Advances in Networking

Part 1

Session 707

Stuart Cheshire, Apple DEST
David Schinazi, Apple Core Networking Engineer
Christoph Paasch, Apple Core Networking Engineer

Advances in Networking

Part 1

Explicit Congestion Notification

IPv6

Networking stack changes

New Network Extension facilities

Multipath protocols for multipath devices

Advances in Networking

Part 2

URLSession Adaptable Connectivity API

URLSessionTask scheduling API

URLSession enhancements

Best practices

Ongoing developments

ECN

Explicit Congestion Notification

ECN

Advantages of Explicit Congestion Notification

Any good transport protocol will maximize network usage

- To the point of congestion

Dropping packets is an expensive way to signal congestion

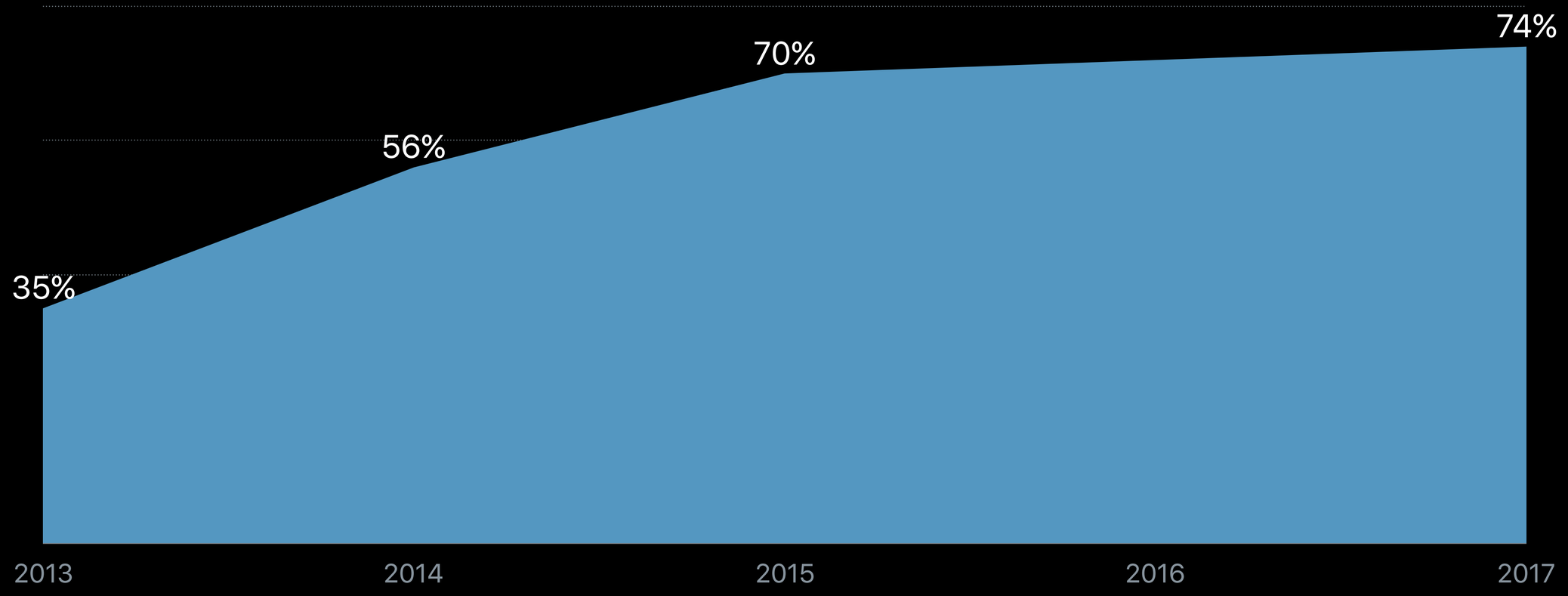
Marking packets “Congestion Experienced” is less destructive

