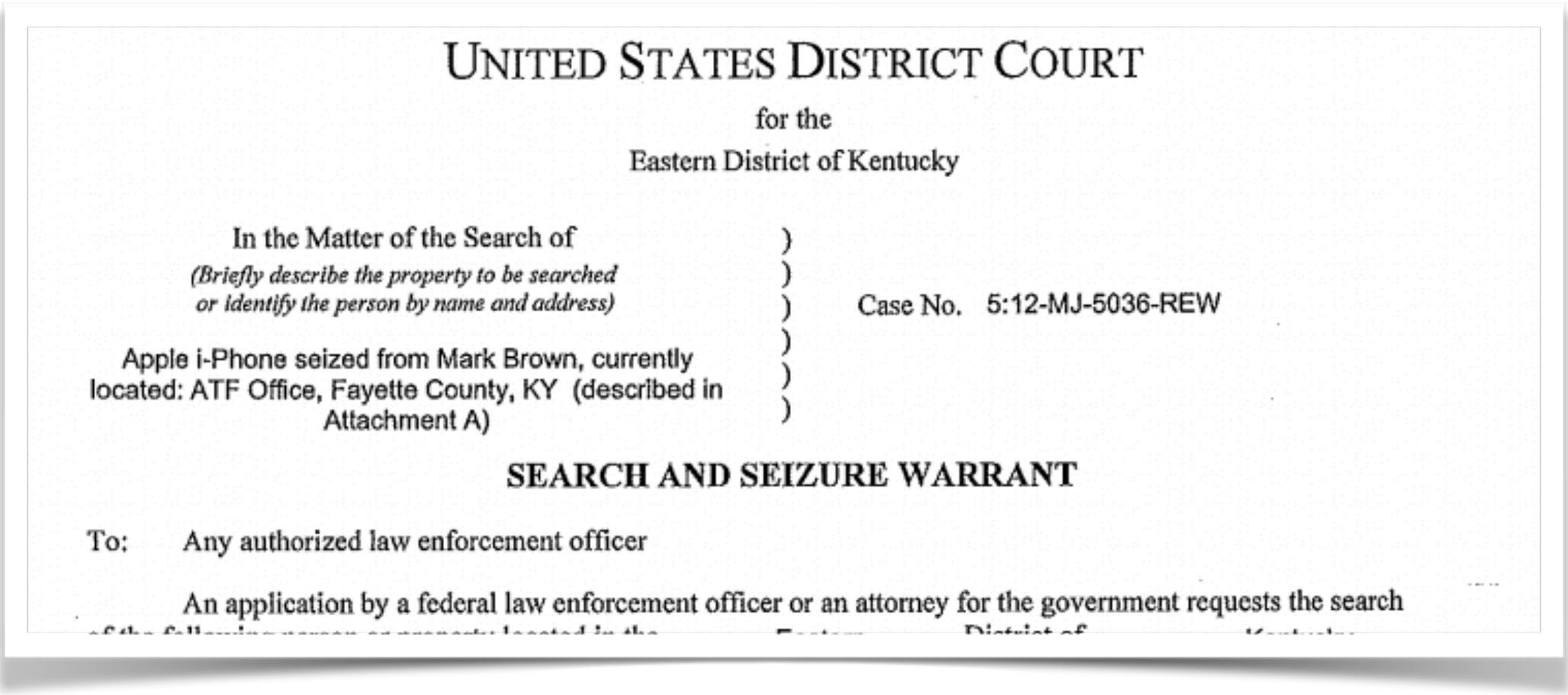
iPhone Encryption, Apple, and The Feds



UNITED STATES DISTRICT COURT

for the

Eastern District of Kentucky

In the Matter of the Search of
(Briefly describe the property to be searched or identify the person by name and address)

Apple i-Phone seized from Mark Brown, currently located: ATF Office, Fayette County, KY (described in Attachment A)

Case No. 5:12-MJ-5036-REW

SEARCH AND SEIZURE WARRANT

To: Any authorized law enforcement officer

An application by a federal law enforcement officer or an attorney for the government requests the search

David Schuetz
@DarthNull
darthnull.org

NoVA Hackers
October 13, 2014

Background

- Apple's new privacy page, "On devices running iOS 8":
  - "Apple cannot bypass your passcode"
  - "…not technically feasible…to respond to government warrants"
- What does that mean? What did they do before?
- What about other forensic ~~attacks~~ Analysis?

On devices running iOS 8, your personal data such as photos, messages (including attachments), email, contacts, call history, iTunes content, notes, and reminders is placed under the protection of your passcode. Unlike our competitors, Apple cannot bypass your passcode and therefore cannot access this data. So it's not technically feasible for us to respond to government warrants for the extraction of this data from devices in their possession running iOS 8.

