# Content Protection for HTTP Live Streaming

Session 502

Roger Pantos HTTP Live Streaming Engineer

# FairPlay Streaming

# Overview of FairPlay Streaming (FPS)

# Overview of FairPlay Streaming (FPS)

Industrial-strength protection for your HTTP Live Streaming audio & video

# Overview of FairPlay Streaming (FPS)

Industrial-strength protection for your HTTP Live Streaming audio & video

Already delivering keys in the premium content industry

# Overview of FairPlay Streaming (FPS)

Industrial-strength protection for your HTTP Live Streaming audio & video

Already delivering keys in the premium content industry

Built into iOS, Apple TV, and OS X

# Overview of FairPlay Streaming (FPS)

Industrial-strength protection for your HTTP Live Streaming audio & video

Already delivering keys in the premium content industry

Built into iOS, Apple TV, and OS X

Power-efficient on mobile devices

# Overview of FairPlay Streaming (FPS)

Industrial-strength protection for your HTTP Live Streaming audio & video

Already delivering keys in the premium content industry

Built into iOS, Apple TV, and OS X

Power-efficient on mobile devices

Integrated with AirPlay

# Overview of FairPlay Streaming (FPS)

Industrial-strength protection for your HTTP Live Streaming audio & video

Already delivering keys in the premium content industry

Built into iOS, Apple TV, and OS X

Power-efficient on mobile devices

Integrated with AirPlay

Offered under the Apple Developer Program License Agreement

# Scope of FairPlay Streaming—What It Is

FairPlay Streaming is:

# Scope of FairPlay Streaming—What It Is

FairPlay Streaming is:

- A secure key delivery mechanism
  - Content Key is protected on the network and on the client during playback

# Scope of FairPlay Streaming—What It Is

FairPlay Streaming is:

- A secure key delivery mechanism
  - Content Key is protected on the network and on the client during playback
- Key delivery is transport agnostic
  - Easy to integrate with existing key server infrastructure

# Scope of FairPlay Streaming—What It Is

FairPlay Streaming is:

- A secure key delivery mechanism
  - Content Key is protected on the network and on the client during playback
- Key delivery is transport agnostic
  - Easy to integrate with existing key server infrastructure
- Requires protected HDMI for external output

# Scope of FairPlay Streaming—What It Isn't

FairPlay Streaming does NOT:

- Provide DRM rights expression or policy enforcement, or
- Provide user authentication or per-device authorization

These can be implemented separately and combined with FPS

# How to Use FairPlay Streaming

# What Do You Need to Do?

# What Do You Need to Do?

Integrate a FairPlay Streaming Key Security Module (KSM) into your key server

# What Do You Need to Do?

Integrate a FairPlay Streaming Key Security Module (KSM) into your key server

Add code to your app to relay key requests and responses

# What Do You Need to Do?

Integrate a FairPlay Streaming Key Security Module (KSM) into your key server

Add code to your app to relay key requests and responses

For each HLS asset that you wish to protect:

- Generate and store a Content Key (CK) in your back-end database
- Encrypt the asset using AES Sample encryption
- Put a reference to the CK into your HLS playlist

# Designing a FairPlay Streaming System

Gianpaolo Fasoli
FairPlay Streaming Engineer

# Designing a FairPlay Streaming System

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

Building blocks and data flows

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

Building blocks and data flows

What we provide, what you have to build

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

Building blocks and data flows

What we provide, what you have to build

Integrating FPS into your Key Server

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

Building blocks and data flows

What we provide, what you have to build

Integrating FPS into your Key Server

Testing your Key Security Module

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

Building blocks and data flows

What we provide, what you have to build

Integrating FPS into your Key Server

Testing your Key Security Module

Integrating FPS into your app

# Designing a FairPlay Streaming System

Purpose and importance of your credentials

Building blocks and data flows

What we provide, what you have to build

Integrating FPS into your Key Server

Testing your Key Security Module

Integrating FPS into your app

Encrypting and testing your content

# FairPlay Streaming Credentials

KSM Credentials differentiate you from other FPS deployments

# FairPlay Streaming Credentials

KSM Credentials differentiate you from other FPS deployments

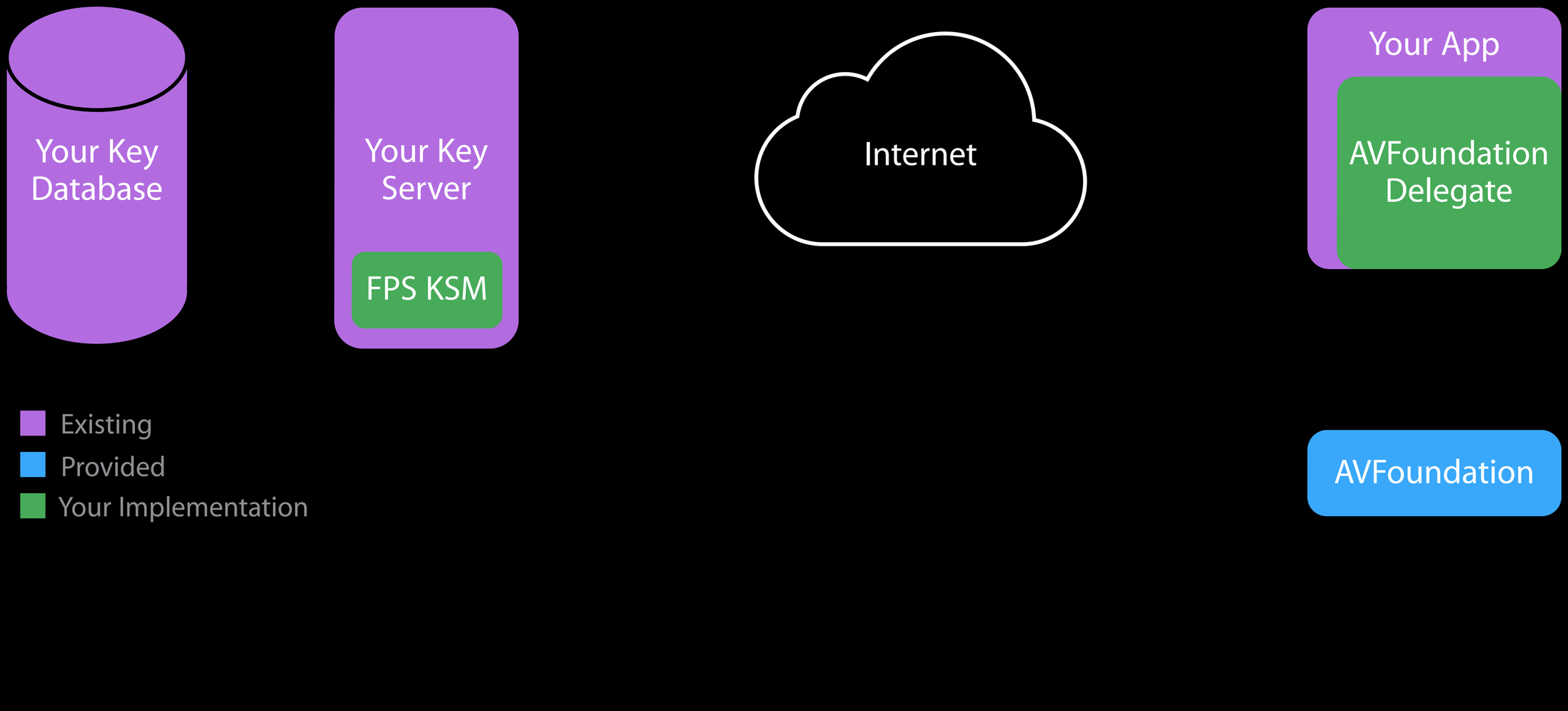
- **Playing content on a customer device requires production credentials**

# FairPlay Streaming Credentials

KSM Credentials differentiate you from other FPS deployments

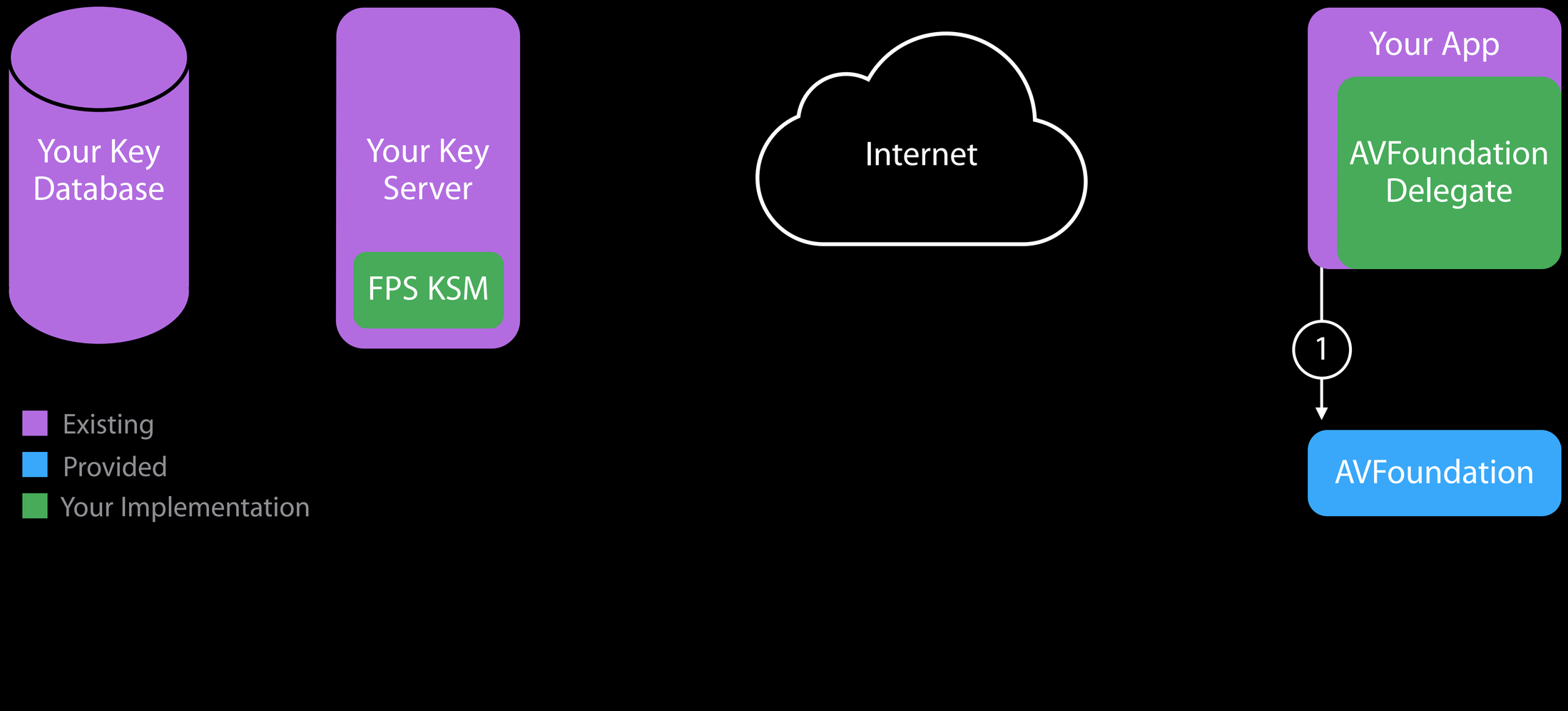
- **Playing content on a customer device requires production credentials**
- You must protect your production credentials

