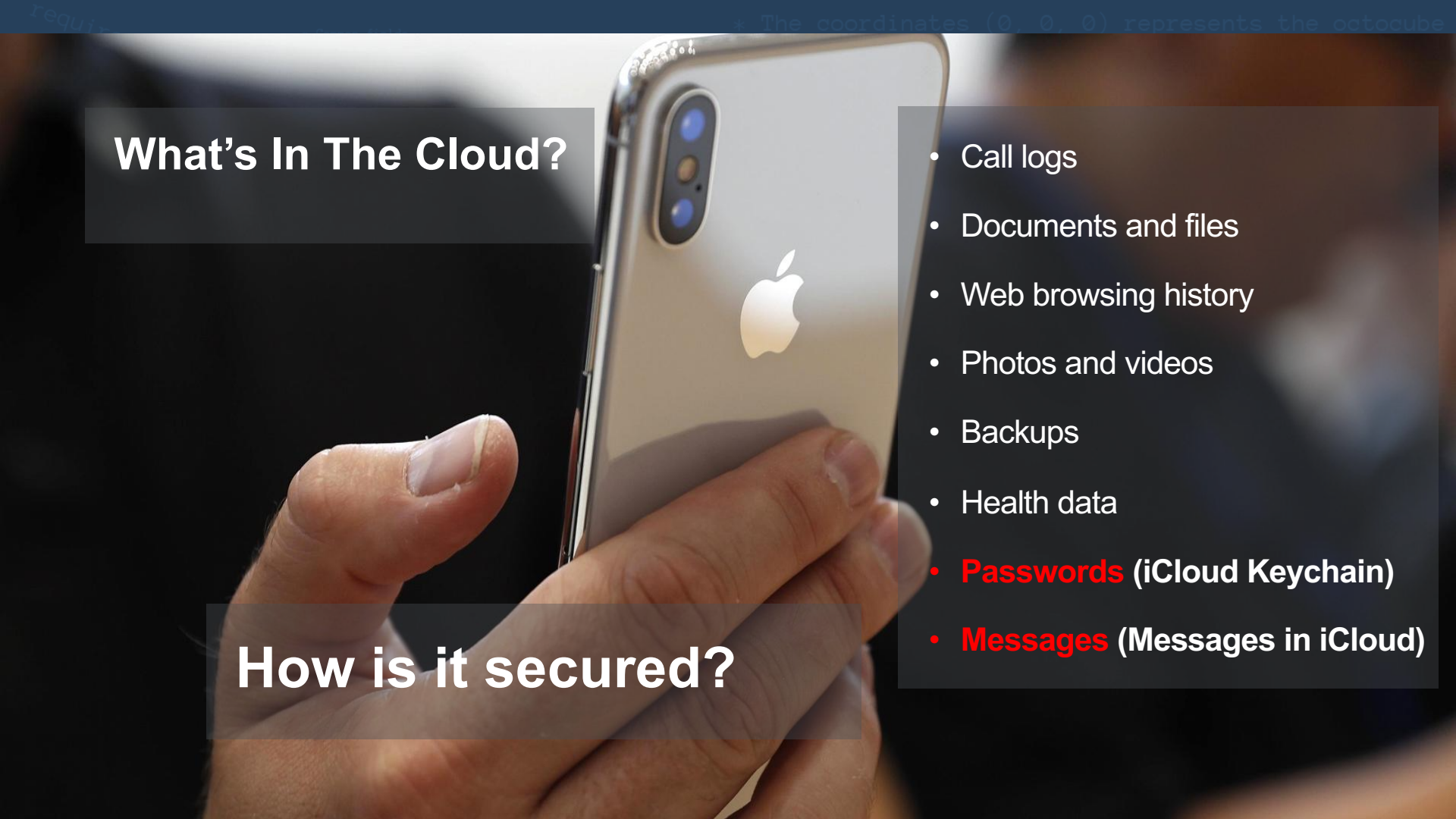
Cloudy With a Chance of Messages

Extracting Messages and attachments from iCloud

Vladimir Katalov, ElcomSoft CEO

- What is in the iCloud
- How secure is the iCloud
- Technical: when Engram meets Manatee
- Issues, risks and profits
- Questions still unanswered



What's In The Cloud?

- Call logs
- Documents and files
- Web browsing history
- Photos and videos
- Backups
- Health data
- **Passwords (iCloud Keychain)**
- **Messages (Messages in iCloud)**

How is it secured?

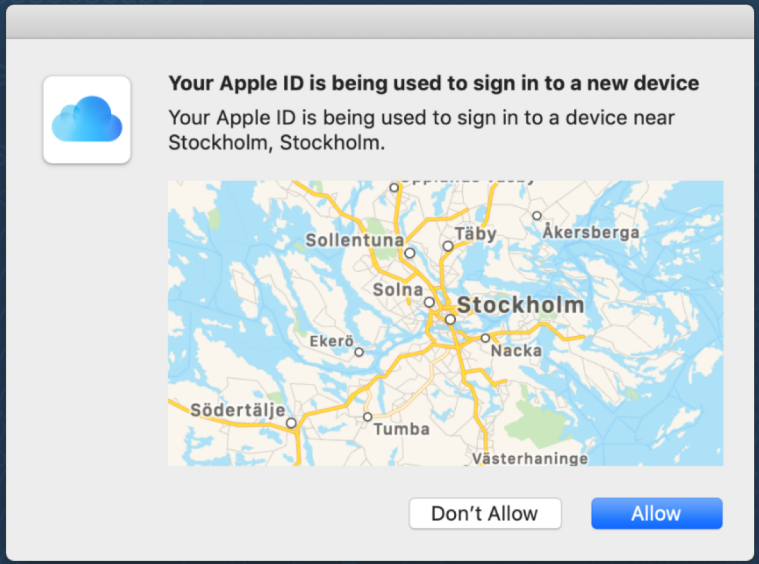
A Bit of History

- Oct’2011: iOS 5; first Apple iCloud release (with iCloud backups)
- May’2012: we allow users to download iCloud backups
- Sep’2013: iCloud Keychain
- Early 2014: two-step verification (not for backups yet; limited countries)
- June 2016: we learned how to extract/decrypt tokens and use them to access iCloud
- Aug’2014: Celebgate; Russian’s Prime Minister iCloud account accessed
- Sep’2014: iOS 8; 2SV now works for backups as well; limiting tokens TTL
- Oct’2014: iOS 8.1; iCloud Photo Library
- Sep’2015: iOS 9; 2FA, iCloud backups on iCloud Drive, tokens work again
- Sep’2017: iOS 11; moving from 2SV to 2FA; Health syncing
- May’2018: iOS 11.4, macOS High Sierra 10.13.5; Messages in iCloud
- Sep’2018: iOS 12; Screen Time (app usage) in iCloud

Stricter 2FA Rules

New This Year

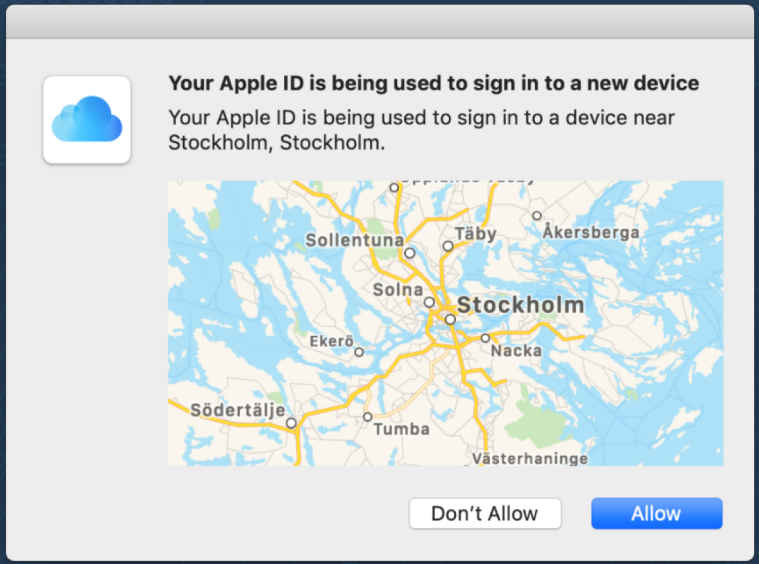
- Two-Factor Authentication now mandatory to sync:
  - Passwords (iCloud Keychain)
  - Messages (iOS 11.4+)
  - Screen Time (iOS 12)



