iCloud syncing and 2FA: friend or foe?

Vladimir Katalov
ElcomSoft Co.Ltd.
Moscow, Russia

About us: our customers

What's inside the smartphone?

- Contacts & calendars
- Call logs and text messages
- Emails and chats
- **Account and application passwords**
- **Web and Wi-Fi passwords**
- Documents, settings and databases
- Web history & searches
- Pictures and videos
- Geolocation history, routes and places
- 3rd party app data
- Cached internet data
- System and application logs
- Social network activities

Data acqusition methods

- **JTAG/chip-off**
  - there is no test access port on many devices
  - full-disk encryption makes offline attacks completely useless
- **Physical**
  - Limited compatibility
  - May alternate data
  - Data may be encrypted
- **Logical**
  - Limited compatibility
  - Bypassing screen lock is needed
- **Cloud**
  - Limited set of data
  - Need credentials
  - Legal problems

Cloud: backup, sync or just storage?

- **Problems**
  - *Different platforms (Apple, Google, Microsoft)*
  - *Many vendor-specific clouds (especially in China: 360, QQ etc)*
  - *3rd party cloud services (Dropbox, Amazon, Azure and more)*
  - *Credentials needed (password or token)*

- **Profits**
  - *No physical access needed*
  - *May be performed silently*

- **Backup**
  - No standard way to get
  - Might not be available
  - Almost all data from device
- **Sync**
  - Limited set of data
  - Most critical real-time data
  - Synced across all devices
- **Storage**
  - Only files/documents
  - Easy to access

Cloud services: backups

- Full device backups are sometimes available (Apple only ☺)
- 3rd party application data is usually not available
- Passwords are not always being saved; might be additionally encrypted
- Daily backups (in best case, until forced from the device)
- Backups cannot be forced remotely
- 3rd party software (like ours ☺) is needed
- Almost no way to manage
- Slow access, long download

Cloud services: synced data

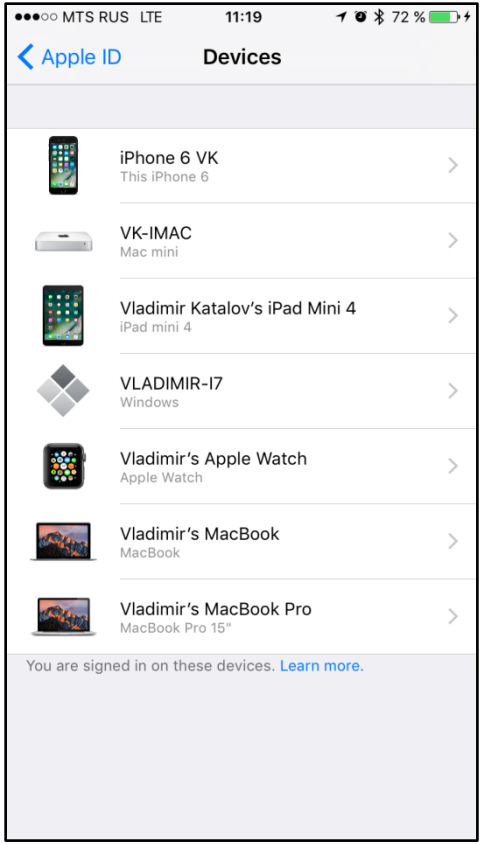
- Contacts
- Call log
- Messages (SMS, iMessage, Hangouts, Skype)
- Calendars
- Mail (only cloud-based)
- Internet activities (visited sites, searches)
- Media files (photos, videos)
- Gaming data
- **Passwords**
- Health data

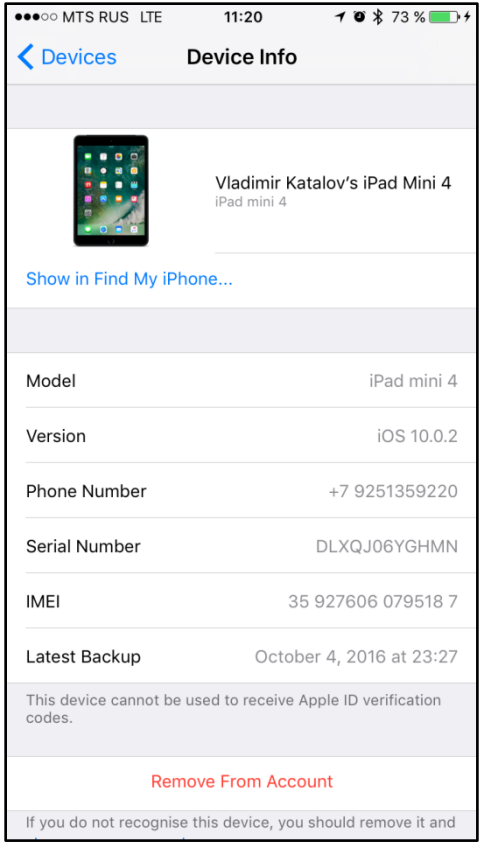
Other

- *Payment info*
- Home devices
- Wallet (Apple-specific)
- Maps (searches, bookmarks, routes)
- Books
- News, weather
- *Location data*

More (i)Cloud data

- Account information
- iCloud storage information
- Contact information (billing/shipping address, emails, credit cards (last 4 digits)
- Connected devices
- Customer service records
- iTunes (purchase/download transactions and connections, update/re-download connections, Match connections, gift cards)
- Retail and online store transactions
- Mail logs
- Family sharing data
- iMessage and FaceTime metadata
- *Deleted data?*





Cloud data by platform

	Apple	Google	Microsoft
Backups	+ (three)	Sort of (single)	Soft of (several)
Contacts/calendars/tasks	+	+	+
Call log	☺	☹	In backups only
Notes	+	+	+
Messages	-	Android N (?)	+
Mail	iCloud mail	Gmail	Outlook
Internet	Safari	Chrome	Edge
Media	iCloud Photo Library	Google Photos	OneDrive
Documents	iCloud Drive	Google Docs	OneDrive
Location	Current/last	Current, history	Current, history
3rd party apps data	iCloud Drive	Google Drive	OneDrive
Other	Health (?), Wallet	Dashboard and more	HealthVault, Skype

