Breaking into the iCloud Keychain

Vladimir Katalov
ElcomSoft Co.Ltd.
Moscow, Russia

[Some of] our customers

What do we want to hack today?

1. All user’s passwords
2. Credit card data

What’s inside the smartphone?

(tip: almost everything)

- Contacts & calendars
- Call logs and text messages
- Emails and chats
- **Account and application passwords**
- **Web and Wi-Fi passwords**
- **Credit card data**
- 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 | Cloud acquisition pros and cons

- **JTAG/chip-off**
  - No test access port on many devices
  - Full disk encryption
- **Physical**
  - Limited compatibility
  - Data may be encrypted
- **Logical**
  - Limited compatibility
  - Bypassing screen lock is needed
- **Cloud**
  - Limited set of data // ***oh, really?*** ☺
  - Need credentials
  - Legal problems

- **Problems**
  - *Different platforms (Apple, Google, Microsoft)*
  - *Many vendor-specific clouds*
  - *3rd party cloud services*
  - *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 [iCloud]

- Full device backups are sometimes available
- 3rd party application data is usually not available
- Passwords are additionally encrypted with hardware-specific key
- Daily backups (in best case, until forced from the device)
- Backups cannot be forced remotely
- 3rd party software is needed
- Almost no way to manage
- Slow access, long download
- Account might be locked due to ‘suspicious activity’

Cloud services: synced data [iCloud]

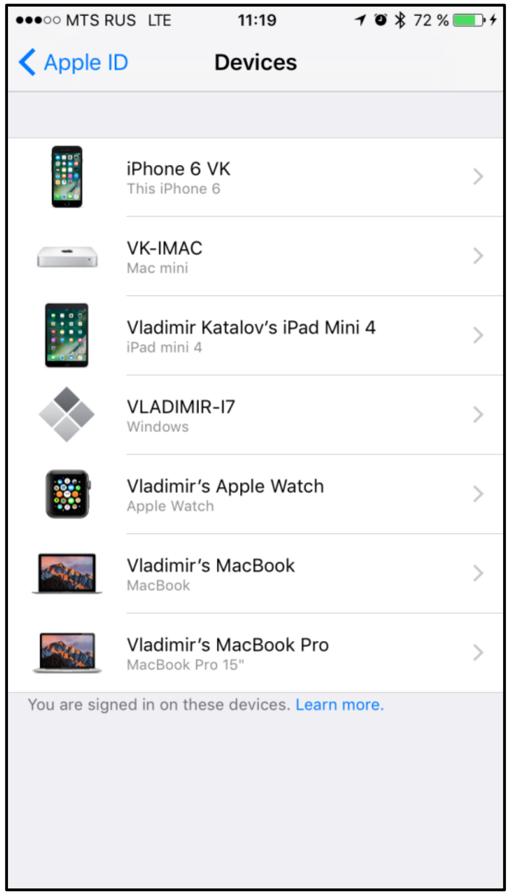
- Contacts
- **Call log**
- Messages (SMS/iMessage, CallKit-compatible apps)
- Calendars
- Mail (only cloud-based)
- Internet activities (visited sites, searches)
- Media files (photos, videos)
- Gaming data
- **Passwords**
- Health data
- **Credit cards**

Other

- ApplePay
- Home devices
- Wallet
- Maps (searches, bookmarks, routes)
- Books
- News, weather
- *Location data*

More iCloud 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?*

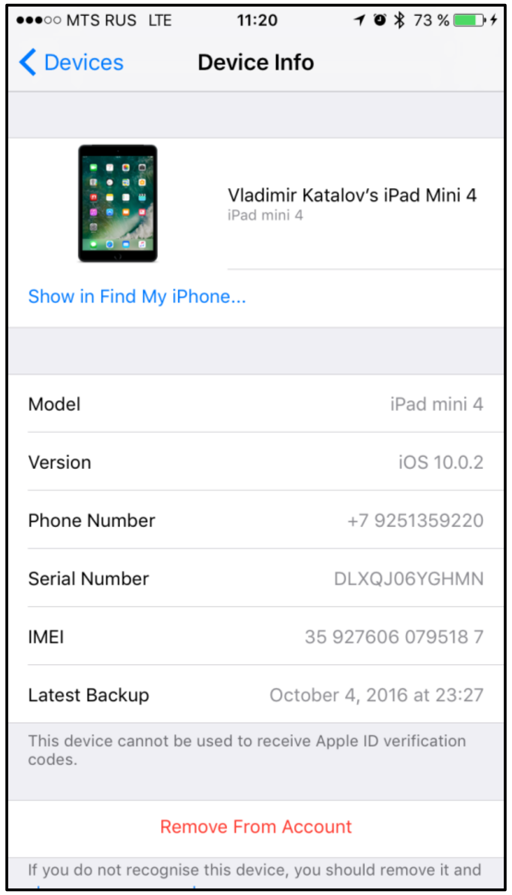


●●●○○ MTS RUS LTE 11:19 72 %

< Apple ID **Devices**

iPhone 6 VK	This iPhone 6
VK-IMAC	Mac mini
Vladimir Katalov's iPad Mini 4	iPad mini 4
VLADIMIR-I7	Windows
Vladimir's Apple Watch	Apple Watch
Vladimir's MacBook	MacBook
Vladimir's MacBook Pro	MacBook Pro 15"

You are signed in on these devices. Learn more.



●●●○○ MTS RUS LTE 11:20 73 %

< Devices **Device Info**

Vladimir Katalov's iPad Mini 4
iPad mini 4

Show in Find My iPhone...

Model	iPad mini 4
Version	iOS 10.0.2
Phone Number	+7 9251359220
Serial Number	DLXQJ06YGHMN
IMEI	35 927606 079518 7
Latest Backup	October 4, 2016 at 23:27

This device cannot be used to receive Apple ID verification codes.

Remove From Account

If you do not recognise this device, you should remove it and

Apple keychains

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

  View (iOS 10): Settings | Safari | Passwords, AutoFill
  View (iOS 11): Settings | Accounts & Passwords | App & Website Passwords
  Protection: <u>it depends</u>
  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: ?

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.

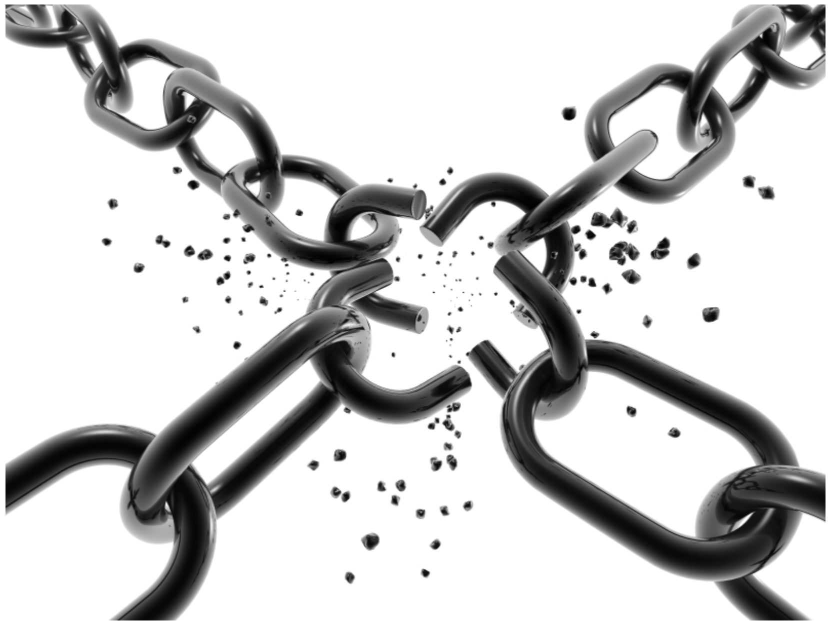
kSecAttrAccessibleWhenUnlockedThisDeviceOnly (9)
The data in the keychain item can be accessed only while the device is unlocked by the user.

- *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

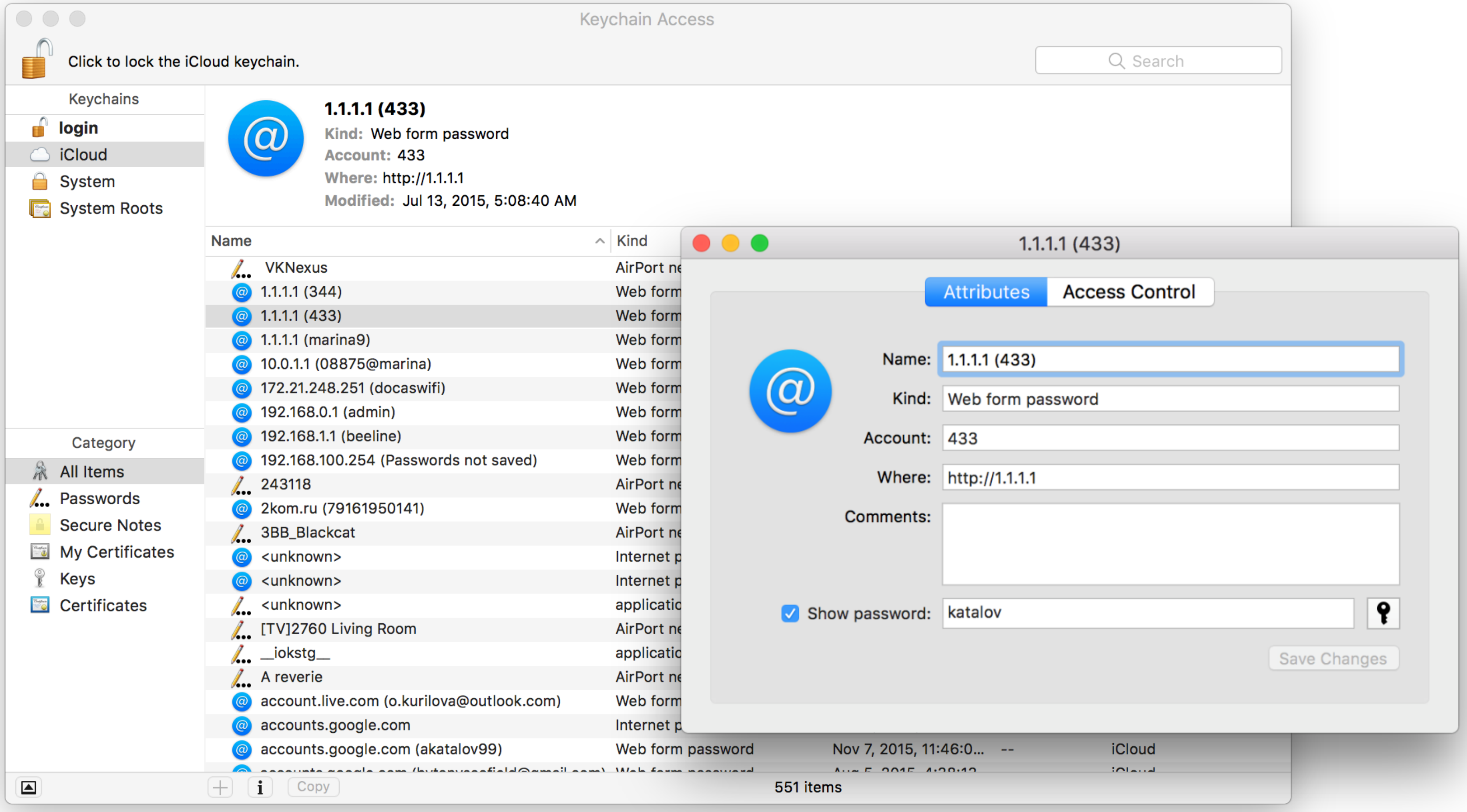
iTunes backup password breaking

- 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



Keychain Access

Click to lock the iCloud keychain.