Stricter Token Policies

New This Year

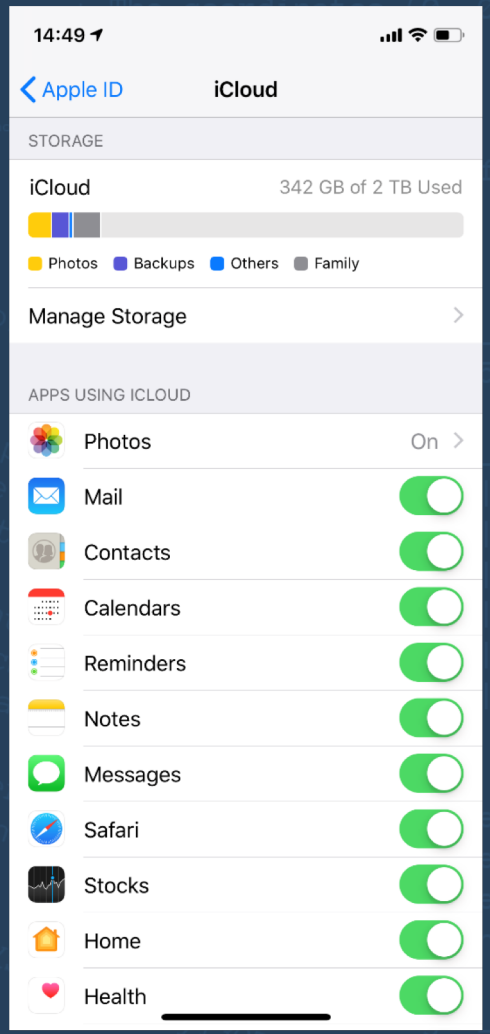
- Let’s talk about tokens
  - New tokens use anisette data to identify hardware
  - Tokens now tied to computer
  - iOS 11.2+2FA (and newer): iCloud backups only on iOS devices

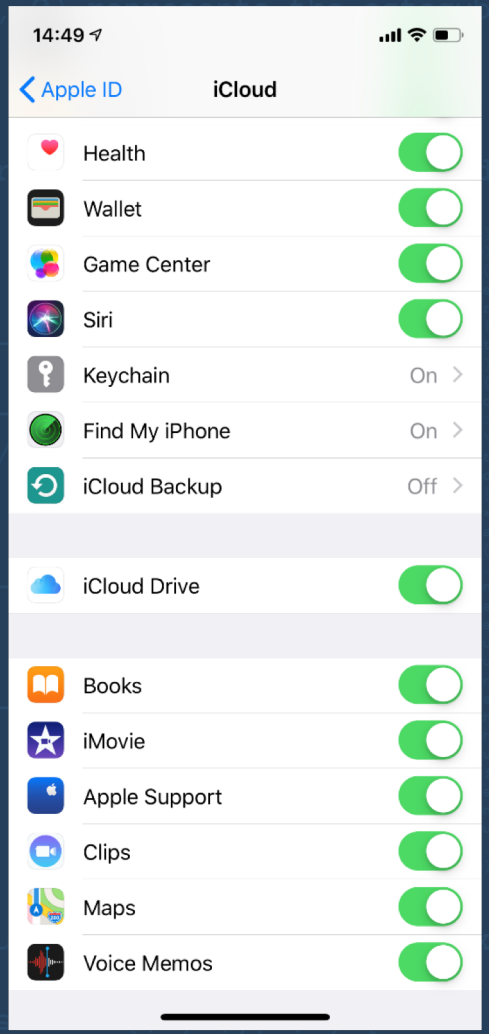


iCloud Sync

Checklist #1: iOS settings

- Not all the categories are listed there (e.g. no call logs, mail signatures, black list, autocorrection dictionaries)
- Some options in fact require the *keychain* to be enabled
- **Messages in iCloud use encryption key stored in iCloud Keychain**





iCloud.com

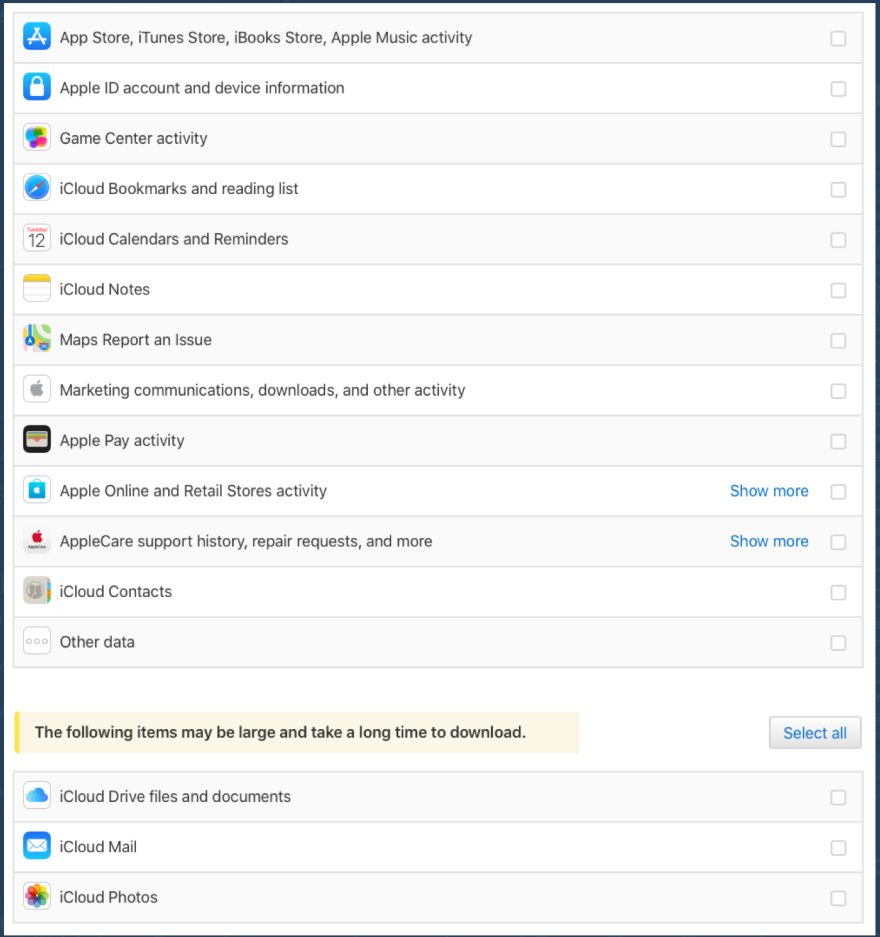
Checklist #2: icloud.com

- Only basic data categories are available
- Immediate notification to the account owner (by email)
- Web browser approach (the token is saved in cookies)
- **Can you see Messages???**

GDPR

Checklist #3: privacy.apple.com

- Available (for now) in selected countries (incl. US, Europe, Russia)
- Takes about a week to prepare data
- Multiple data formats (txt, csv, xml, json)
- Some “internal” Apple data is here (not available by other means)
- The most interesting is hidden under *Other data*
- **Still no messages**
  - For a good reason



Law Enforcement: Still No Messages

Checklist #4: Gov requests

iii. Email Content and Other iCloud Content. My Photo Stream, iCloud Photo Library, iCloud Drive, Contacts, Calendars, Bookmarks, Safari Browsing History, Maps Search History, Messages, iOS Device Backups

iCloud stores content for the services that the subscriber has elected to maintain in the account while the subscriber's account remains active. Apple does not retain deleted content once it is cleared from Apple's servers. iCloud content may include email, stored photos, documents, contacts, calendars, bookmarks, Safari browsing history, Maps Search History, Messages and iOS device backups. iOS device backups may include photos and videos in the Camera Roll, device settings, app data, iMessage, Business Chat, SMS, and MMS messages and voicemail. All iCloud content data stored by Apple is encrypted at the location of the server. When third-party vendors are used to store data, Apple never gives them the keys. Apple retains the encryption keys in its U.S. data centers. iCloud content, as it exists in the subscriber's account, may be provided in response to a search warrant issued upon a showing of probable cause.

