering

NEFilterProvider

Previous solutions for school content filtering

On-site content filter
⊗

NEFilterProvider

Previous solutions for school content filtering

On-site content filter	Global proxy
⊗	⊗

NEFilterProvider

Previous solutions for school content filtering

On-site content filter	Global proxy	Full tunnel VPN
⊗	⊗	⊗

NEFilterProvider

Previous solutions for school content filtering

On-site content filter	Global proxy	Full tunnel VPN	On-device content filter
✗	✗	✗	✓

NEFilterProvider

On-device network content filtering

Dynamic evaluation of network content

NEFilterProvider

On-device network content filtering

Dynamic evaluation of network content

Ability to update filtering rules on the fly from the Internet

NEFilterProvider

On-device network content filtering

Dynamic evaluation of network content

Ability to update filtering rules on the fly from the Internet

Customizable block page

NEFilterProvider

On-device network content filtering

Dynamic evaluation of network content

Ability to update filtering rules on the fly from the Internet

Customizable block page

For supervised devices

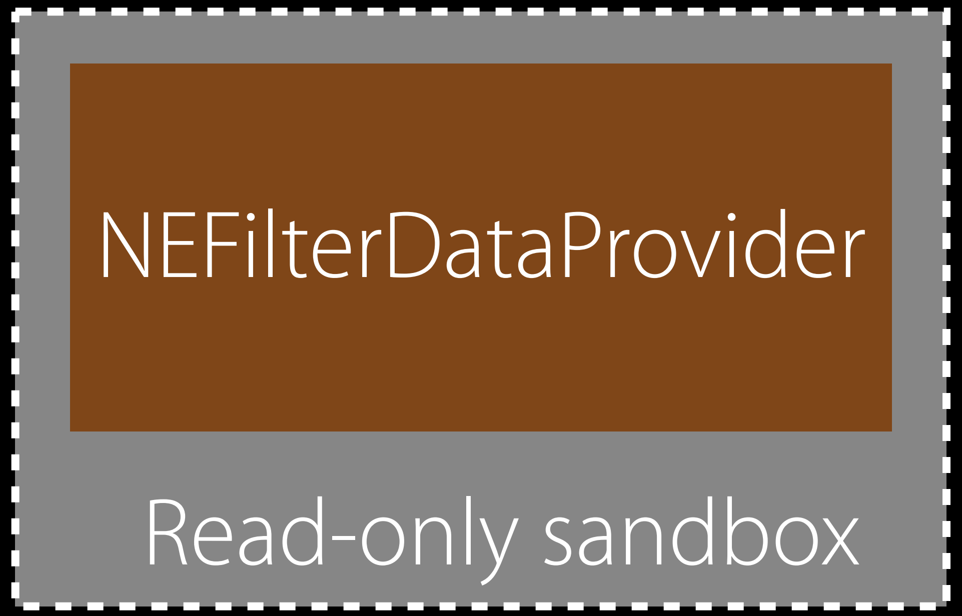
NEFilterProvider

Filter data path

NEFilterDataProvider

NEFilterProvider

Filter data path

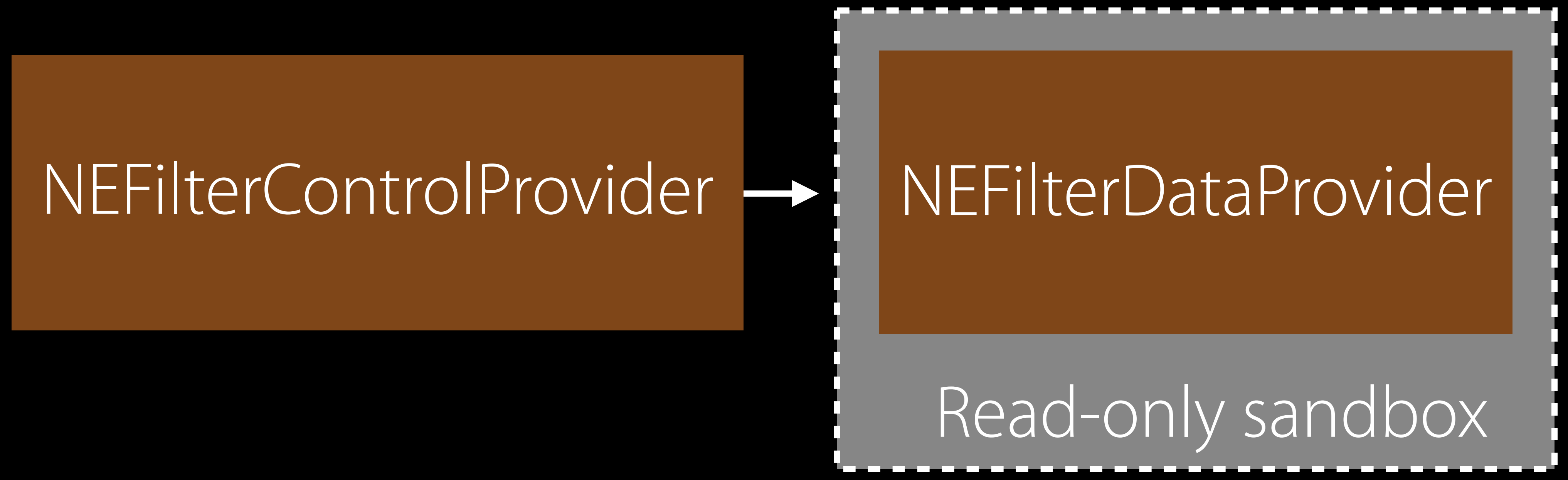


NEFilterDataProvider

Read-only sandbox

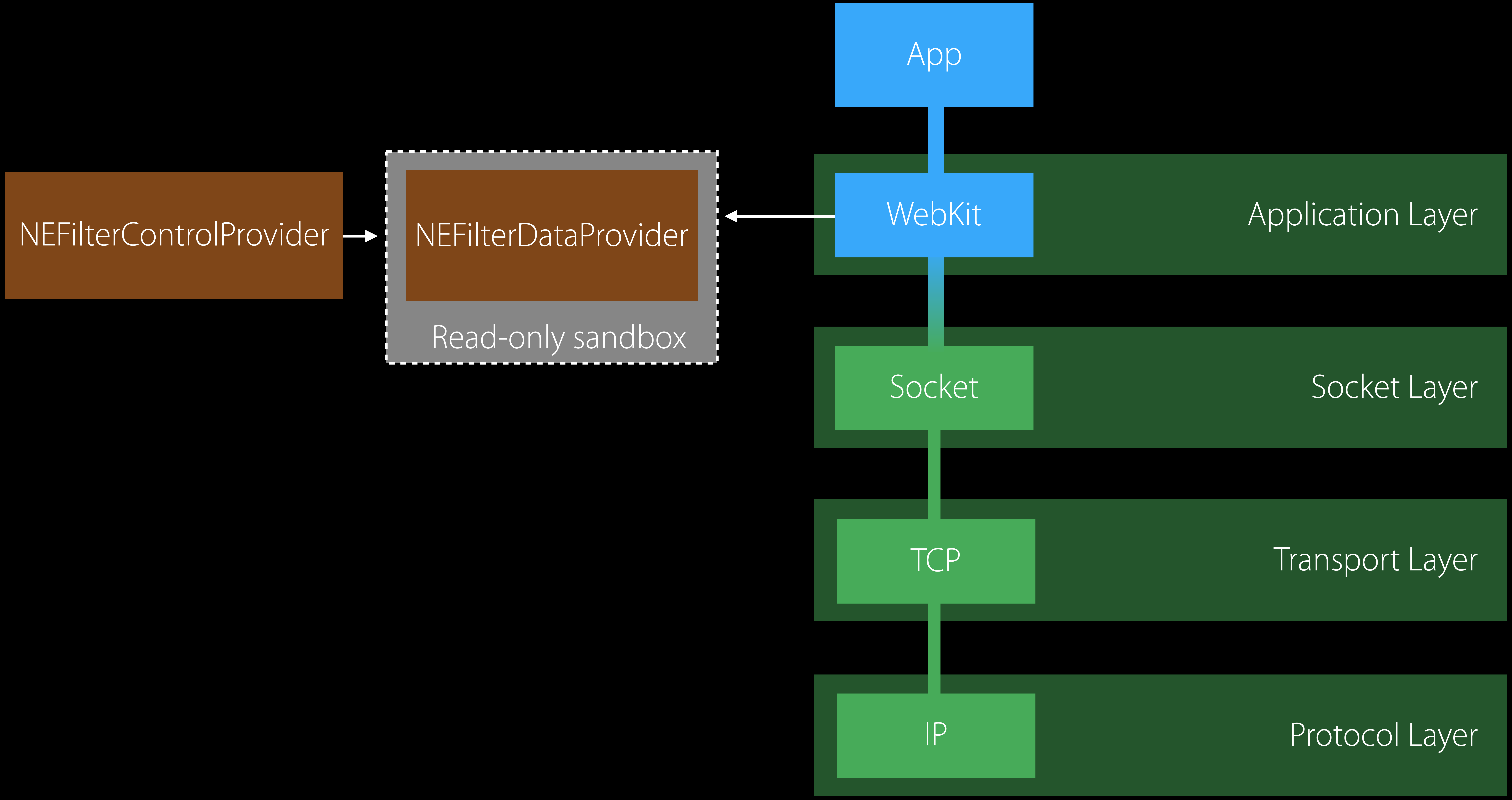