1.1.1.1 (433)
Kind: Web form password
Account: 433
Where: http://1.1.1.1
Modified: Jul 13, 2015, 5:08:40 AM

Name	Kind
VKNexus	AirPort ne
1.1.1.1 (344)	Web form
1.1.1.1 (433)	Web form
1.1.1.1 (marina9)	Web form
10.0.1.1 (08875@marina)	Web form
172.21.248.251 (docaswifi)	Web form
192.168.0.1 (admin)	Web form
192.168.1.1 (beeline)	Web form
192.168.100.254 (Passwords not saved)	Web form
243118	AirPort ne
2kom.ru (79161950141)	Web form
3BB_Blackcat	AirPort ne
<unknown>	Internet p
<unknown>	Internet p
<unknown>	applicatio
[TV]2760 Living Room	AirPort ne
__iokstg__	applicatio
A reverie	AirPort ne
account.live.com (o.kurilova@outlook.com)	Web form
accounts.google.com	Internet p
accounts.google.com (akatalov99)	Web form password Nov 7, 2015, 11:46:0... -- iCloud

551 items

1.1.1.1 (433)

Name:	1.1.1.1 (433)
Kind:	Web form password
Account:	433
Where:	http://1.1.1.1
Comments:	
[✓] Show password:	katalov

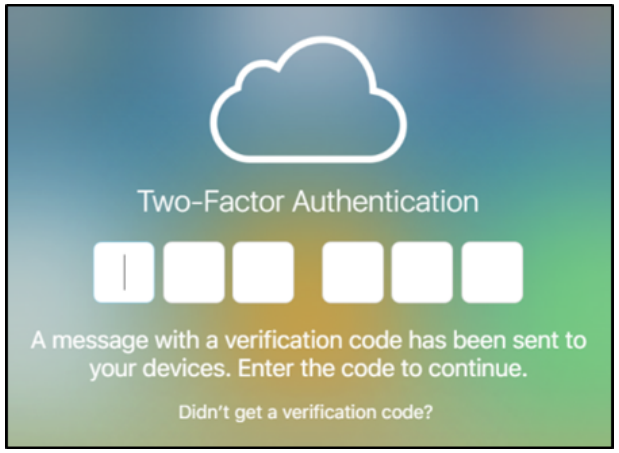
iCloud data protection

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)!

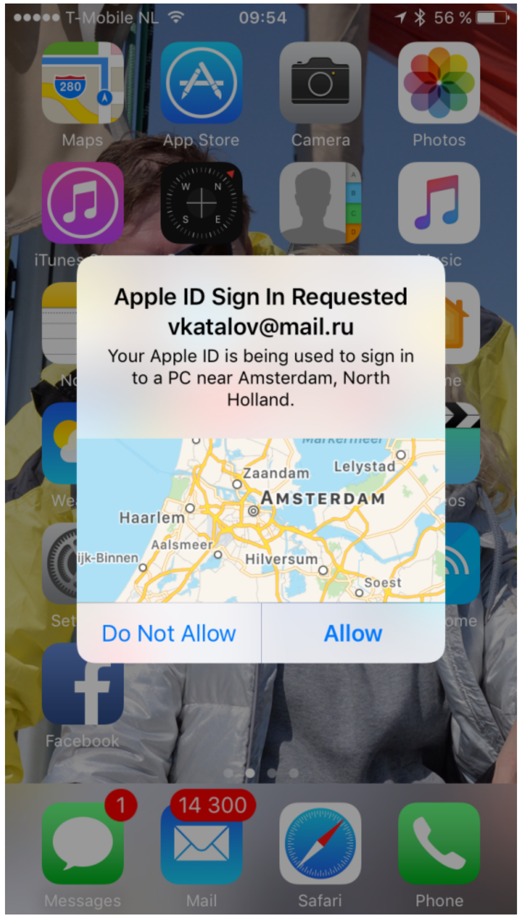
- Notification to email when the data is accessed
- Account might be blocked due to suspicious activity (new!)
- Two-step verification (legacy, not recommended)
- **Two-factor authentication**
  - Immediate push notification to all trusted devices
  - Have to allow access
  - Security code
    - As push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

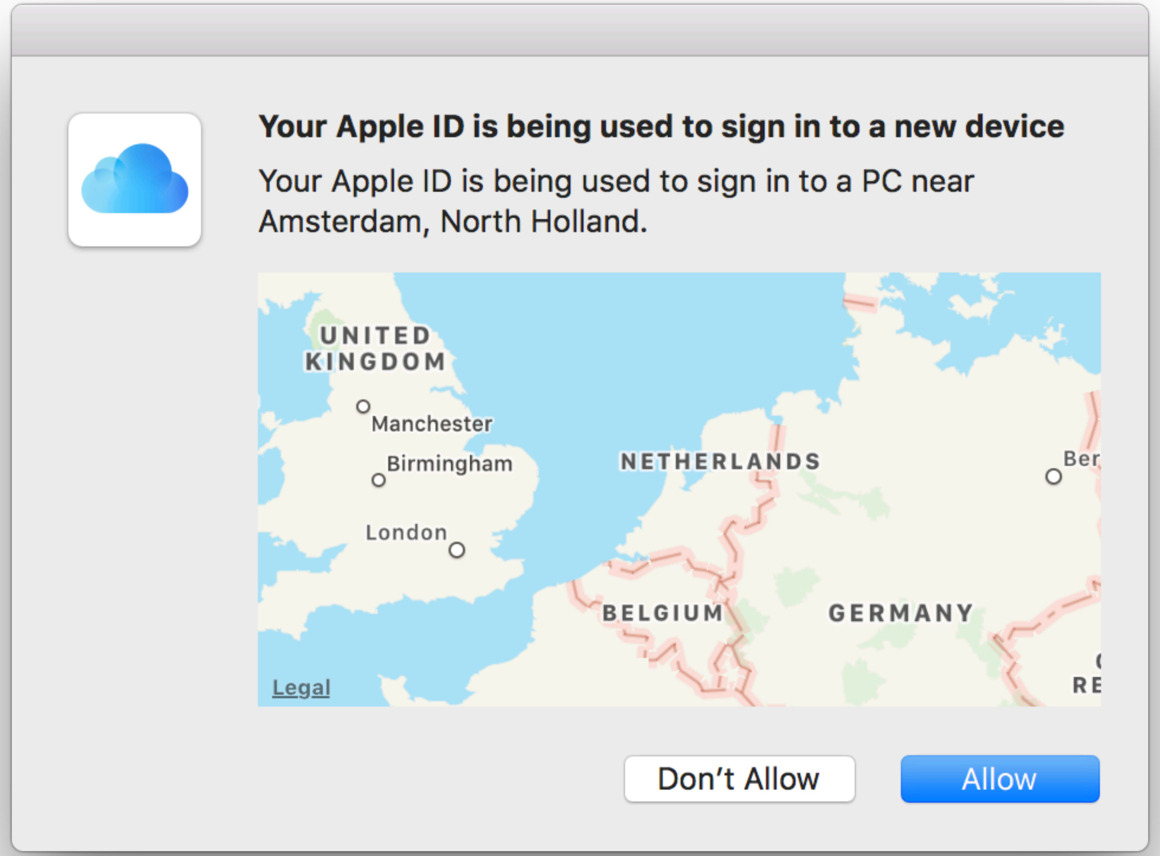


A message with a verification code has been sent to your devices. Enter the code to continue.

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

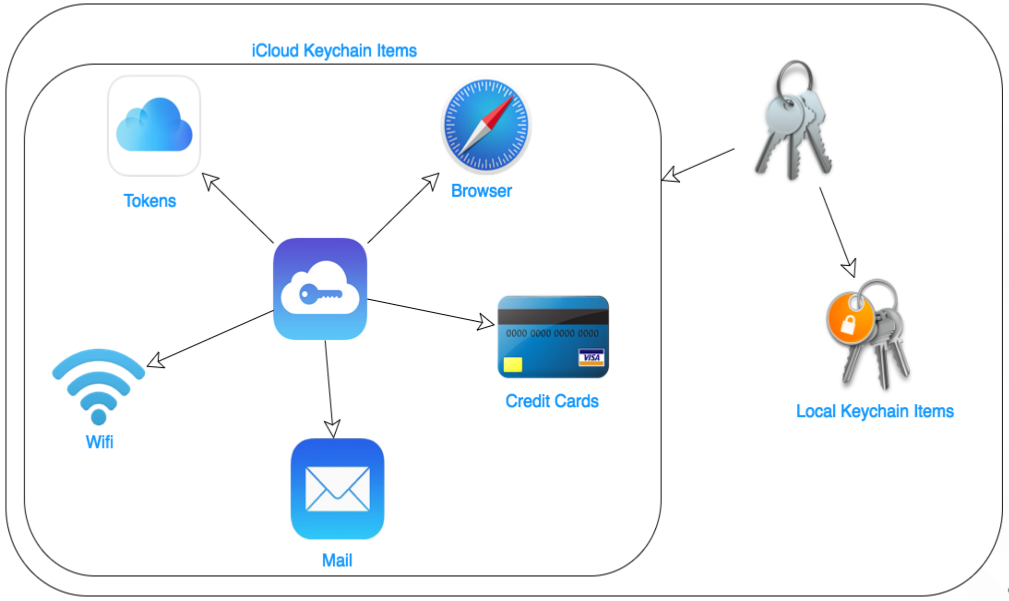
iCloud sign-in

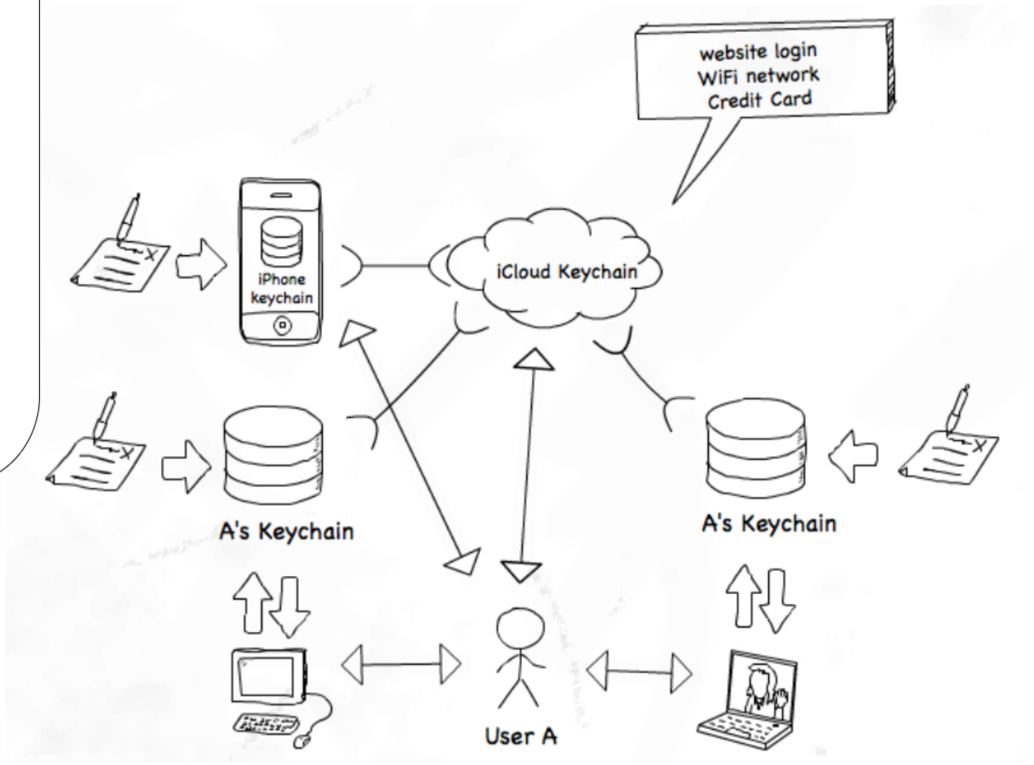

Apple ID Sign In Requested
vkatalov@mail.ru
Your Apple ID is being used to sign in to a PC near Amsterdam, North Holland.