# FairPlay Streaming—Request Flow



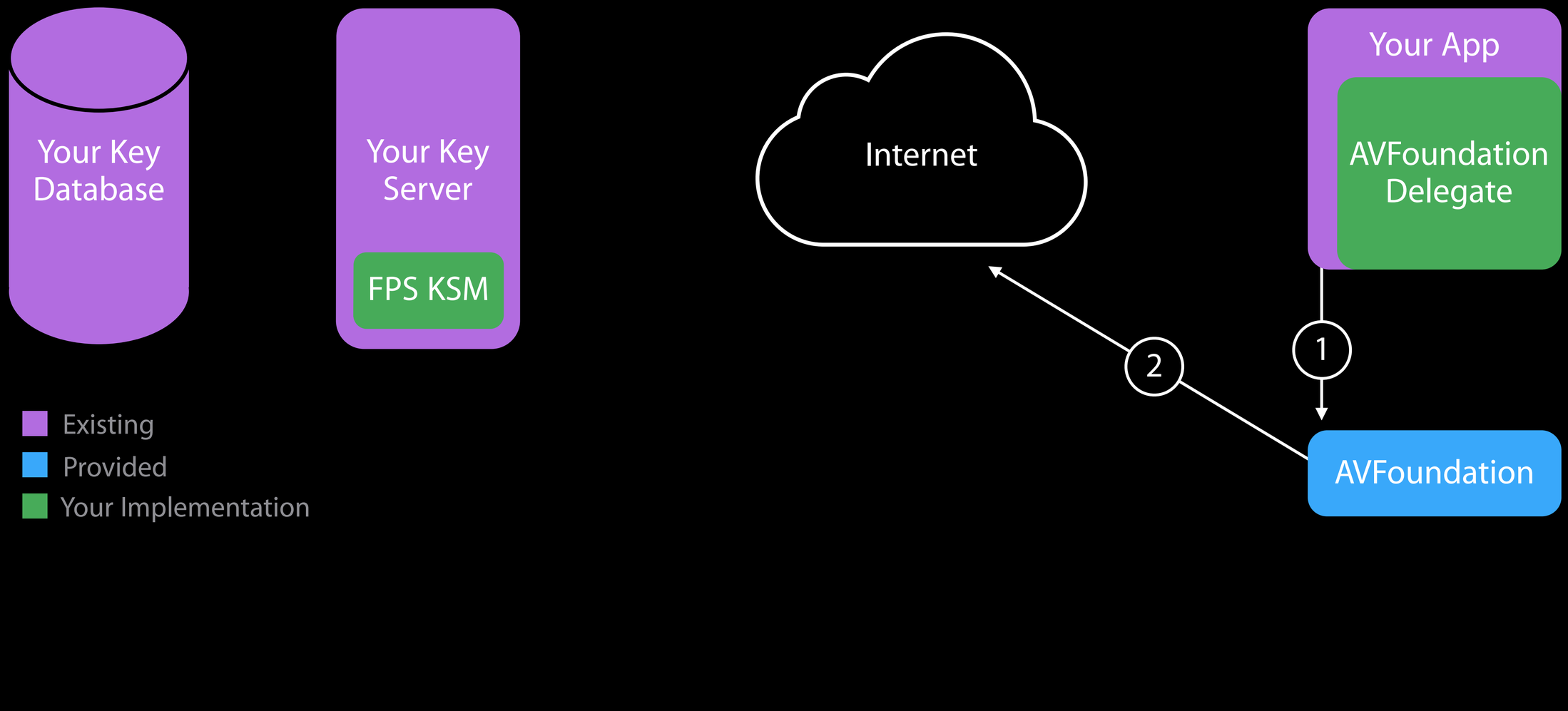
# FairPlay Streaming—Request Flow

① Your app asks AVFoundation to play your protected HLS asset



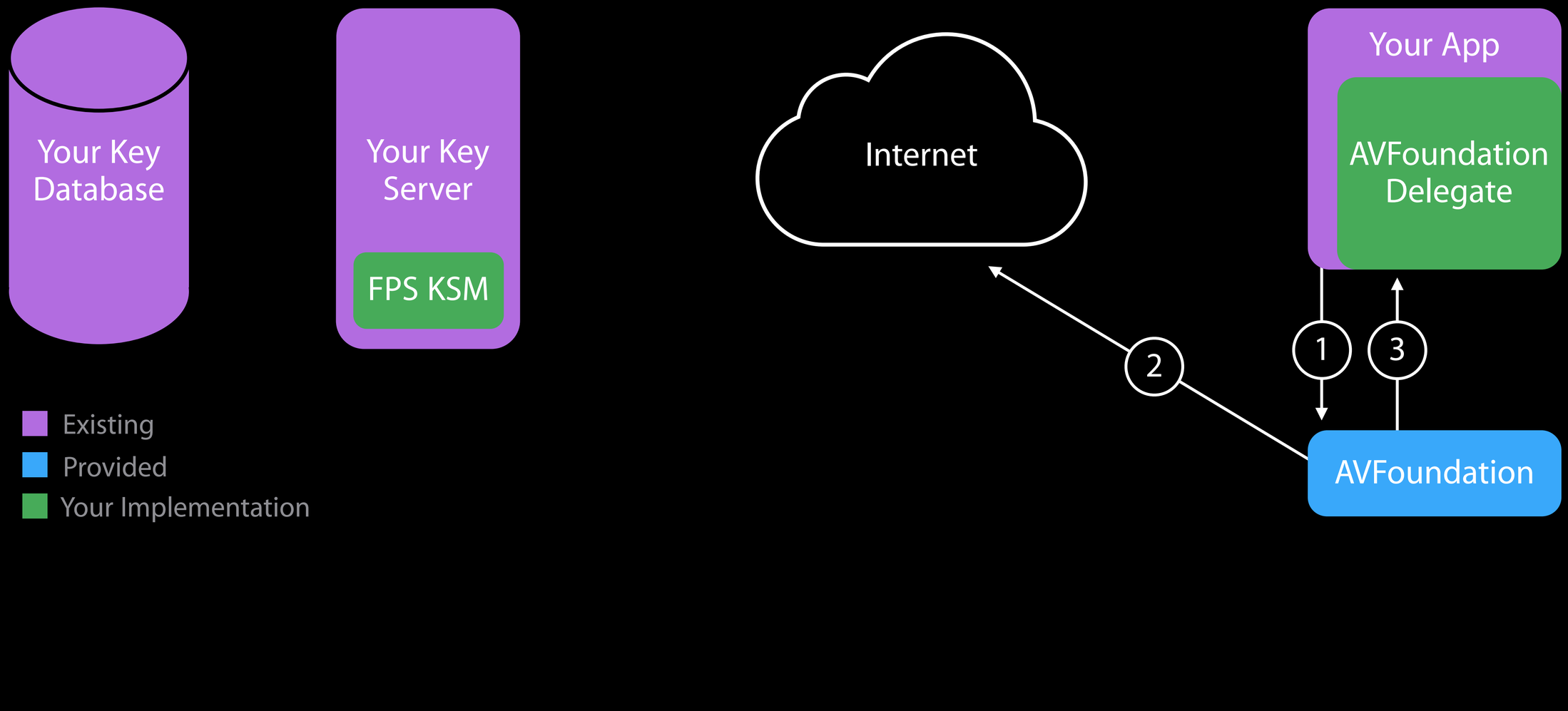
# FairPlay Streaming—Request Flow

② AVFoundation will download your `m3u8` playlist containing the KEY tag



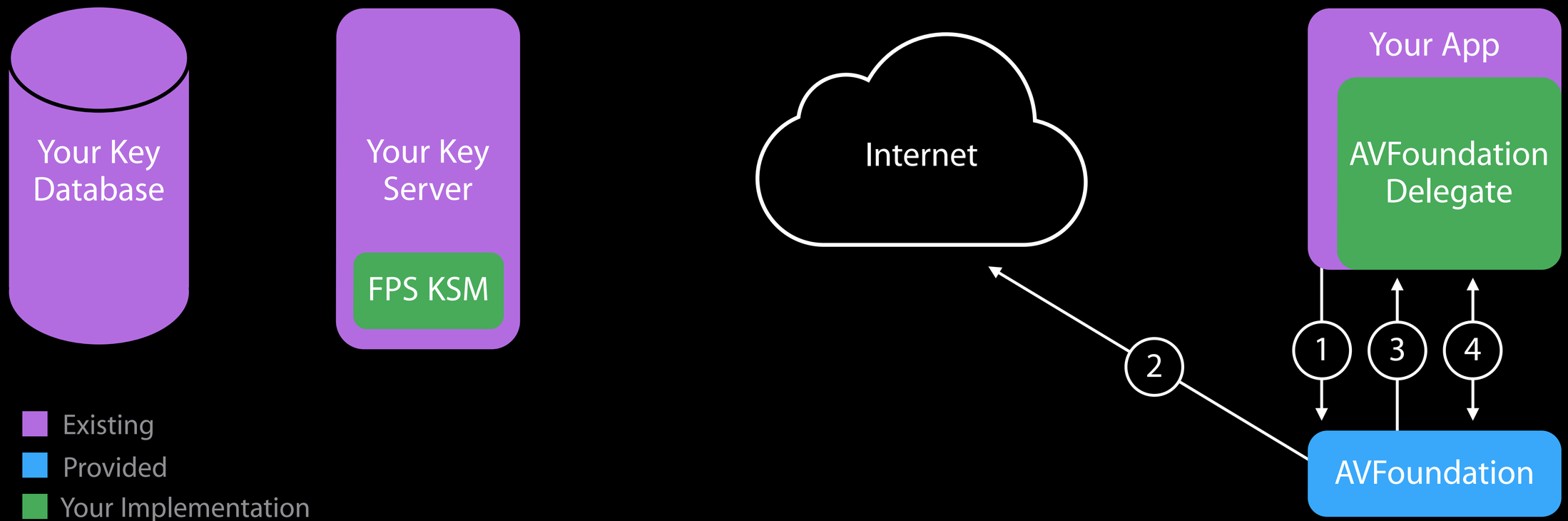
# FairPlay Streaming—Request Flow

3 AVFoundation will call your app delegate to request the key



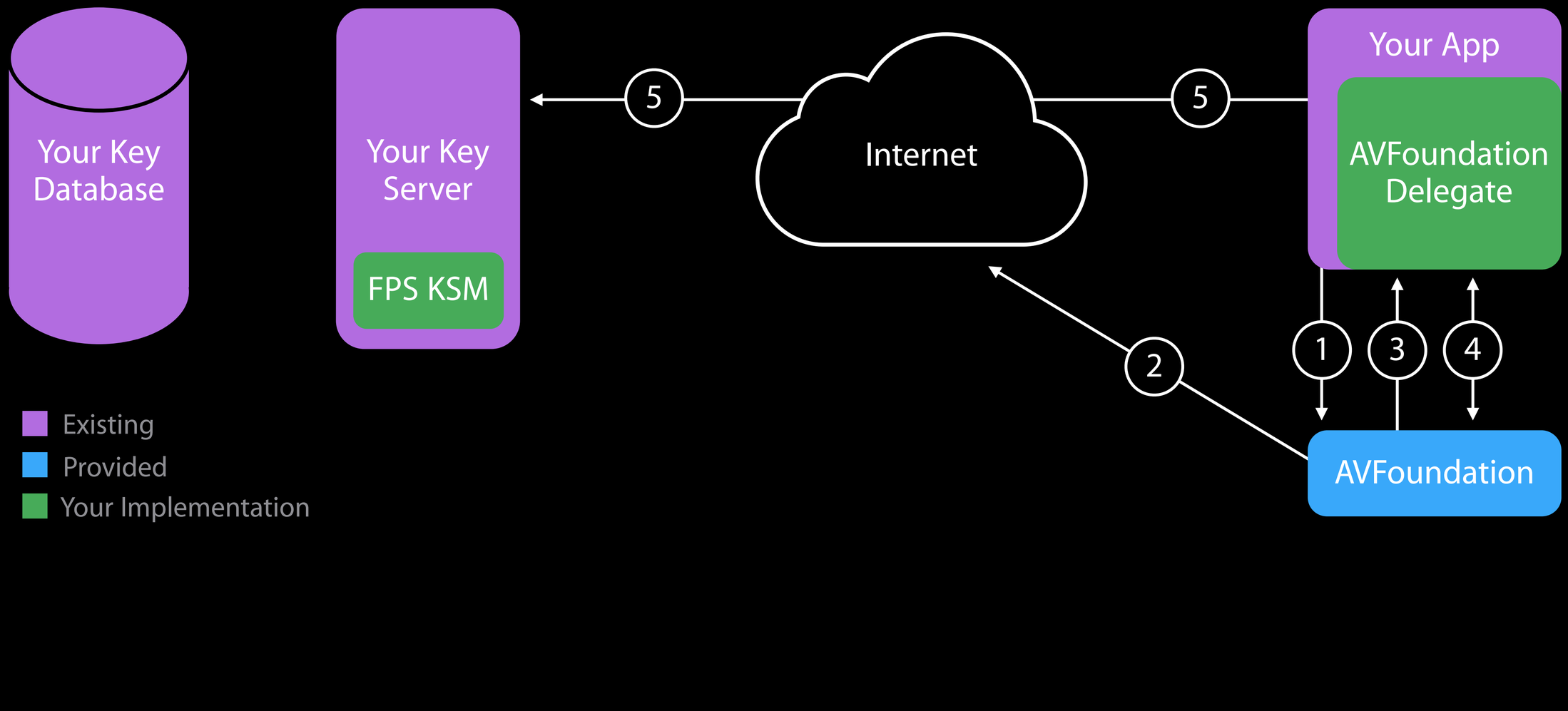
# FairPlay Streaming—Request Flow

④ Your app delegate calls AVFoundation to create an FPS Server Playback Context request