NEFilterProvider

Filter data path



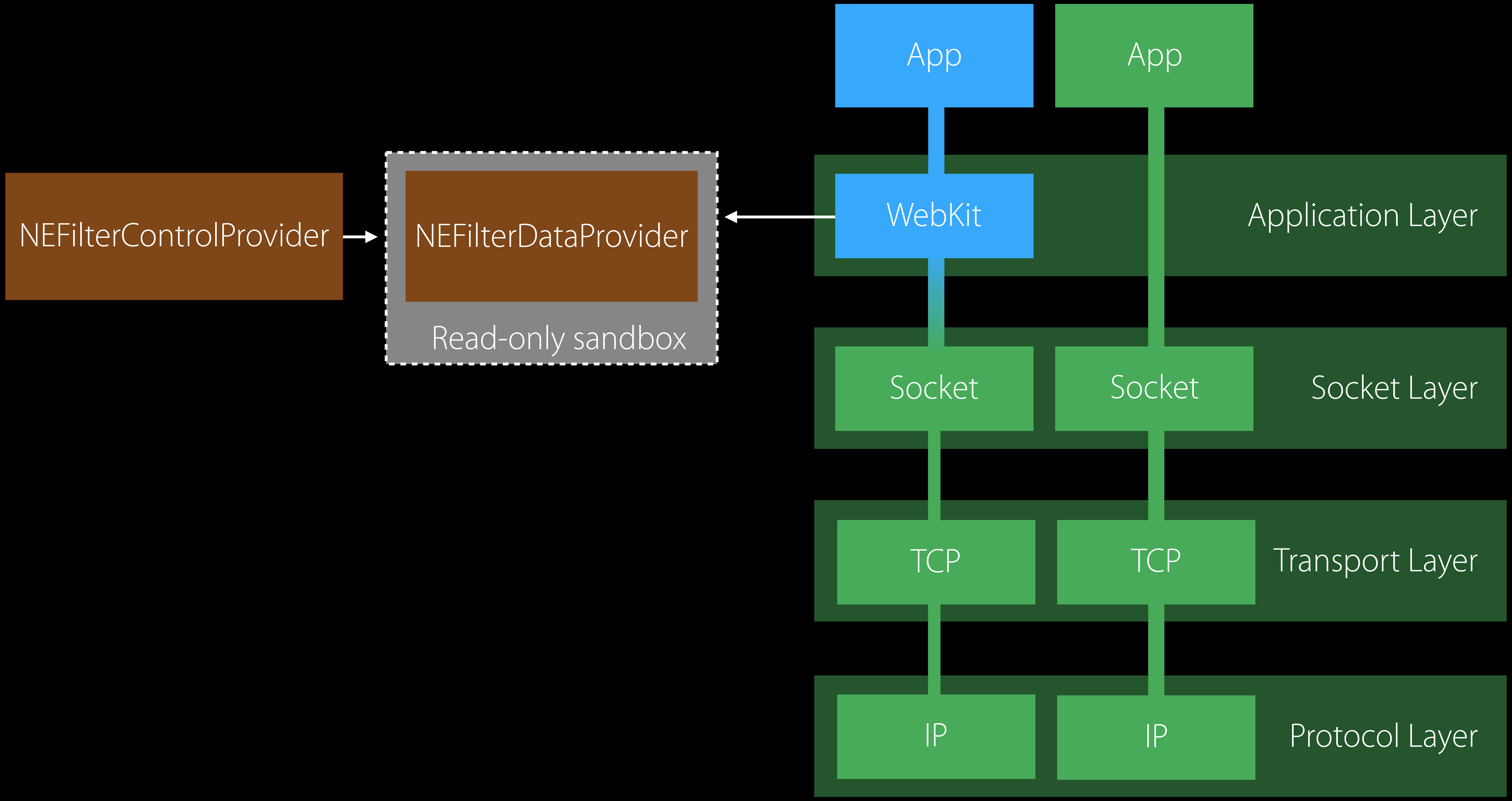
NEFilterProvider

Filter data path



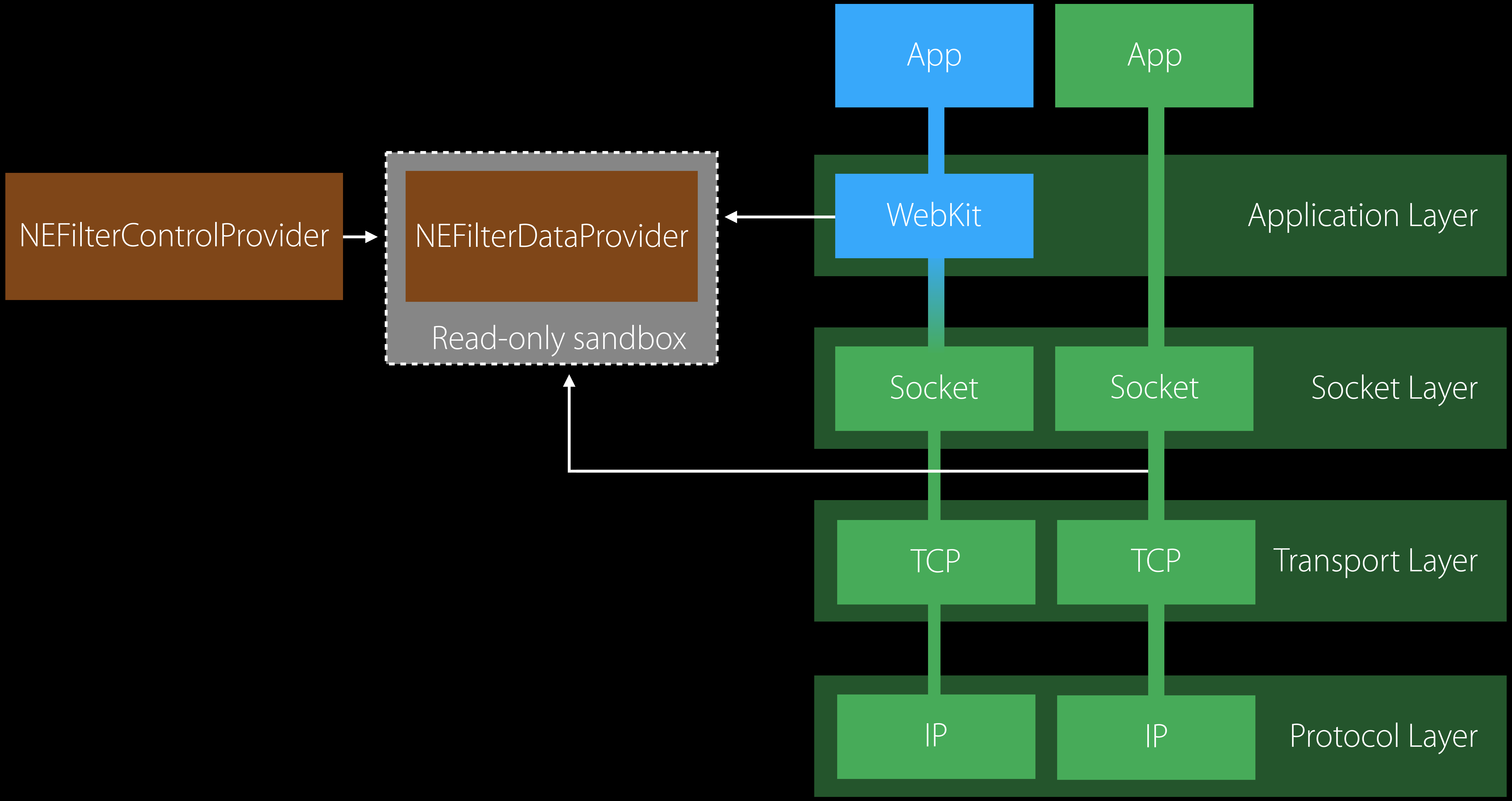
NEFilterProvider

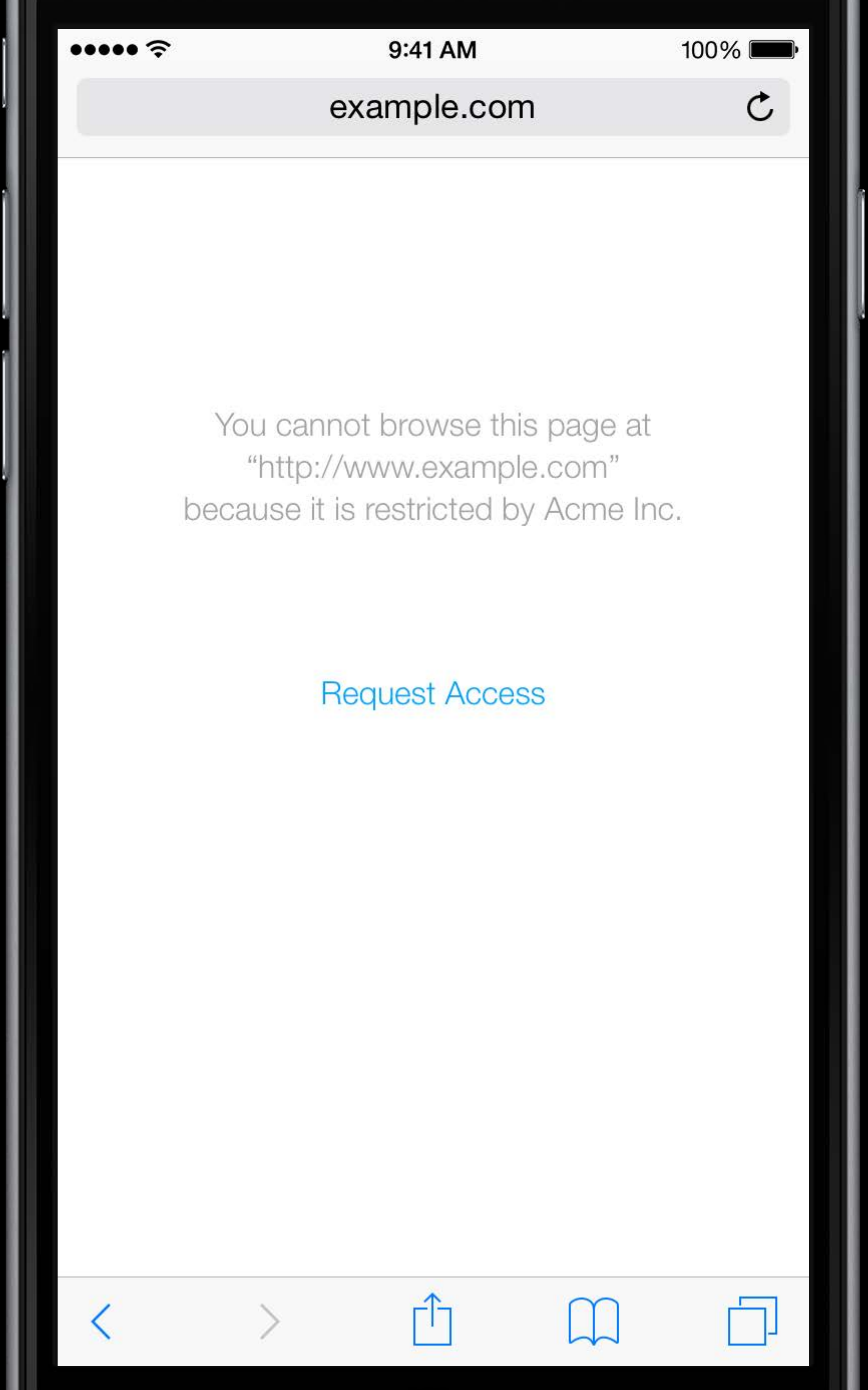
Filter data path



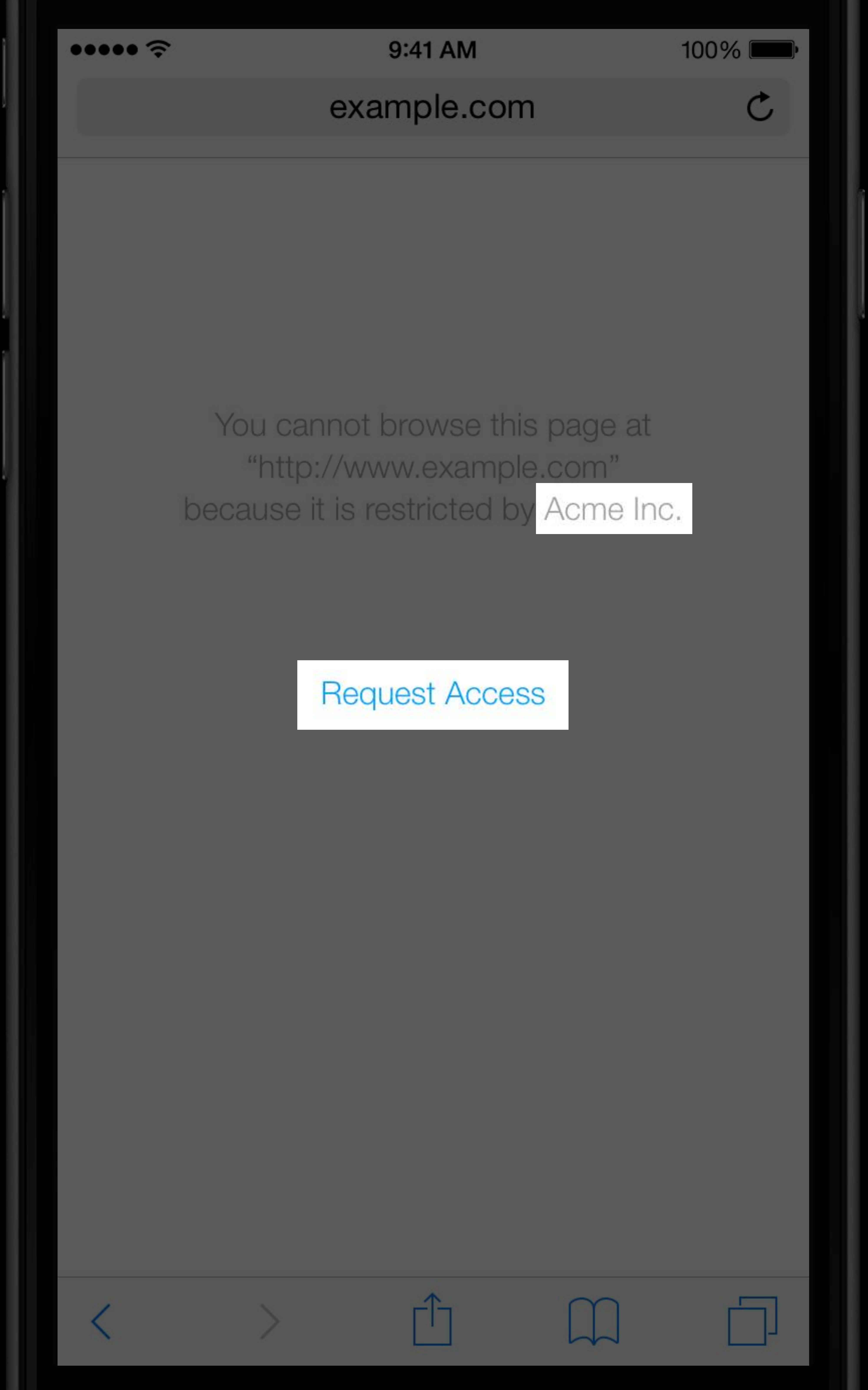
NEFilterProvider

Filter data path

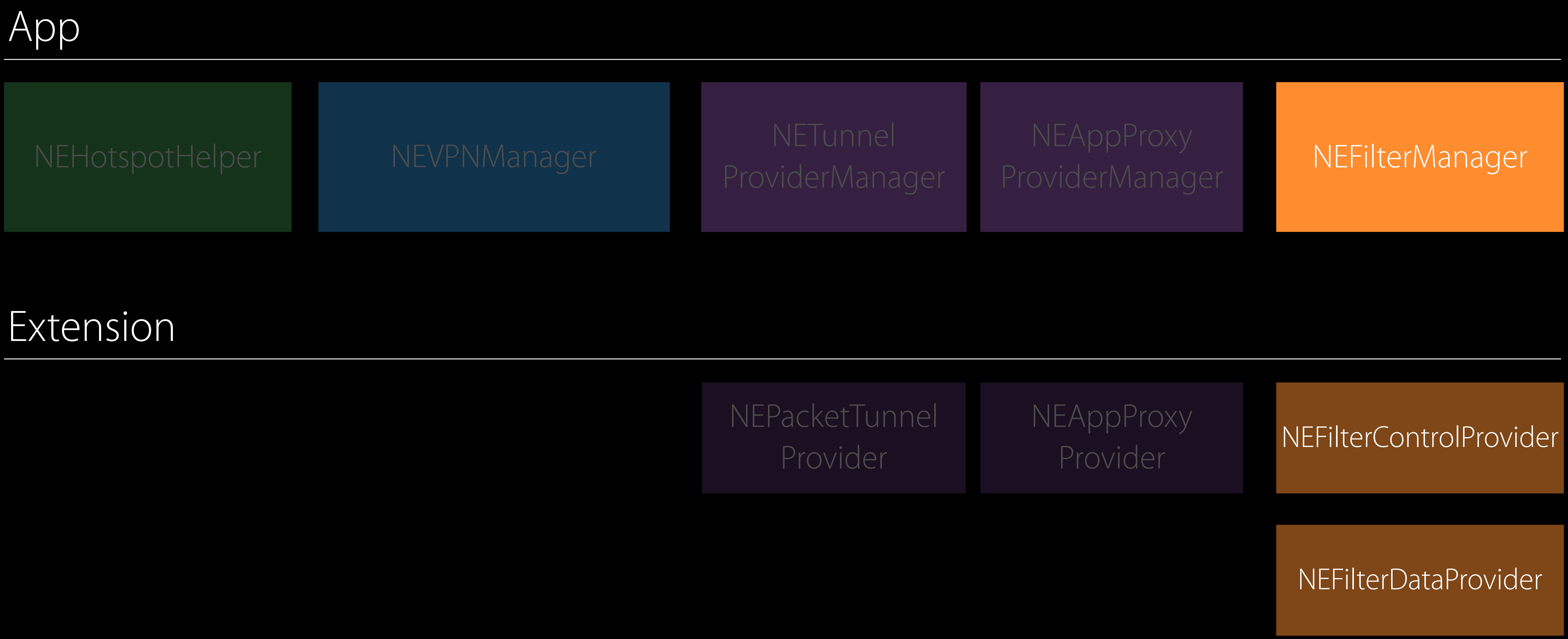