Your Apple ID is being used to sign in to a new device

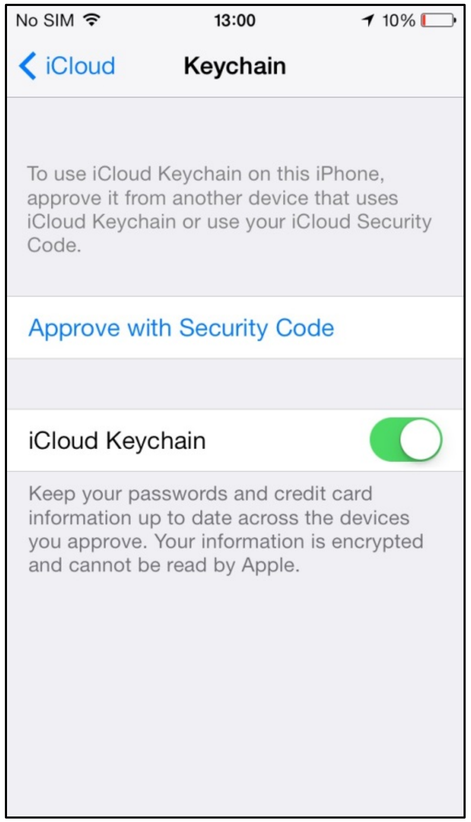
Your Apple ID is being used to sign in to a PC near Amsterdam, North Holland.

About iCloud keychain





Set up iCloud keychain – no 2FA



No SIM 13:00 10%

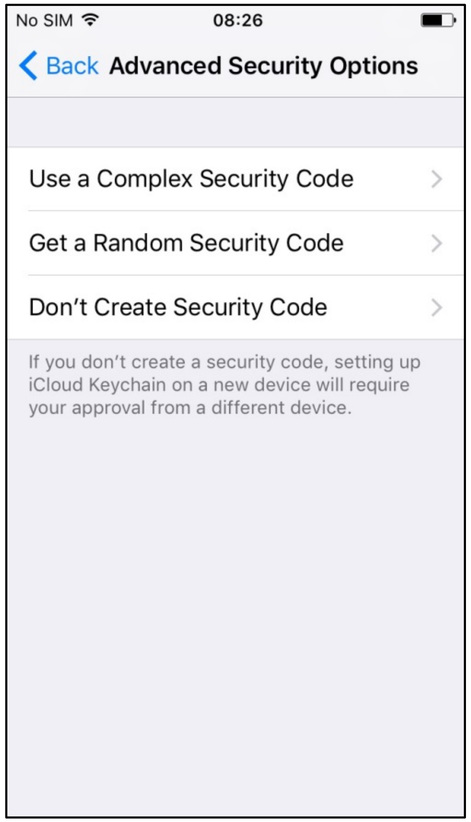
iCloud Keychain

To use iCloud Keychain on this iPhone, approve it from another device that uses iCloud Keychain or use your iCloud Security Code.

Approve with Security Code

iCloud Keychain

Keep your passwords and credit card information up to date across the devices you approve. Your information is encrypted and cannot be read by Apple.



No SIM 08:26

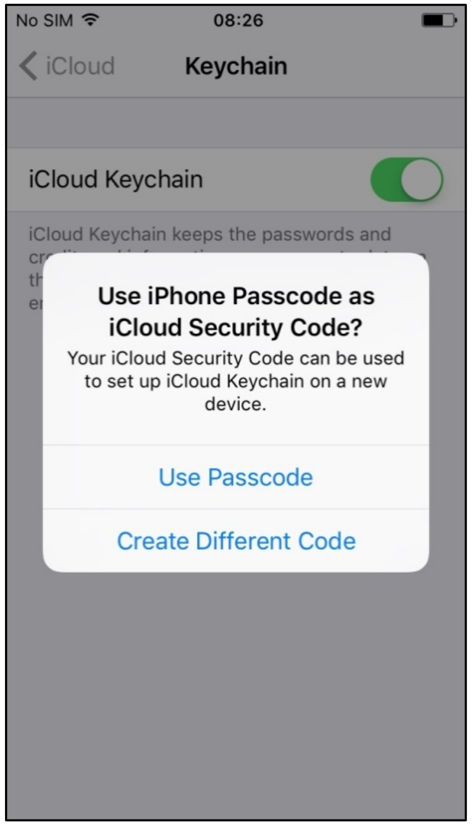
Back Advanced Security Options

Use a Complex Security Code

Get a Random Security Code

Don’t Create Security Code

If you don’t create a security code, setting up iCloud Keychain on a new device will require your approval from a different device.



No SIM 08:26

iCloud Keychain

iCloud Keychain

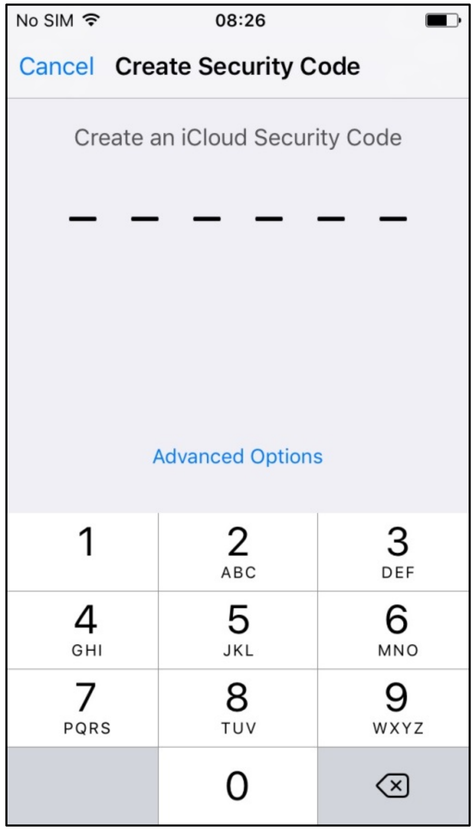
iCloud Keychain keeps the passwords and

Use iPhone Passcode as iCloud Security Code?

Your iCloud Security Code can be used to set up iCloud Keychain on a new device.

Use Passcode

Create Different Code



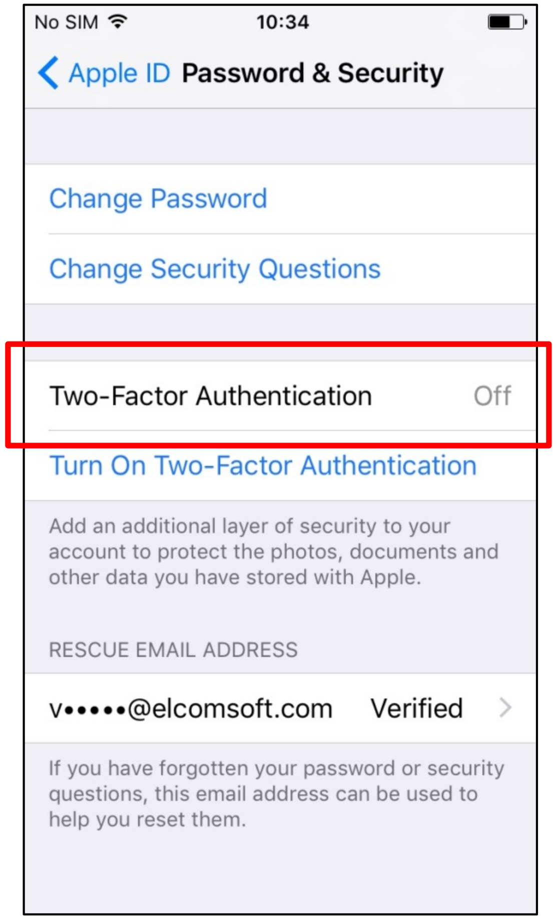
No SIM 08:26

Cancel Create Security Code

Create an iCloud Security Code

Advanced Options

Set up 2FA


No SIM 10:34

Apple ID Password & Security

Change Password

Change Security Questions

Two-Factor Authentication Off

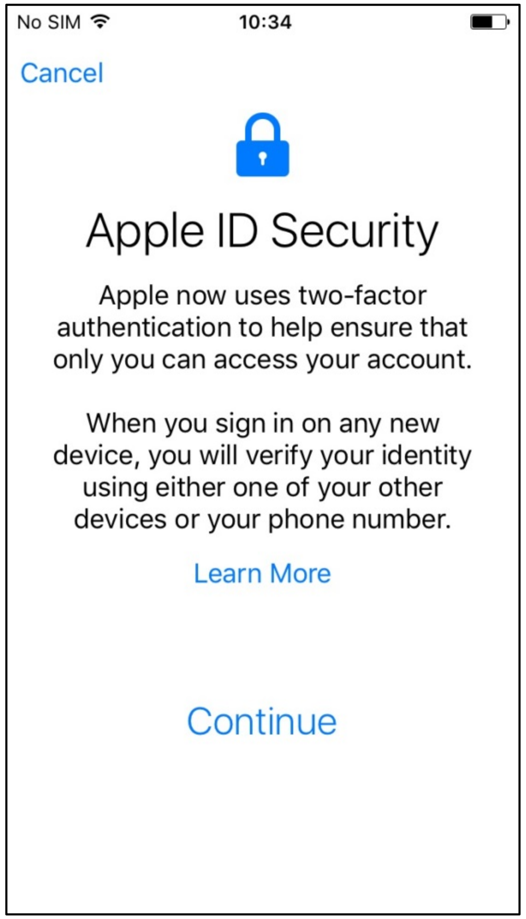
Turn On Two-Factor Authentication

Add an additional layer of security to your account to protect the photos, documents and other data you have stored with Apple.