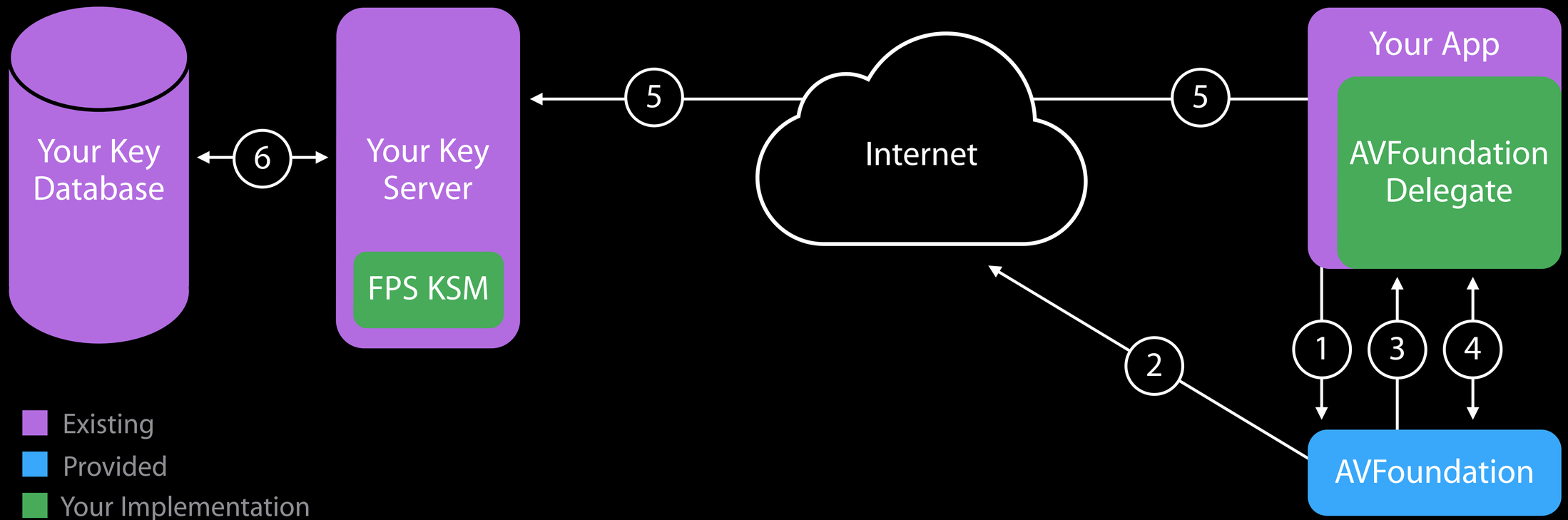
# FairPlay Streaming—Request Flow

5 Your app delegate sends the FPS SPC to your key server



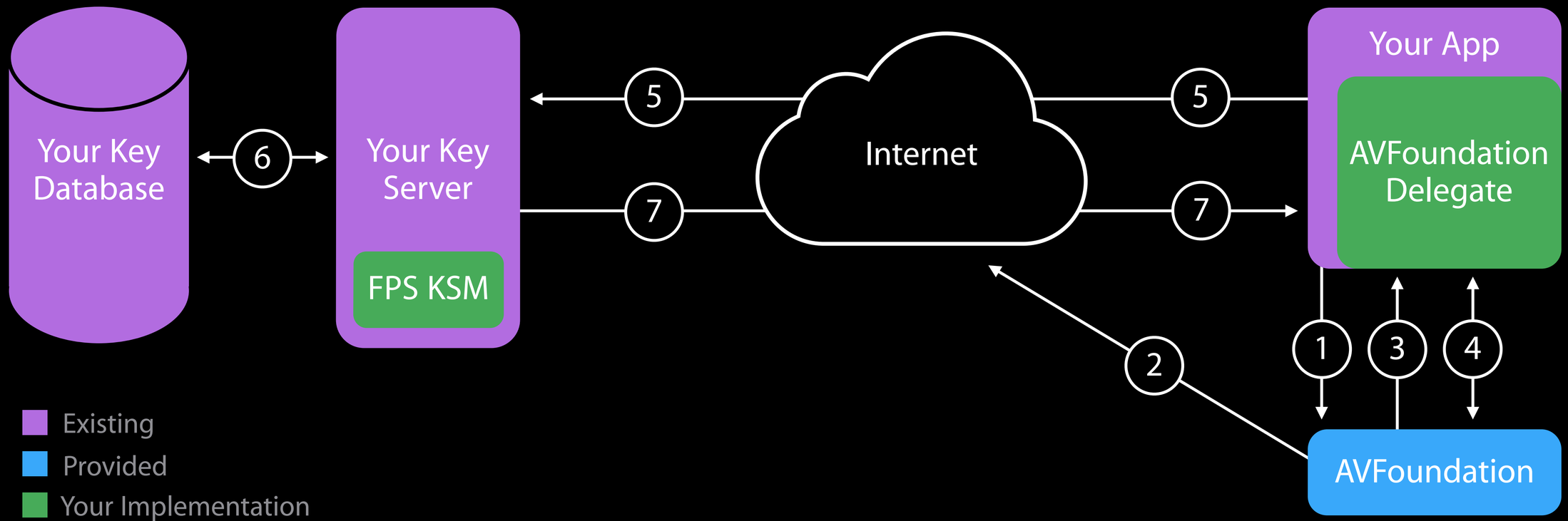
# FairPlay Streaming—Request Flow

6 Key server unwraps the SPC with your FPS KSM and performs CK lookup



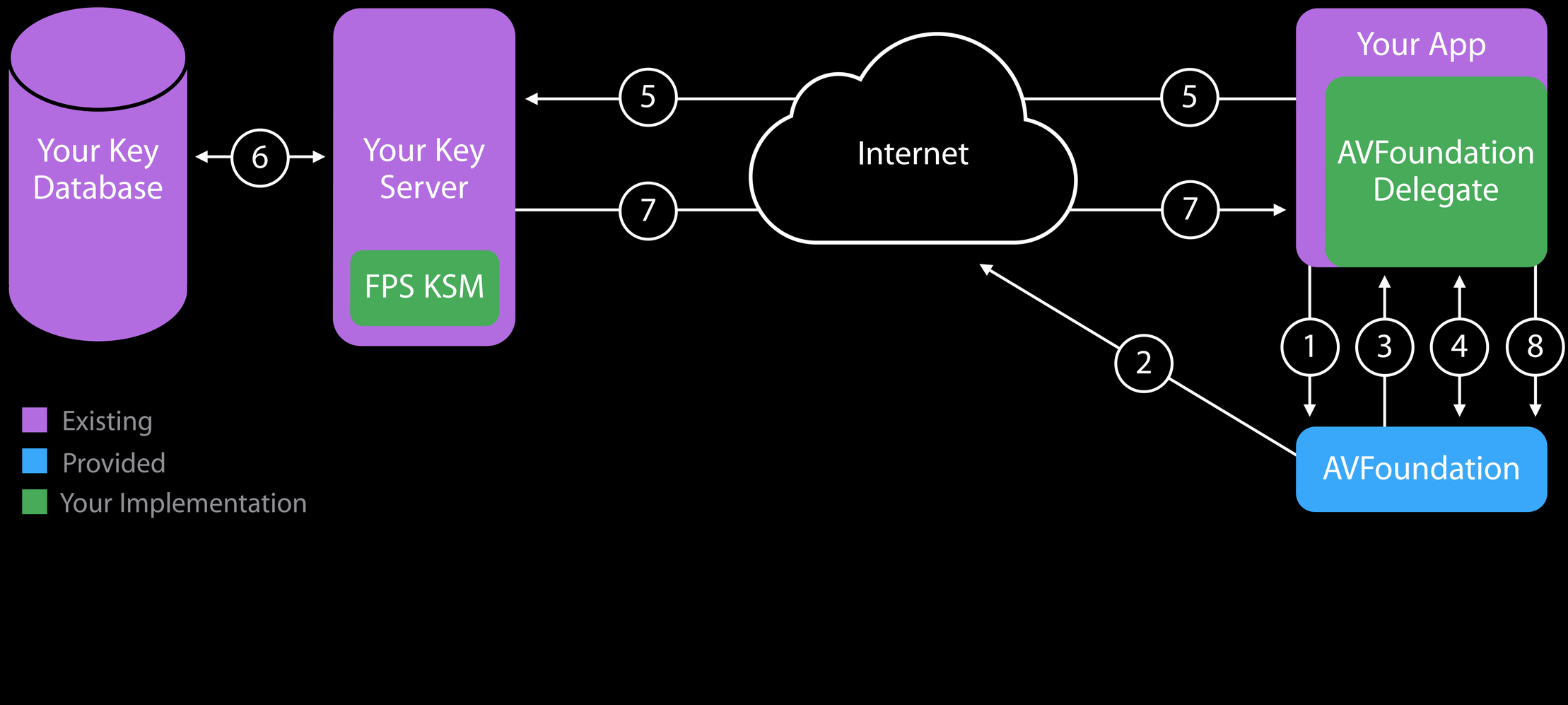
# FairPlay Streaming—Request Flow

7 After lookup, your FPS KSM wraps the content key into a Content Key Context response



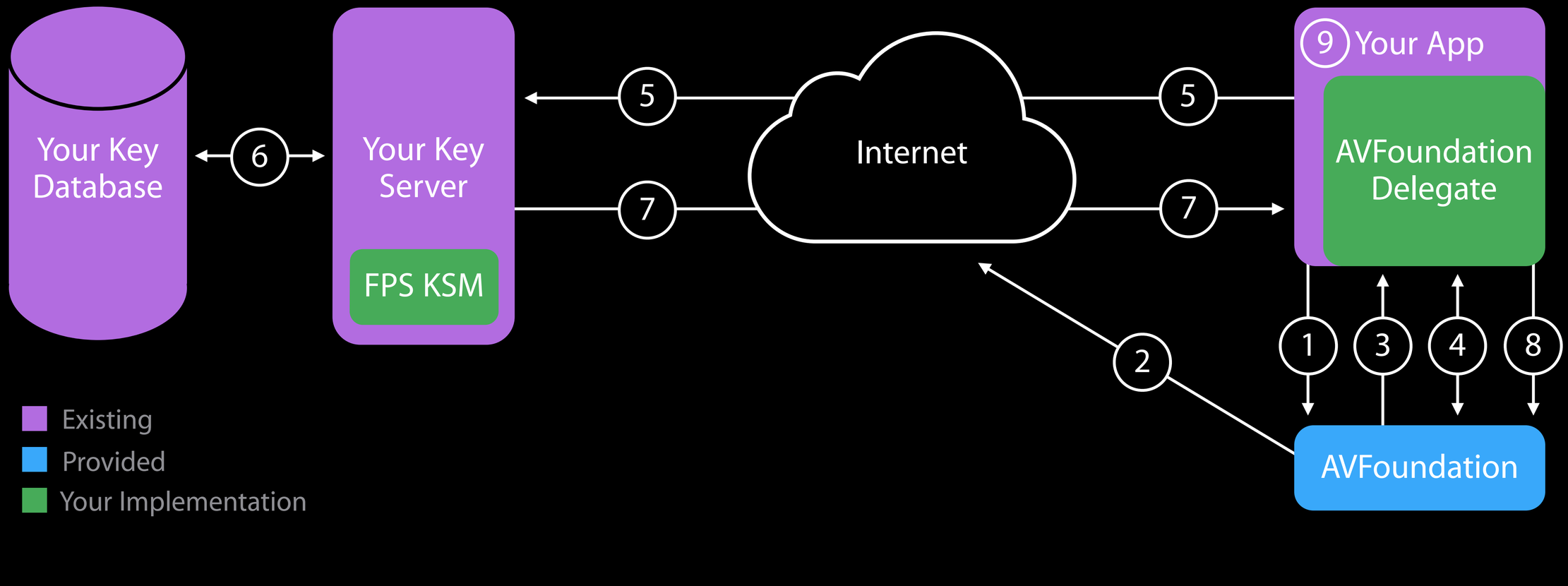
# FairPlay Streaming—Request Flow

8 Your app delegate provides the CKC to AVFoundation



# FairPlay Streaming—Request Flow

9 Now the device can decrypt and play the content



# What Is Provided

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

FairPlay Streaming SDK

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

FairPlay Streaming SDK

- Protocol specification

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

FairPlay Streaming SDK

- Protocol specification
- Server reference implementation

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

FairPlay Streaming SDK

- Protocol specification
- Server reference implementation
- Server test vectors and validation tools

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

FairPlay Streaming SDK

- Protocol specification
- Server reference implementation
- Server test vectors and validation tools
- Example content

# What Is Provided

AVFoundation, including API for `AVAssetResourceLoader` delegate

FairPlay Streaming SDK

- Protocol specification
- Server reference implementation
- Server test vectors and validation tools
- Example content
- Client example code

# Integrating FPS Into Your Key Server

Your Key Server must:

- Decrypt and validate SPC request
- Lookup CK by the asset identifier
- Produce CKC response

# Integrating FPS Into Your Key Server

Your Key Server must:

- **KSM**: Decrypt and validate SPC request
- Lookup CK by the asset identifier
- Produce CKC response

# Integrating FPS Into Your Key Server

Your Key Server must:

- **KSM**: Decrypt and validate SPC request
- Lookup CK by the asset identifier
- **KSM**: Produce CKC response

# Integrating FPS Into Your Key Server

Your Key Server must:

- **KSM**: Decrypt and validate SPC request
- Lookup CK by the asset identifier
- **KSM**: Produce CKC response

Implement KSM logic from scratch using protocol specification, or

# Integrating FPS Into Your Key Server

Your Key Server must:

- **KSM**: Decrypt and validate SPC request
- Lookup CK by the asset identifier
- **KSM**: Produce CKC response

Implement KSM logic from scratch using protocol specification, or

Customize the C reference implementation in the SDK (language, integration)

# Testing Your Key Security Module

# Testing Your Key Security Module

Supplied test vectors should be used to validate correctness of responses produced

# Testing Your Key Security Module

Supplied test vectors should be used to validate correctness of responses produced

- Your KSM implementation will consume test SPC request and produce response

# Testing Your Key Security Module

Supplied test vectors should be used to validate correctness of responses produced

- Your KSM implementation will consume test SPC request and produce response
- Supplied tool will validate your produced CKC response

# Testing Your Key Security Module

Supplied test vectors should be used to validate correctness of responses produced

- Your KSM implementation will consume test SPC request and produce response
- Supplied tool will validate your produced CKC response

Test vectors are based on non-functional development credentials

# Testing Your Key Security Module

Supplied test vectors should be used to validate correctness of responses produced

- Your KSM implementation will consume test SPC request and produce response
- Supplied tool will validate your produced CKC response

Test vectors are based on non-functional development credentials

End-to-end playback test on device requires production credentials!