Cloud passwords, keys etc

	Apple	Google	Microsoft
Wi-Fi	+	+	In backups
Web sites	+	+	+
Credit cards	+	*CVV is needed*	?
Credit cards (2)	*Apple Pay (Wallet): last 4 digits only*	*Google Pay (?)*	Wallet (?)
App-specific	*It depends*	*Sometimes* ☺	-
Authentication tokens	+	+	-
Encryption keys	+	-	-
Certificates	+	-	-
Autocomplete	+	+	+

Apple keychains

- **iOS keychain**
  - Local (encrypted backup)
  - Local (not encrypted backup)
  - iCloud

  View: Settings | Safari | Passwords,
  Settings | Safari | AutoFill
  Protection: it depends
  Decrypt/export: no way (3rd party software only)

- **OS X (macOS) keychain**

  View: Keychain utility (one by one)
  Protection: password (by default, same as log on)
  Decrypt/export: 3rd party software only

- **iCloud keychain**

  View: Only when/if synced with local device
  Protection: well, strong ☺
  Decrypt/export: no way

Backup vs iCloud keychains

	Backup	iCloud
Wi-Fi	+	+
Web sites	+	+
Credit cards	+	+
App-specific	+	*It depends*
AirPlay/AirPort	+	+
Encryption keys & tokens	+	*It depends*
Autocomplete	+	-

Keychain in iCloud backups have most data encrypted with device-specific key

iOS keychain – passwords (Wi-Fi, email, web form)

<Name>AirPort (**AP name**)</Name>
<Service>AirPort</Service>
<Account>**AP name**</Account>
<Data>**AP password**</Data>
<Access Group>apple</Access Group>
<Creation Date>20121231120800.529226Z</Creation Date>
<Modification Date>20121231120800.529226Z</Modification Date>
<Protection Class>CLASS: 7</Protection Class>

<Name>imap.gmail.com (vkatalov@gmail.com)</Name>
<Server>imap.gmail.com</Server>
<Account>**email**</Account>
<Data>**password**</Data>
<Protocol>IMAP</Protocol>
<Port>143</Port>
<Access Group>apple</Access Group>
<Creation Date>20121231124745.097385Z</Creation Date>
<Modification Date>20121231124745.097385Z</Modification Date>
<Protection Class>CLASS: 7</Protection Class>

<Name>accounts.google.com (**email**)</Name>
<Server>accounts.google.com</Server>
<Account>**email**</Account>
<Data>**password**</Data>
<Protocol>HTTPS</Protocol>
<Authentication Type>form</Authentication Type>
<Description>Web form password</Description>
<Access Group>com.apple.cfnetwork</Access Group>
<Creation Date>20150705071047.78112Z</Creation Date>
<Modification Date>20150805133813.889686Z</Modification Date>
<Label>accounts.google.com (**email**)</Label>
<Protection Class>CLASS: 6</Protection Class>

iOS keychain (credit card data)

<Name>SafariCreditCardEntries (BBA00CB1-9DFA-4964-B6B8-3F155D88D794)</Name>
<Service>SafariCreditCardEntries</Service>
<Account>BBA00CB1-9DFA-4964-B6B8-3F155D88D794</Account>
<Data>
 <Dictionary>
  <CardholderName>**NAME**</CardholderName>
  <ExpirationDate>**DATE**</ExpirationDate>
  <CardNameUIString>Visa</CardNameUIString>
  <CardNumber>**NUMBER**</CardNumber>
 </Dictionary>
</Data>
<Comment>This keychain item is used by Safari to automatically fill credit card information in web forms.</Comment>
<Access Group>com.apple.safari.credit-cards</Access Group>
<Creation Date>20131016100432.283795Z</Creation Date>
<Modification Date>20150826181627.118539Z</Modification Date>
<Label>Safari Credit Card Entry: Visa</Label>
<Protection Class>CLASS: 6</Protection Class>

iOS [backup] keychain protection classes

kSecAttrAccessibleAfterFirstUnlock (7)
The data in the keychain item cannot be accessed after a restart until the device has been unlocked once by the user.

kSecAttrAccessibleAfterFirstUnlockThisDeviceOnly (10)
The data in the keychain item cannot be accessed after a restart until the device has been unlocked once by the user.

kSecAttrAccessibleAlways (8)
The data in the keychain item can always be accessed regardless of whether the device is locked.

kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly
The data in the keychain can only be accessed when the device is unlocked. Only available if a passcode is set on the device.

kSecAttrAccessibleAlwaysThisDeviceOnly (11)
The data in the keychain item can always be accessed regardless of whether the device is locked.

kSecAttrAccessibleWhenUnlocked (6)
The data in the keychain item can be accessed only while the device is unlocked by the user.

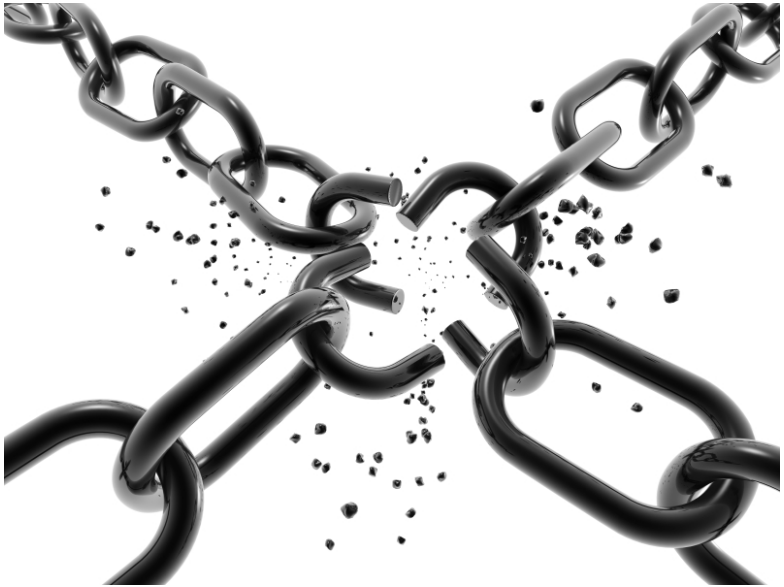
kSecAttrAccessibleWhenUnlockedThisDeviceOnly (9)
The data in the keychain item can be accessed only while the device is unlocked by the user.

