WELLCOME

How to secure apiKeys used on apps with support of iCloudKit

NSCoder Night Madrid, July 19th, 2023

@pmunoz08

ApiKeys Management on apps

- Hard coded ApiKeys in source code

ApiKeys Management on apps

- Hard coded ApiKeys in source code
- Store ApiKeys in Xcode Configuration and Info.plist

ApiKeys Management on apps

- Hard coded ApiKeys in source code
- Store ApiKeys in Xcode Configuration and Info.plist
- Obfuscate ApiKeys using some kind of Encryption

ApiKeys Management on apps

- Hard coded ApiKeys in source code
- Store ApiKeys in Xcode Configuration and Info.plist
- Obfuscate ApiKeys using some kind of Encryption
- Don’t Store ApiKeys On-Device

Hard coded ApiKeys in source code

✅ Easy implementation

❌ They will forme part of your source

The act of compilation transforms human-readable text into machine-executable binary, leaving no trace of the secret

Hard coded ApiKeys in source code

✅ Easy implementation

❌ They will forme part of your source

❌ The act of compilation transforms human-readable text into machine-executable binary, leaving no trace of the secret. (No really)

Hard coded ApiKeys in source code

Using a reverse-engineering tool like Radare2, that secret is plainly visible from inside the compiled executable.

```
[0x100003a38]> q
pedro@iMac-de-Pedro-141 radare2 % r2 /Users/pedro/Desktop/UnSplash_sample\ 24-6-23,\ 16.26.xcarchive/Products/Applications/UnSplash_sample
 -- 99 bugs, take one down pass it around. 100 bugs...
[0x100005eb0]> iz
[Strings]
nth paddr      vaddr       len size section                type    string
―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
0   0x00017aa4 0x100017aa4 11  12   2.__TEXT.__const        ascii   ContentView
1   0x00017bc4 0x100017bc4 9   10   2.__TEXT.__const        ascii   DataModel
2   0x00017cf0 0x100017cf0 9   10   2.__TEXT.__const        ascii   PhotoView
3   0x00017d96 0x100017d96 9   10   2.__TEXT.__const        ascii   \e %,4=AFI
4   0x00017da4 0x100017da4 8   9    2.__TEXT.__const        ascii   \e %,4=AF
5   0x00017e1c 0x100017e1c 12  13   2.__TEXT.__const        ascii   UnplashPhoto
6   0x00017e48 0x100017e48 4   5    2.__TEXT.__const        ascii   User
7   0x00017e6c 0x100017e6c 5   6    2.__TEXT.__const        ascii   Links
8   0x00017e90 0x100017e90 4   5    2.__TEXT.__const        ascii   Urls
9   0x0001802c 0x10001802c 10  11   2.__TEXT.__const        ascii   CodingKeys
10  0x000185e4 0x1000185e4 13  14   2.__TEXT.__const        ascii   UnplashClient
11  0x00018688 0x100018688 13  14   2.__TEXT.__const        ascii   PhotosPreview
12  0x000186e0 0x1000186e0 15  16   2.__TEXT.__const        ascii   UnSplash_sample
13  0x00018700 0x100018700 18  19   2.__TEXT.__const        ascii   UnSplash_sampleApp
14  0x000187b0 0x1000187b0 12  13   2.__TEXT.__const        ascii   NSURLSession
15  0x000187c0 0x1000187c0 18  19   2.__TEXT.__const        ascii   HTTPDataDownloader
16  0x000187f4 0x1000187f4 12  13   2.__TEXT.__const        ascii   UnplashError
17  0x00018894 0x100018894 15  16   2.__TEXT.__const        ascii   PhotoDetailView
0   0x00018a13 0x100018a13 4   5    3.__TEXT.__cstring      ascii   slug
1   0x00018a18 0x100018a18 10  11   3.__TEXT.__cstring      ascii   created_at