- Reduces retransmissions
- Reduces delay
- Improves user experience

Requires SQM (Smart Queue Management) algorithm

Server Support for ECN

Alexa Top Million websites



Client Support for ECN

iOS, macOS, tvOS, watchOS

iOS 10.3 requests ECN for 50% of eligible TCP connections

- No problems reported from customers or service providers

We are seeing Congestion Experienced marking in multiple networks

- United States: 0.2%
- China: 1%
- Mexico: 3%
- France: 6%
- Argentina: 30%

Ongoing Deployment of ECN

Clients and servers

Clients: iOS 11 seed requests ECN for 100% of eligible TCP connections

- Wi-Fi and Ethernet
- Select Carriers—contact Apple to be added to the list

Servers: 74% of Alexa Top Million web sites support ECN

Internet is ready for network operators to deploy SQM+ECN at bottleneck links

- Immediate user experience improvement

IPv6

Making sure your apps keep up with the Internet

David Schinazi, Apple Core Networking Engineer

World IPv6 Launch 5 Years Ago Yesterday

6/6/2012

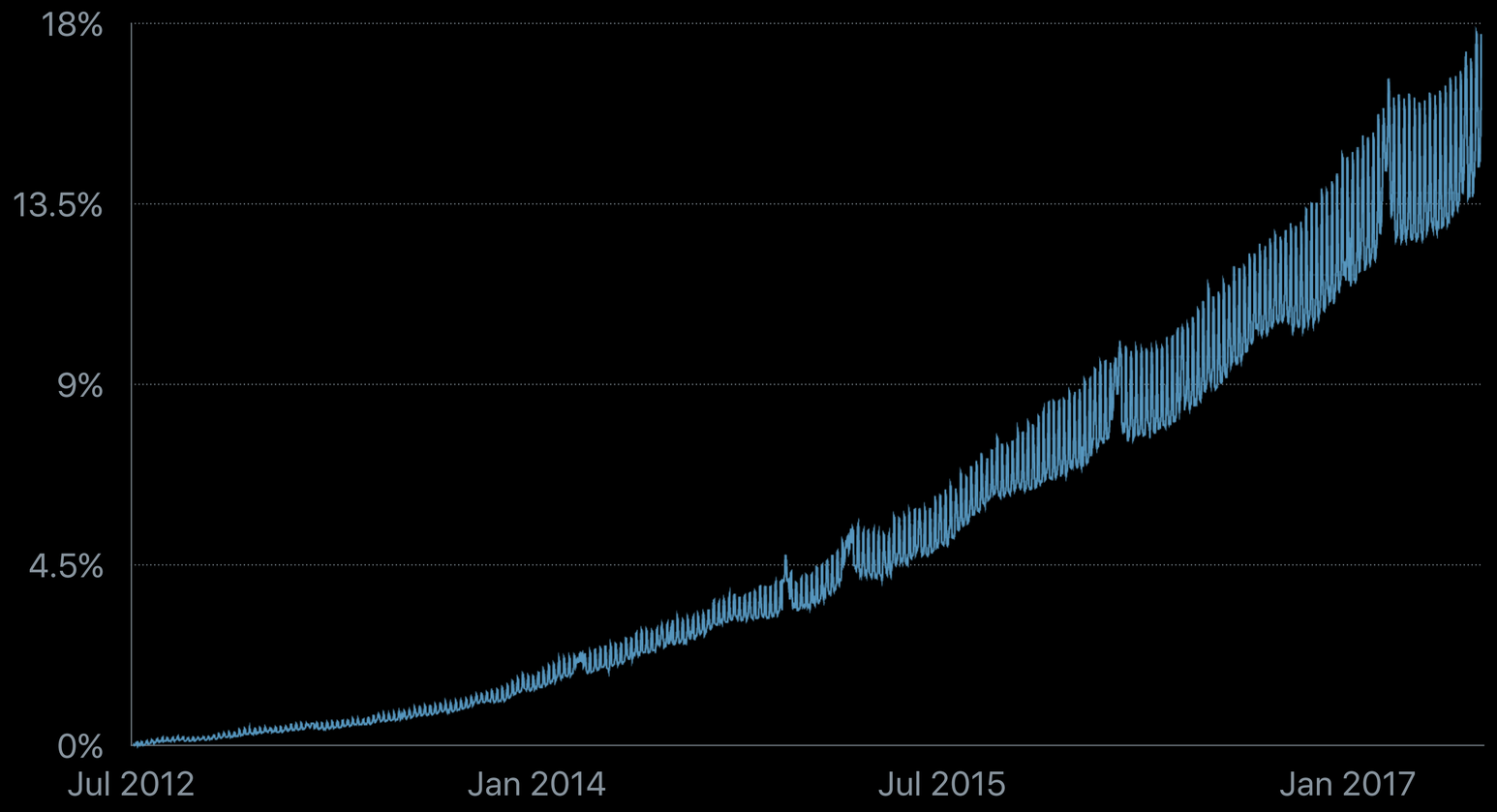
Belgium: 49%

United States: 32%

United Kingdom: 17%

India: 23%

Japan: 18%

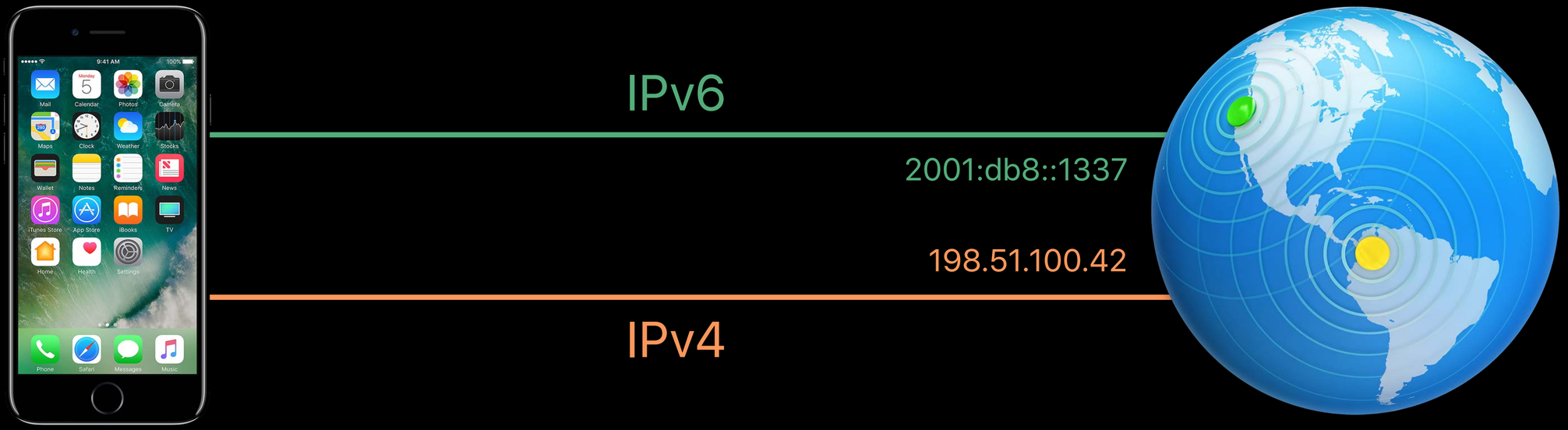


Data Credit: Google, Inc.

Dual-Stack Connectivity More Prevalent

Your app most likely has IPv6 connectivity

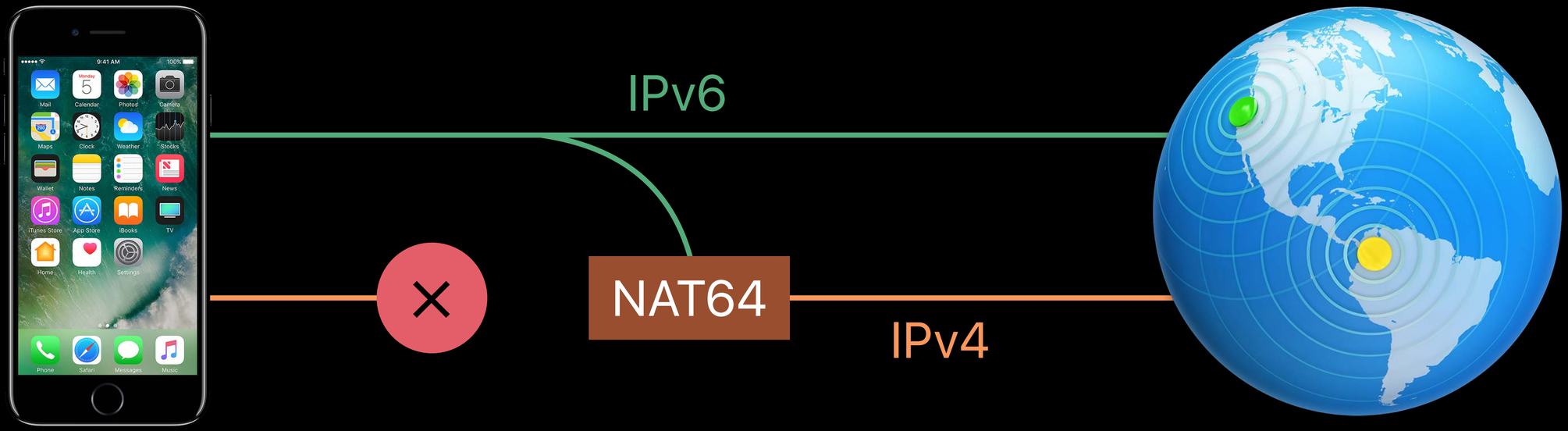
HTTPS request times are still 15-30% faster



NAT64 Connectivity is Emerging

One major carrier only offers IPv6

Your app needs to work without IPv4 addresses



Best Practices

Recap from WWDC 15 and 16

Use higher-layer networking frameworks

- URLSession, CFNetwork

Avoid

- Legacy IPv4-only APIs
- Direct use of IPv4 addresses
- Preflight checks

App Review Enforcement

IPv6 (and NAT64) compatibility is an App Store submission requirement

- All apps are tested on a NAT64 network

Rejections now very rare

Check out the developer website

- Supporting IPv6 DNS64/NAT64 Networks

Your App and Next Generation Networks	WWDC 2015
Networking for the Modern Internet	WWDC 2016