III. Information Available from Apple

A. Device Registration
B. Customer Service Records
C. iTunes
D. Apple Retail Store Transactions
E. Apple Online Store Purchases
F. Gift Cards
G. iCloud
H. Find My iPhone
I. Extracting Data from Passcode Locked iOS Devices
J. Other Available Device Information
K. Requests for Apple Retail Store CCTV Data
L. Game Center
M. iOS Device Activation
N. Sign-on Logs
O. My Apple ID and iForgot Logs
P. FaceTime
Q. iMessage

iCloud Sync

iCloud Sync a Lot of Data

- Call logs (cannot be disabled)
- Safari history, tabs, bookmarks
- Contacts, calendars, notes, reminders
- Apple Maps: routes, searches and places
- Wallet: boarding passes, bookings and reservations, loy
- iBooks, podcasts
- News, Weather, Stocks
- iMovie, Clips, voice memos, Siri shortcuts
- Health, Home
- Photos & videos
- Passwords and payment data
- *FileVault2 recovery token*

General iCloud Security

Encrypted with a key stored alongside

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

Most of the data: *A minimum of 128-bit AES encryption*

The key is stored with the data (on Apple servers)

No problem extracting and using the key

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.*

Encryption based on the key stored in iCloud Keychain:

- **Messages**
- Screen Time
- Home (?)
- Health (iOS 12)

Additional iCloud Security

What Apple Says about iCloud Data Protection

End-to-end encrypted data

End-to-end encryption provides the highest level of data security. Your data is protected with a key derived from information unique to your device, combined with your device passcode, which only you know. No one else can access or read this data.

These features and their data are transmitted and stored in iCloud using end-to-end encryption:

- Home data
- Health data
- iCloud Keychain (includes all of your saved accounts and passwords)
- Payment information
- Siri information
- Wi-Fi network information

To use end-to-end encryption, you must have two-factor authentication turned on for your Apple ID. To access your data on a new device, you might have to enter the passcode for an existing or former device.

Messages in iCloud also uses end-to-end encryption. If you have iCloud Backup turned on, your backup includes a copy of the key protecting your Messages. This ensures you can recover your Messages if you lose access to iCloud Keychain and your trusted devices. When you turn off iCloud Backup, a new key is generated on your device to protect future messages and isn't stored by Apple.

iCloud Security Basics

iCloud Data Protection

- **Apple ID Password**
  - Protects against unauthorized access
- **Two-Factor Authentication**
  - Required for syncing iCloud Messages, Screen Time
  - Highly recommended and heavily pushed by Apple
- **Data encryption**
  - However, encryption key stored alongside the data

iCloud Security

iCloud Data Protection in Detail

- **iCloud Keychain (and some specific categories such as Messages) have additional encryption**
  - 256-bit AES
  - Keys are stored in CKKS (CloudKey Keychain Sync)
  - Key are encrypted with TLK (top local/level key) stored in keychain
  - To access the cloud keychain, passcode or system password from already enrolled device is needed
  - Data **not provided** to LE or via GDPR requests
- **Authentication tokens are short-lived**
  - Just not for synced data
  - There's more about tokens

iCloud Security

iCloud Data Protection in Detail

- **iCloud tokens have stronger protection than ever**
  - Lifespan even shorter than before
  - New format tokens can only be used on the same computer
  - Hardware ID with *anisette* data
  - **iOS 11.2 and up + 2FA**: iCloud backups only accessible from iOS devices
  - Still possible (theoretically) to get *continuation token* that would allow access to all data, including the keychain
  - All code is hardly obfuscated, a lot of kernel functions used

iCloud Security

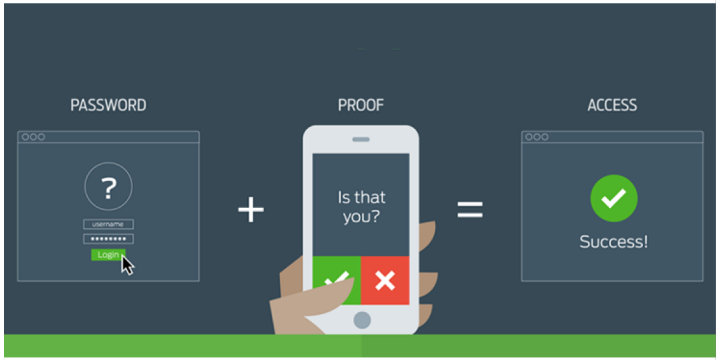
iCloud Data Protection in Detail

- **Apple may lock iCloud accounts**
  - This happens AFTER the backup is downloaded!
- **Data is stored in chunks with third-party cloud services**
  - Microsoft Azure, Google, Amazon AT&T
  - Encryption keys stored on Apple own servers (still not clear about China)
    - Apple drinks Putin's Kool-Vodka, shoves Russians' iCloud data into Russia

      https://www.theregister.co.uk/2015/09/11/apple_icloud_russia/
  - Third-party providers have no access to encryption keys
  - However, these keys are still accessible to anyone with login/password/2FA

iCloud Security

Two-Factor Authentication



- Access to one of the following is required:
  - Access to trusted device
  - SIM card
  - Recovery Key
- Two-step authentication only required once:
  - Authentication token can be saved for future access without login, password or 2FA

iCloud Messages

Messages Benefit from Additional Protection

iCloud Keychain and Messages

- Encrypted and protected
- Passcode or system password of an already enrolled device required to access

iCloud Messages

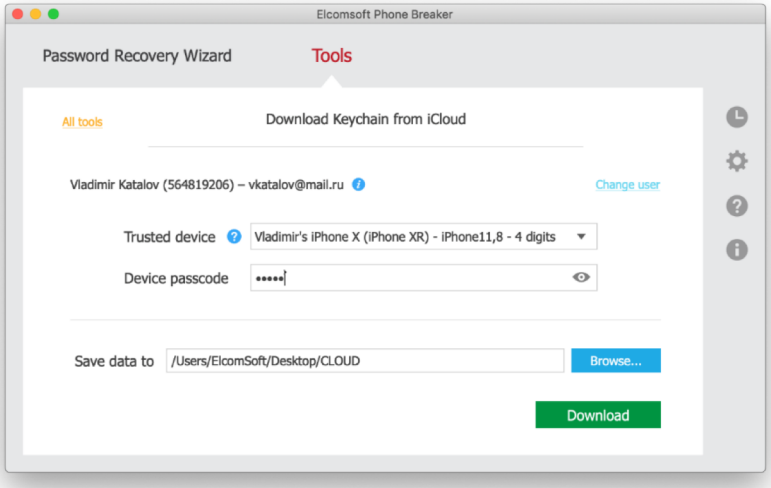
Messages in iCloud

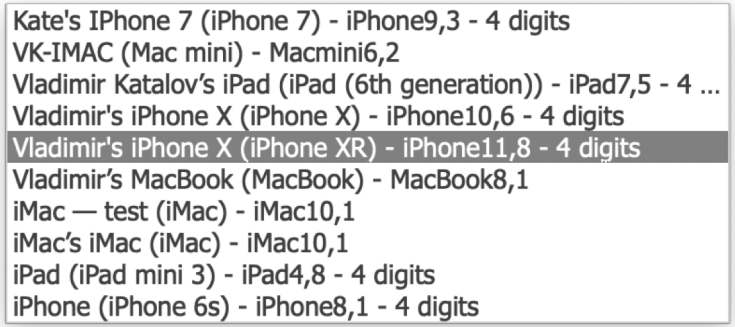
- iOS 11.4 and newer may sync messages (iMessages, SMS) through iCloud
- Protection is pinned to iCloud Keychain
  - AES256 encryption, passcode required
- Apple ID, password and 2FA required
  - 2FA **not required** if acquiring from a trusted Mac
- Passcode or system password from an already enrolled device required
- If messages are synced, they are excluded from iCloud backups

iCloud Keychain

iCloud keychain access

- Passcode or password of a trusted device is needed
- For iOS devices, passcode length is known
- Sometimes old devices still work
- Sometimes old passcodes still work