2   0x00018a23 0x100018a23 10  11   3.__TEXT.__cstring      ascii   updated_at
3   0x00018a2e 0x100018a2e 11  12   3.__TEXT.__cstring      ascii   promoted_at
4   0x00018a3a 0x100018a3a 5   6    3.__TEXT.__cstring      ascii   width
5   0x00018a40 0x100018a40 6   7    3.__TEXT.__cstring      ascii   height
6   0x00018a47 0x100018a47 5   6    3.__TEXT.__cstring      ascii   color
7   0x00018a4d 0x100018a4d 9   10   3.__TEXT.__cstring      ascii   blur_hash
8   0x00018a57 0x100018a57 11  12   3.__TEXT.__cstring      ascii   description
9   0x00018a63 0x100018a63 15  16   3.__TEXT.__cstring      ascii   alt_description
10  0x00018a73 0x100018a73 4   5    3.__TEXT.__cstring      ascii   urls
11  0x00018a78 0x100018a78 5   6    3.__TEXT.__cstring      ascii   links
12  0x00018a7e 0x100018a7e 4   5    3.__TEXT.__cstring      ascii   user
13  0x00018a83 0x100018a83 4   5    3.__TEXT.__cstring      ascii   self
14  0x00018a88 0x100018a88 4   5    3.__TEXT.__cstring      ascii   html
15  0x00018a8d 0x100018a8d 8   9    3.__TEXT.__cstring      ascii   download
16  0x00018aa0 0x100018aa0 17  18   3.__TEXT.__cstring      ascii   download_location
17  0x00018ab6 0x100018ab6 4   5    3.__TEXT.__cstring      ascii   full
18  0x00018abb 0x100018abb 7   8    3.__TEXT.__cstring      ascii   regular
19  0x00018ac3 0x100018ac3 5   6    3.__TEXT.__cstring      ascii   small
20  0x00018ac9 0x100018ac9 5   6    3.__TEXT.__cstring      ascii   thumb
21  0x00018acf 0x100018acf 8   9    3.__TEXT.__cstring      ascii   small_s3
22  0x00018ad8 0x100018ad8 4   5    3.__TEXT.__cstring      ascii   name
23  0x00018ae0 0x100018ae0 16  17   3.__TEXT.__cstring      ascii   twitter_username
24  0x00018b00 0x100018b00 24  25   3.__TEXT.__cstring      ascii   https://api.unsplash.com
25  0x00018b20 0x100018b20 64  65   3.__TEXT.__cstring      ascii   9657b2982a53f[redacted]bbcfec8a021e
26  0x00018b70 0x100018b70 22  23   3.__TEXT.__cstring      ascii   yyyy-MM-dd'T'HH:mm:ssZ
27  0x00018b90 0x100018b90 31  32   3.__TEXT.__cstring      ascii   _TtC15UnSplash_sample9DataModel
28  0x00018bb0 0x100018bb0 7   8    3.__TEXT.__cstring      ascii   _photos
29  0x00018bb9 0x100018bb9 6   7    3.__TEXT.__cstring      ascii   client
30  0x00018bc0 0x100018bc0 4   5    3.__TEXT.__cstring      ascii   page
31  0x00018bd0 0x100018bd0 42  43   3.__TEXT.__cstring      ascii   Contradictory frame constraints specified.
32  0x00018c00 0x100018c00 36  37   3.__TEXT.__cstring      ascii   _TtC15UnSplash_sample13UnplashClient
33  0x00018c25 0x100018c25 13  14   3.__TEXT.__cstring      ascii   $defaultActor
34  0x00018c34 0x100018c34 4   5    3.__TEXT.__cstring      ascii   host
35  0x00018c39 0x100018c39 13  14   3.__TEXT.__cstring      ascii   apiKeyUnplash
36  0x00018c47 0x100018c47 10  11   3.__TEXT.__cstring      ascii   downloader
37  0x00018c60 0x100018c60 36  37   3.__TEXT.__cstring      ascii   _TtC15UnSplash_sample13PhotosPreview
38  0x00018c90 0x100018c90 132 133  3.__TEXT.__cstring      ascii   Accessing Environment<%s>'s value outside of being installed on a Vie
0   0x00018d20 0x100018d20 15  16   5.__TEXT.__objc_methname ascii   dateFromString:
1   0x00018d30 0x100018d30 13  14   5.__TEXT.__objc_methname ascii   initWithData:
```