- *xxxThisDeviceOnly*: encrypted using device-specific hardware key (can be extracted from 32-bit devices only)
- All others: in password-protected local backups, encrypted with the key derived from backup password

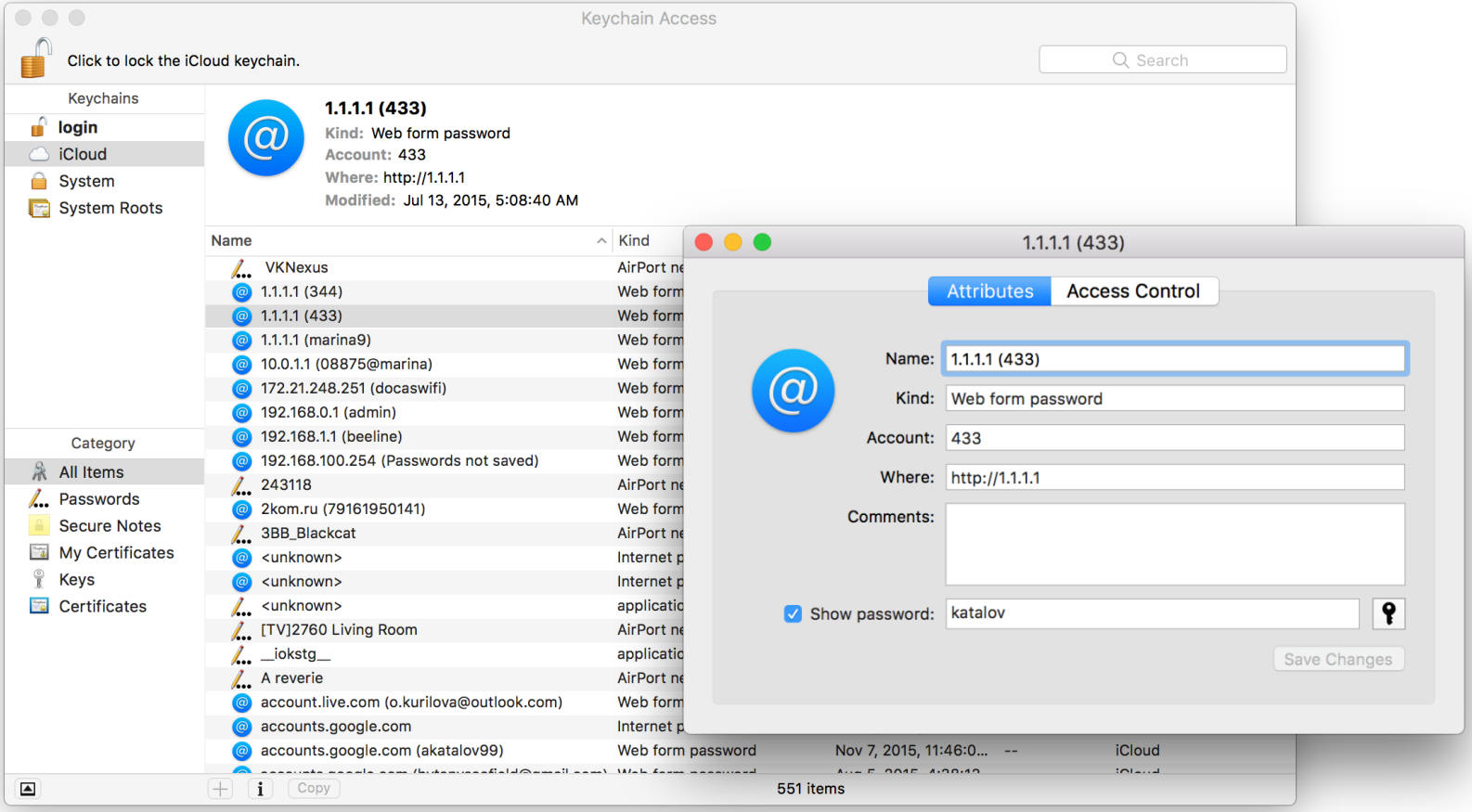
iTunes backup password breaking

- Get manifest.plist
- Get *BackupKeyBag*
- Check password
  - iOS 3
    - pbkdf2_sha1(2,000)
  - iOS 4 to 10.1 (but 10.0)
    - Same as above, but 10,000 iterations
  - iOS 10.0
    - Same as above works
    - *Single sha256 hash is also stored*
  - iOS 10.2+
    - pbkdf2_sha256(10,000,000)
    - pbkdf2_sha1(10,000)
  - Unwrap AES key from KeyBag
- Decrypt keychain (+other files?)