User-Space Networking

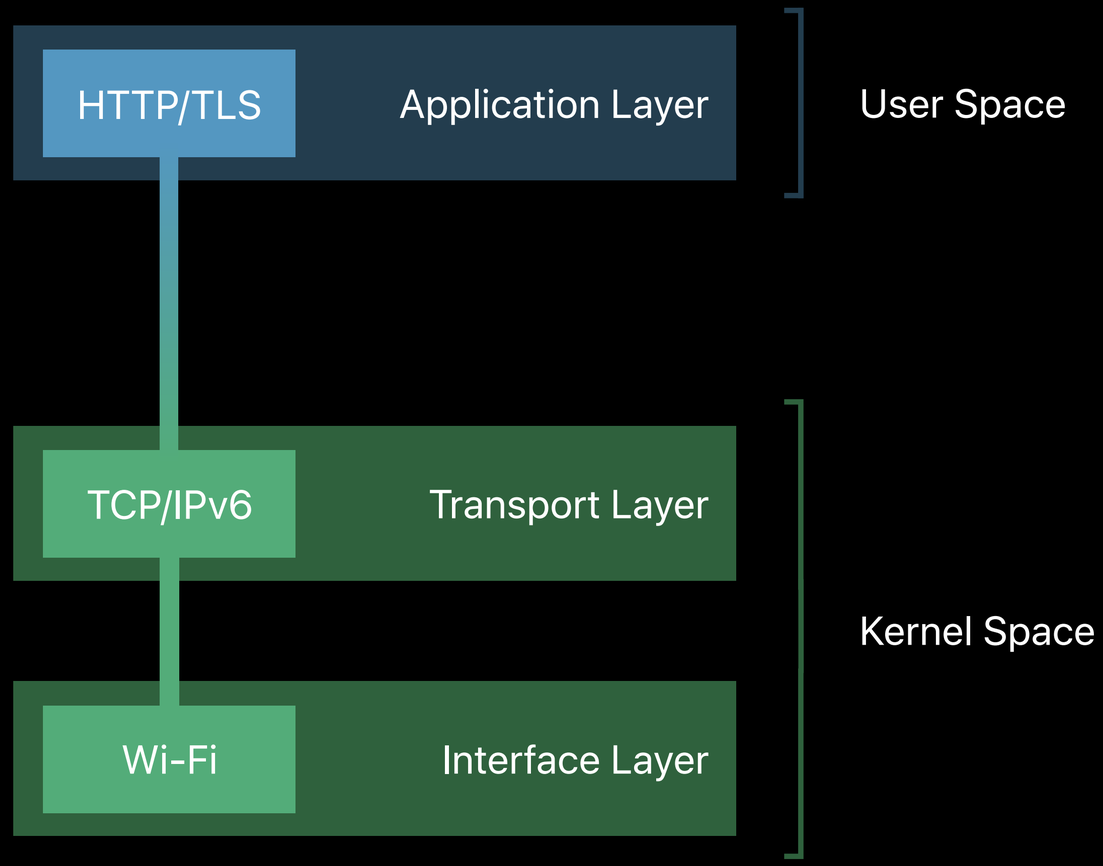
Network Stack Evolution

Traditional Model

Transport protocols in the kernel

Remaining protocols handled in apps

Context switch between kernel and user space to transfer data

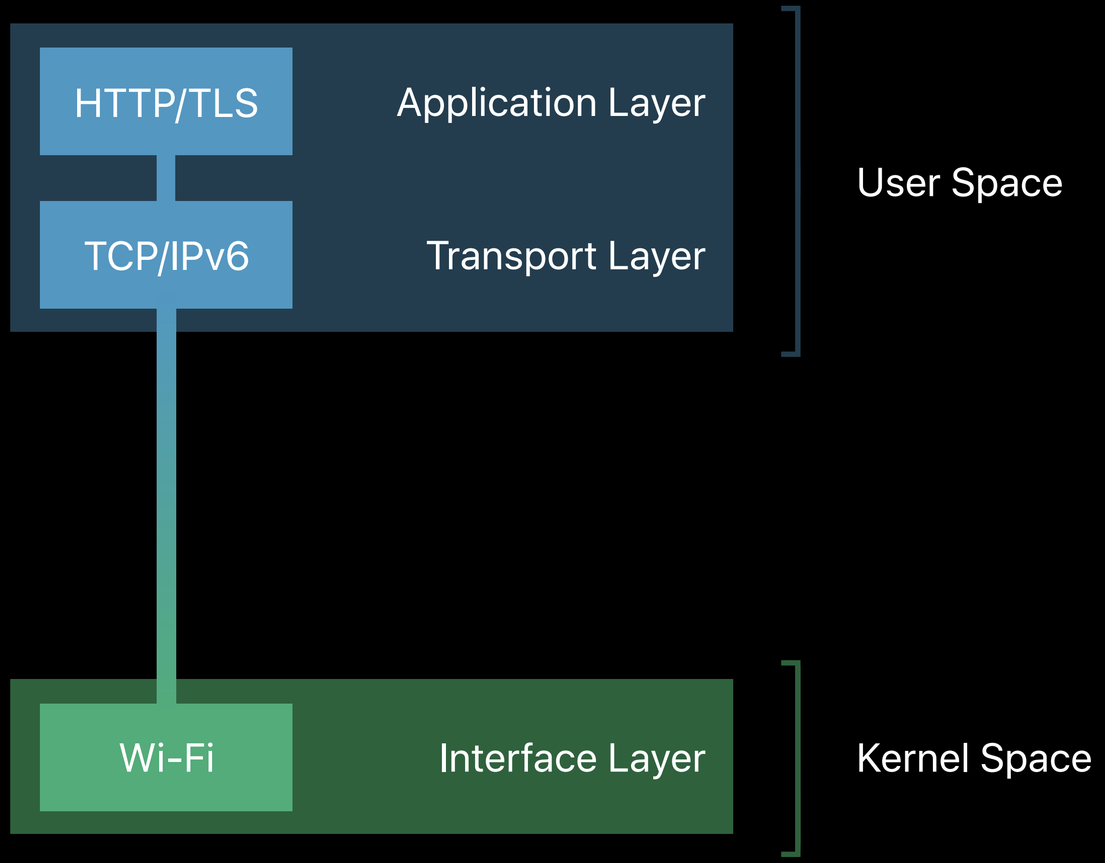


Network Stack Evolution

User-Space Networking

NEW

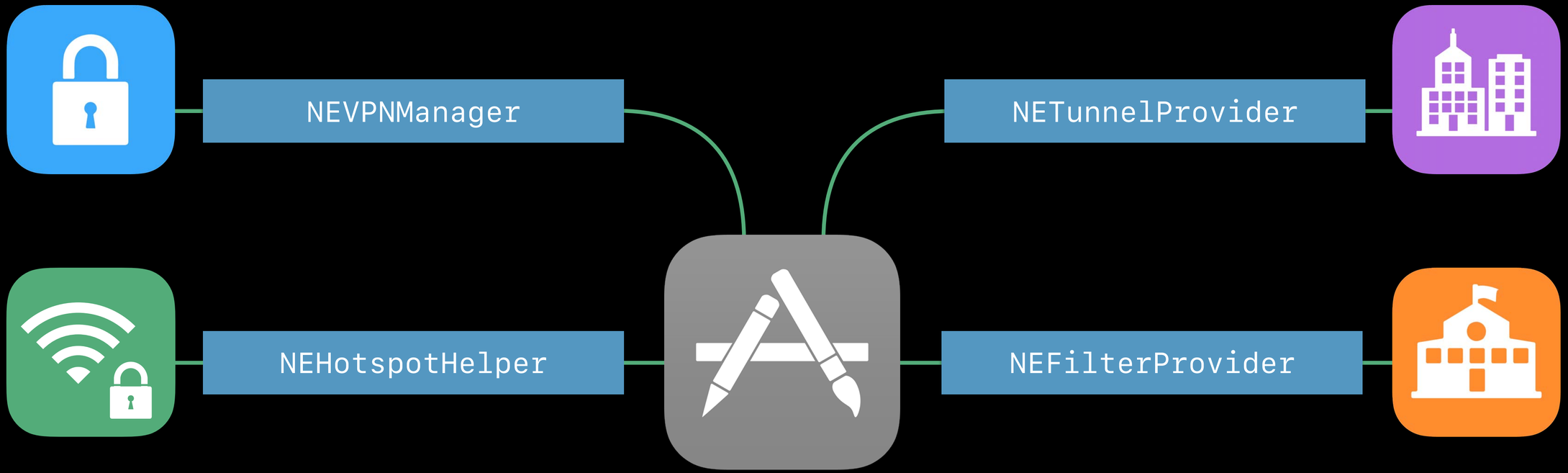
- Protocol stack unified within app
- Improved system efficiency
- Available when using URLSession on iOS, watchOS, tvOS
- Not available when using BSD sockets
- Network Kernel Extensions will be deprecated in a future release