RESCUE EMAIL ADDRESS

v•••••@elcomsoft.com Verified

If you have forgotten your password or security questions, this email address can be used to help you reset them.


No SIM 10:34

Cancel

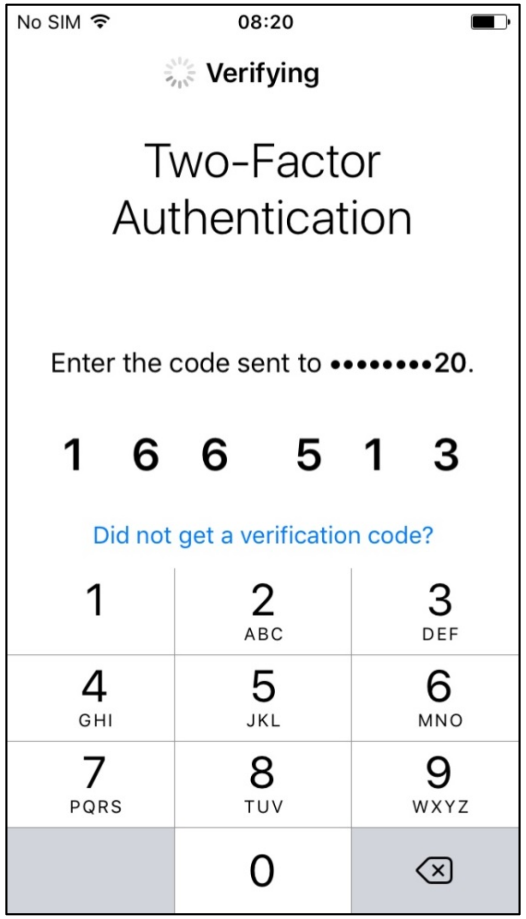
Apple ID Security

Apple now uses two-factor authentication to help ensure that only you can access your account.

When you sign in on any new device, you will verify your identity using either one of your other devices or your phone number.

Learn More

Continue


No SIM 08:20

Verifying

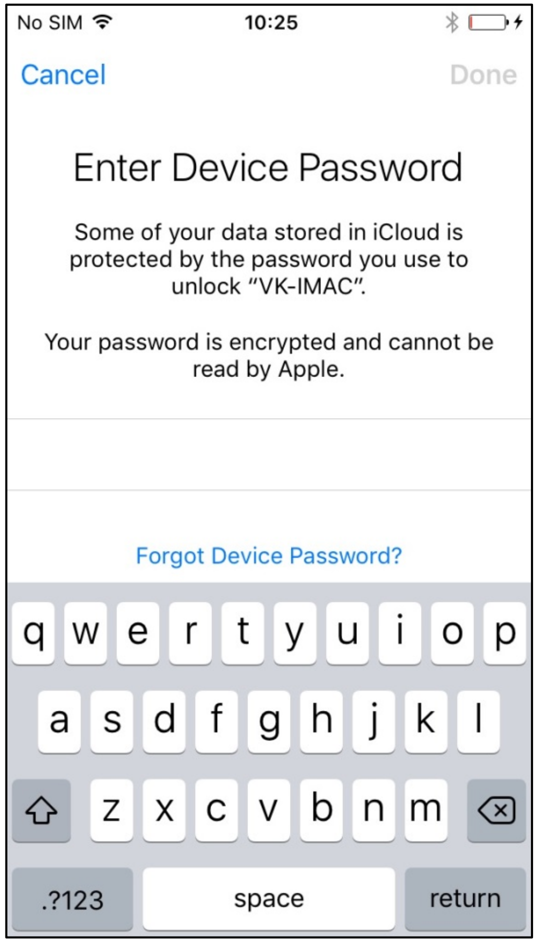
Two-Factor Authentication

Enter the code sent to ••••••••20.

1 6 6 5 1 3

Did not get a verification code?

Set up iCloud keychain –2FA


No SIM 10:25

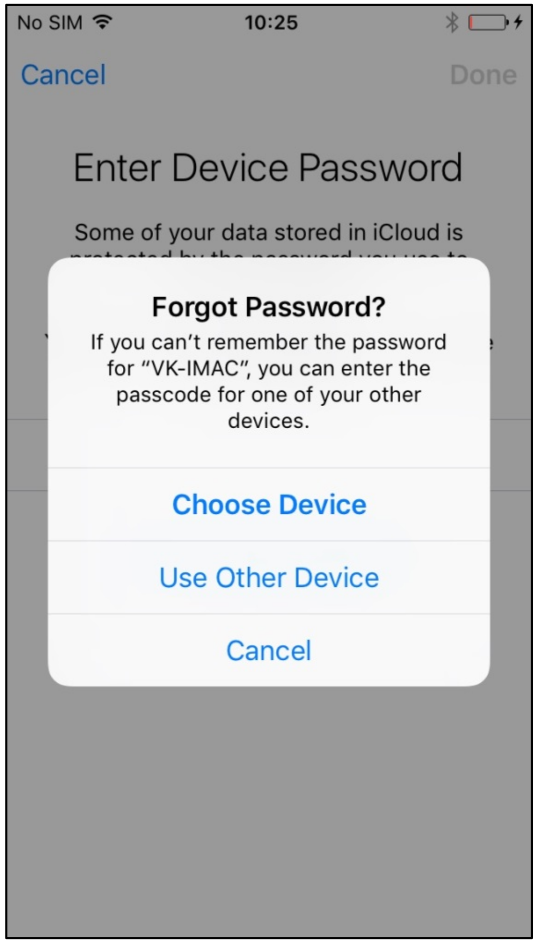
Cancel Done

Enter Device Password

Some of your data stored in iCloud is protected by the password you use to unlock "VK-IMAC".

Your password is encrypted and cannot be read by Apple.

Forgot Device Password?


No SIM 10:25

Cancel Done

Enter Device Password

Some of your data stored in iCloud is

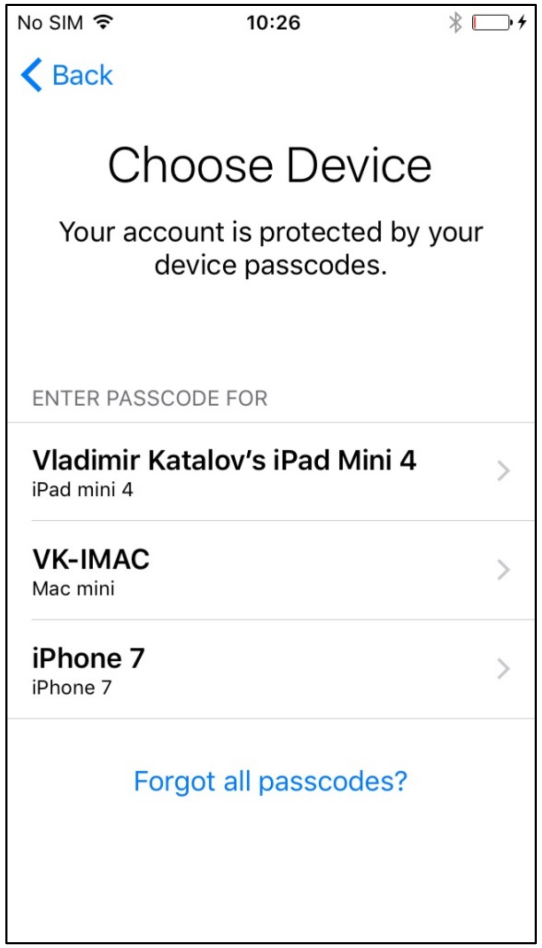
Forgot Password?

If you can't remember the password for "VK-IMAC", you can enter the passcode for one of your other devices.

Choose Device

Use Other Device

Cancel


No SIM 10:26

Back

Choose Device

Your account is protected by your device passcodes.

ENTER PASSCODE FOR

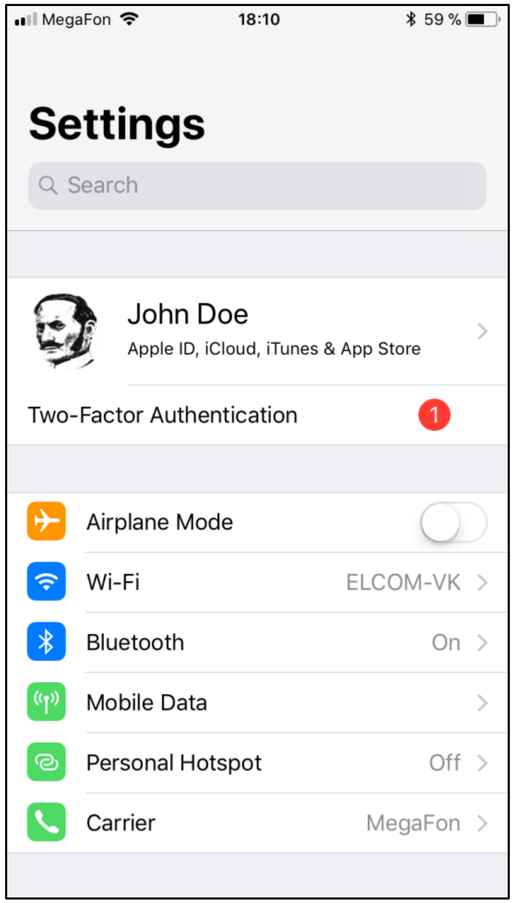
Vladimir Katalov's iPad Mini 4
iPad mini 4

VK-IMAC
Mac mini

iPhone 7
iPhone 7

Forgot all passcodes?

iOS 11 and 2FA



MegaFon 18:10 59 %

Settings

Search

John Doe
Apple ID, iCloud, iTunes & App Store

Two-Factor Authentication 1

Airplane Mode

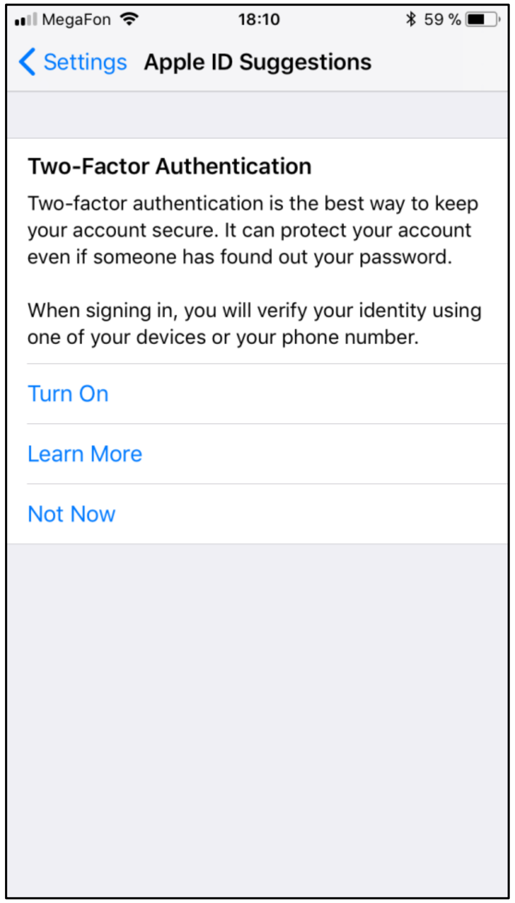
Wi-Fi ELCOM-VK

Bluetooth On

Mobile Data

Personal Hotspot Off

Carrier MegaFon



MegaFon 18:10 59 %

Settings Apple ID Suggestions

Two-Factor Authentication

Two-factor authentication is the best way to keep your account secure. It can protect your account even if someone has found out your password.