You cannot browse this page at “http://www.example.com” because it is restricted by Acme Inc.



You cannot browse this page at “http://www.example.com” because it is restricted by Acme Inc.



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

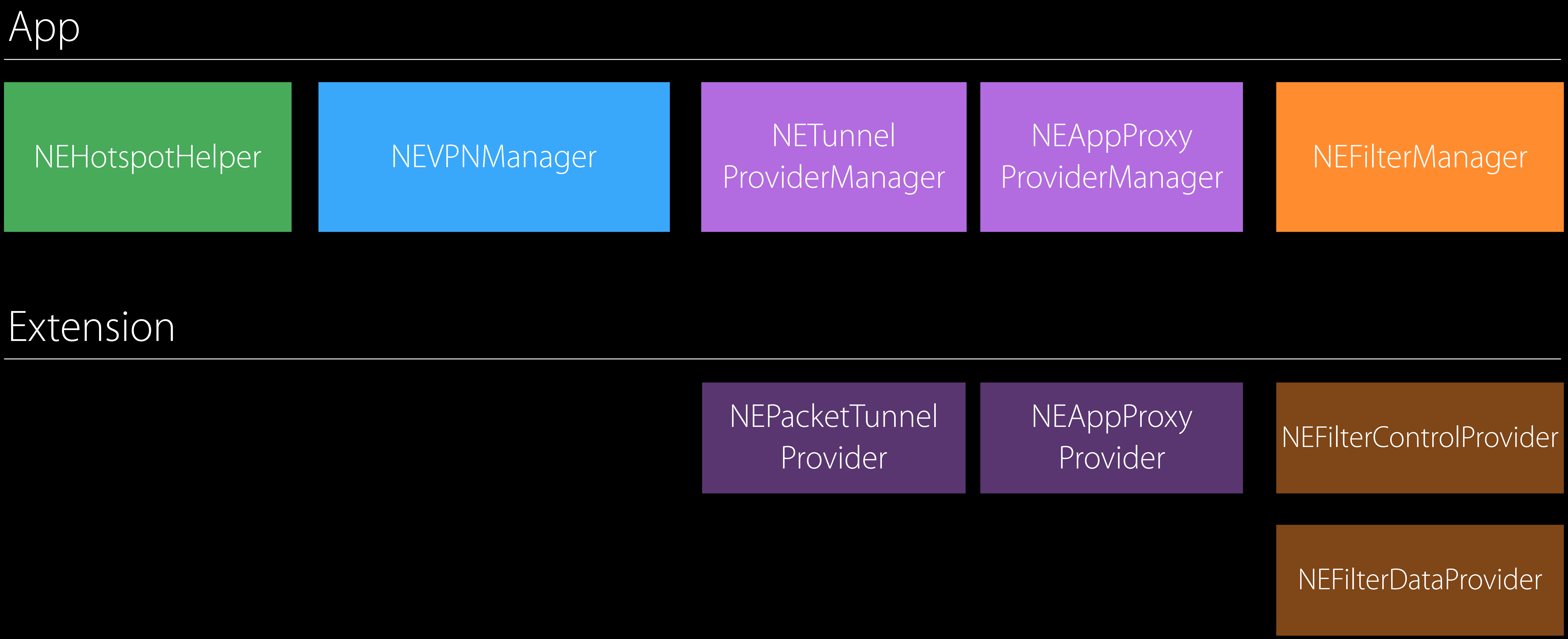
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App