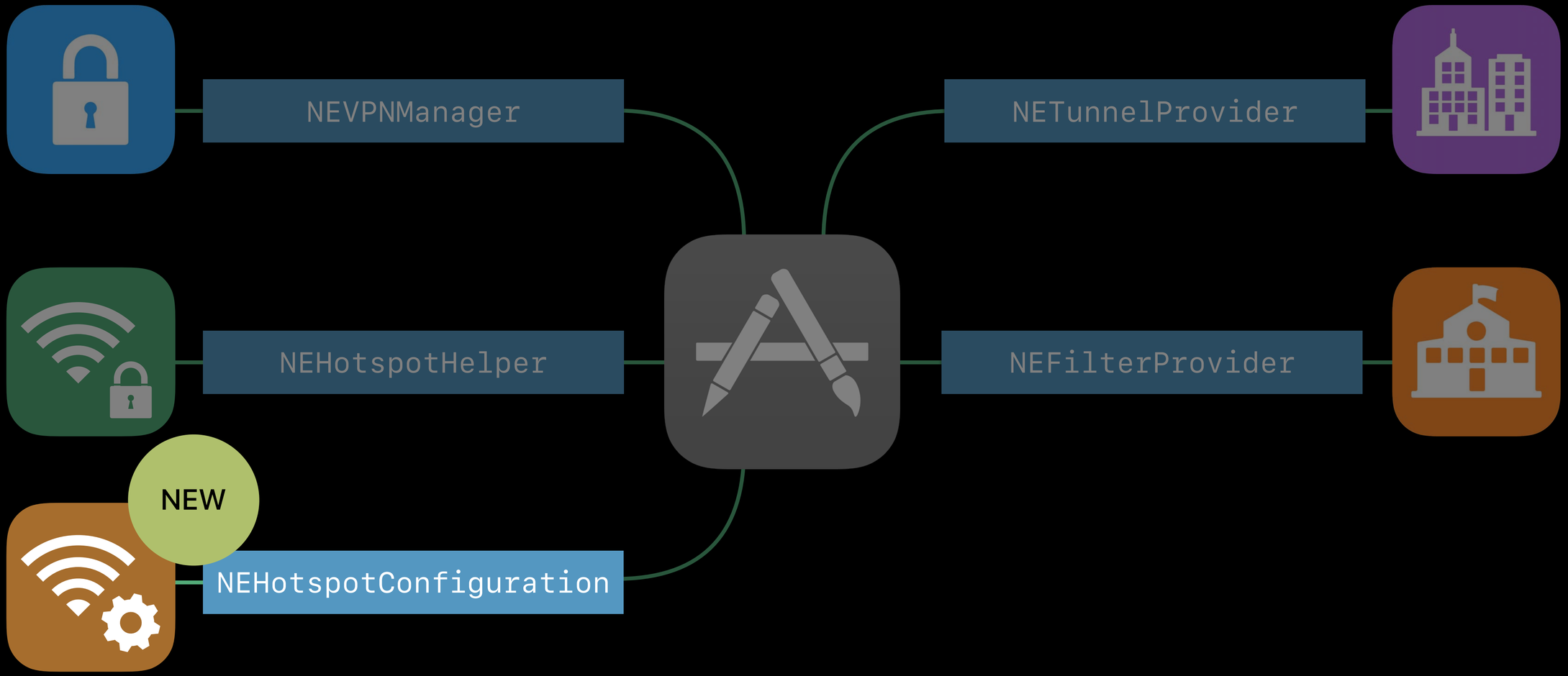
Network Extension

New APIs for Wi-Fi Configuration and DNS Proxy

Network Extension

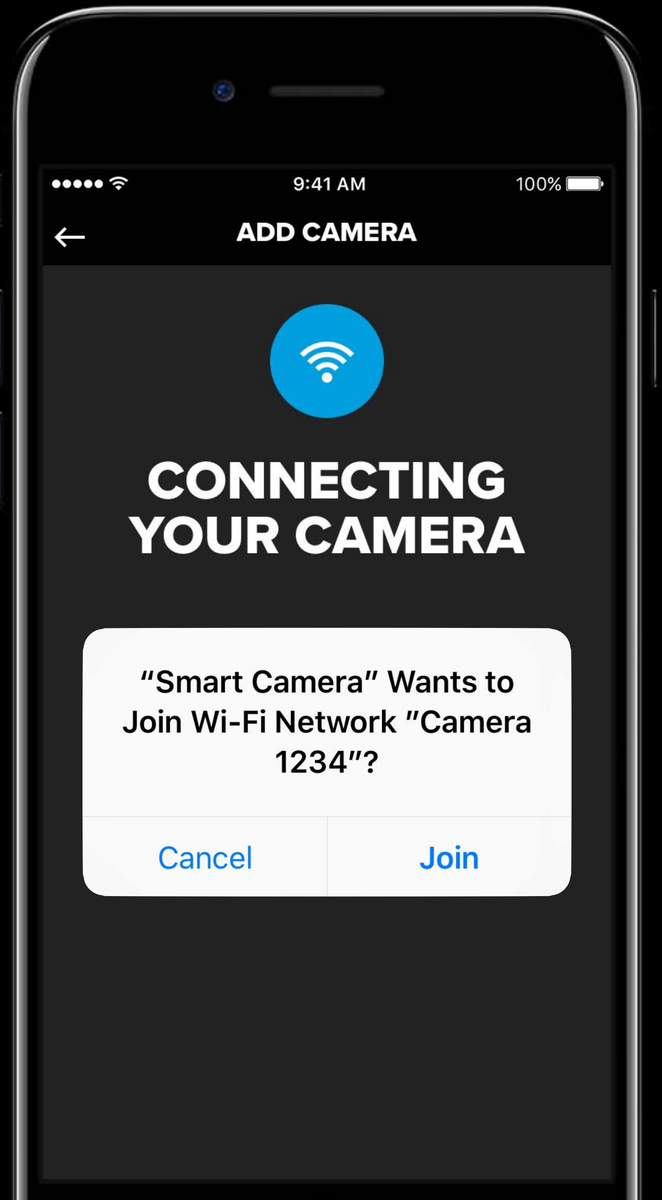


Network Extension



Simplify Setting up Smart Devices

Simplify Setting up Smart Devices



NEHotspotConfiguration

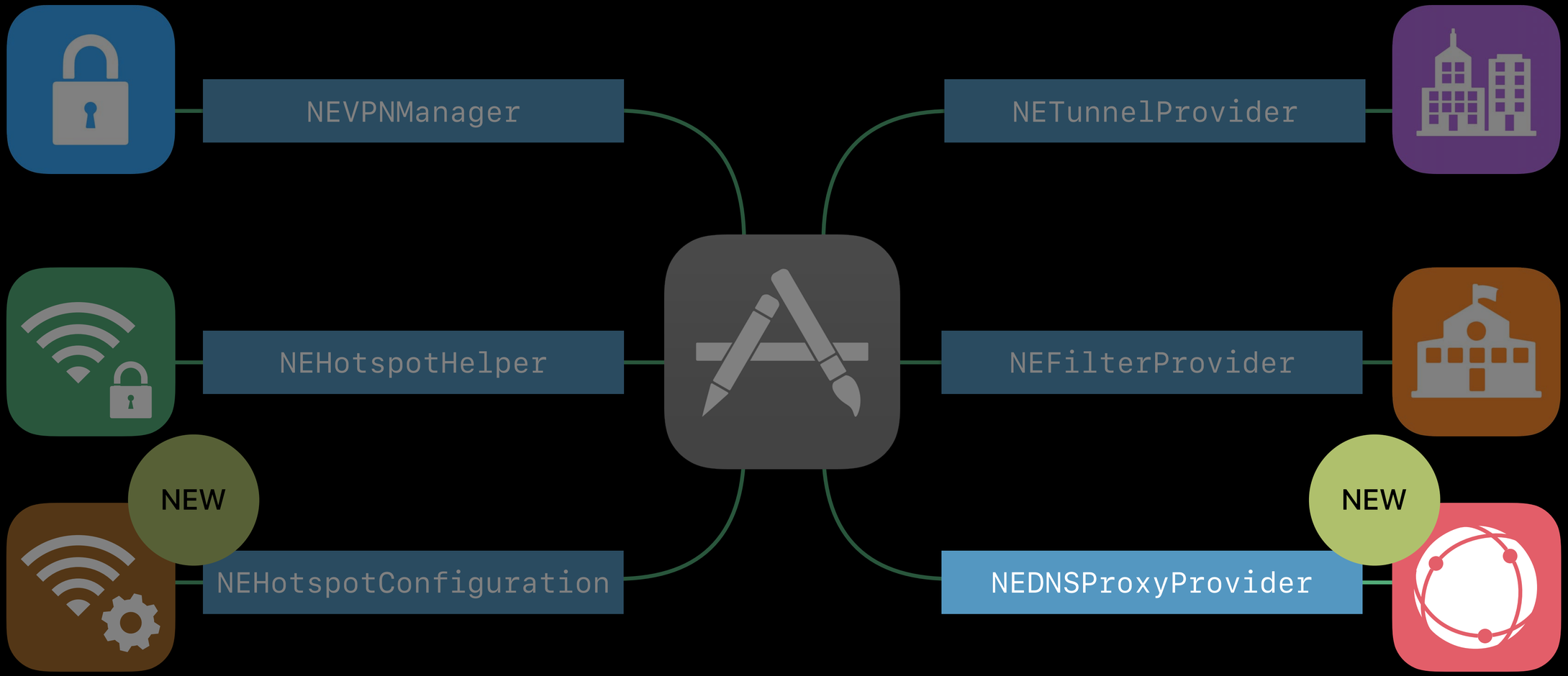
Simplifies connections to Wi-Fi networks

Can be temporary or persistent

Supports authentication (Open, WEP, WPA, EAP, Hotspot 2.0)

```swift
// Network Extension Wi-Fi Configuration API

import NetworkExtension

let cameraWiFiConfig = NEHotspotConfiguration(ssid: "Camera 1234",
                                              passphrase: "correcthorsebatterystaple",
                                              isWEP: false)
cameraWiFiConfig.joinOnce = true

NEHotspotConfigurationManager.shared.apply(cameraWiFiConfig) { error in
    // Handle error or success
}

let caffeWiFiConfig = NEHotspotConfiguration(ssid: "I Love Coffee")

NEHotspotConfigurationManager.shared.apply(caffeWiFiConfig) { error in
    // Handle error or success
}
```