When signing in, you will verify your identity using one of your devices or your phone number.

Turn On

Learn More

Not Now

iCloud sync modes

Recovery: recovery from keychain backup/storage in the iCloud

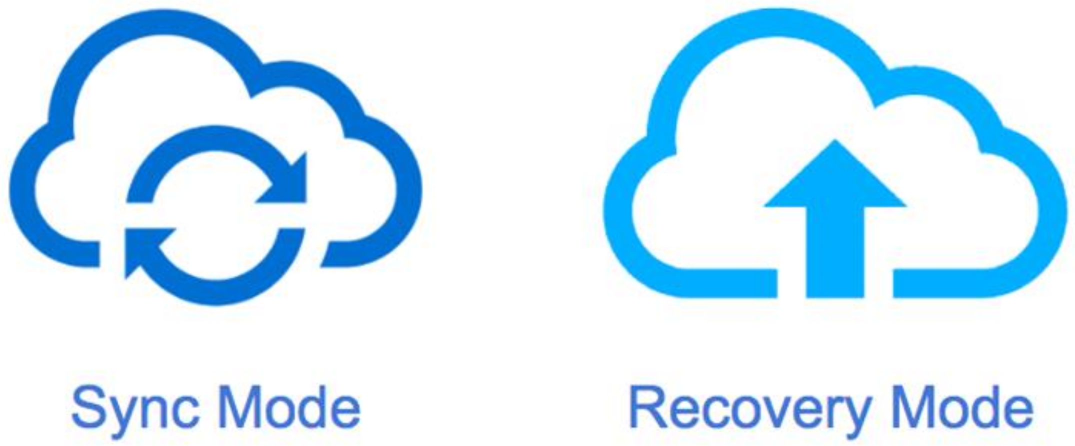
com.apple.sbd3 (Secure Backup Daemon)

Keep backup of keychain records, and copying to new devices (when there are new trusted ones)

Sync: real-time syncing across cloud and devices

com.apple.security.cloudkeychainproxy3

Support for “trusted circle”, adding new devices to it etc

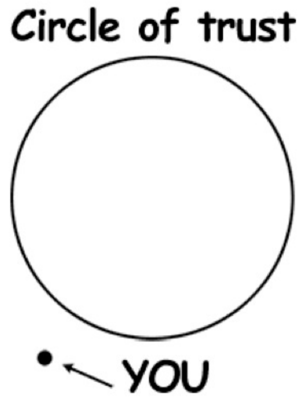


iCloud circle of trust

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

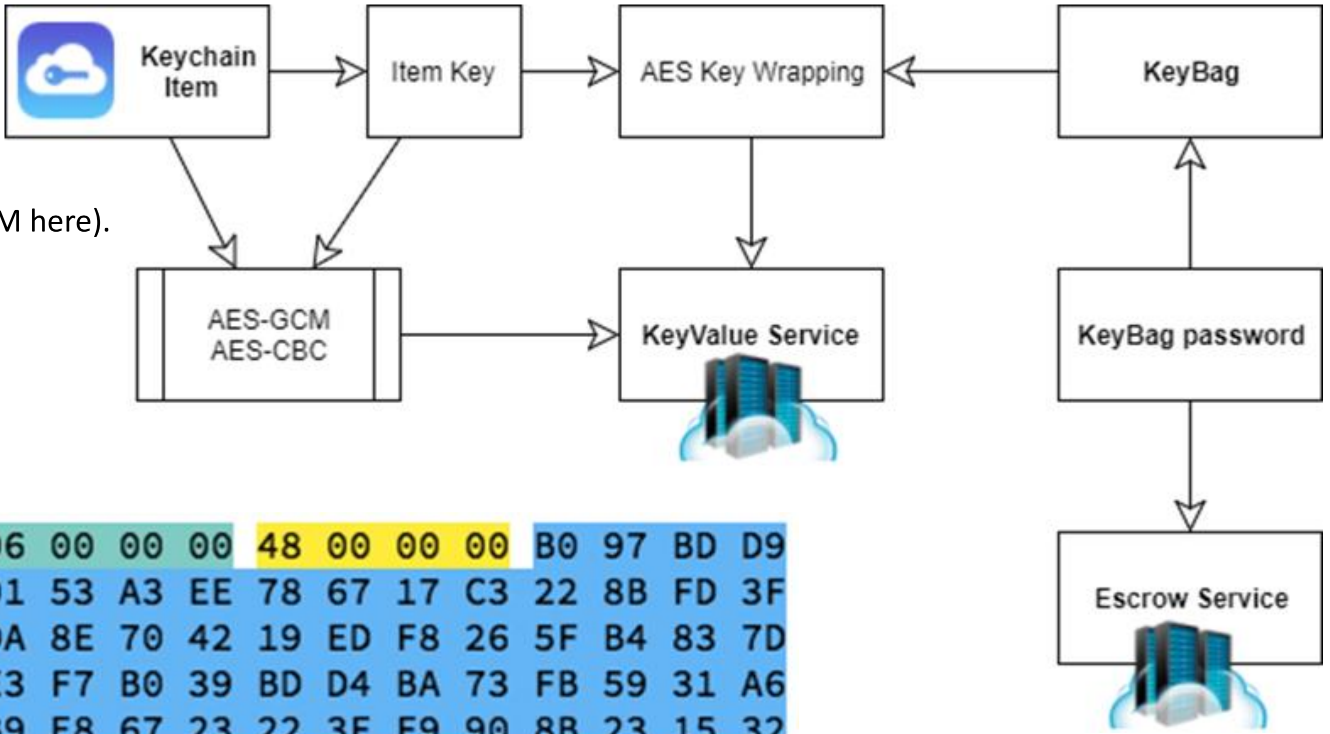
- **Keychain syncing**
  - Circle of trust
  - Public key: syncing identity (specific to device)
  - Private key (elliptical P256), derived from iCloud password
  - Each synced item is encrypted specifically for the device (cannot be decrypted by other devices)
  - Only items with *kSecAttrSynchronizable*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

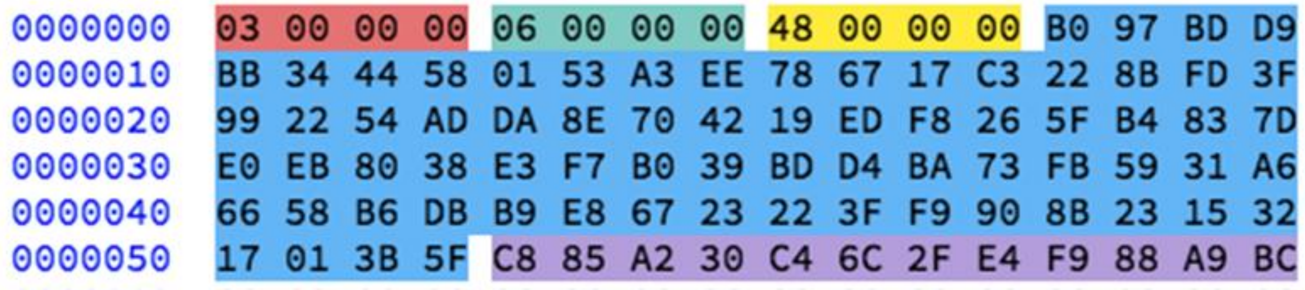
- **Keychain recovery**
  - Secure escrow service (*optional*)
  - No 2FA: iCloud security code is needed (+SMS)
  - No 2FA, no iCSC: recovery is not possible
  - 2FA: device passcode is needed
  - Hardware Security Module (WTF is that? ☺)



iCloud keychain recovery mode

3: key version (GCM or CBC algorythm; GCM here).
6: record protection class (KeyBag #6 here)
0x48: wrapped key size
Next: encrypted key data





```
0000000  03 00 00 00 06 00 00 00 48 00 00 00 B0 97 BD D9
0000010  BB 34 44 58 01 53 A3 EE 78 67 17 C3 22 8B FD 3F
0000020  99 22 54 AD DA 8E 70 42 19 ED F8 26 5F B4 83 7D
0000030  E0 EB 80 38 E3 F7 B0 39 BD D4 BA 73 FB 59 31 A6
0000040  66 58 B6 DB B9 E8 67 23 22 3F F9 90 8B 23 15 32
0000050  17 01 3B 5F C8 85 A2 30 C4 6C 2F E4 F9 88 A9 BC
```

iCloud keychain recovery protection (no 2FA)

<u>iCSC - iCloud Security code</u>

No iCSC

Sync mode only. Keychain records are not stored in the iCloud and cannot be recovered if all trusted devices are lost/ Access is possible only through push notification to the trusted device. *The most safe/secure config? ;)*

iCSC is set

- Push notification to trusted device (as above)
- iCSC plus code from SMS (6 digits)

Note: iCSC is not stored anywhere in the cloud, just its hash (in Escrow). Three options are available:

- Simple (4 or 6 digits, depends on iOS version)
- Complex (any symbols, up to 32)
- Device-generated/random (24 symbols)

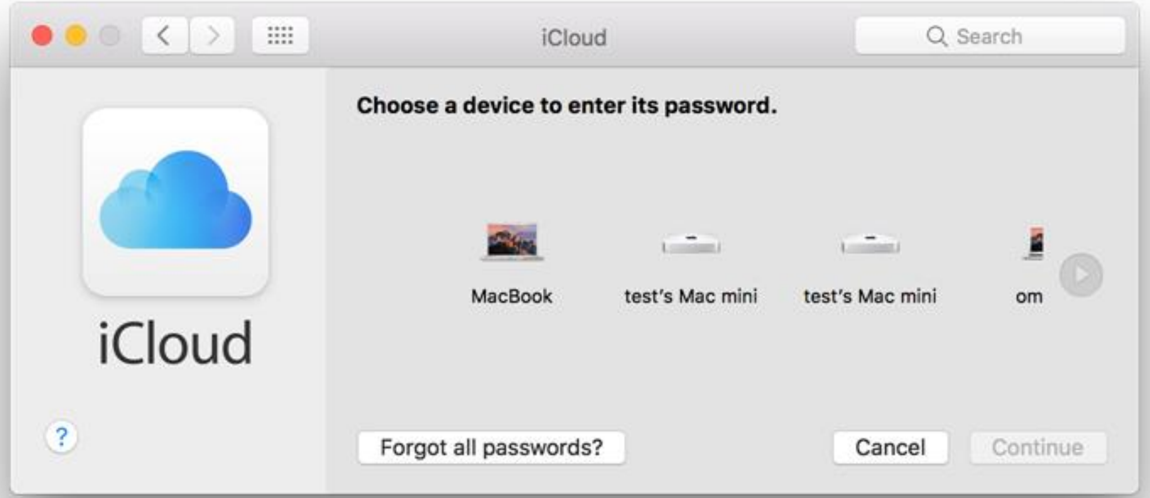
iCloud keychain recovery protection (2FA)