# Integrating FPS Into Your App

Register an `AVAssetResourceLoader` delegate with `AVAsset`

# Integrating FPS Into Your App

Register an `AVAssetResourceLoader` delegate with `AVAsset`

`AVAssetResourceLoader` delegate must:

# Integrating FPS Into Your App

Register an `AVAssetResourceLoader` delegate with `AVAsset`

`AVAssetResourceLoader` delegate must:

- Generate the SPC
  - handle `shouldWaitForLoadingOfRequestedResource:` for key requests
  - call -`[AVAssetResourceLoadingRequest streamingContentKeyRequestDataForApp: contentIdentifier: options: error: ]` to produce SPC

# Integrating FPS Into Your App

Register an `AVAssetResourceLoader` delegate with `AVAsset`

`AVAssetResourceLoader` delegate must:

- Generate the SPC
  - handle `shouldWaitForLoadingOfRequestedResource:` for key requests
  - call - `[AVAssetResourceLoadingRequest streamingContentKeyRequestDataForApp: contentIdentifier: options: error: ]` to produce SPC
- Send SPC request to your Key Server

# Integrating FPS Into Your App

Register an `AVAssetResourceLoader` delegate with `AVAsset`

`AVAssetResourceLoader` delegate must:

- Generate the SPC
  - handle `shouldWaitForLoadingOfRequestedResource:` for key requests
  - call - `[AVAssetResourceLoadingRequest streamingContentKeyRequestDataForApp: contentIdentifier: options: error: ]` to produce SPC
- Send SPC request to your Key Server
- Provide CKC response (or error) to `AVAssetResourceLoadingRequest`

# Encrypting and Testing Your Content

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

- `METHOD=SAMPLE-AES`

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

- `METHOD=SAMPLE-AES`
- `KEYFORMAT="com.apple.streamingkeydelivery"`

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

- `METHOD=SAMPLE-AES`
- `KEYFORMAT="com.apple.streamingkeydelivery"`

Many 3rd-party encoders support HLS sample encryption

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

- `METHOD=SAMPLE-AES`
- `KEYFORMAT="com.apple.streamingkeydelivery"`

Many 3rd-party encoders support HLS sample encryption

To check your encryption workflow

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

- `METHOD=SAMPLE-AES`
- `KEYFORMAT="com.apple.streamingkeydelivery"`

Many 3rd-party encoders support HLS sample encryption

To check your encryption workflow

- SDK contains an example of sample-encrypted content for comparison

# Encrypting and Testing Your Content

Encrypt your content with HLS Sample Encryption

- `METHOD=SAMPLE-AES`
- `KEYFORMAT="com.apple.streamingkeydelivery"`

Many 3rd-party encoders support HLS sample encryption

To check your encryption workflow

- SDK contains an example of sample-encrypted content for comparison
- HLS **`mediafilesegmenter`** can produce encrypted content for comparison

# FairPlay Streaming with AirPlay

# FairPlay Streaming with AirPlay

# FairPlay Streaming with AirPlay

AirPlay Video will transfer streaming operation to Apple TV

# FairPlay Streaming with AirPlay

AirPlay Video will transfer streaming operation to Apple TV

No additional code needs to be written!

# FairPlay Streaming with AirPlay

AirPlay Video will transfer streaming operation to Apple TV

No additional code needs to be written!

SPC request is generated by FPS on Apple TV and CKC response is for Apple TV

# FairPlay Streaming with AirPlay

AirPlay Video will transfer streaming operation to Apple TV

No additional code needs to be written!

SPC request is generated by FPS on Apple TV and CKC response is for Apple TV

- Your app on the sending device relays messages between Apple TV and your key server

# FairPlay Streaming with AirPlay

AirPlay Video will transfer streaming operation to Apple TV

No additional code needs to be written!

SPC request is generated by FPS on Apple TV and CKC response is for Apple TV

- Your app on the sending device relays messages between Apple TV and your key server

Provides the same level of security as local playback

# FairPlay Streaming with AirPlay

AirPlay Video will transfer streaming operation to Apple TV

No additional code needs to be written!

SPC request is generated by FPS on Apple TV and CKC response is for Apple TV

- Your app on the sending device relays messages between Apple TV and your key server

Provides the same level of security as local playback

FPS content is disabled by AirPlay Mirroring, not rendered in screenshots or recordings

# FairPlay Streaming in Safari on OS X

# FairPlay Streaming in Safari on OS X

NEW

# FairPlay Streaming in Safari on OS X

NEW

FairPlay Streaming accessed through HTML5 Encrypted Media Extensions

# FairPlay Streaming in Safari on OS X

NEW

FairPlay Streaming accessed through HTML5 Encrypted Media Extensions

Key delivery code must be written in JavaScript

# FairPlay Streaming in Safari on OS X

NEW

FairPlay Streaming accessed through HTML5 Encrypted Media Extensions

Key delivery code must be written in JavaScript

- Example provided with FPS SDK

# FairPlay Streaming in Safari on OS X

NEW

FairPlay Streaming accessed through HTML5 Encrypted Media Extensions

Key delivery code must be written in JavaScript

- Example provided with FPS SDK

Same KSM can support both iOS clients and Safari on OS X

# FairPlay Streaming in Safari on OS X

NEW

FairPlay Streaming accessed through HTML5 Encrypted Media Extensions

Key delivery code must be written in JavaScript

- Example provided with FPS SDK

Same KSM can support both iOS clients and Safari on OS X

Supports AirPlay

# Integrating FPS Into Your Web Page

# Integrating FPS Into Your Web Page

Set **`m3u8`** URL as src attribute of HTML **`<video>`** tag (as usual)

# Integrating FPS Into Your Web Page

Set `m3u8` URL as src attribute of HTML `<video>` tag (as usual)

Add `EventListener` for ‘`webkitneedkey`‘ to video element:

# Integrating FPS Into Your Web Page

Set `m3u8` URL as src attribute of HTML `<video>` tag (as usual)

Add `EventListener` for ‘`webkitneedkey`‘ to video element:

Set EME CDM `keySystem` (`video.webkitSetMediaKeys`) to “`com.apple.fps.1_0`“

# Integrating FPS Into Your Web Page

Set `m3u8` URL as src attribute of HTML `<video>` tag (as usual)

Add `EventListener` for ‘`webkitneedkey`‘ to video element:

Set EME CDM `keySystem` (`video.webkitSetMediaKeys`) to “`com.apple.fps.1_0`“

Create `keySession` on “`video/mp4`” to relay messages with the `keySystem`

# Integrating FPS Into Your Web Page

Set `m3u8` URL as src attribute of HTML `<video>` tag (as usual)

Add `EventListener` for '`webkitneedkey`' to video element:

Set EME CDM `keySystem` (`video.webkitSetMediaKeys`) to "`com.apple.fps.1_0`"

Create `keySession` on "`video/mp4`" to relay messages with the `keySystem`

Add Event handler for '`webkitkeymessage`' to `keySession`:

# Integrating FPS Into Your Web Page

Set `m3u8` URL as src attribute of HTML `<video>` tag (as usual)

Add `EventListener` for '`webkitneedkey`' to video element:

Set EME CDM `keySystem` (`video.webkitSetMediaKeys`) to "`com.apple.fps.1_0`"

Create `keySession` on "`video/mp4`" to relay messages with the `keySystem`

Add Event handler for '`webkitkeymessage`' to `keySession`:

Send SPC request to your Key Server

# Integrating FPS Into Your Web Page

Set `m3u8` URL as src attribute of HTML `<video>` tag (as usual)

Add `EventListener` for '`webkitneedkey`' to video element:

Set EME CDM `keySystem` (`video.webkitSetMediaKeys`) to "`com.apple.fps.1_0`"

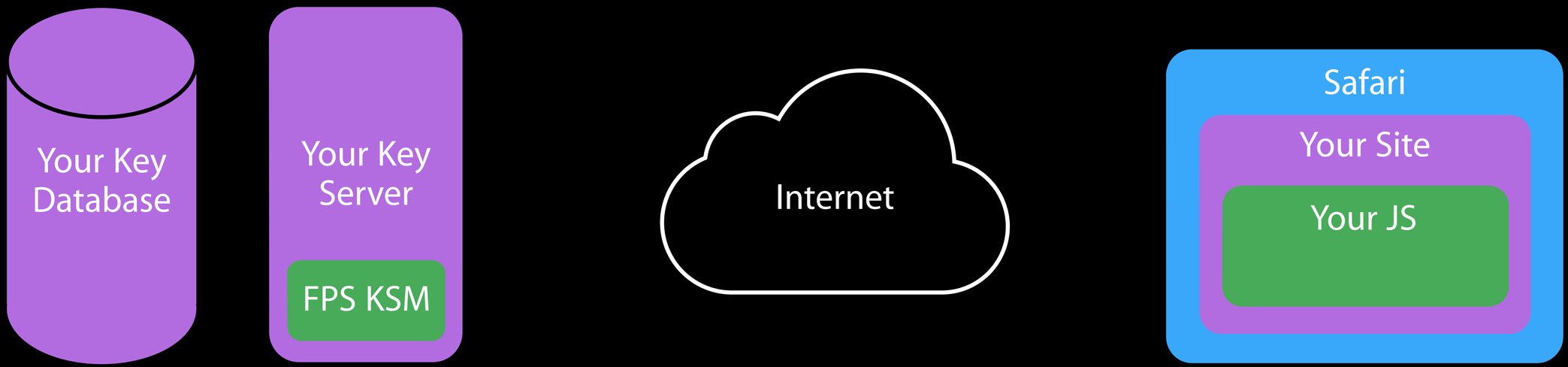
Create `keySession` on "`video/mp4`" to relay messages with the `keySystem`

Add Event handler for '`webkitkeymessage`' to `keySession`:

Send SPC request to your Key Server

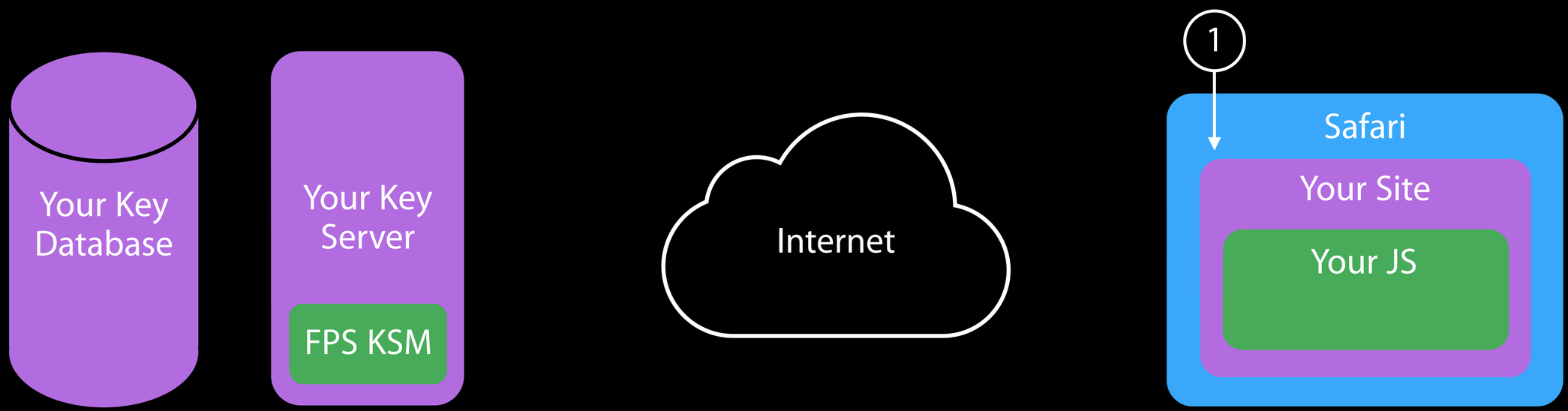
Provide CKC response to `keySession.update`()