Using Xcode Configuration and Info.plist

✅ Easy implementation

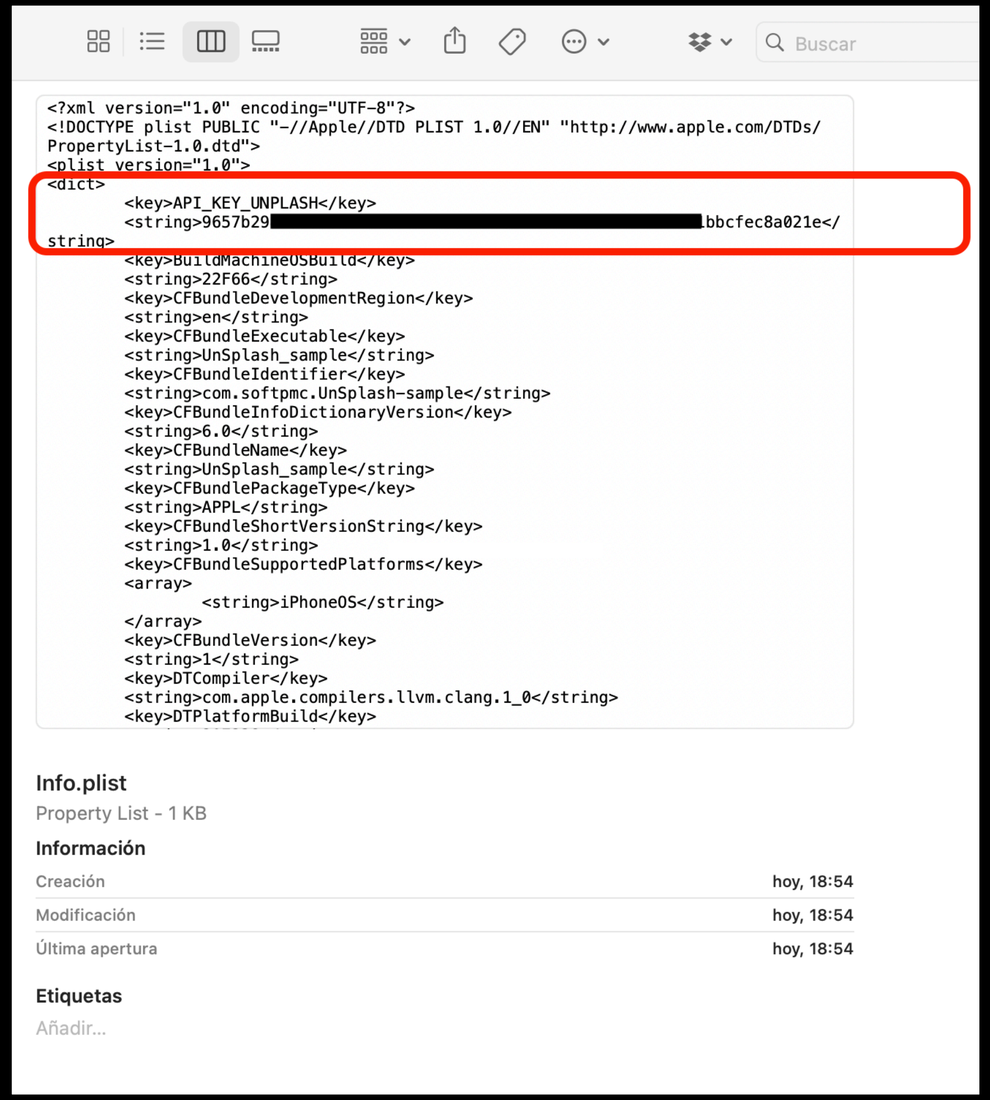
✅ They won’t forme part of your source

✅ Can’t be retrieved using a reverse-engineering tool

But .....