NEHotspotHelper

NEVPNManager

NETunnel ProviderManager

NEAppProxy ProviderManager

NEFilterManager

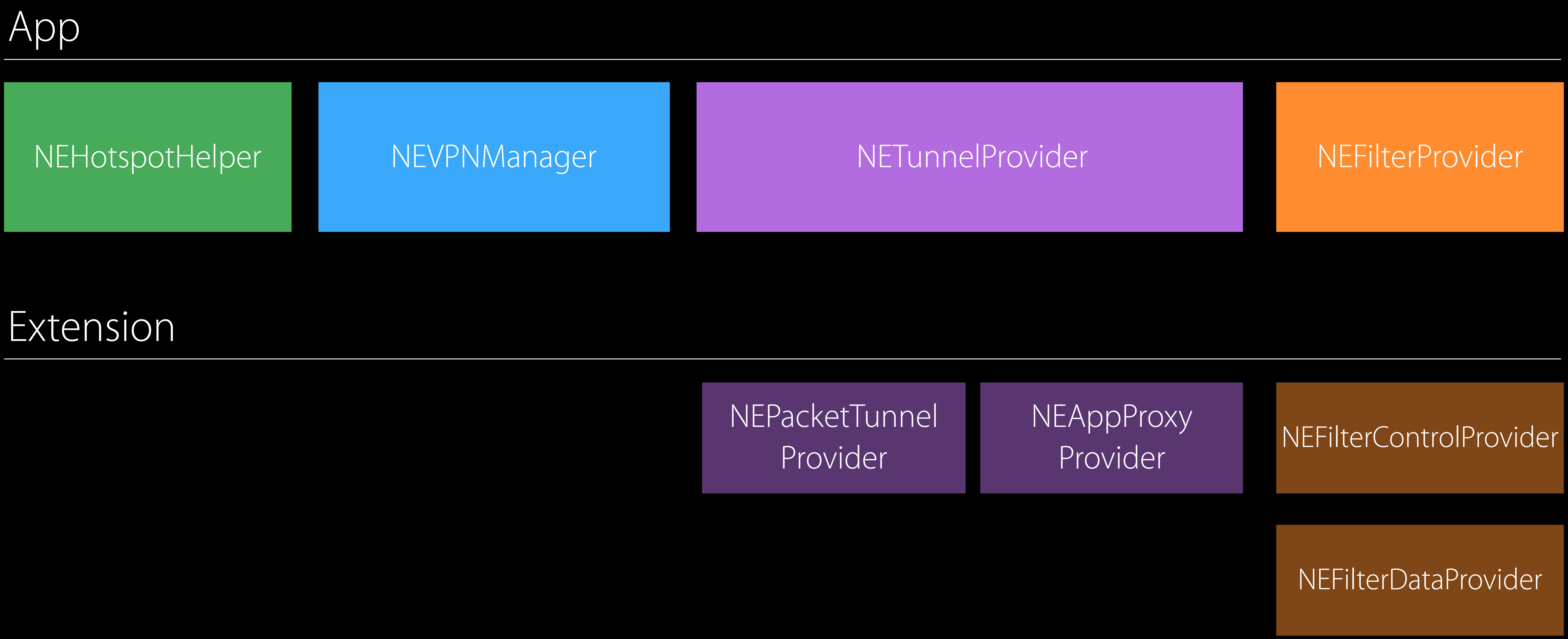
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