# Safari Request Flow



# Safari Request Flow

① User hits Play

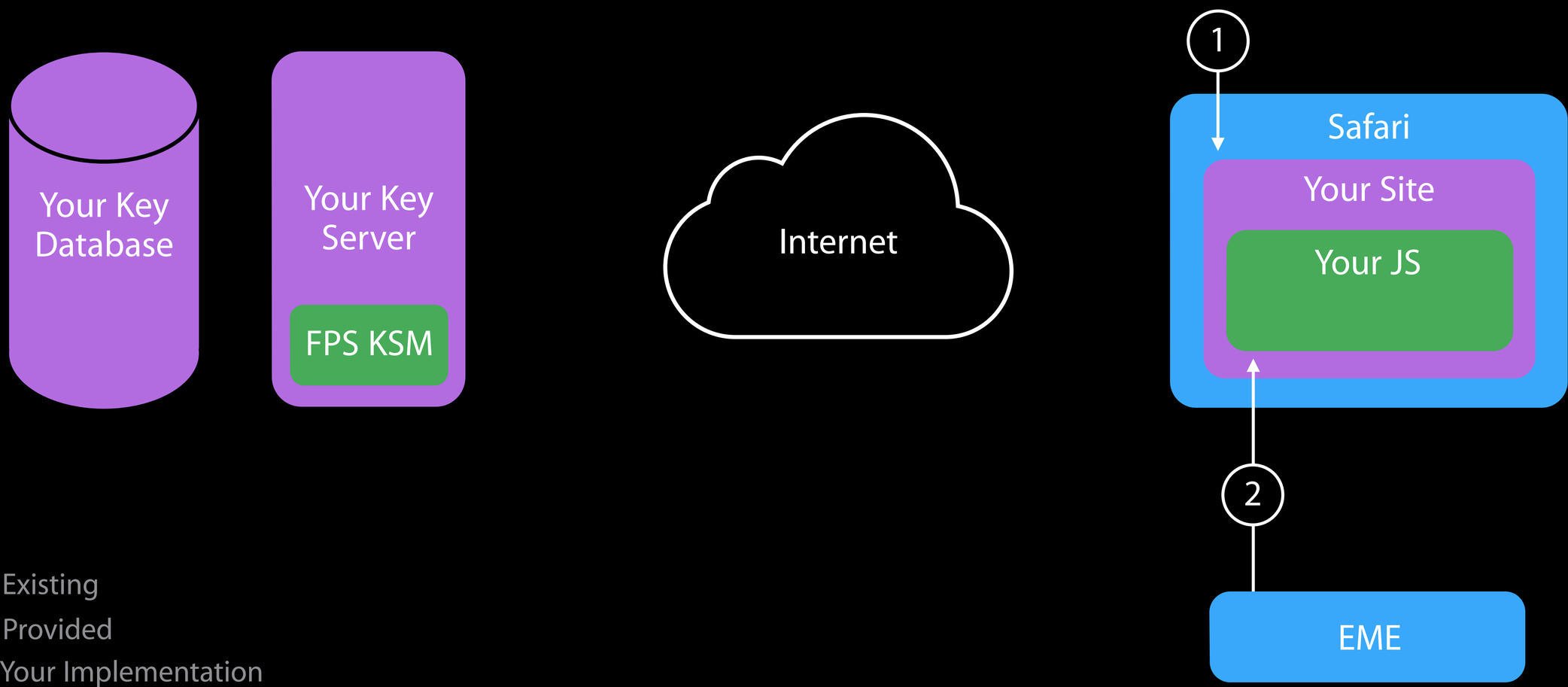



Existing
Provided
Your Implementation

EME

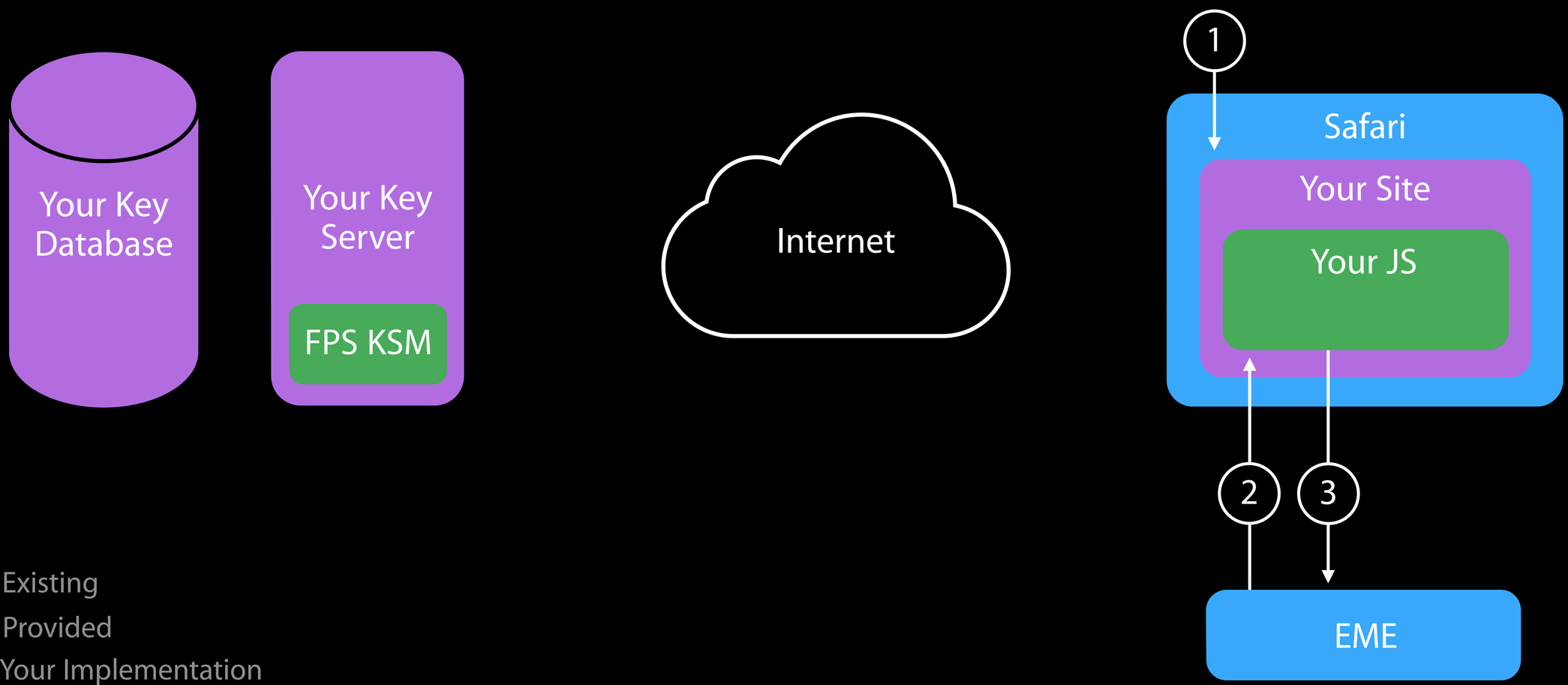
# Safari Request Flow

② Your Event Listener receives '`webkitneedkey`' message



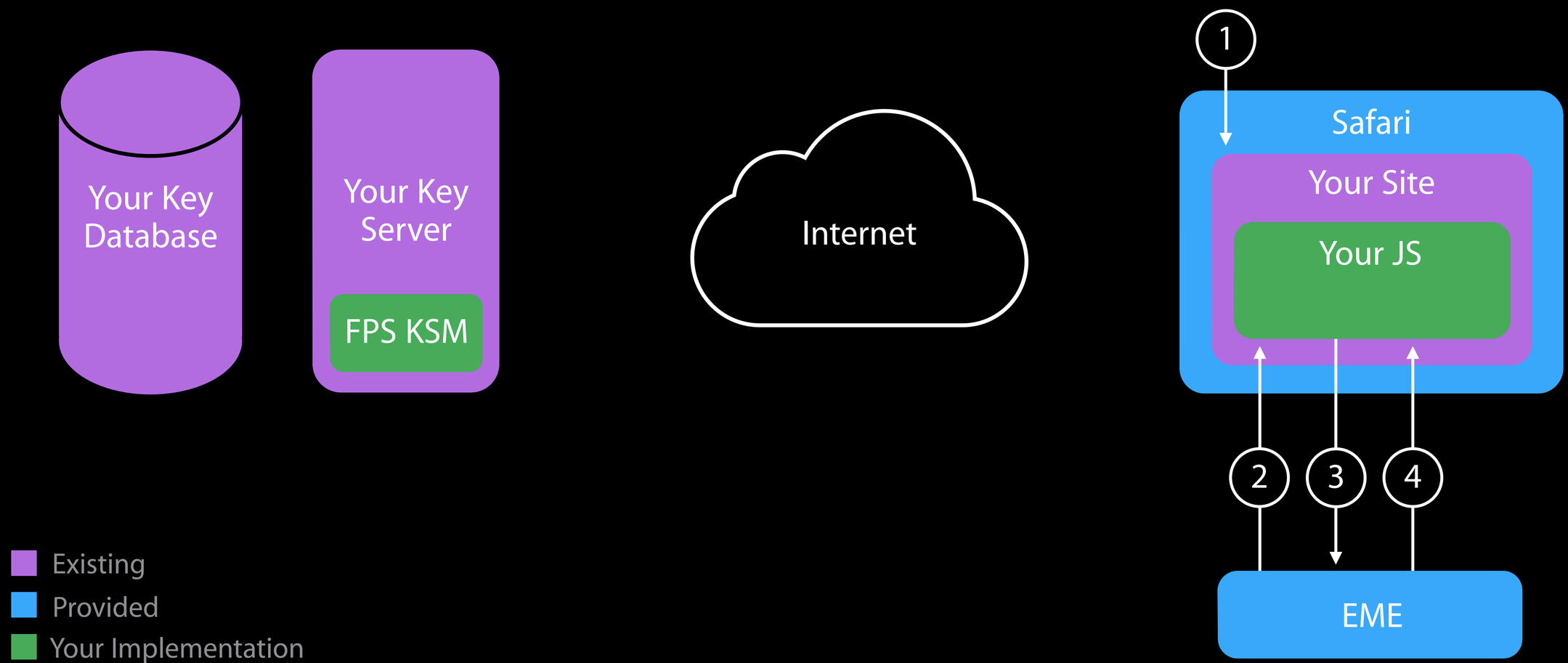
# Safari Request Flow

3 Your Event Listener creates keySession and waits for '`webkitkeymessage`' Event



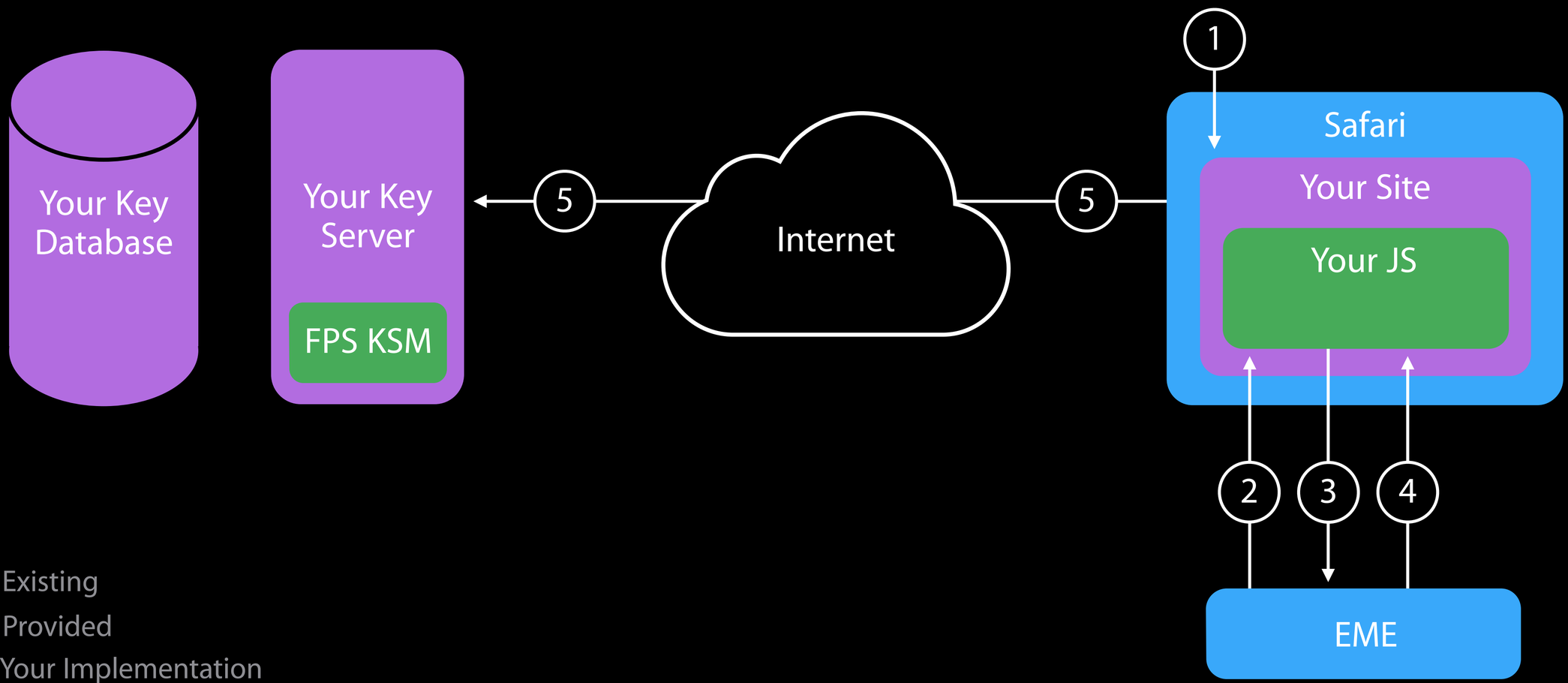
# Safari Request Flow

④ Your '`webkitkeymessage`' Event Handler receives message containing SPC



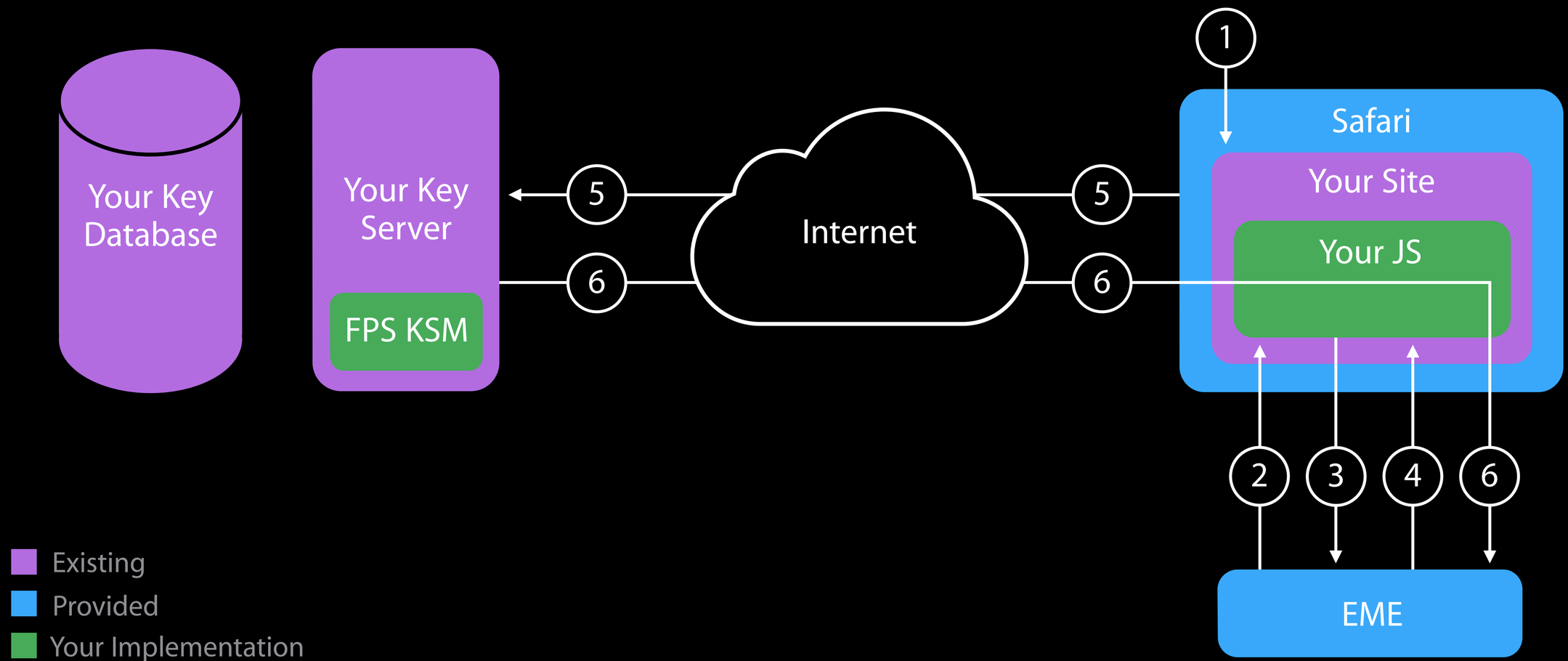
# Safari Request Flow

5 Your Event Handler sends SPC to your Key Server