Hashes are salted, so no rainbow tables ☹



macOS keychain



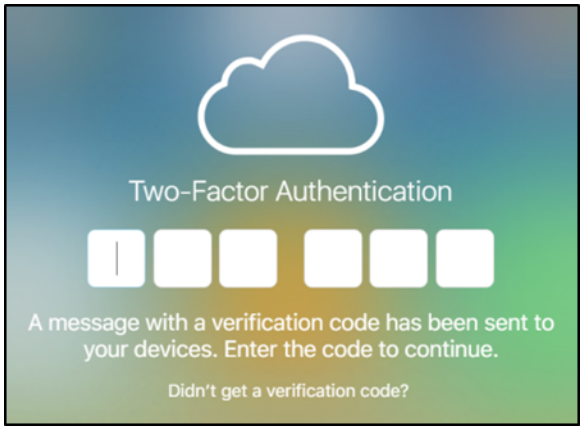
iCloud data protection

https://support.apple.com/en-us/HT202303

Most of the data: *A minimum of 128-bit AES encryption*
iCloud Keychain: *Uses 256-bit AES encryption to store and transmit passwords and credit card information. Also uses elliptic curve asymmetric cryptography and key wrapping.*

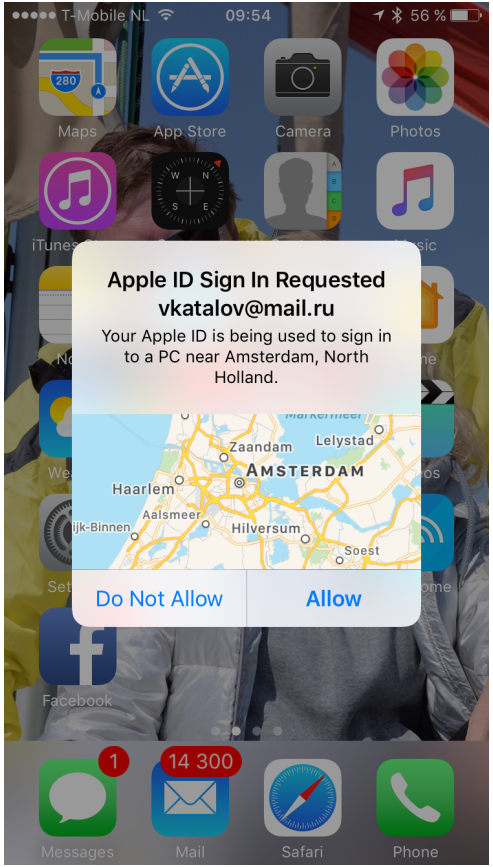
Key is stored along with the data (except just the iCloud keychain)!

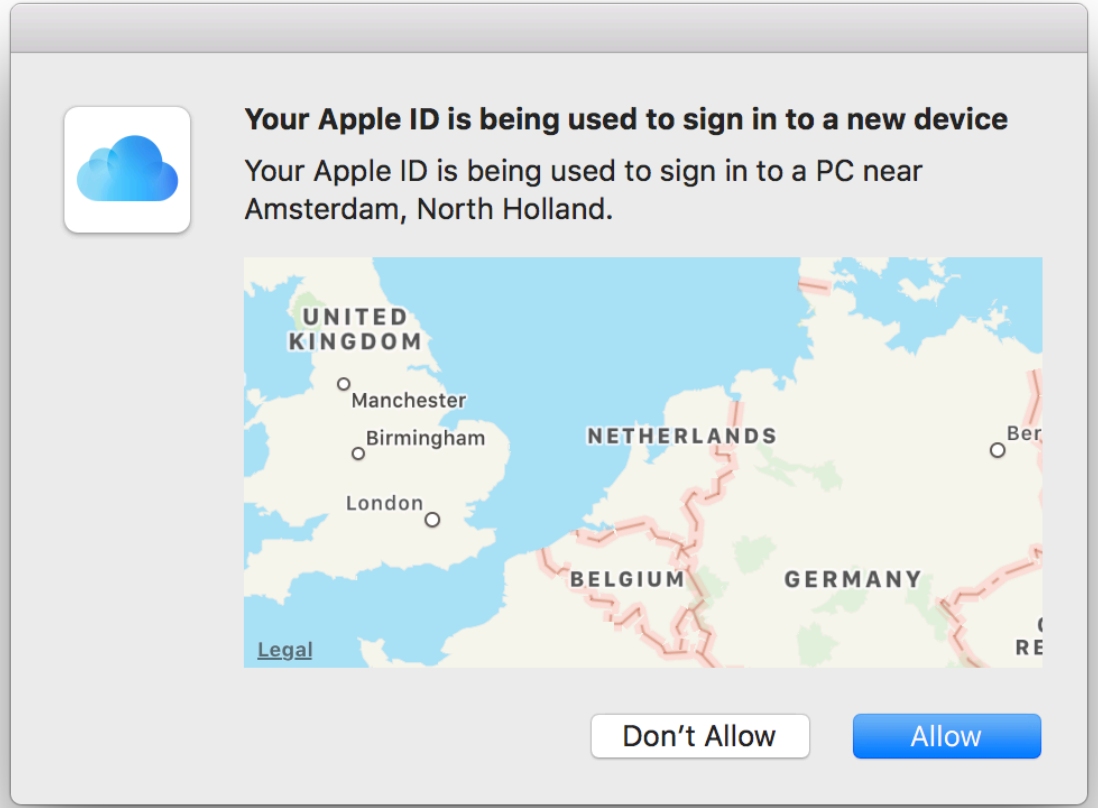
- Notification to email when the data is accessed
- Account might be blocked due to suspicious activity (new!)
- Two-step verification (legacy, not recommended)
- **Two-factor authentication**
  - Immediate push notification to all trusted devices
  - Have to allow access
  - Security code
    - As push notification
    - By SMS to trusted phone number
    - Generated by trusted device



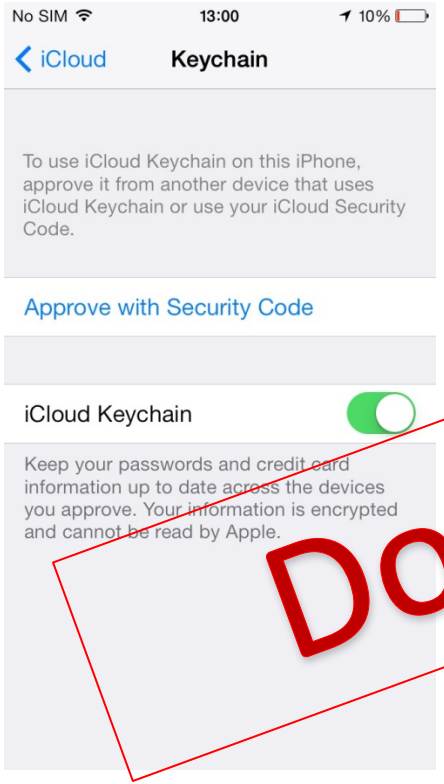
Workaround for 2FA: use authentication token from the device (iPhone/iPad/iPod), PC or Mac