iCloud Keychain

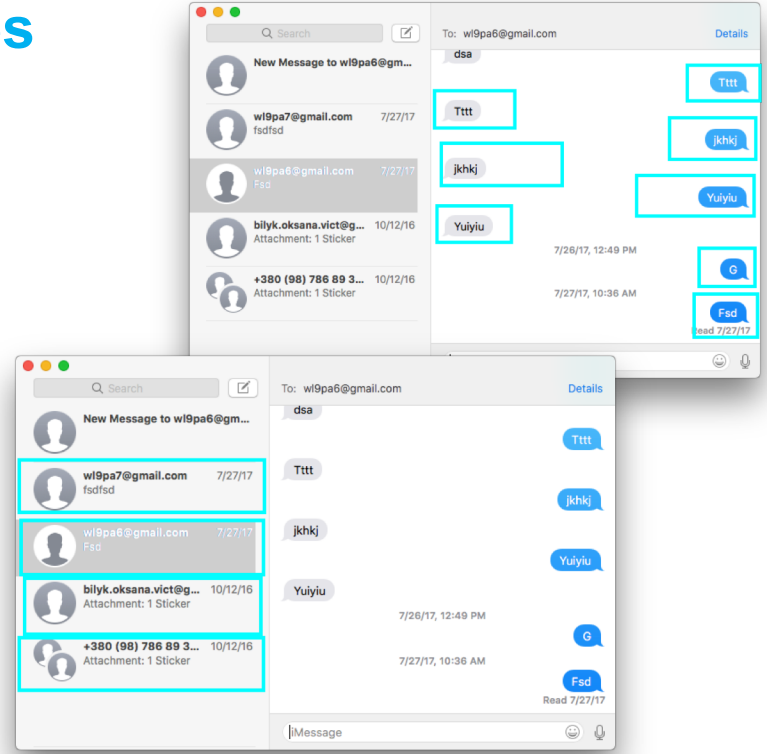
Apple IDs (44) Wi-Fi accounts (293) Mail accounts (19) Browser passwords (529) Credit cards (24) DSIDs & Tokens (193)

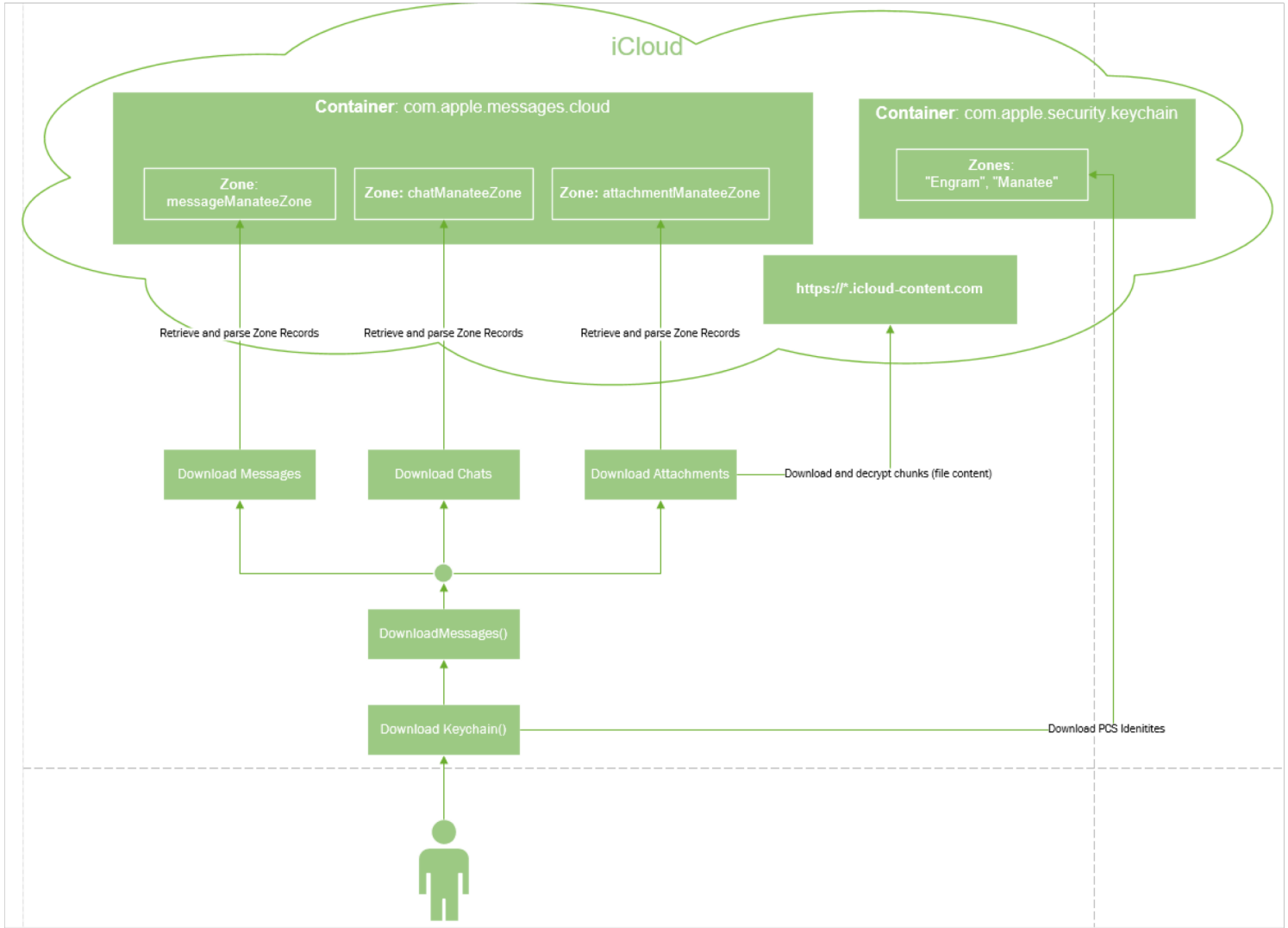
Name	Creation date	Modification date	Token
com.afisha.Restaurants.com.apple.faceboo...	2016.09.09 14:11:18	2016.12.13 12:29:02	CAAAANldxfDABACUh1YPYtHA4...
com.viber.com.apple.facebook.oauth-token...	2016.09.09 07:00:20	2017.01.26 16:29:36	EAACvCTS5cQwBAIG3OSrBBVtC...
com.apple.twitter.oauth-token-secret (Savi...	2016.11.07 19:36:45	2016.12.13 12:29:02	Q9df663MO6tFGCQIAoW2251NF...
com.apple.facebook.oauth-token (ekaterin...	2016.09.08 20:47:18	2016.12.13 12:29:02	EAADCOCzKN18BALMI7AZAZAr4...
net.whatsapp.WhatsApp.com.apple.facebo...	2016.11.22 08:38:14	2016.12.13 12:29:02	CAAAAR0Mp3UkBAF9jwL2ZBI5G...
com.apple.twitter.oauth-token (KostumerG...	2016.11.07 19:36:46	2016.12.13 12:29:02	465424707-3EW4Xp6hB8DEFu0...
com.facebook.PageAdminApp.com.apple.fa...	2016.11.08 06:52:33	2016.12.12 19:52:35	EAACW5Fg5N2IBAAdbQYBDoAL...
com.expedia.booking.com.apple.facebook....	2017.01.08 08:03:01	2017.01.08 15:44:30	EAAB3ohnwUAIBAIZCl6gCojLGEf...
com.apple.linkedin.oauth-token-secret (vk...	2017.01.11 03:33:45	2017.01.11 03:33:45	b582f0fd-5dba-4066-b479-2dafe...
pinterest.com.apple.facebook.oauth-token ...	2016.10.05 08:04:53	2016.12.13 12:29:02	CAAAAP9uIENwBANl2quOjsS3gt...
com.apple.facebook.oauth-token (savich@...	2014.02.21 18:49:12	2017.10.22 16:50:03	CAAB712VtSu8BAI1lhUtqyFzMII...