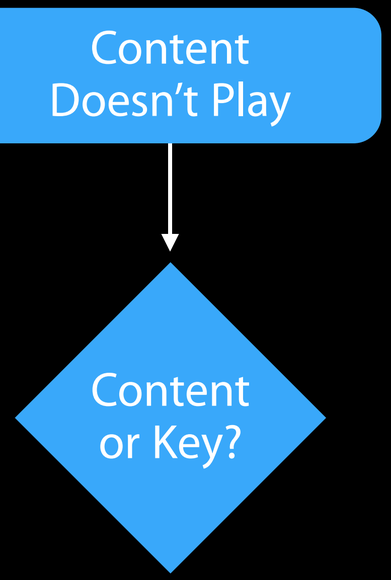
# Safari Request Flow

6 You update **`keySession`** upon receipt of CKC response

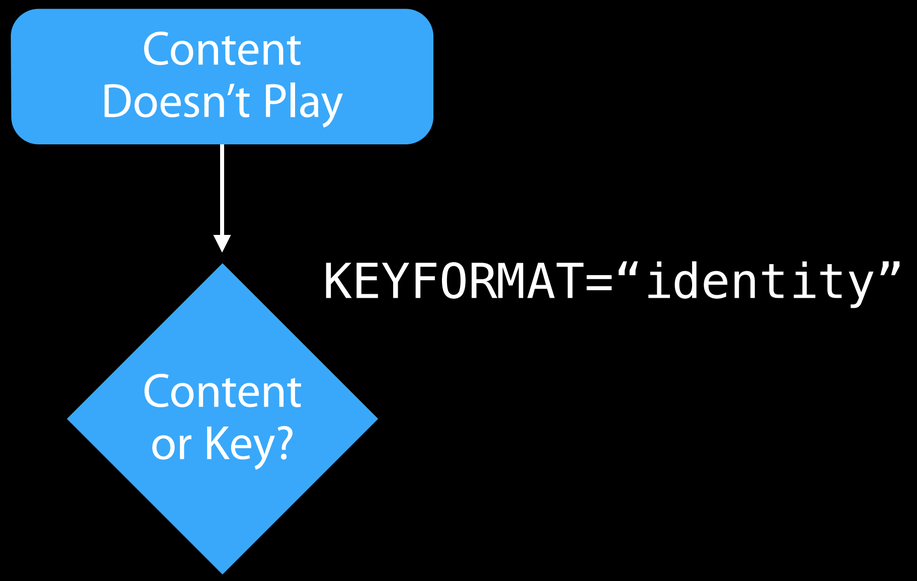


# FairPlay Streaming Integration Troubleshooting

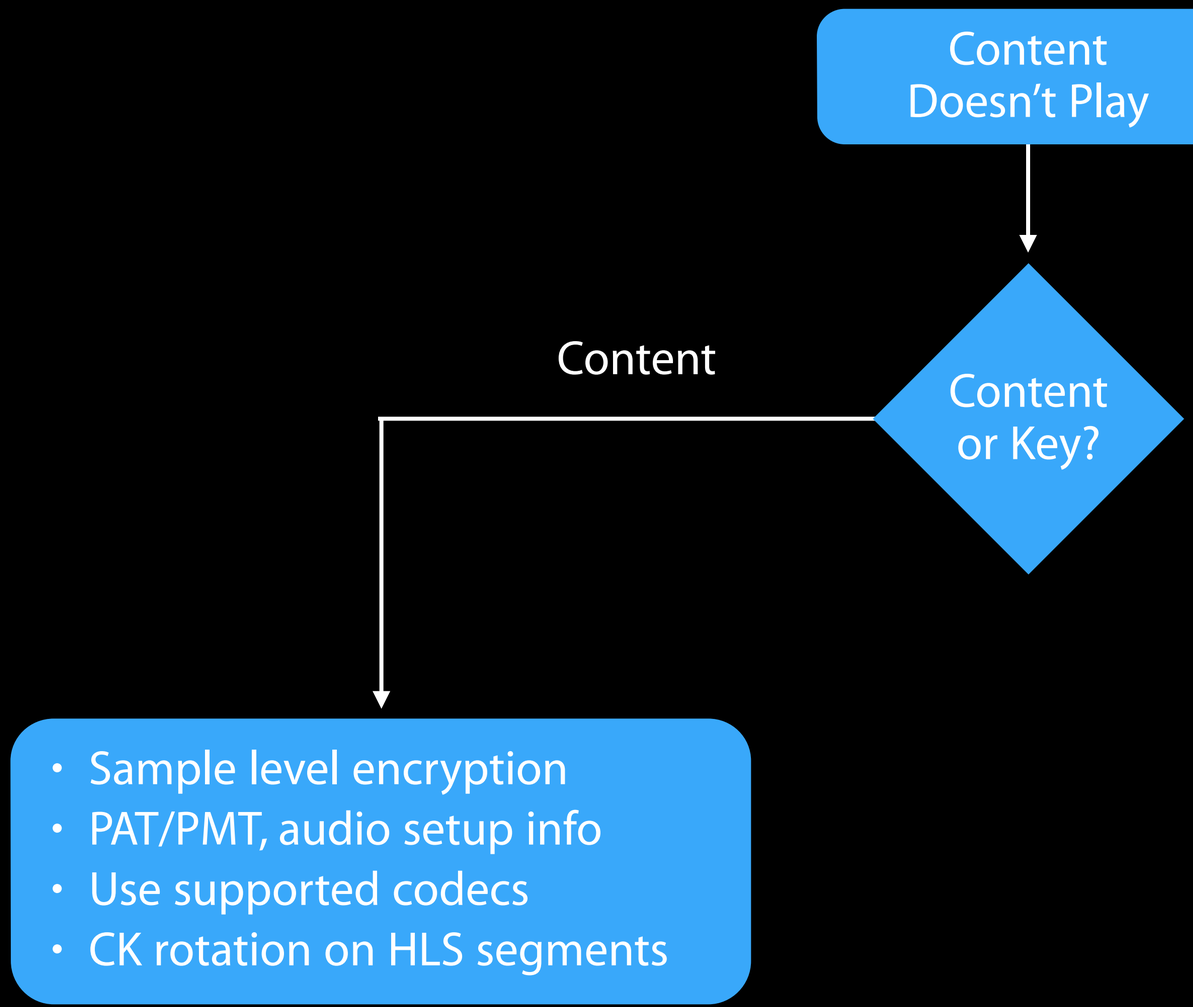
# Troubleshooting



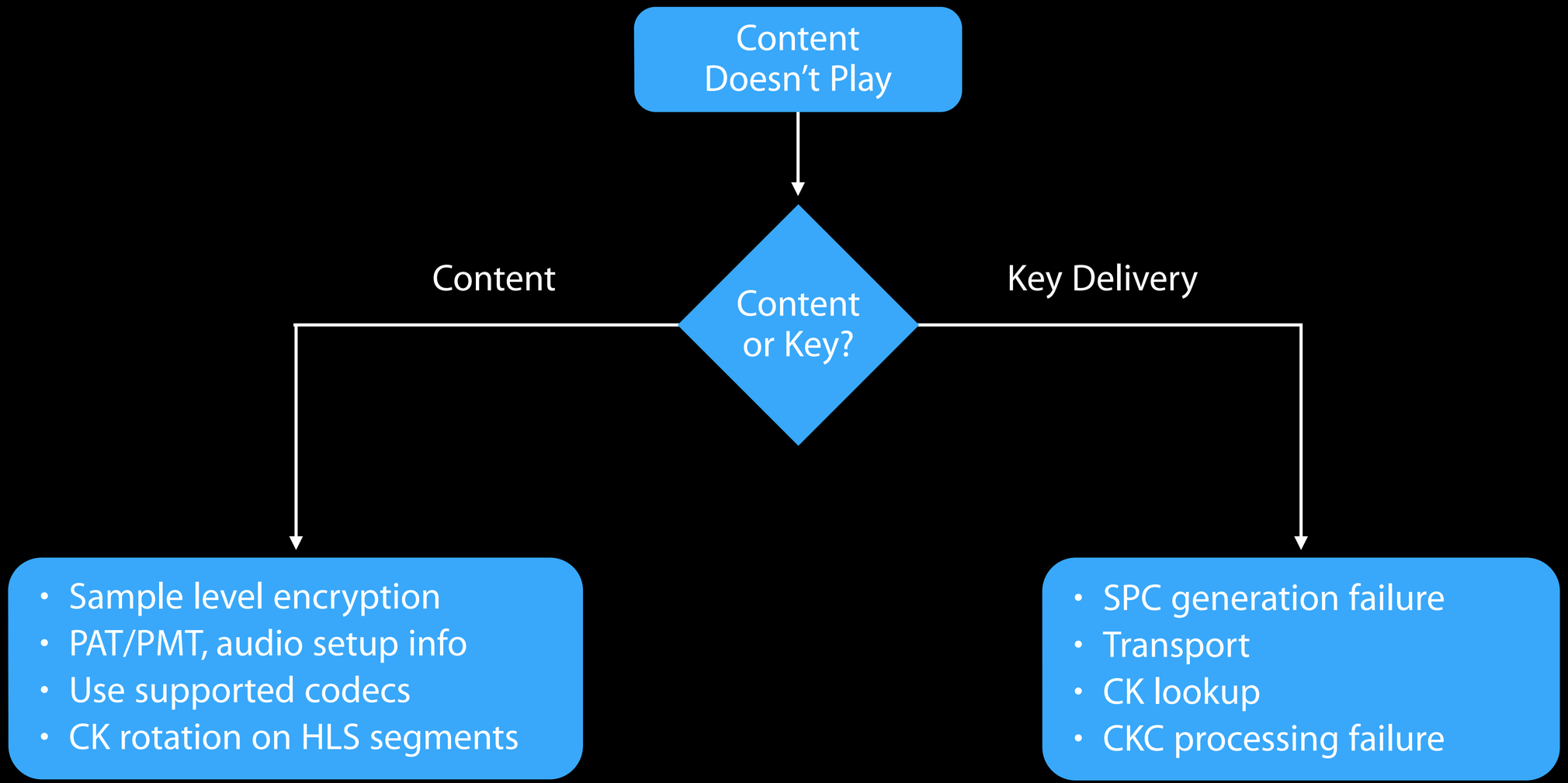
# Troubleshooting



# Troubleshooting



# Troubleshooting



# Summary of FairPlay Streaming

FairPlay Streaming provides industrial-strength content protection for HLS

Built into on iOS, Apple TV and Safari on OS X

Deeply integrated into the OS

Designed for power-efficient playback

Supports platform features such as AirPlay, external output protection, and HTML5

# More Information

Documentation and Videos

FairPlay Streaming

http://developer.apple.com/streaming/fps/

Technical Support

Apple Developer Forums

http://developer.apple.com/forums

Developer Technical Support

http://developer.apple.com/support/technical

# Labs

| | | |
|---|---|---|
| HTTP Live Streaming Lab | Graphics, Games and Media Lab B | Tuesday 11:00AM |
| AirPlay Lab | Graphics, Games and Media Lab B | Tuesday 3:30PM |
| AVKit and AV Foundation Lab | Graphics, Games and Media Lab A | Wednesday 1:30PM |
| AVKit and AV Foundation Lab | Graphics, Games and Media Lab B | Thursday 11:00AM |
| HTTP Live Streaming Lab | Graphics, Games and Media Lab C | Thursday 11:00AM |


WWDC15