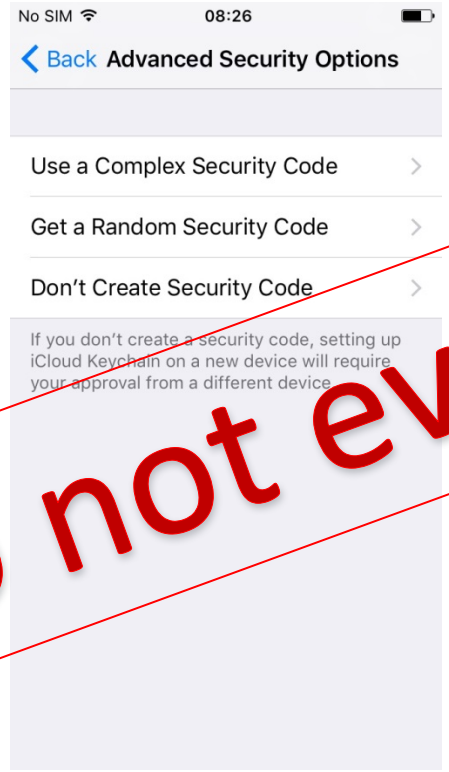
iCloud sign-in

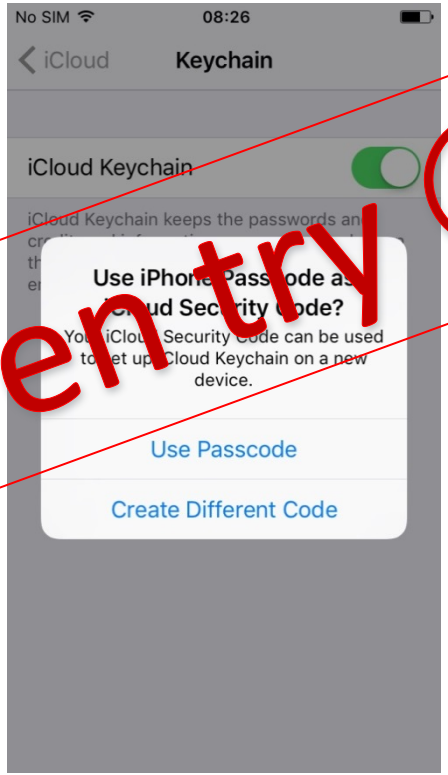


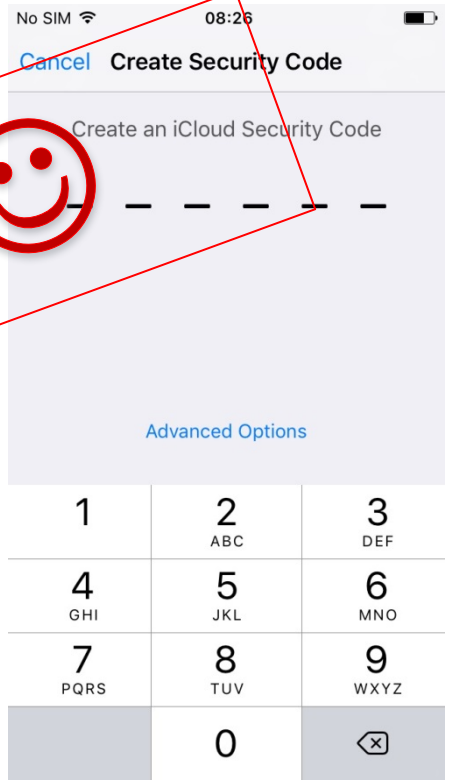


Set up iCloud keychain – no 2FA



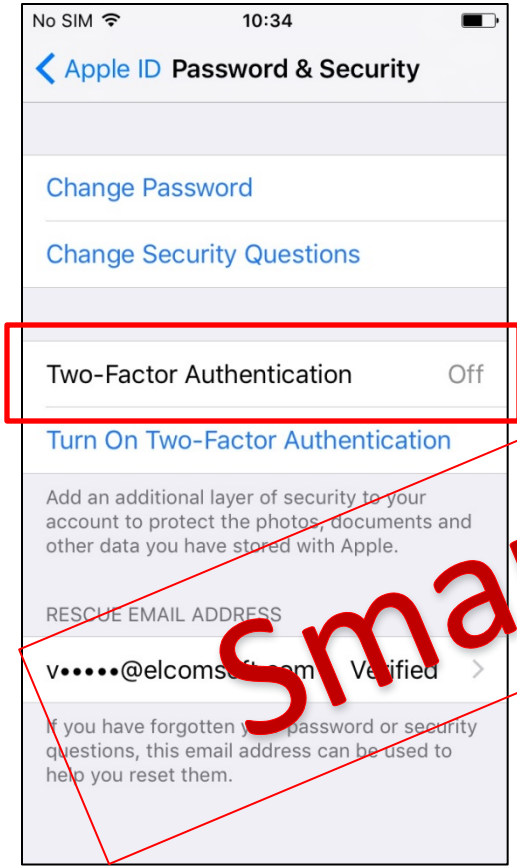


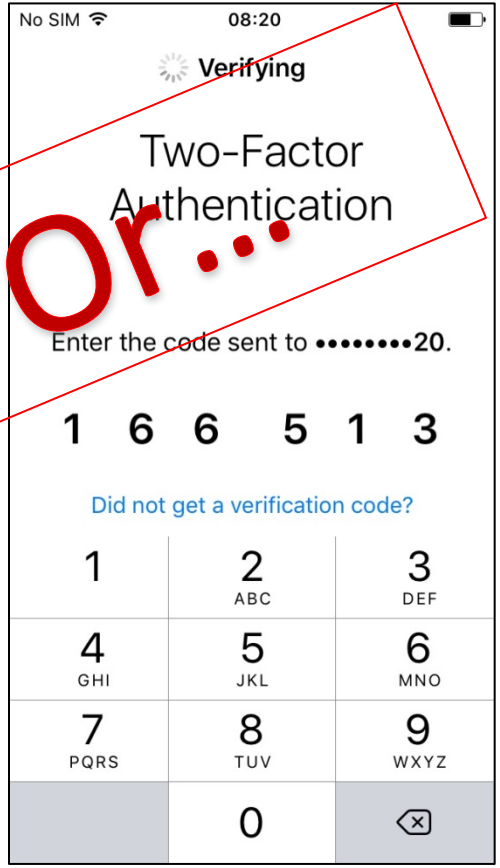




Do not even try

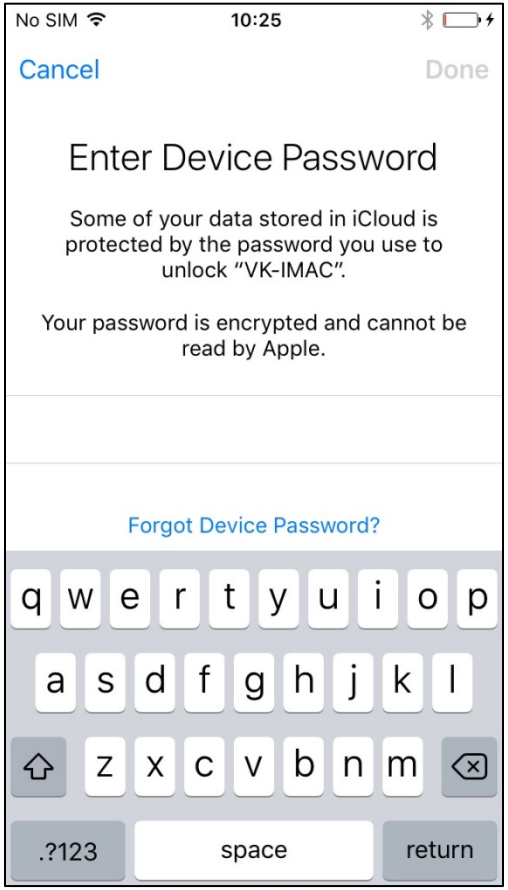
Set up 2FA

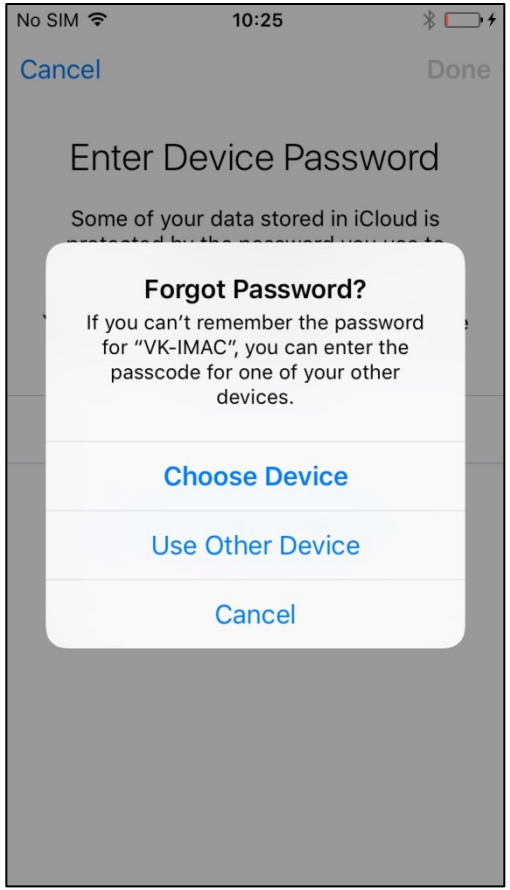


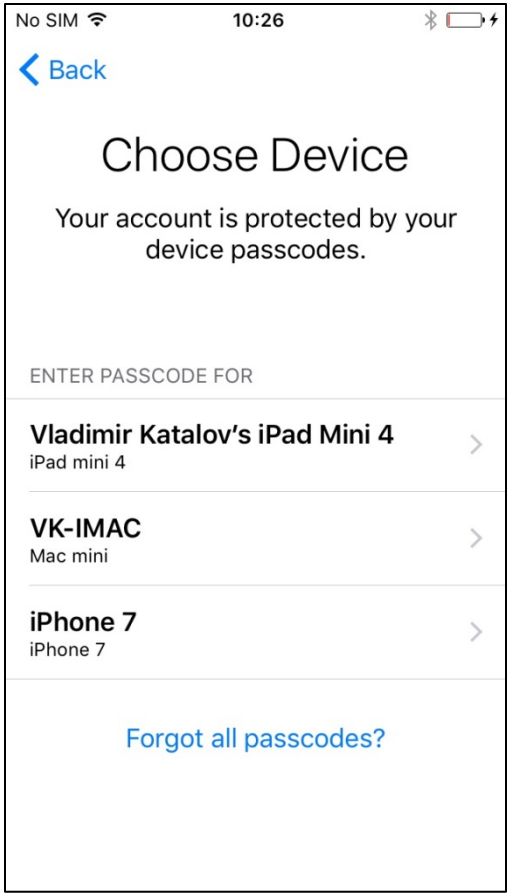


Smart choice! Or…

Set up iCloud keychain –2FA





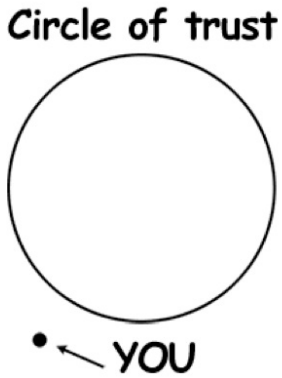


iCloud keychain inside out

iOS Security Guide:
https://www.apple.com/business/docs/iOS_Security_Guide.pdf

- **Keychain syncing**
  - Circle of trust
  - Public key: syncing identity (specific to device)
  - Private key (elliptical P256), derived from iCloud password
  - Each synced item is encrypted specifically for the device (cannot be decrypted by other devices)
  - Only items with *kSecAttrSynchronizable* are synced