Using Xcode Configuration and Info.plist

The Info.plist file is used to store our apiKeys!. And is easily accesible from the app payload



Obfuscate ApiKeys using Encryption

✅ More secure than previous options.

❌ They will forme part of your source

❌ Using a reverse-engineering tool, that obfuscate apiKey is plainly visible as well as the salt used for obfuscation.

Don’t Store ApiKeys On-Device

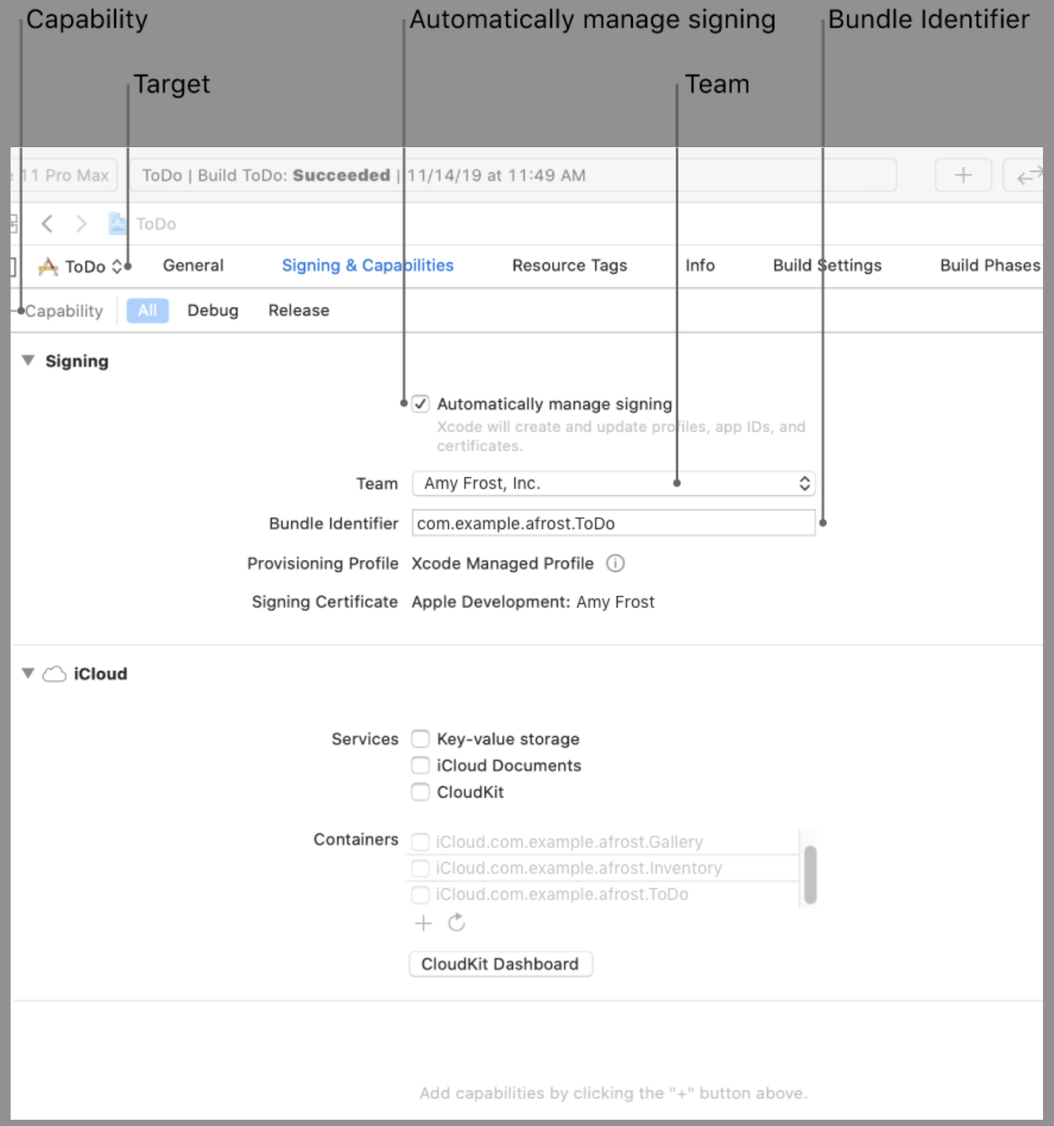
- ✅ More secure than previous options.
- ✅ They won’t forme part of your source
- ✅ Can’t be retrieved using a reverse-engineering tool