Accessing iCloud Messages

Extracting and Decrypting Messages

- Obtain PCS Identities structure containing encryption keys
- PCS Identities extracted from two iCloud areas: “Engram” and “Manatee“
  - *MessageManateeZone* for message data
  - *ChatManateeZone* for chat data
  - *AttachmentManateeZone* for attachments
- The keys are required to decrypt iCloud Messages

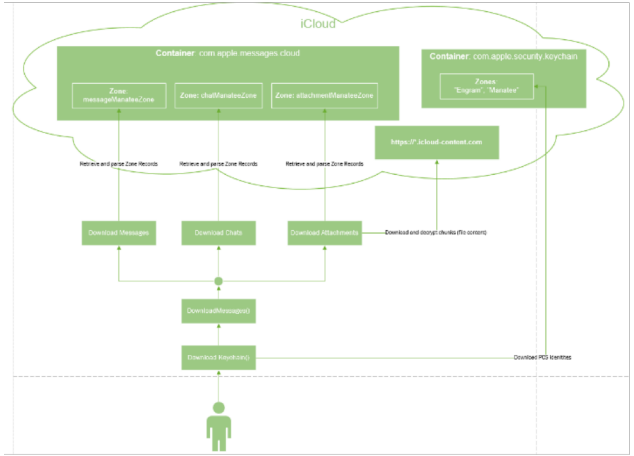



iCloud
Container: com.apple.messages.cloud
Zone: messageManateeZone
Zone: chatManateeZone
Zone: attachmentManateeZone
Container: com.apple.security.keychain
Zones: "Engram", "Manatee"
https://*.icloud-content.com
Retrieve and parse Zone Records
Retrieve and parse Zone Records
Retrieve and parse Zone Records
Download Messages
Download Chats
Download Attachments
Download and decrypt chunks (file content)
DownloadMessages()
Download Keychain()
Download PCS Idenitites

At a Glance

First, let us have a quick look

- Retrieve container "com.apple.messages.cloud", bundle "com.apple.imagent“ from iCloud
- Save publicID required for decrypting the zone key
- publicID identifies the particular PCS Identity required to decrypt data
- These keys will be used to decrypt the data

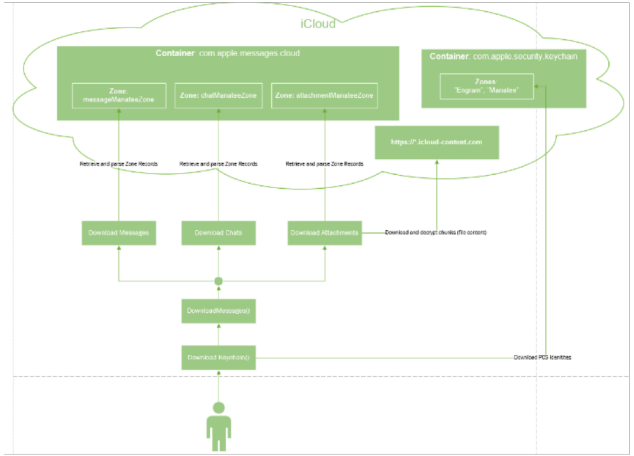


- Most text information (chats, messages, attachment metadata) accessible via CloudKit protocol from the dedicated container:
  - container id = "**com.apple.messages.cloud**"
  - bundle id = "**com.apple.imagent**"

At a Glance

Messages, chats and meta data

- Messages, chats, attachment metadata is available in separate zones:
  - **messageManateeZone**
  - **chatManateeZone**
  - **attachmentManateeZone**



- Non-text content (attachments) accessed separately
- Standard chunk download protocol with modified encryption
- Apple frameworks now define “Ford chunks” (with the keys for data chunks)
- “Ford chunks” must be decrypted before decrypting main chunks

At a Glance

The role of iCloud Keychain

- Some PCS Idenitites that are required to decrypt iCloud Drive data is stored in "**com.apple.security.keychain**“ container, "**com.apple.securityd**“ bundle
- These are encrypted with keys stored in iCloud Keychain
- As a result, one must obtain iCloud Keychain to decrypt Messages

ckzone	identifier	
Engram	com.apple.ProtectedCloudStorage-Messages3	9E47F
Manatee	com.apple.ProtectedCloudStorage-BluetoothCloudPairing2	42944
Manatee	com.apple.ProtectedCloudStorage-Rawhide3	88010
Manatee	com.apple.ProtectedCloudStorage-com.apple.health.sync2	BBF6(
Manatee	com.apple.ProtectedCloudStorage-com.apple.knowledgestore2	FAA8
Manatee	com.apple.ProtectedCloudStorage-com.apple.routined2	DEA4
Manatee	com.apple.ProtectedCloudStorage-com.apple.siri.knowledge2	FC06
Manatee	com.apple.ProtectedCloudStorage-com.apple.siri.profile2	627D
Manatee	com.apple.ProtectedCloudStorage-com.apple.textinput.KeyboardServices.Secure2	1777

At a Glance

iCloud Messages: what’s inside

- messageSubject
- timeDelivered
- messageBody – message body (main text)
- messageBodyData – raw text data in plist format
- timeRead – time the message was read by recepient
- expireState – whether or not the message has expired
- parentChatId – grouping parameter for chats. Identified used to link messages to chats.
- destinationCallerId – message recipient (receiver)

At a Glance

iCloud Messages: what’s inside

- errorCode – error code when sending message
- guid – message identifier used to link messages to attachments
- sender – who sent the message
- service – service that sent the message (e.g. iMessage)
- messageType – the type of message
- time – time of sending the message
- version – message version
- Some other data

At a Glance

Retrieving messages

- Using CloudKit protocol, request records from **messageManateeZone** https://*-ckdatabase.icloud.com/api/client/record/sync

```
POST /api/client/record/sync HTTP/1.1
Host: p66-ckdatabase.icloud.com
Accept-Encoding: deflate, gzip
X-Apple-I-MD-M: **********
X-Apple-I-MD: **********
X-Apple-I-MD-RINFO: **********
X-CloudKit-DatabaseScope: Private
X-Apple-I-Client-Time: **********
Accept: application/x-protobuf
X-CloudKit-UserId: **********
X-CloudKit-ProtocolVersion: client=1;comments=1;device=1;presence=1;records=1;sharing=1;subscriptions=1;users=1;mes
User-Agent: CloudKit/482.30 (13G36)
X-Apple-Request-UUID: **********
X-CloudKit-AuthToken: **********
Connection: keep-alive
X-CloudKit-BundleId: com.apple.imagent
X-CloudKit-ContainerId: com.apple.messages.cloud
Content-Type: application/x-protobuf; desc="https://p33-ckdatabase.icloud.com:443/static/protobuf/CloudDB/CloudDBCl
X-MMe-Client-Info: <iPad3,3> <iPhone OS;11.1;13G36> <com.apple.cloudkit.CloudKitDaemon/482.30 (com.apple.cloudd/482
Content-Length: ***

record_retrieve_changes_request {
    zoneHash: "****************************************"
    header {
        container {
            str: "messageManateeZone"
            num: 6
        }
        userId {
            str: "**********"
            num: 7
        }
    }
    max_records_in_response: 1000
    unknown: 1
    didMigration {
    unknown1: 1
    }
}
```

At a Glance

Retrieving messages

- Decrypt and parse the following fields:
  - "msgProto"
  - "chatID"
  - "dcId"
  - "eCode"
  - "flags"
  - "guid"
  - "msgType"
  - "sender"
  - "svc"
  - "time"