App

NEHotspotHelper

NEVPNManager

NETunnelProvider

NEFilterProvider

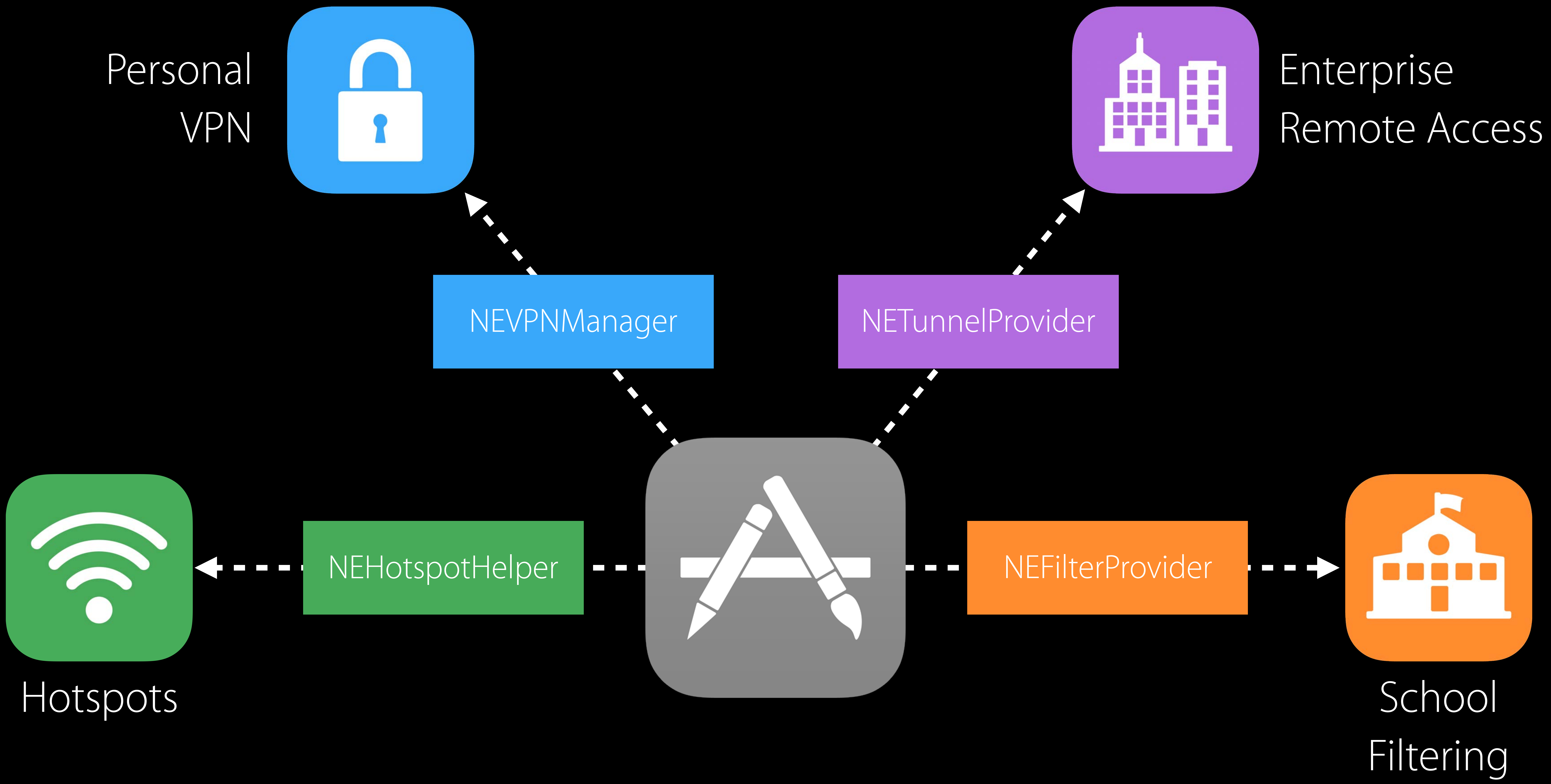
Extension

NEPacketTunnel Provider

NEAppProxy Provider

NEFilterControlProvider

NEFilterDataProvider



Personal VPN

Enterprise Remote Access

NEVPNManager

NETunnelProvider

Hotspots

NEHotspotHelper

NEFilterProvider

School Filtering

Using Network Extension APIs

Select the "Personal VPN" capability in Xcode

NEVPNManager

Special entitlements required

Send request to:

networkextension@apple.com

NEHotspotHelper

NETunnelProvider

NEFilterProvider

Summary

Summary

Port your existing Captive Network apps to use the NEHotspotHelper API

Summary

Port your existing Captive Network apps to use the NEHotspotHelper API

Adopt the NEVPNManager API in your personal VPN apps

Summary

Port your existing Captive Network apps to use the NEHotspotHelper API

Adopt the NEVPNManager API in your personal VPN apps

Use the NETunnelProvider APIs to create custom enterprise VPN apps

Summary

Port your existing Captive Network apps to use the NEHotspotHelper API

Adopt the NEVPNManager API in your personal VPN apps

Use the NETunnelProvider APIs to create custom enterprise VPN apps

Use MDM to deploy Per-App VPN for BYOD

Summary

Port your existing Captive Network apps to use the NEHotspotHelper API

Adopt the NEVPNManager API in your personal VPN apps

Use the NETunnelProvider APIs to create custom enterprise VPN apps

Use MDM to deploy Per-App VPN for BYOD

Use the NEFilterProvider APIs to create on-device network content filtering apps

More Information

Documentation
Network Extension Documentation
Coming Soon

Sample Code
SimpleTunnel Sample Code
Coming Soon

More Information

Technical Support

Apple Developer Forums
http://devforums.apple.com

Developer Technical Support
http://developer.apple.com/support/technical

General Inquiries

Paul Danbold, Core OS Evangelist
danbold@apple.com

Network Extension Permission Requests
networkextension@apple.com

Related Sessions

What's New in Managing Apple Devices	Nob Hill	Tuesday 11:00AM
Networking with NSURLSession	Pacific Heights	Thursday 9:00AM
Your App and Next Generation Networks	Mission	Friday 11:00AM

Labs

Networking Lab	Frameworks Lab B	Friday 1:30PM
Education and Enterprise Development and Deployment Lab	Frameworks Lab C	Friday 2:30PM

WWDC15