Don’t Store ApiKeys On-Device

- Store them on the server
- No server? Maybe some Apple services can help us:
  1. On-Demand Resources
  2. Background Updates: Using silent APNS
  3. **CloudKit Database:** Set apiKeys in a database in your CloudKit Dashboard

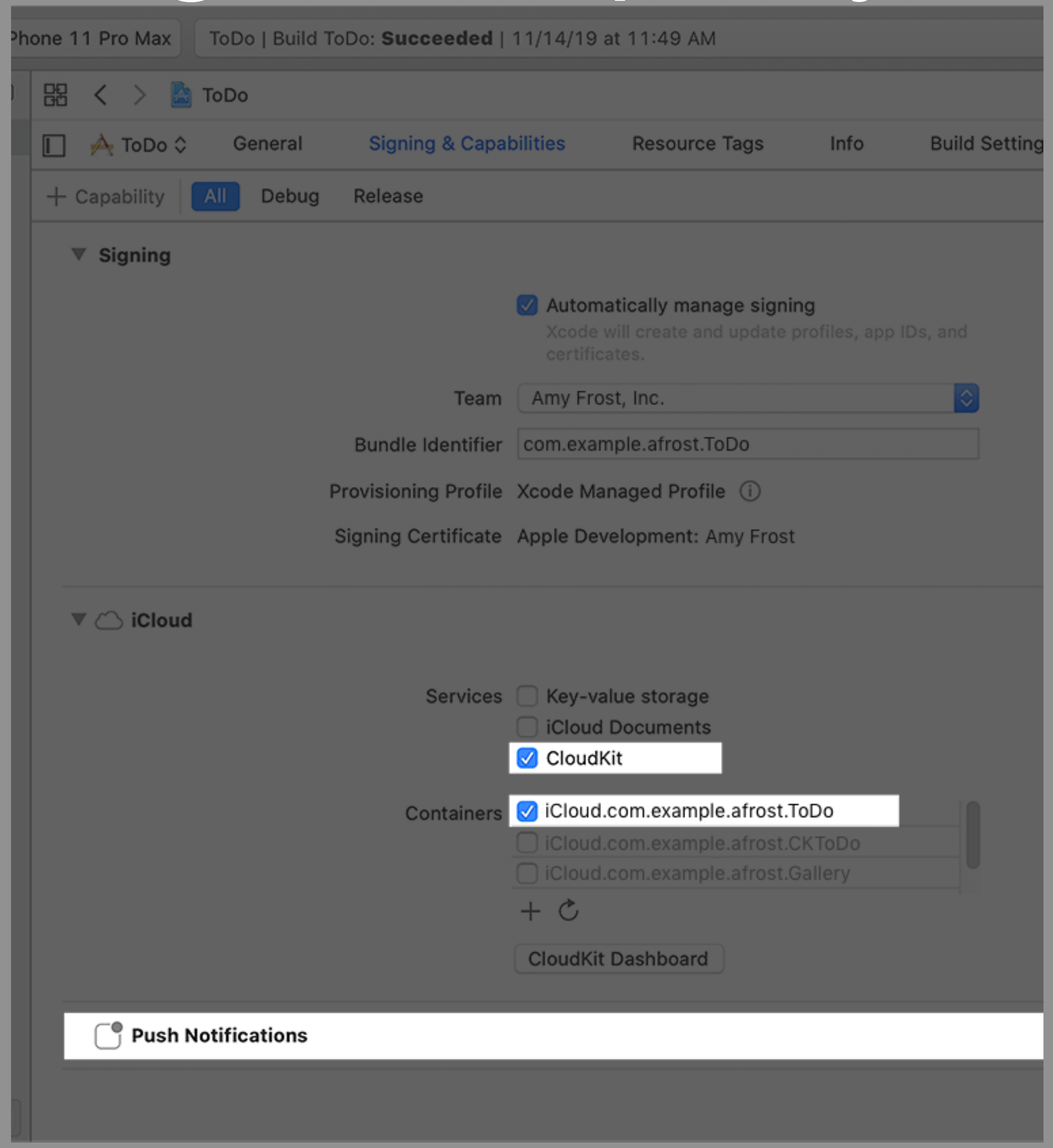
Using CloudKit as storage for ApiKeys

- Add the iCloud Capability to Your Xcode Project



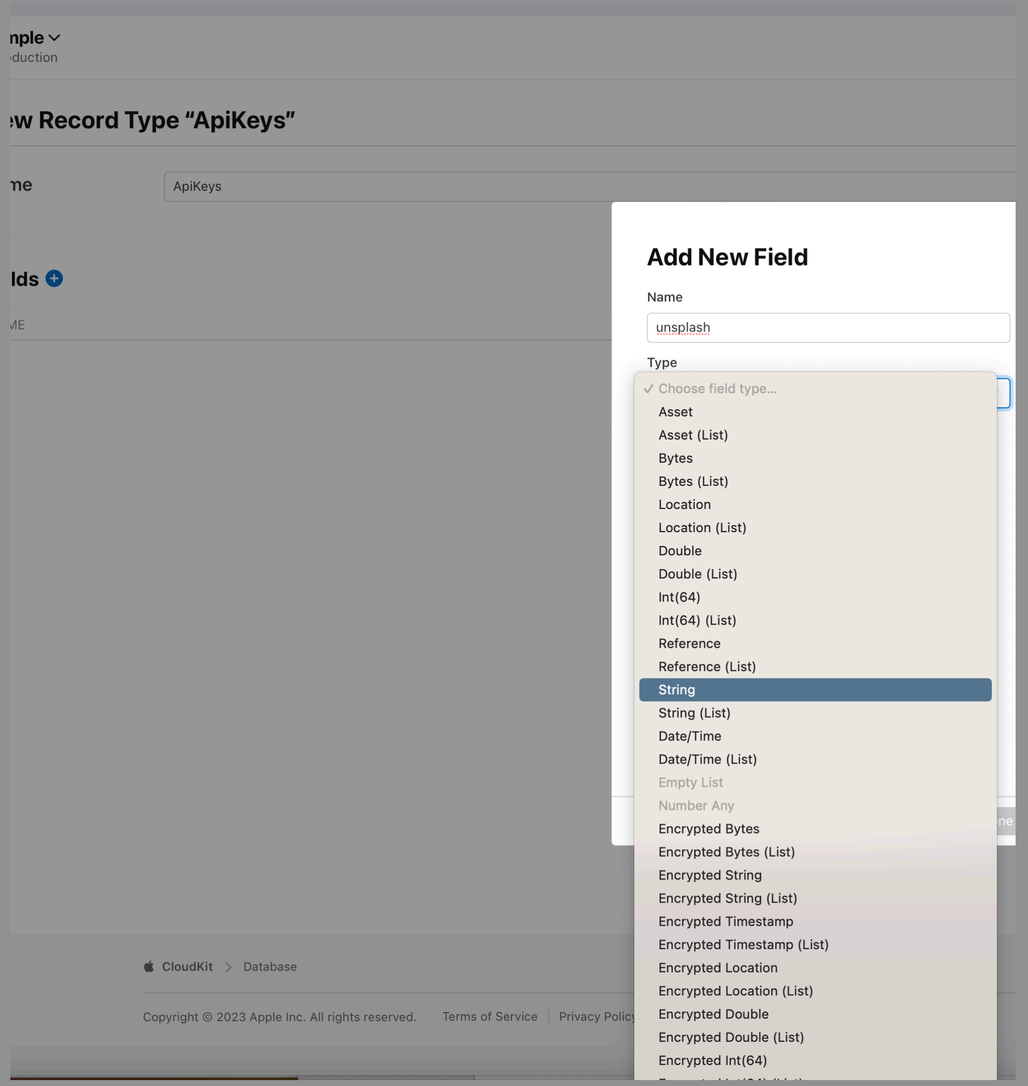
Using CloudKit as storage for ApiKeys

- Create Your Container
- Select or Create an iCloud Account for Development



Using CloudKit as storage for ApiKeys

- In Consosle, we create a new Record type
- We give it a name for example “ApiKeys” and add a new field. We’ll named “unplash”
- Set RecordName an field as queryables
- Add a new record item with the apiKey value



Using CloudKit as storage for ApiKeys

- Now with make use of CloudKit to retrieve the apiKey:

```
let apiKeyRecordID = CKRecord.ID(recordName: ckRecordId)
let publicDatabase = CKContainer.default().database(with: .public)

let record = try await publicDatabase.record(for: apiKeyRecordID)
let key = record[ckField] as? String
```