At a Glance

Retrieving chats

- Using CloudKit protocol, send request to retrieve records from **chatManateeZone** https://*-ckdatabase.icloud.com/api/client/record/sync.

```
POST /api/client/record/sync HTTP/1.1
Host: p66-ckdatabase.icloud.com
Accept-Encoding: deflate, gzip
X-Apple-I-MD-M: **********
X-Apple-I-MD: **********
X-Apple-I-MD-RINFO: **********
X-CloudKit-DatabaseScope: Private
X-Apple-I-Client-Time: 2018-11-20T08:49:46Z
Accept: application/x-protobuf
X-CloudKit-UserId: **********
X-CloudKit-ProtocolVersion: client=1;comments=1;device=1;presence=1;records=1;sharing=1;subscriptions=1;users=1;mes
User-Agent: CloudKit/482.30 (13G36)
X-Apple-Request-UUID: **********
X-CloudKit-AuthToken: **********
Connection: keep-alive
X-CloudKit-BundleId: com.apple.imagent
X-CloudKit-ContainerId: com.apple.messages.cloud
Content-Type: application/x-protobuf; desc="https://p33-ckdatabase.icloud.com:443/static/protobuf/CloudDB/CloudDBCl
X-MMe-Client-Info: <iPad3,3> <iPhone OS;11.1;13G36> <com.apple.cloudkit.CloudKitDaemon/482.30 (com.apple.cloudd/482
Content-Length: ***

record_retrieve_changes_request {
  header {
    container {
      str: "chatManateeZone"
      num: 6
    }
    userId {
      str: "**********"
      num: 7
    }
  }
  max_records_in_response: 1000
  unknown: 1
  didMigration {
    unknown1: 1
  }
}
```

At a Glance

Retrieving chats

- Decrypt and parse the following fields:
  - "arch"
  - "cid"
  - "eid"
  - "filt"
  - "gid"
  - "guid"
  - "lah"
  - "name"
  - "ogid"
  - "prop"
  - "ptcpts"
  - "rmn"
  - "rwm"
  - "sqry"
  - "ste"
  - "stl"
  - "svc"
  - "v"

At a Glance

Retrieving attachments

Attachments are downloaded in two steps:

1. Obtain attachment metadata using CloudKit protocol. This includes file name, type and size, date/time etc. This step is similar to steps described above.
2. Download actual content as chunks; decrypt chunks

At a Glance

Retrieving attachments 1: Metadata

Obtain the following data:

mimeType – attachment type

totalBytes – file size

transferState – sending status

isSticker – is it a sticker?

attachmentGuid – attachment identifier

filePath – path to attached file on the device it was sent from

isOutgoing – whether the attachment is originated from this account

UTI – file type identifier (Uniform Type Identifier)

transferName – internal name of attached file

version – attachment version

filename – attached file name

createdDate – attachment creation date

md5Hash - md5 hash of the attached file

Other data

At a Glance

Retrieving attachments 1: Metadata

- Using CloudKit protocol, send request to retrieve records from **attachmentManateeZone** https://*-ckdatabase.icloud.com/api/client/record/sync
- Retrieve list of records
- Each record corresponds to one attachment
- Each attachment may contain several BLOBs (e.g. live photos, still photos and videos)
- Obtain “**cm**” fields
- Decrypt and parse “**cm**” fields
- Parse all fields that contain authInfo structures (e.g. "lqa“)


```
fields {
    propertyName {
      name: "lqa"
    }
    propertyValue {
      valueType: 6
      authInfo {
        owner1Dsid: "********************"
        fileChecksum: "******************************"
        structSize: 81920
        token: "******************************"
        url: "https://p53-content.icloud.com:443"
        owner2Dsid: "********************"
        wrapped_key {
          name: "******************************"
        }
        fileSignature: "******************************"
        downloadTokenExpiration: 1535030520
      }
    }
  }
```

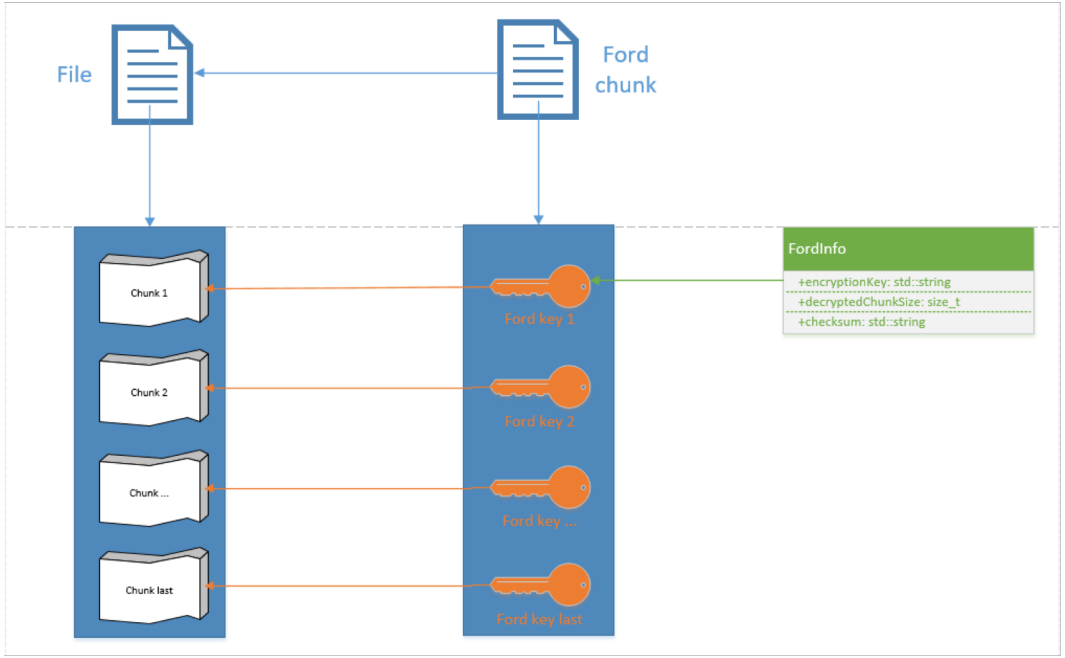
At a Glance

Retrieving attachments 2: Downloading chunks

1. Send request to download attachment file/receive chunks: https://*-content.icloud.com/11179350243/authorizeGet и https://detxl-edge-*.icloud-content.com

2. Receive list of chunks and corresponding Ford chunks.

   Ford chunks are Apple's **new development** for message attachments. They contain decryption keys for the corresponding data chunks



At a Glance

Content chunks and Ford chunks

- Content chunk (attachments):

```
chunk_info {
  chunk_checksum: "\204r\034\362S\314\222g\022\307\245\022s\363\311\265V\037\b\025\244"
  chunk_length: 57344
  chunk_offset: 0
}
```

- Ford chunks (encryption keys for attachment data chunks):

```
chunk_info {
  chunk_checksum: "\f\017\001\004\254{\343\254\323@9,\233\331\311\241y\323/\376V\26
7\214\343\001\277J\374\207(\017\210\374\307.-^b\023ls\3755\257\247"
  chunk_length: 112
  chunk_offset: 0
}
```

- Content chunk (iCloud backups, for comparison):

```
chunk_info {
  chunk_checksum: "\201\366\376-\026\\\360\000\245\374\252y\224C\201`P\030\345\315S"
  chunk_encryption_key: "******************************"
  chunk_length: 181
  4: 0
}
```

At a Glance

Detecting and decrypting Ford chunks

- Ford chunks can be detected by chunk_checksum (beginning with 0x0C, 0x0F, 0x01(\f\017\001\004)
- Assembling the key and decrypting ford chunks to obtain encryption keys for content chunks

```
void DecryptFordKeys()
{
    fordEncryptedData  = read chunk content

    //Create derived key
    assetKey = file->GetToken().unwrappingKey;
    derivedKey = Hkdf(EVP_sha256, assetKey, "PCSMMCS2", assetKey)

    //Decrypt ford chunk
    AesSivCryptor cryptor(derivedKey);

    authData = fordEncryptedData
    cryptor.SetNonce(authData);

    cipherText = fordEncryptedData
    fordDecryptedData = cryptor.Decrypt(cipherText)
}
```

```
chunk_info {
  chunk_checksum: "\f\017\001\004\254{\343\254\323@9,\233\331\311\241y\323/\376V\26
7\214\343\001\277J\374\207(\017\210\374\307.-^b\023ls\3755\257\247"
  chunk_length: 112
  chunk_offset: 0
}
```