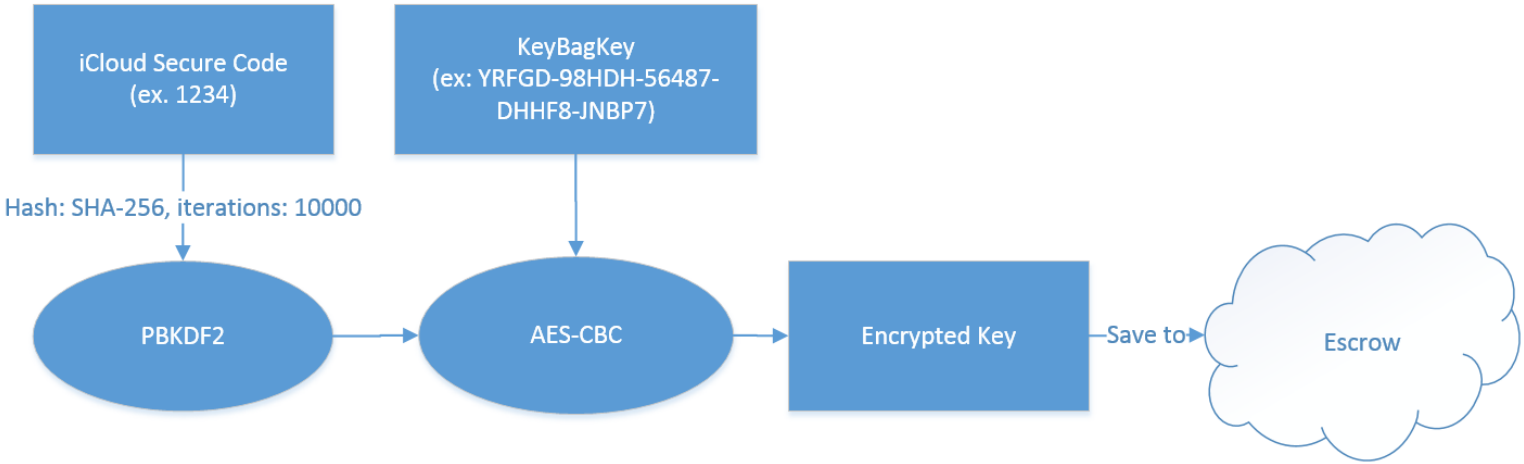
- **Keychain recovery**
  - Secure escrow service (*optional*)
  - iCloud security code is needed (not with 2FA!)
  - Hardware Security Module (WTF is that? ☺)



Escrow proxy architecture

Escrow proxy

- SRP (Secure Remote Password) protocol
- Safe from MITM
- Does not need password to be transferred in plain text
- Does not keep password on server



Escrow proxy protocol

- **enroll**
  to add new records
- **get_records**
  to get data
- **get_sms_targets**
  get trusted phone numbers
- **generate_sms_challenge**
  start verification by sms
- **srp_init**
  first authentication step under SRP
- **Recover**
  second SRP step

<u>What we can get from Escrow record</u>

- Info on key used for protection
- Number of failed retries
- Device data (model, version, password strength)
- List of keys for KeyBag decryption
- Protected Storage Services list


```
<plist version="1.0">
<dict>
    <key>BackupKeybagDigest</key>
    <data>
    JAfmiRjR3IUw5SQga2J1sh40coQ=
    </data>
    <key>ClientMetadata</key>
    <dict>
        <key>SecureBackupMetadataTimestamp</key>
        <string>2017-03-31 14:10:22</string>
        <key>SecureBackupNumericPassphraseLength</key>
        <integer>0</integer>
        <key>SecureBackupUsesComplexPassphrase</key>
        <integer>1</integer>
        <key>SecureBackupUsesNumericPassphrase</key>
        <false/>
        <key>device_mid</key>
        <string>mIZ3Nrg+ISj2...rPx9UsEcOotM0NZ</string>
        <key>device_model</key>
        <string>MacBook Air</string>
        <key>device_model_class</key>
        <string>MacBook Air</string>
        <key>device_model_version</key>
        <string>MacBookAir3,2</string>
        <key>device_name</key>
        <string>omgwtf</string>
        <key>device_platform</key>
        <integer>2</integer>
    </dict>
    <key>SecureBackupUsesMultipleiCSCs</key>
    <true/>
    <key>com.apple.securebackup.timestamp</key>
    <string>2017-03-31 14:10:22</string>
    <key>peerInfo</key>
    <data>
    MIIECzGCA74w...kClZJEdg==
    </data>
</dict>
</plist>
```