Network Extension



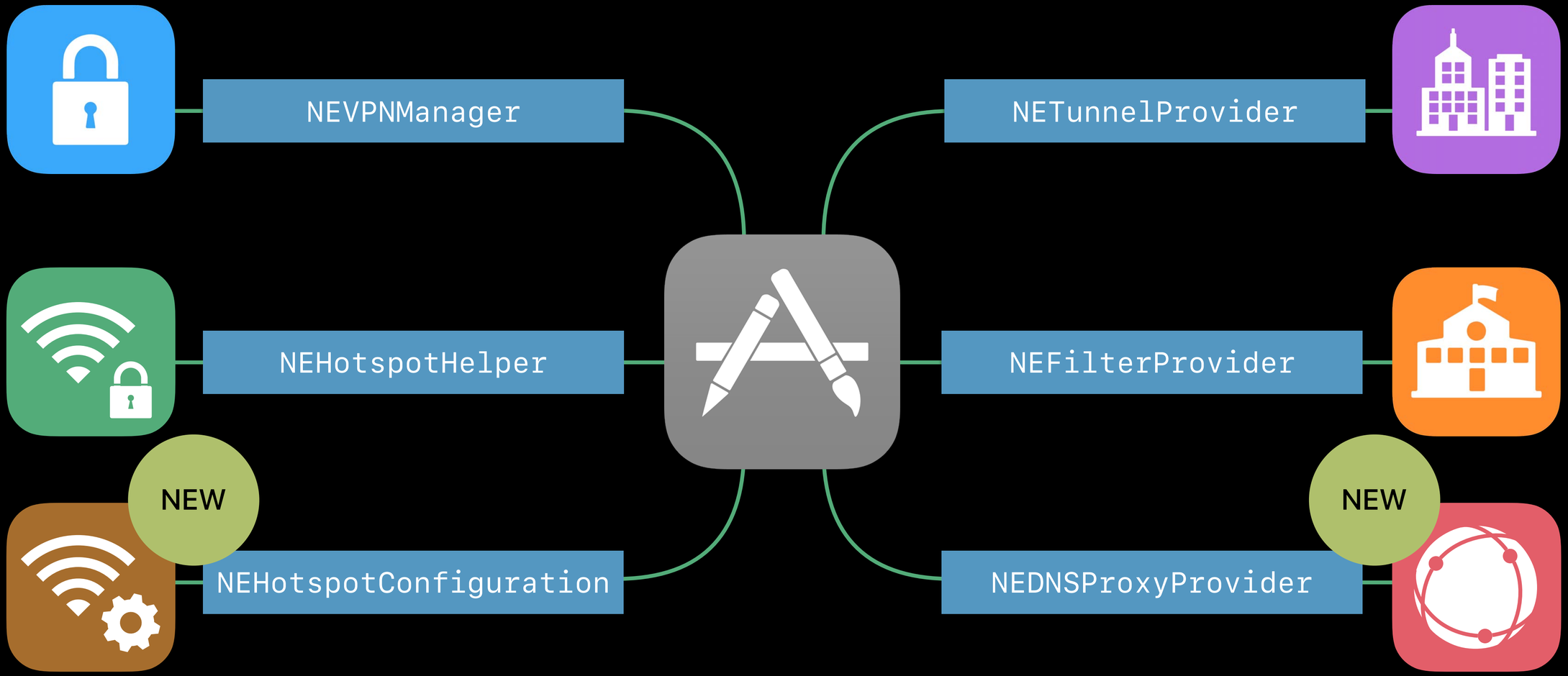
NEDNSProxyProvider

Receives the system’s DNS query messages

Handles them as it wishes

- Can send to recursive resolver of its choice
- Can send using protocol of its choice
  - DNS over TLS
  - DNS over HTTP

Network Extension



Multipath Protocols for Mobile Devices

Wi-Fi Assist and Multipath Transport Protocols

Christoph Paasch, Apple Core Networking Engineer

Multipath Protocols for Mobile Devices

Internet access in the mobile world

- Today’s protocols only use one interface

Wi-Fi Assist and Multipath Transport Protocols

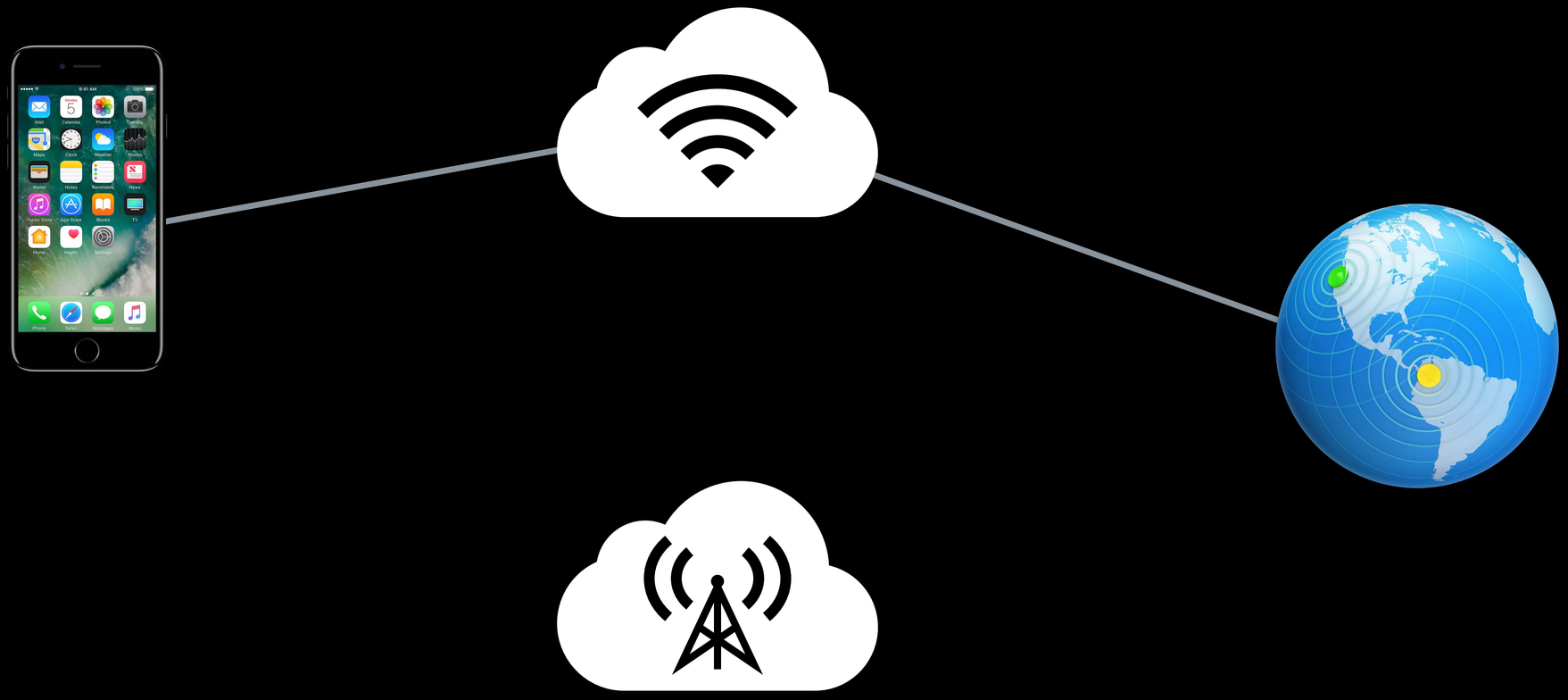
- Improve reliability and user experience