At a Glance

Parsing Ford chunks

- On successful decryption, we get a protobuf structure of several versions. Let's keep parsing:

```
message Ford {
  FordInfo ford_info = 1;
  FordInfoArray ford_array = 2;
}

message FordInfo {
  repeated FordEncryptionData enc_data = 1;
  bytes checksum = 2;
}

message FordEncryptionData {
  bytes ford_key = 1;
  bytes chunk_len = 2;
}
```

```
message FordInfoArray {
  bytes checksum = 1;
  repeated FordInfoElement fordInfos = 2;
}

message FordInfoElement {
  bytes chunk_len = 1;
  repeated FordKeyData keyData = 2;
}

message FordKeyData {
  bytes key = 1;
}
```

At a Glance

Detecting content chunks

- Parsing the protobuf structure into an array of encryption keys for data chunks:

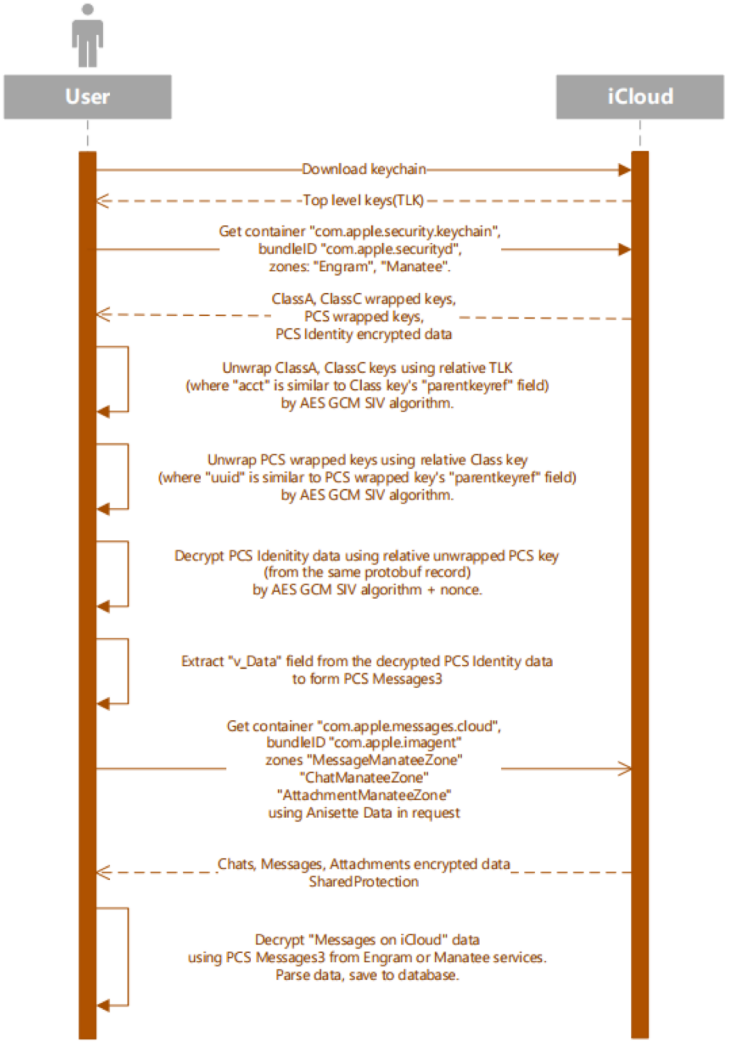
```
struct FordInfo
{
    std::string encryptionKey;
    std::string checksum;
    size_t chunkDecryptedLength;
};
```

- Decrypt content chunks with keys received on step above. Chunks are matched to keys via chunk checksums.
- Assemble chunks into files
- Save files, check decryption and integrity. Done!

Extracting Messages

Steps

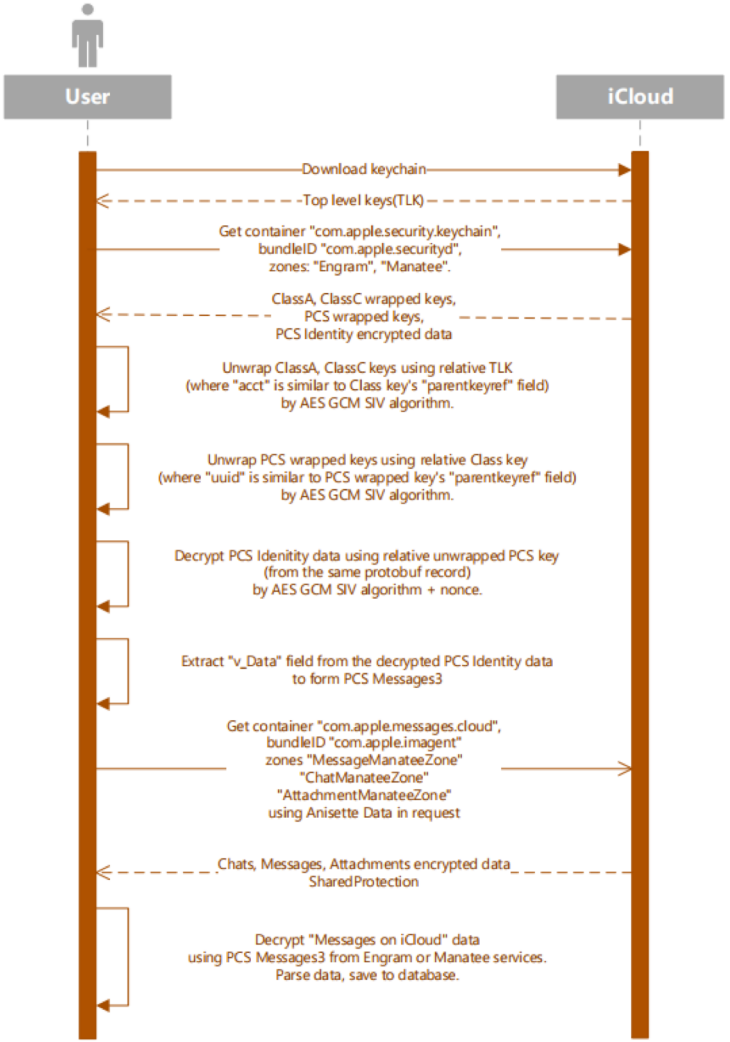
- Access container "com.apple.security.keychain", bundle "com.apple.securityd".
- Download data from zones Engram and Manatee from "com.apple.security.keychain“ container
- Parse zone data, obtain ClassRecord, ZoneRecord



Extracting Messages

Steps

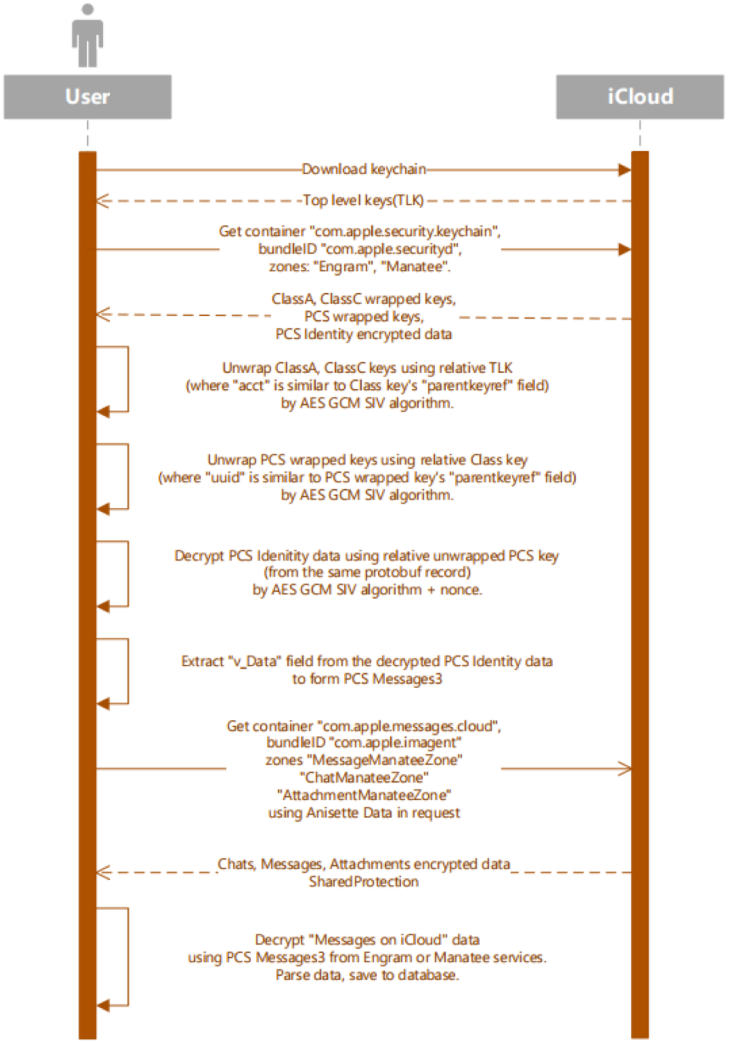
- Locate the required publicID in ZoneRecord
- Obtain wrapped EngramKey/ManateeKey from ZoneRecord that corresponds to the publicID
- Use *parentUdid* and *udid* (ZoneRecord.udid == ClassRecord.parentUdid) to locate ClassRecord corresponding to ZoneRecord
- Extract wrapped ClassAKey/ClassCKey from this ClassRecord