For every device, separate record is created (at EscrowProxy):

com.apple.icdp.<deviceHash>

Contents: BackupBagPassword (randomly generated)

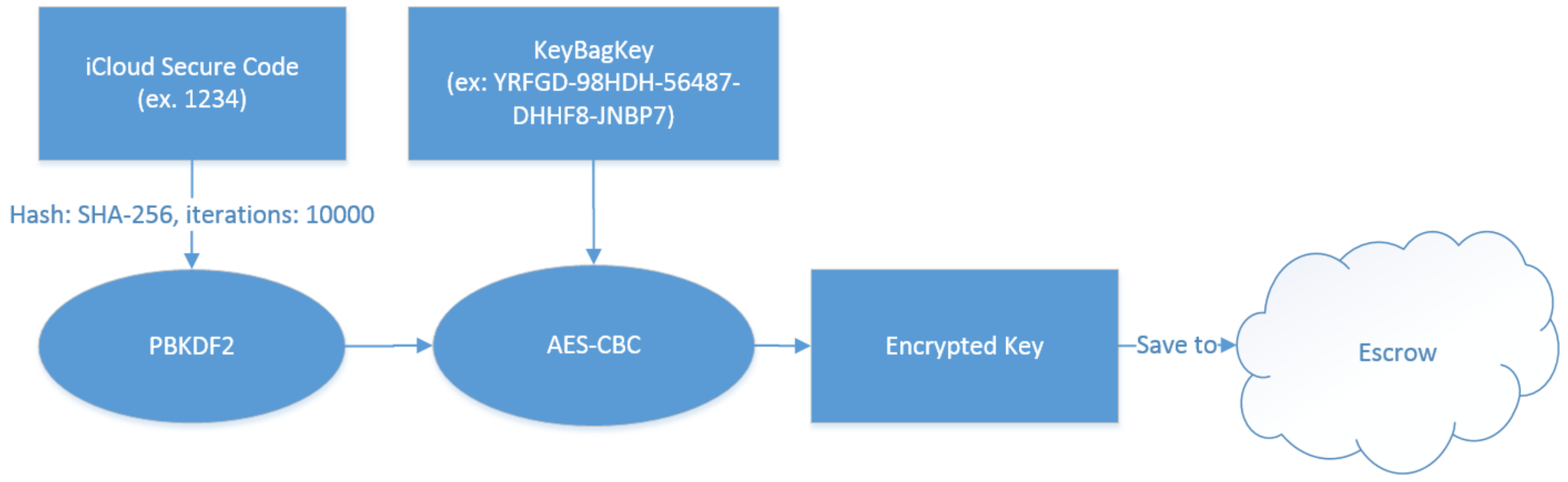
Usage: RFC6637 to encrypt keys from iCloud Keychain Keybags



Escrow proxy architecture (1)

Escrow proxy

- SRP (Secure Remote Password) protocol
- Safe from MITM
- Does not need password to be transferred at all (even hash)
- Does not keep password on server

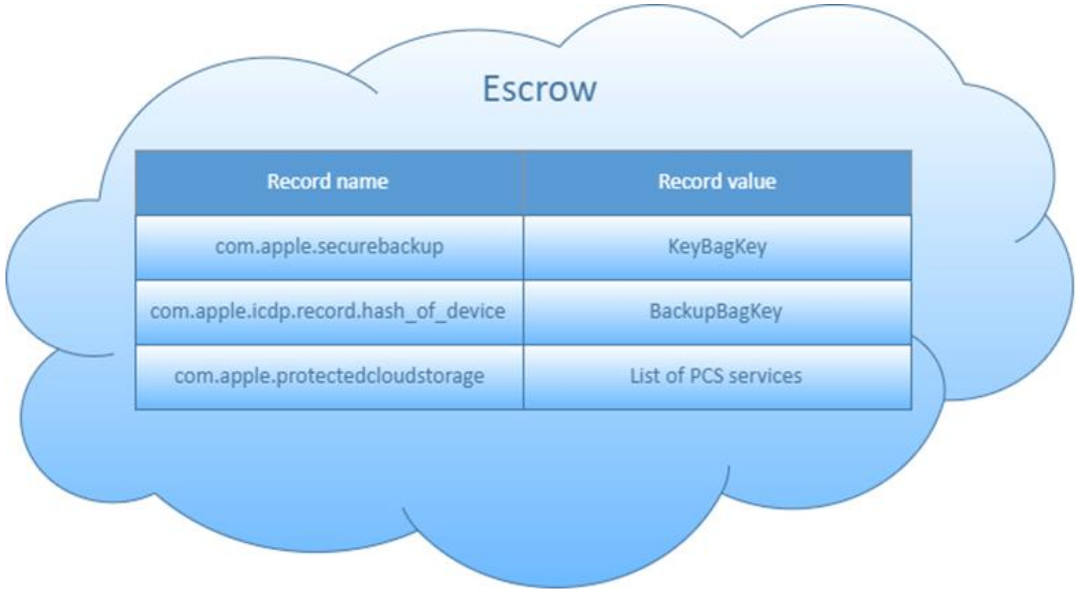

iCloud Secure Code (ex. 1234)
KeyBagKey (ex: YRFGD-98HDH-56487-DHHF8-JNBP7)
Hash: SHA-256, iterations: 10000
PBKDF2
AES-CBC
Encrypted Key
Save to
Escrow

Escrow proxy architecture (2)

Cloud Keychain records of interest at ***EscrowProxy***

- com.apple.securebackup: keep BackupBagPassword from Keybag, where iCloudKeychain is stored for 'full restore'

- com.apple.icdp.<deviceHash>: BackupBagPassword from iCloudKeychain individual records from given devices, stored for partial recovery



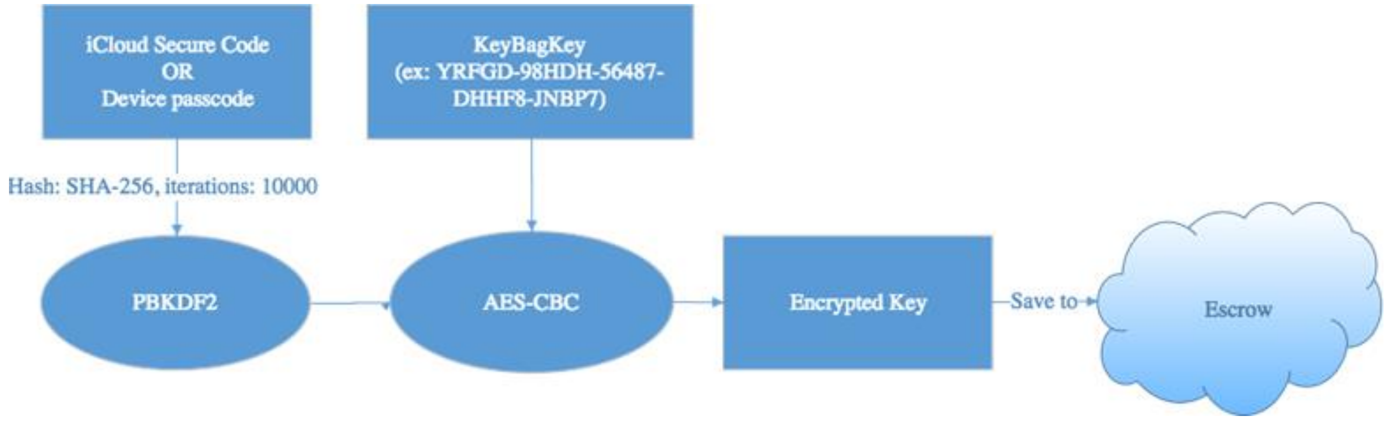
Record name	Record value
com.apple.securebackup	KeyBagKey
com.apple.icdp.record.hash_of_device	BackupBagKey
com.apple.protectedcloudstorage	List of PCS services

Escrow proxy architecture (3)

No 2FA (iCSC) and 2FA (Device Passcode):

- Client generates random 25-symbol KeyBagKey
- PBKDF2(SHA256, 10000) to generate iCSC/passcode hash
- KeyBagKey is encrypted with AES-CBC using hash as a key
- Encrypted KeyBagKey is stored in EscrowProxy

Note: if 'random' option is selected as iCSC, then it is not hashed, and saved 'as is' It is further used for encrypting KeyBag with set of keys for iCloud Keychain.

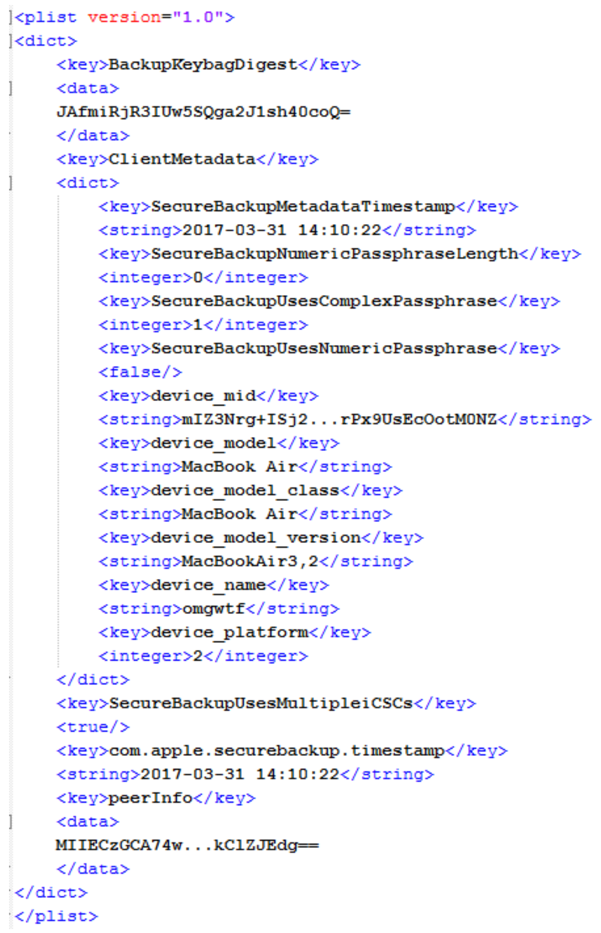

iCloud Secure Code OR Device passcode
KeyBagKey (ex: YRFGD-98HDH-56487-DHHF8-JNBP7)
Hash: SHA-256, iterations: 10000
PBKDF2
AES-CBC
Encrypted Key
Save to
Escrow

Escrow proxy API

Command	Action
/get_club_cert	Returns certificate, associated with account
/enroll	Add new secure record
/get_records	Get list of stored records
/get_sms_targets	Get phone number, associated with account
/generate_sms_challenge	Sends approval code via sms to associated number
/srp_init	Initializes authentication via SRP-6a protocol
/recover	SRP authentication finalization. returns secure records on success
/update_record	Updates records information associated with account