Public API for Multipath TCP in iOS 11

Internet Access in the Mobile World

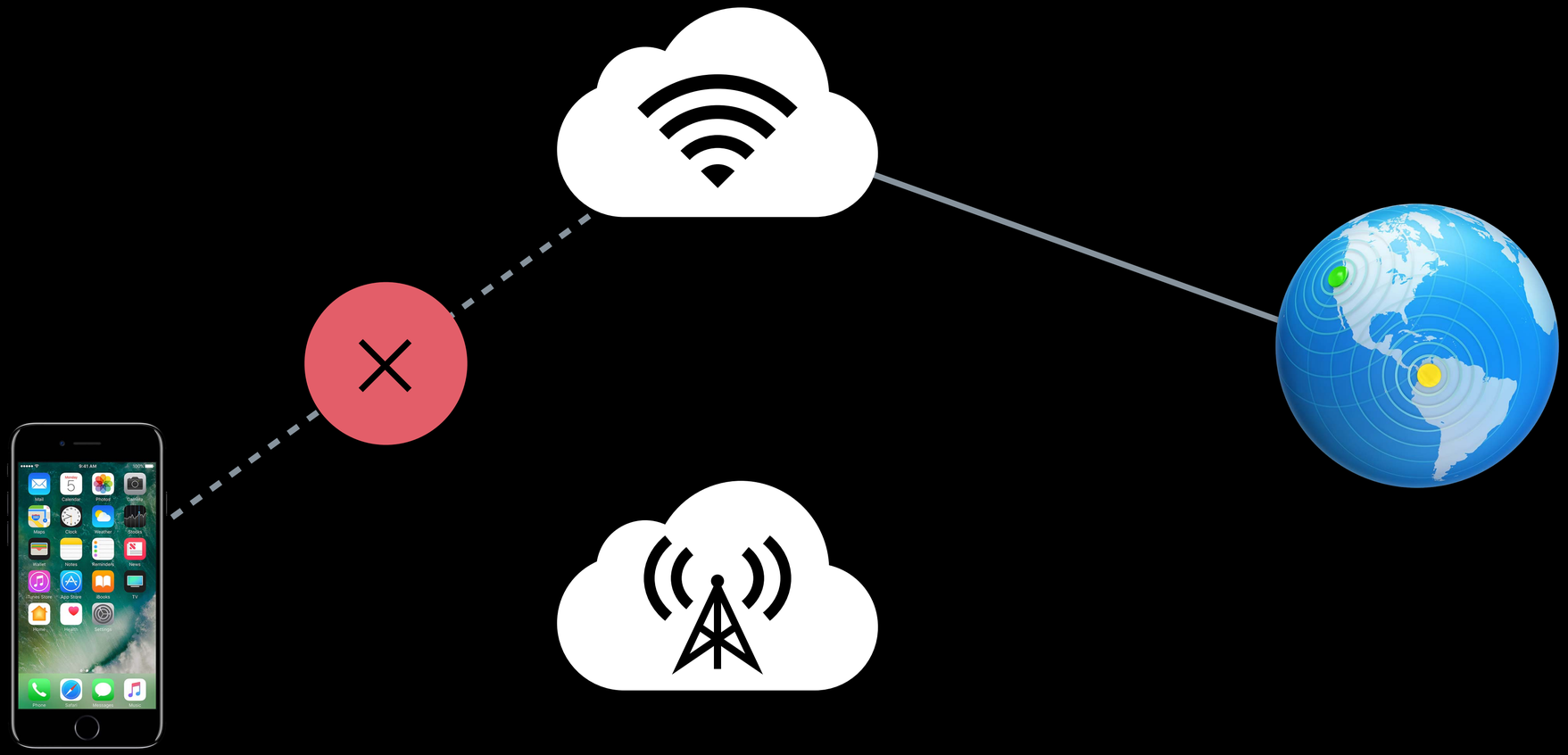
Internet Access in the Mobile World

Leaving home



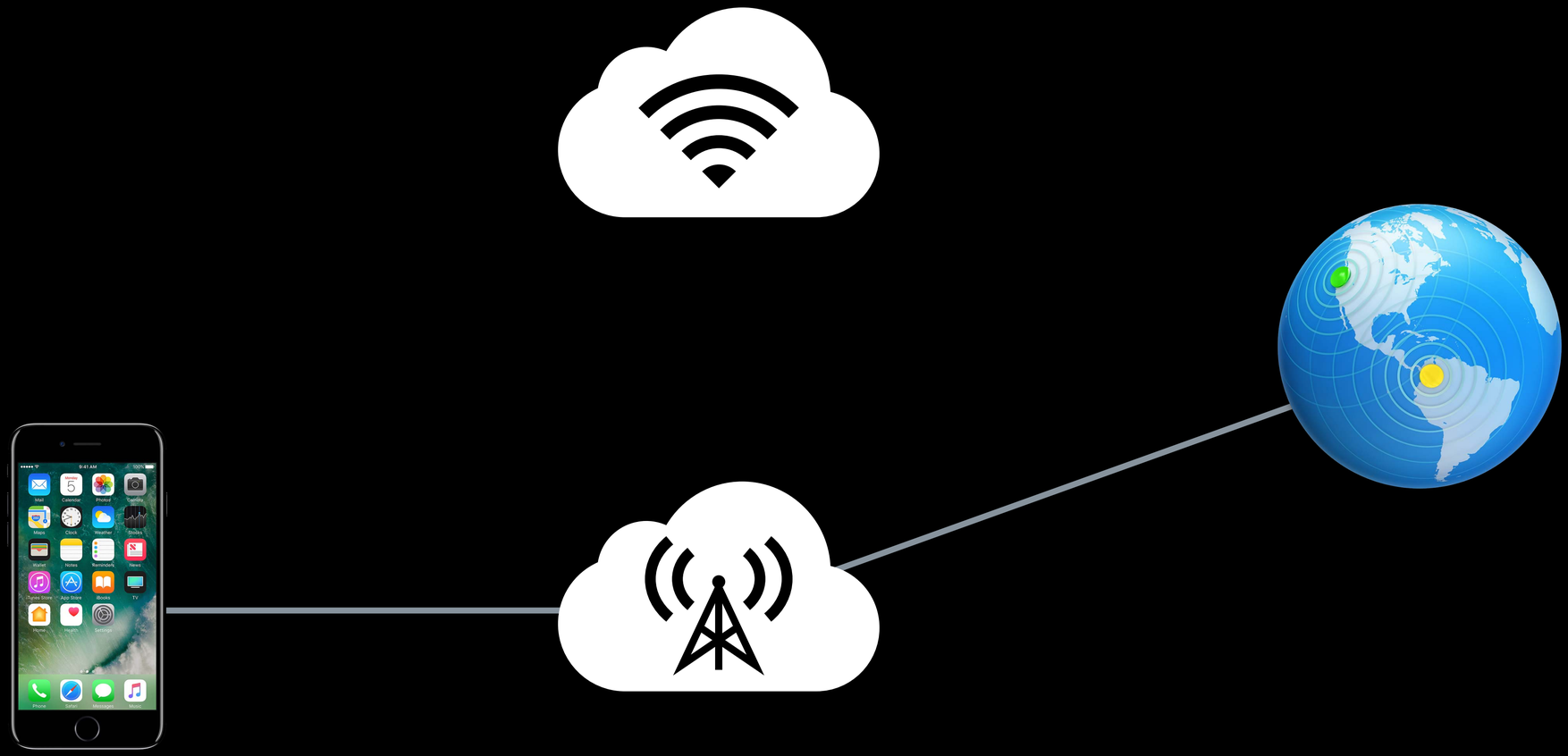
Internet Access in the Mobile World

Leaving home



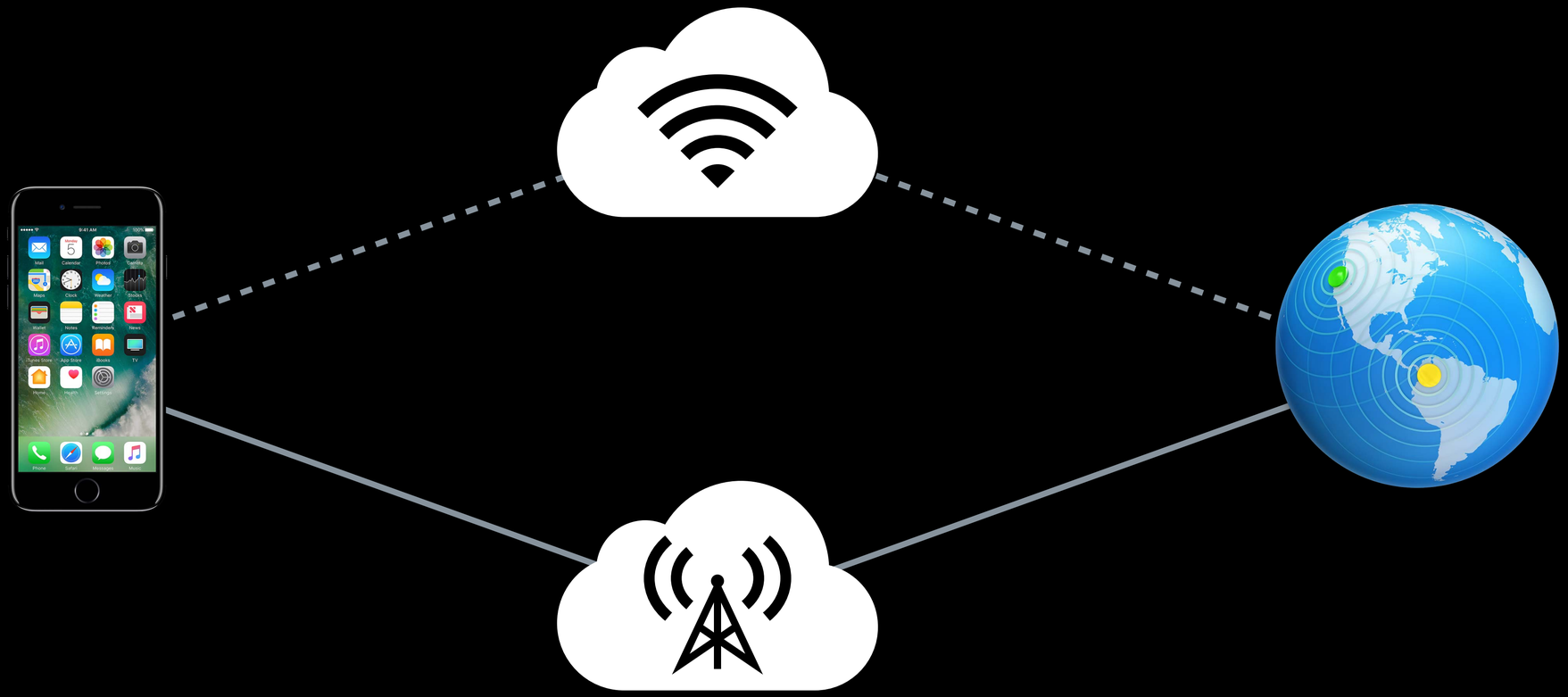
Internet Access in the Mobile World

Leaving home



Internet Access in the Mobile World

Poor Wi-Fi



Wi-Fi Assist

Choosing the right interface

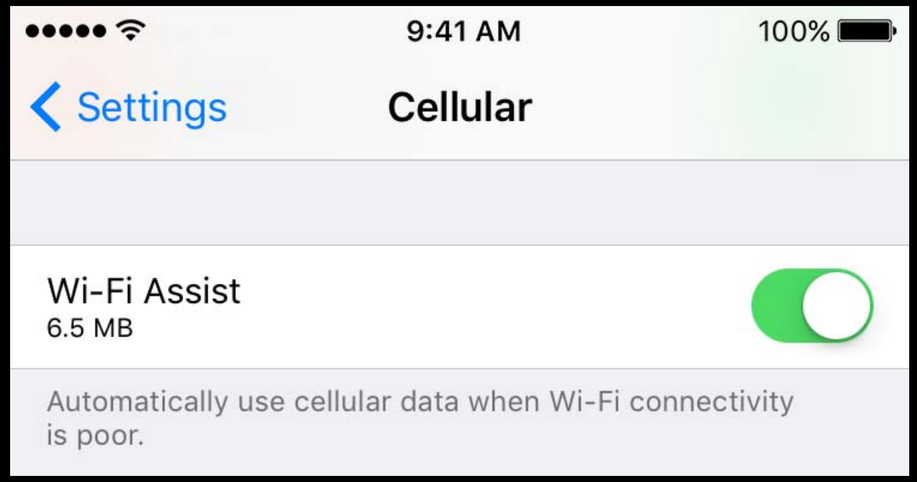
Wi-Fi Assist

Triggered by

- Marginal Wi-Fi