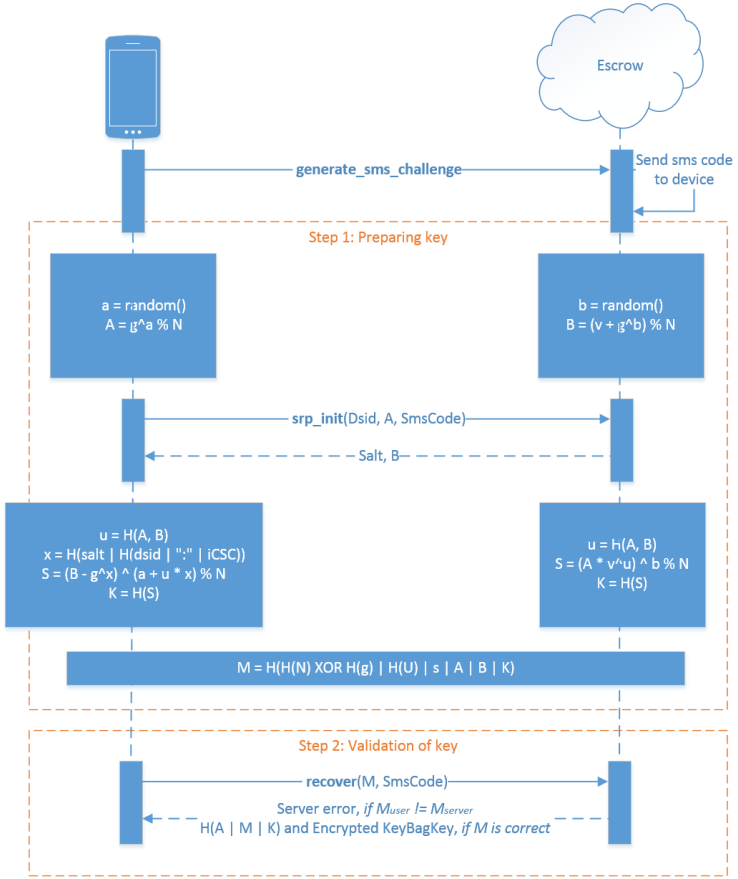
SRP protocol

iCSC - iCloud Secure Code
H – SHA256
N, g – 2048-bit generator of the multiplicative group (RFC 5054)

The user enroll password verifier and salt to EscrowCache. EscrowCache stores password verifier and salt.

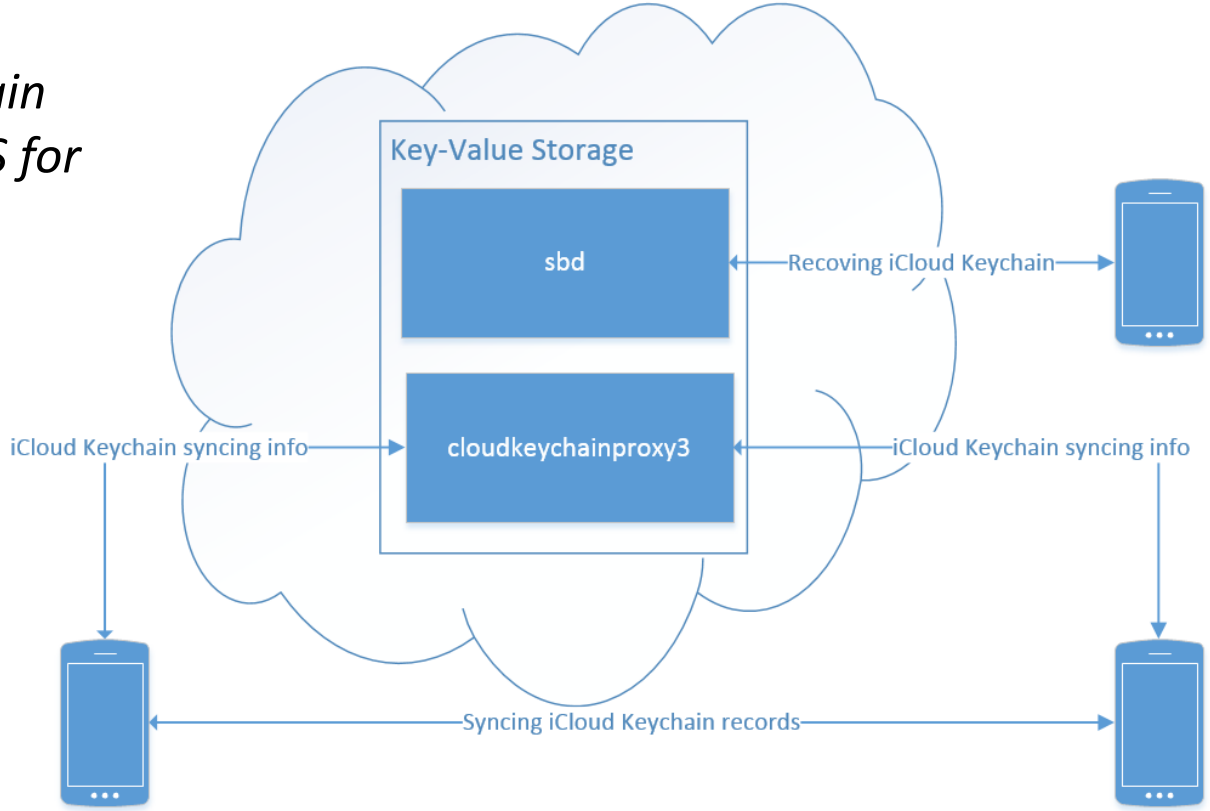
<salt> = random()
x = SHA(<salt> | SHA(<dsid> | ":" | <iCSC>))
<password verifier> = v = g^x % N

If *com.apple.securebackup* record exists, that means that iCloud Security Code is set. Otherwise, EscrowProxy contains *com.apple.icdp.record.hash_of_device* records, so iCloud Keychain can be synced when one of device passwords is provided.



Key-Value Storage

If 2FA is enabled, keychain data are copied into KVS for circle synchronization



Keychain recovery

- **GetAccountSettings (get token)**

- **Sync**

Registry-version: if empty, get the whole keychain (plus current state); if not, get only new data
Returns keychain and BackupKeyBag

- **SRP authentication**

get_sms_targets
generate_sms_challenge

- **srp_init**

Get data for Recovery request

- **Recover** (get *KeyBagKey*)

- **Decrypt KeyBagKey**
- **Decrypt KeyBag**
- **Decrypt KeyChain**

If we have 2FA passed and obtained the token:

- **No need to have trusted device**
- **No iCloud Security Code**
- **No notification to trusted devices**
- **Get all the passwords and CC data ☺**
- **One of device' passcodes is still needed ☹**

iCloud Keychain access - alternatives

- **Add new device to “circle of trust”**
  - Need to pass 2FA
  - Notifications to all devices

- **Get iCloud backup**
  - Same as above
  - Might not exist (or too old)
  - Need to get *securityd* key (physical acquistion only, 32-bit devices)
  - No real-time access

- **Get local backup**
  - Physical access to PC/Mac is needed
  - Backup might be password-protected

- **Break *circle* protocol?** ☺

Thanks!
Questions?