ckRecordId. The id of record we have created in iCloud console before
ckField. The name of record field que want to retrieve “unsplash”

- The apikey can be stored in Keychain using Secure Enclave as an added layer of protection

Using CloudKit as storage for ApiKeys

DEMO

Using CloudKit as storage for ApiKeys

ONE MORE THING

- We should to protect the app from the called it “MAN IN THE MIDDLE”

- We should to protect the app from the called it “MAN IN THE MIDDLE”
- It’s easiest to capture the app network traffic using a proxy between the app and the server

- We should to protect the app from the called it “MAN IN THE MIDDLE”
- It’s easiest to capture the app network traffic using a proxy between the app and the server
- We can use a tool as Charles to do it

- We should to protect the app from the called it “MAN IN THE MIDDLE”
- It’s easiest to capture the app network traffic using a proxy between the app and the server
- We can use a tool as Charles to do it
- We will use SSL Pinning, to protect app from “MAN IN THE MIDDLE”

SSL Pinning

- Certificate or a Hash from Public Key of this can be used for pinning
- Certificate can be drag ad from from Safari to a folder in Finder

SSL Pinning

- Create Hash Key:

```
 openssl s_client -servername servername -connect servername:443 | openssl x509 -pubkey -noout | openssl pkey -pubin -outform der | openssl dgst -sha256 -binary | openssl enc -base64
```

Example

```
 openssl s_client -servername api.unsplash.com -connect  api.unsplash.com:443| openssl x509 -pubkey -noout | openssl pkey -pubin -outform der | openssl dgst -sha256 -binary | openssl enc -base64

xYGlcA+pIzAH9M7nNK5YelX5qcAryQ+tbkKmBL+o4X0=X0=
```

- Alternatively Qualys SSL Labs can be used

SSL Pinning

- For pinning Certificate we'll use URLSession delegate function:

```
func urlSession(_ session: URLSession, task: URLSessionTask,
didReceive challenge: URLAuthenticationChallenge) async ->
(URLSession.AuthChallengeDisposition, URLCredential?)
```

- Validate certificate of hash of public key and let continue requests only if validation is success

Certificate Pinning

DEMO

LINKS

Unsplash
NSHister secrets
Radare
CloudKit overview
Secure Enclave docuemntation
KeychainPasswordItem Keychain helper SPM
Charles
Qualys SSL Labs

Using CloudKit as storage for ApiKeys

Using CloudKit as storage for ApiKeys

THAT IS ALL
Thanks