Escrow proxy: ‘public’ records

- 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...kC1ZJEdg==
    </data>
</dict>
</plist>
```

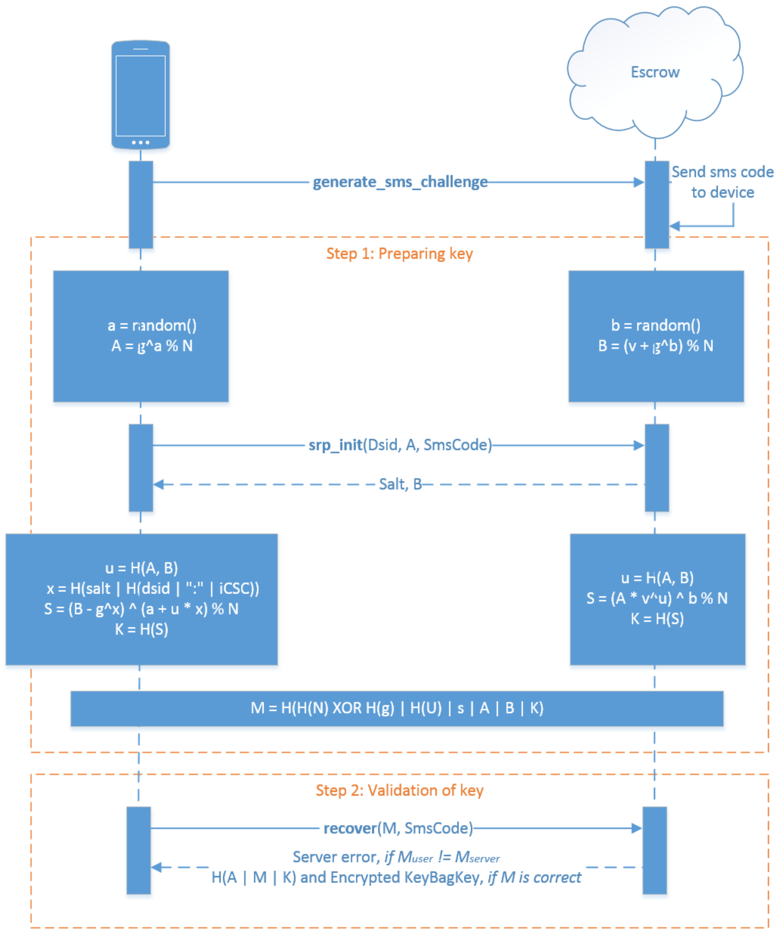
SRP protocol (v6)

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.


Escrow
generate_sms_challenge
Send sms code to device
Step 1: Preparing key
a = random()
A = g^a % N
b = random()
B = (v + g^b) % N
srp_init(Dsid, A, SmsCode)
Salt, B
u = H(A, B)
x = H(salt | H(dsid | ":" | iCSC))
S = (B - g^x) ^ (a + u * x) % N
K = H(S)
u = H(A, B)
S = (A * v^u) ^ b % N
K = H(S)
M = H(H(N) XOR H(g) | H(U) | s | A | B | K)
Step 2: Validation of key
recover(M, SmsCode)
Server error, if Muser != Mserver
H(A | M | K) and Encrypted KeyBagKey, if M is correct

Escrow proxy – access tokens

- No 2FA, iCloud Security Code: MME token is enough; validation uses SMS to trusted number set in account

How to obtain: same as for backups, synced data, iCloud Photo Library etc

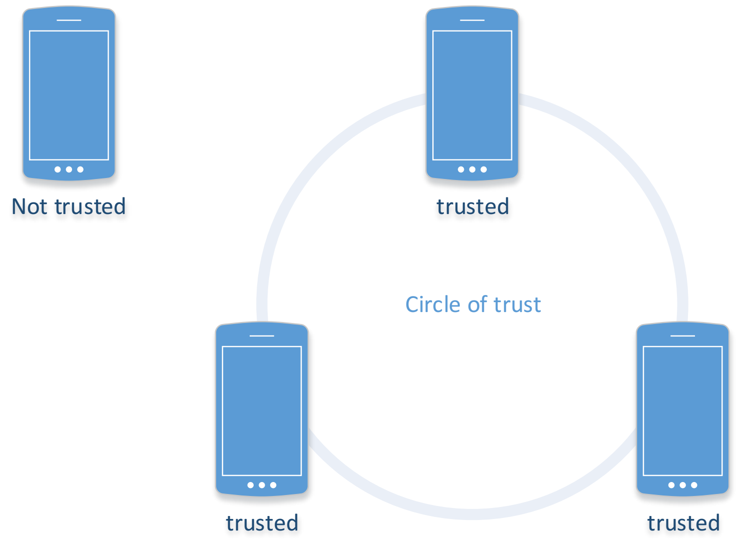
- 2FA, device passcode: PET (Password Equivalent Token); TTL=5 minutes

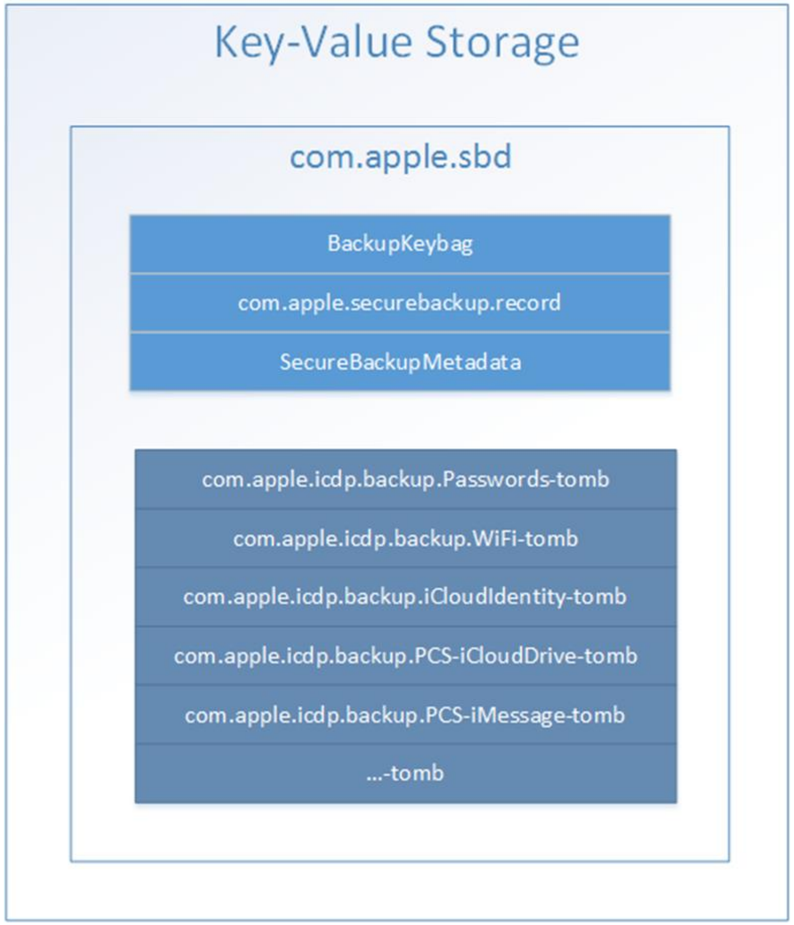
How to obtain: pass GSA authentication (to approve short-time access from the given device); new in macOS 10.11

Authentication type for access of Escrow record

Record name	Authentication Type
com.apple.securebackup	MME + SMS
com.apple.icdp.record.hash_of_device	PET
com.apple.protectedcloudstorage	MME

Keychain is sync mode




Key-Value Storage
com.apple.sbd
BackupKeybag
com.apple.securebackup.record
SecureBackupMetadata
com.apple.icdp.backup.Passwords-tomb
com.apple.icdp.backup.WiFi-tomb
com.apple.icdp.backup.iCloudIdentity-tomb
com.apple.icdp.backup.PCS-iCloudDrive-tomb
com.apple.icdp.backup.PCS-iMessage-tomb
...-tomb

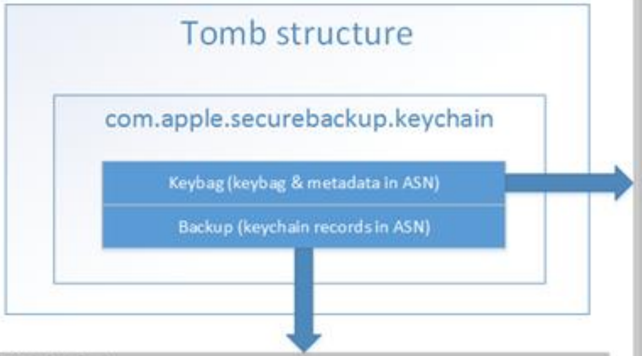
In sync mode, KeyBag may contain as full records in recovery mode (BackupKeyBag, com.apple.securebackup.record) or *tombs*, unique for every domain (HomeKit, Wi-Fi etc)

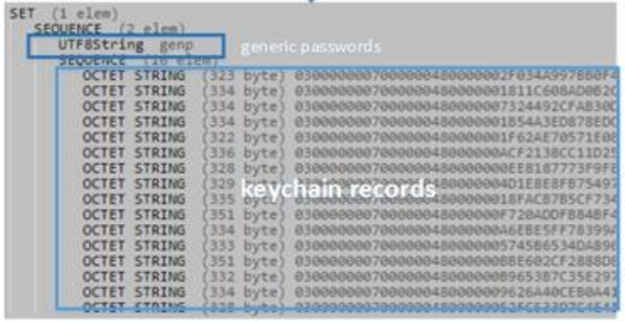
Tombs

- Keybag & metadata (ASN.1 format)
- Keychain: records for the given domain, encrypted with Keybag
- Wrapped Key (for every RecordID): Keybag key wrapped with RFC 6637

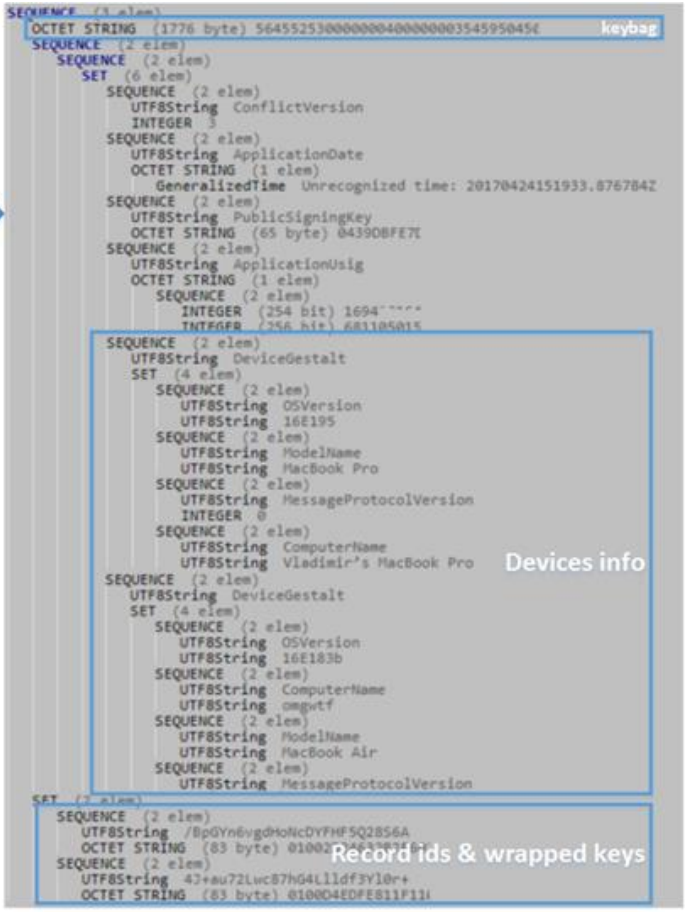
To decrypt

- get tombs from com.apple.sbd
- find all RecordIDs
- get BackupBagPassword for the given RecordID, using passcode of the device
- unwrap KeyBag key
- decrypt keys from KeyBag
- Decrypt Keychain records


Tomb structure
com.apple.securebackup.keychain
Keybag (keybag & metadata in ASN)
Backup (keychain records in ASN)