"Fittest Wins Out" contest between Wi-Fi and Cell

- Wi-Fi has head start over Cell
- On a flow by flow basis, at flow setup time



Your App and Next Generation Networks	WWDC 2015
Networking for the Modern Internet	WWDC 2016

Multipath TCP

An end-to-end transport for mobile devices

Multipath TCP

Multipath TCP (MPTCP—RFC 6824 “TCP Extensions for Multipath Operation”)

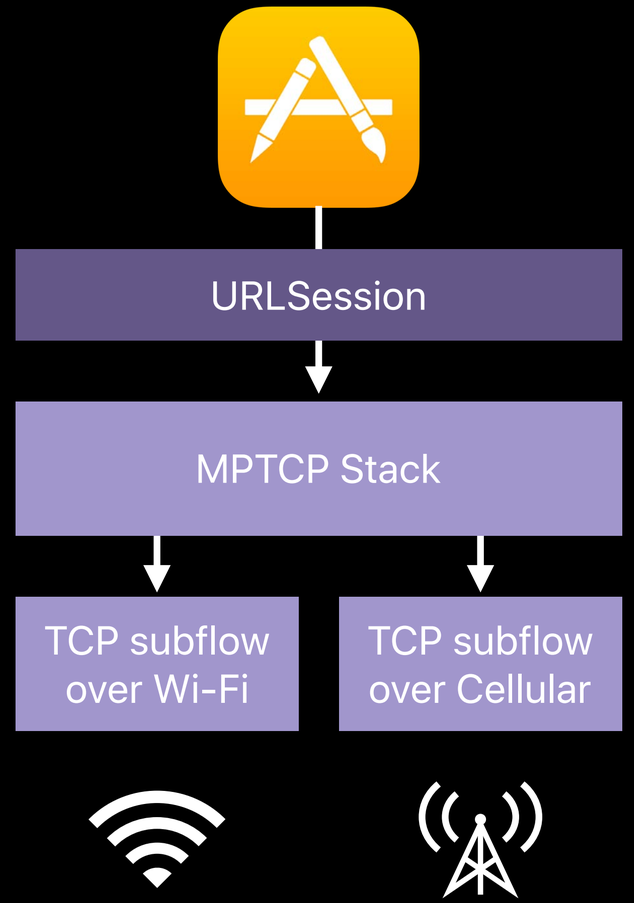
- Built on top of TCP
  - Reliability
  - Congestion control
- Seamless handover from Wi-Fi to Cell
- Chooses optimal interface for latency-sensitive flows