Extracting Messages

Steps

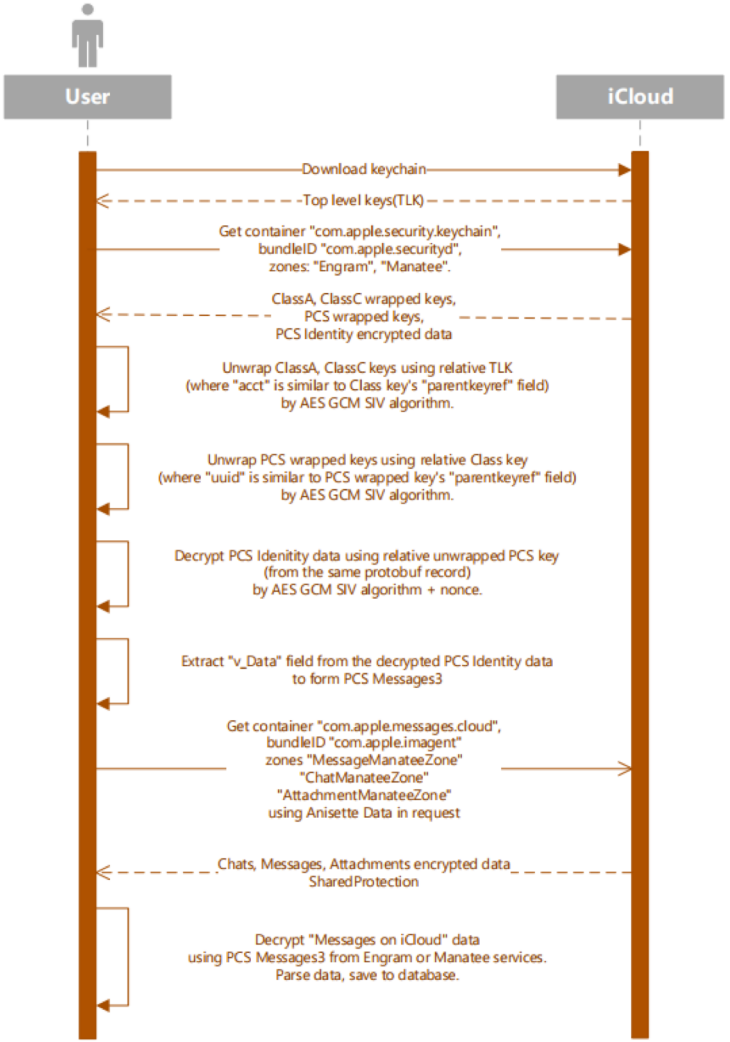
- Download iCloud Keychain
- Analyse iCloud Keychain
  - Use *parentUdid* and *acct* (ClassRecord.udid == TlkRecord.acct) to locate *TlkRecord* corresponding to that *ClassRecord*
- Using *tlk* as a decryption key, unwrap ClassAKey/ClassCKey (AES GCM SIV algorithm)



Extracting Messages

Steps

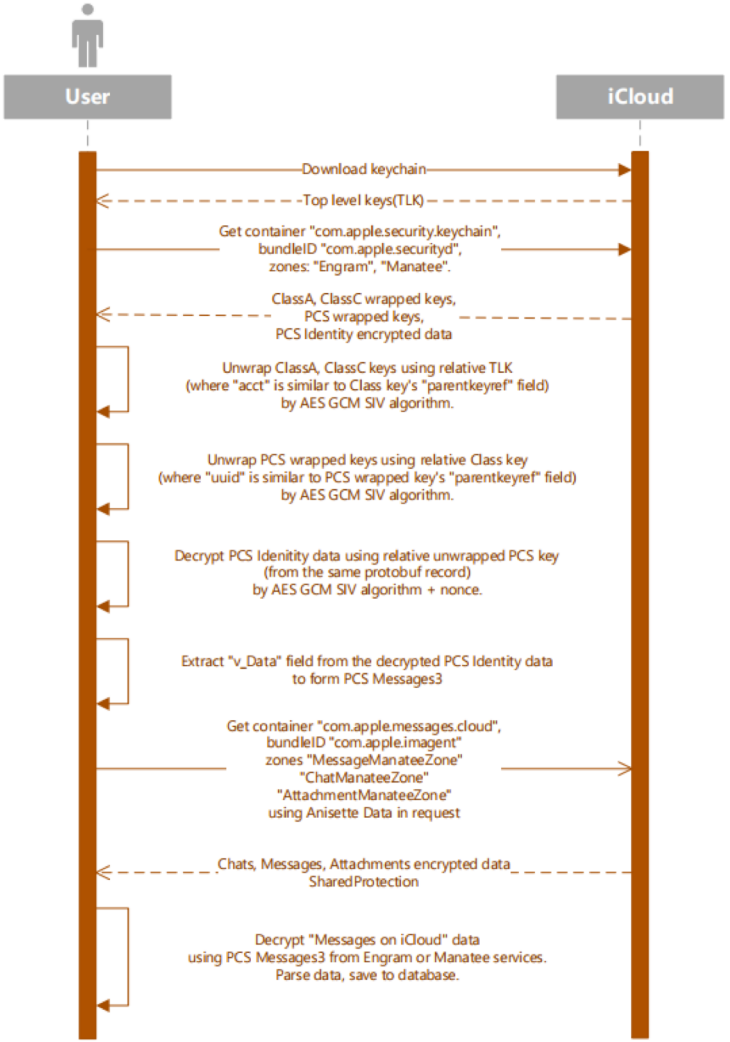
- Using unwrapped ClassAKey/ClassCKey as a decryption key, unwrap EngramKey/ManateeKey data (AES GCM SIV algorithm)
- Using unwrapped EngramKey/ManateeKey as a decryption key, unwrap ManateeEncItem/EngramEncItem data (AES GCM SIV algorithm)
  - First 16 bytes == nonce
  - Remaining data == ciphertext



Extracting Messages

Steps

- From decrypted ManateeEncItem/EngramEncItem data in bplist format, extract ManateePCS/EngramPCS
  - These are PCS Messages3
- Using Messages3 PCS keys, decrypt *messageManateeZone*, *chatManateeZone*, *attachmentManateeZone* from *com.apple.messages.cloud* container
- We have successfully decrypted ***messages***, ***chats*** and ***attachments***



Q&A (1)

- Is it secure after all?
  - *Well, almost*
- Can Apple access your messages?
  - *Seems that not (until backdoor exists)*
- Can Law Enforcement access your messages
  - *Only with forensic software*
- What is needed to get access to messages?
  - *Apple ID, password, second factor, passcode*
- Is real-time surveillance possible?
  - *Yes, but the different way*
- Any chance to access messages w/o password and second factor?
  - *Yes*

Q&A (2), other issues

- Syncing is not reliable (and not in real time)
- Syncing is only possible with 2FA and iCloud keychain
- Syncing of messages is disabled by default
- If sync is enabled, messages & attachments are NOT included into iCloud backups
- What is the risk (or profit, depending on what side you are on)?
- Think of message attachments (media files with EXIF data, link previews etc)
- How to protect yourself?

QUESTIONS?

Cloudy With a Chance of Messages

Extracting Messages from iCloud

Vladimir Katalov, ElcomSoft CEO