```
SET  (1 elem)
  SEQUENCE  (2 elem)
    UTF8String  genp
    SEQUENCE  (16 elem)
      OCTET STRING  (323 byte) 0300000007000000480000002F034A997BB0F4
      OCTET STRING  (334 byte) 030000000700000048000000001811C608AD0B2C
      OCTET STRING  (334 byte) 03000000070000004800000007324492CFAB30C
      OCTET STRING  (334 byte) 030000000700000048000000001B54A3ED878EDC
      OCTET STRING  (322 byte) 030000000700000048000000001F62AE70571E08
      OCTET STRING  (336 byte) 03000000070000004800000000ACF2138CC11D25
      OCTET STRING  (328 byte) 03000000070000004800000000EE8187773F9F8
      OCTET STRING  (329 byte) 030000000700000048000000004D1E8E8FB75497
      OCTET STRING  (335 byte) 0300000007000000480000000018FACB7B5CF734
      OCTET STRING  (351 byte) 03000000070000004800000000F720ADDFB848F4
      OCTET STRING  (334 byte) 03000000070000004800000000A6EBE5FF78399A
      OCTET STRING  (333 byte) 030000000700000048000000005745B6534DA896
      OCTET STRING  (351 byte) 03000000070000004800000000B8E602CF2888D8
      OCTET STRING  (332 byte) 03000000070000004800000000B9653B7C35E297
      OCTET STRING  (334 byte) 030000000700000048000000009626A40CEB0A41
      OCTET STRING  (328 byte) [illegible]
```



```
SEQUENCE  (3 elem)
  OCTET STRING  (1776 byte) 56455253000000004000000035459504 5€
  SEQUENCE  (2 elem)
    SEQUENCE  (2 elem)
      SET  (6 elem)
        SEQUENCE  (2 elem)
          UTF8String  ConflictVersion
          INTEGER  3
        SEQUENCE  (2 elem)
          UTF8String  ApplicationDate
          OCTET STRING  (1 elem)
            GeneralizedTime  Unrecognized time: 20170424151933.876784Z
        SEQUENCE  (2 elem)
          UTF8String  PublicSigningKey
          OCTET STRING  (65 byte) 04390BFE7E
        SEQUENCE  (2 elem)
          UTF8String  ApplicationUsig
          OCTET STRING  (1 elem)
            SEQUENCE  (2 elem)
              INTEGER  (254 bit) 1694
              INTEGER  (256 bit) 681105015
        SEQUENCE  (2 elem)
          UTF8String  DeviceGestalt
          SET  (4 elem)
            SEQUENCE  (2 elem)
              UTF8String  OSVersion
              UTF8String  16E195
            SEQUENCE  (2 elem)
              UTF8String  ModelName
              UTF8String  MacBook Pro
            SEQUENCE  (2 elem)
              UTF8String  MessageProtocolVersion
              INTEGER  0
            SEQUENCE  (2 elem)
              UTF8String  ComputerName
              UTF8String  Vladimir's MacBook Pro
        SEQUENCE  (2 elem)
          UTF8String  DeviceGestalt
          SET  (4 elem)
            SEQUENCE  (2 elem)
              UTF8String  OSVersion
              UTF8String  16E183b
            SEQUENCE  (2 elem)
              UTF8String  ComputerName
              UTF8String  omgwtf
            SEQUENCE  (2 elem)
              UTF8String  ModelName
              UTF8String  MacBook Air
            SEQUENCE  (2 elem)
              UTF8String  MessageProtocolVersion
    SET  (2 elem)
      SEQUENCE  (2 elem)
        UTF8String  /BpGYn6vgdHoNcDYFHF5Q28S6A
        OCTET STRING  (83 byte) 01002[illegible]
      SEQUENCE  (2 elem)
        UTF8String  4J+au72Lwc87hG4Llldf3Yl0r+
        OCTET STRING  (83 byte) 0100D4EDFE811F11
```

Other components and alternative approaches

GSA (GrandSlam Authentication)

- gsa.apple.com
- based on SRP protocol
- introduced in macOS 10.10 (basic)
- improved in macOS 10.11

AnisietteData

- MachineID + OTP
- MachineID (60 bytes): unique for device
- OTP (24 bytes): random; refreshed every 90 seconds
- code is hardly obfuscated
- implemented in Apple Private API

Continuation token

- obtained through GSA
- means to get tokens for other services
- no need to keep Apple ID and password on device
- can be used to get *updated* tokens with short TTL
- for further requests: use **AlternateDSID** & **Continuation token** instead of AppleID & password

Demo

No 2FA

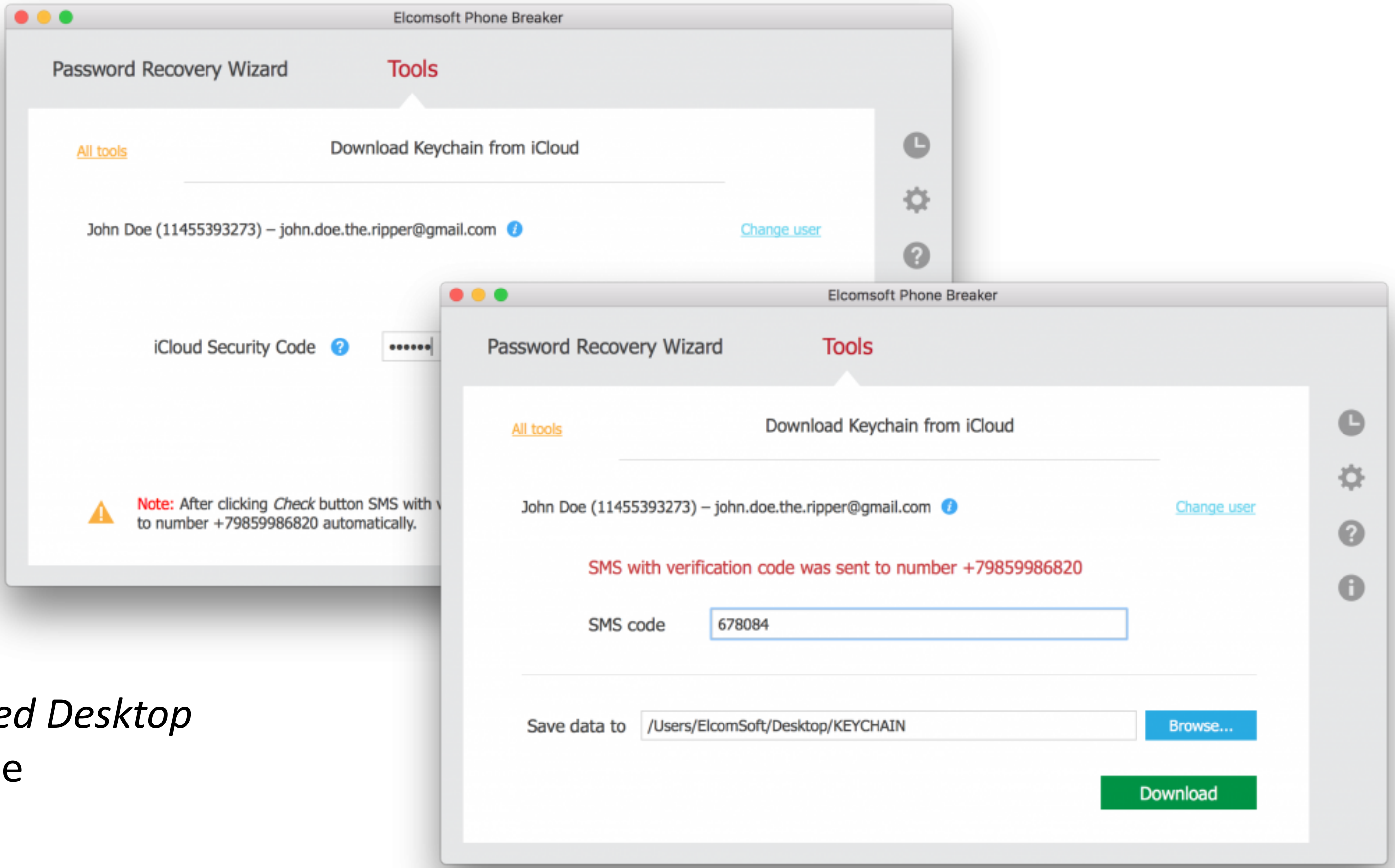
- Apple ID
- Password
- iCloud security code
- SMS to trusted number

2FA

- Apple ID
- Password

no need to pass 2FA on trusted Desktop

- Passcode of enrolled device



Elcomsoft Phone Breaker

Password Recovery Wizard Tools

All tools

Download Keychain from iCloud

John Doe (11455393273) – john.doe.the.ripper@gmail.com

Change user

iCloud Security Code ••••••

Note: After clicking *Check* button SMS with v
to number +79859986820 automatically.

Elcomsoft Phone Breaker

Password Recovery Wizard Tools

All tools

Download Keychain from iCloud

John Doe (11455393273) – john.doe.the.ripper@gmail.com

Change user

SMS with verification code was sent to number +79859986820

SMS code 678084

Save data to /Users/ElcomSoft/Desktop/KEYCHAIN

Browse...

Download

Conclusions / risks

- Sync and recovery: different approaches
- Trusted circle: not hard to get in, but leaves traces
- Both sync and recovery can be used (mixed)
- Need to have credentials
- Need to have trusted device

  …or SMS
- Need to know iCSC

  …or device passcode
- Legacy 2SV: forget it
- With 2FA, keychain is always stored in iCloud
- **No 2FA, no iCSC: most safe from TLA?**

- **Get *Continuation token (+machine ID)* to obtain full access without anything else!**
- ***…implementation is still relatively secure ☺***

Wait, one more thing…

- iCloud Keychain contains more data than officially documented: not just passwords, but also tokens (e.g. to 2FA-protected social network accounts)
- iCloud Keychain is being activated right when you enable 2FA (or even always exist??), though contains only system keys, not user data
- iCloud Keychain contains encryption key used to lock some new iCloud data (iOS 11)
- iCloud Keychain approach can be used effectively when local keychain is not easily accessible

What else do you hide from us, Apple? :)

Thanks!
Questions?

ELCOMSOFT