Multipath TCP

MPTCP schedules traffic across the interfaces

One “TCP subflow” per interface

MPTCP creates/destroys subflows

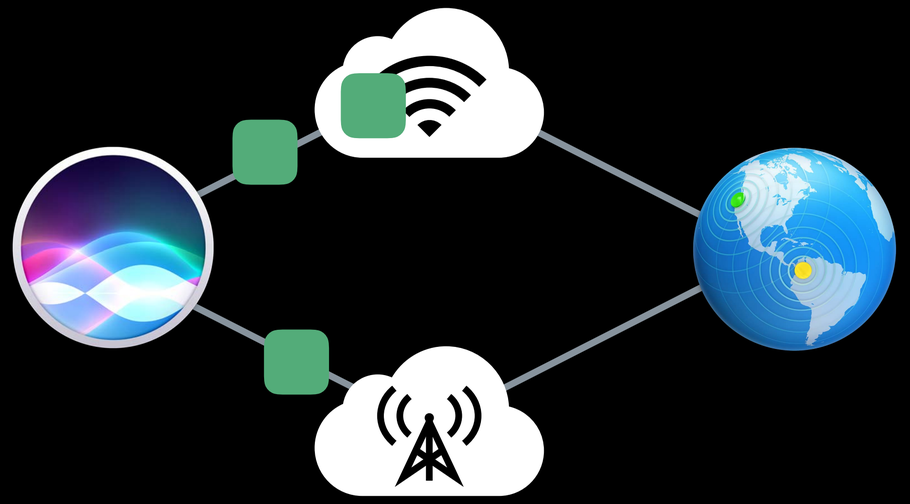


Multipath TCP at Apple

Implemented since iOS 7 for Siri

User feedback (time to first word)
20% faster in the 95th percentile

5x reduction in network failures



NEW

Multipath TCP in iOS 11

New public API

Multipath TCP in iOS 11

Server support

Multipath service types

URLSession API

Multipath TCP

Server support

Multipath TCP on your Servers

Requires MPTCP-capable servers

New Linux kernel

- https://multipath-tcp.org

AWS and GCE images available

Multipath TCP on your Servers

Multipath TCP

Choosing the Multipath Service Type

Multipath TCP in iOS 11

NEW

Public API in iOS 11 to enable MPTCP

- Handover Mode for high reliability
- Interactive Mode for low latency

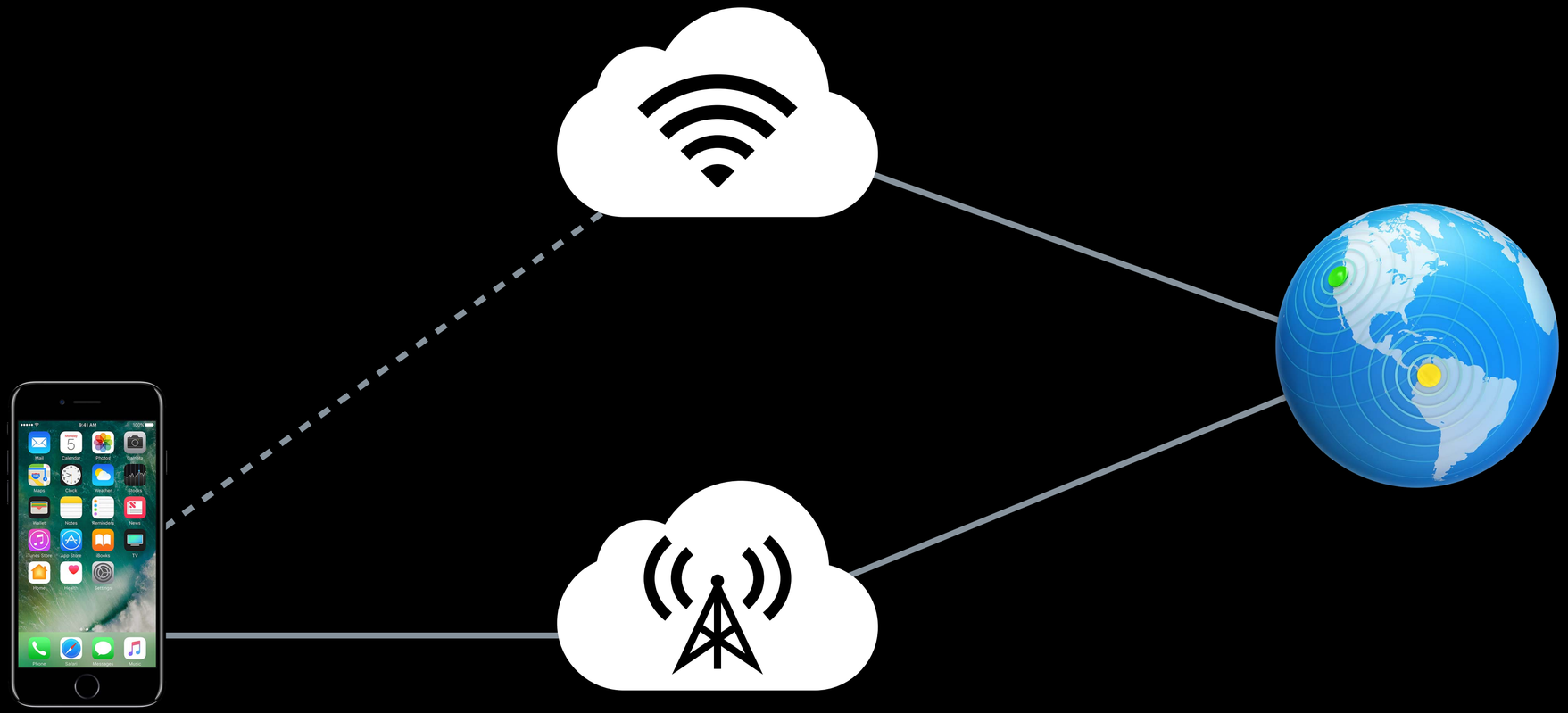
Handover Mode

From Wi-Fi to cellular and back

Multipath TCP in iOS 11

Handover Mode

NEW



Multipath TCP in iOS 11

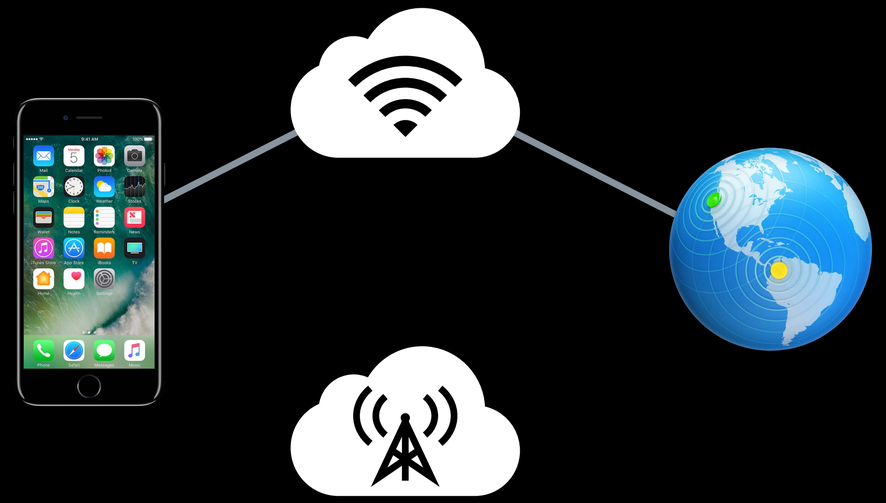
Handover Mode

NEW

Reliability for persistent connections

Minimal cell usage

Available in Beta 1



Interactive Mode

Reducing latency for our end users

Multipath TCP in iOS 11

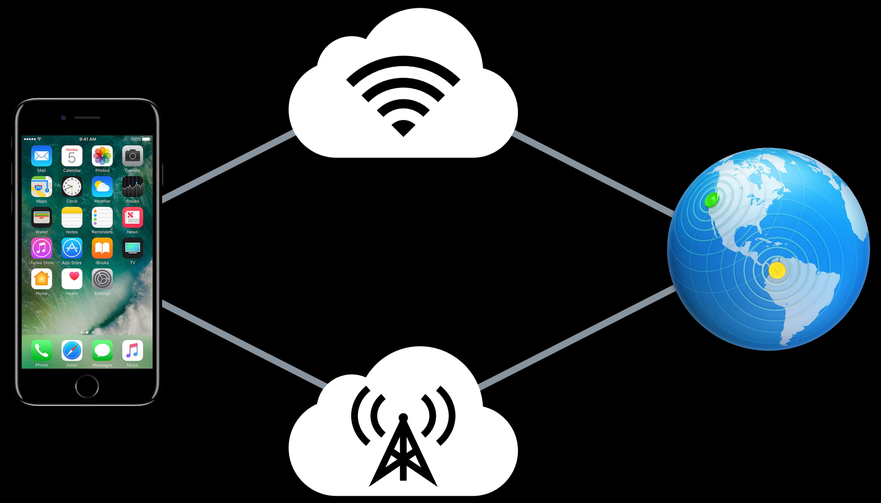
Interactive Mode

NEW

Low latency for low-volume interactive flows

Wi-Fi and cellular

Available in an upcoming Beta



Multipath TCP in iOS 11

URLSessionConfiguration property

Multipath TCP in iOS 11

URLSession

NEW

Exposed as URLSessionConfiguration property

```
var multipathServiceType: URLSessionConfiguration.MultipathServiceType

    none = 0,
    handover = 1,
    interactive = 2,
```

Add Capability “Multipath” in Xcode

Aggregation Mode

Available for experimentation

Aggregation Mode

Combines link capacities

Available through developer settings

Starting in an upcoming Beta

Multipath Protocols for Mobile Devices

Wi-Fi Assist provides better networking on mobile devices

iOS 11 public API for Multipath TCP

Seamless handover between Wi-Fi and Cellular

Most efficient interface for data transfer

More Information

Part 1
https://developer.apple.com/wwdc17/707
Part 2
https://developer.apple.com/wwdc17/709