Apple “deluged” by police

- CNET, May 2013, claims “Apple can bypass the security software”:
  - Big backlog (7 weeks, one case took 4 months)

CNET › Tech Industry › Apple deluged by police demands to decrypt iPhones

Apple deluged by police demands to decrypt iPhones

ATF says no law enforcement agency could unlock a defendant's iPhone, but Apple can "bypass the security software" if it chooses. Apple has created a police waiting list because of high demand.

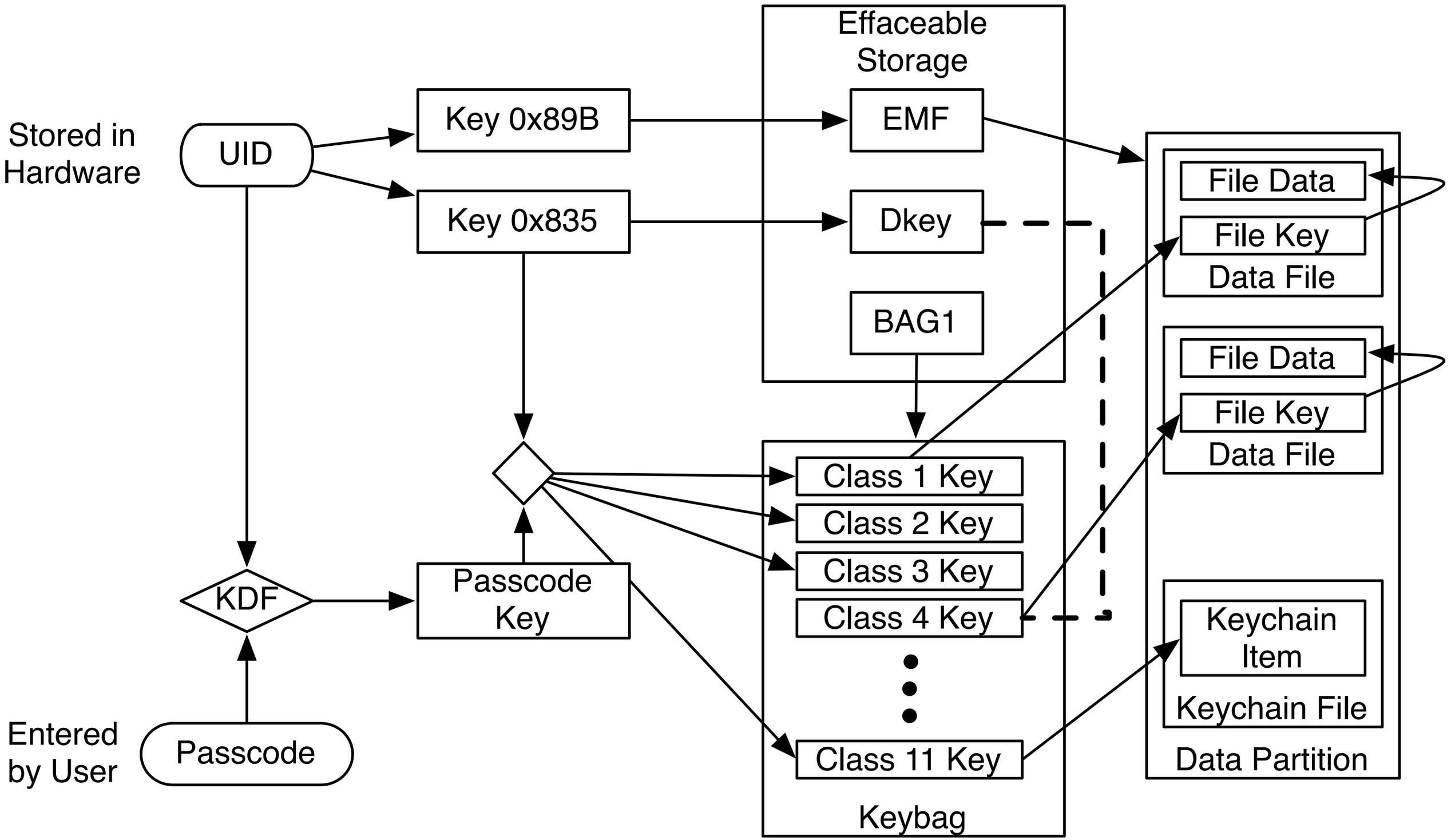
by Declan McCullagh @declanm / May 10, 2013 4:00 AM PDT

What does it MEAN?!?

- Inference:
  - Can’t just plug in and use a magic key
    - Could brute force passcodes, conceivably
- “Apple can afford a LOT of GPU crackers…”
  - It doesn’t work that way

How iOS encryption works

NoVAHackers
Oct 13, 2014

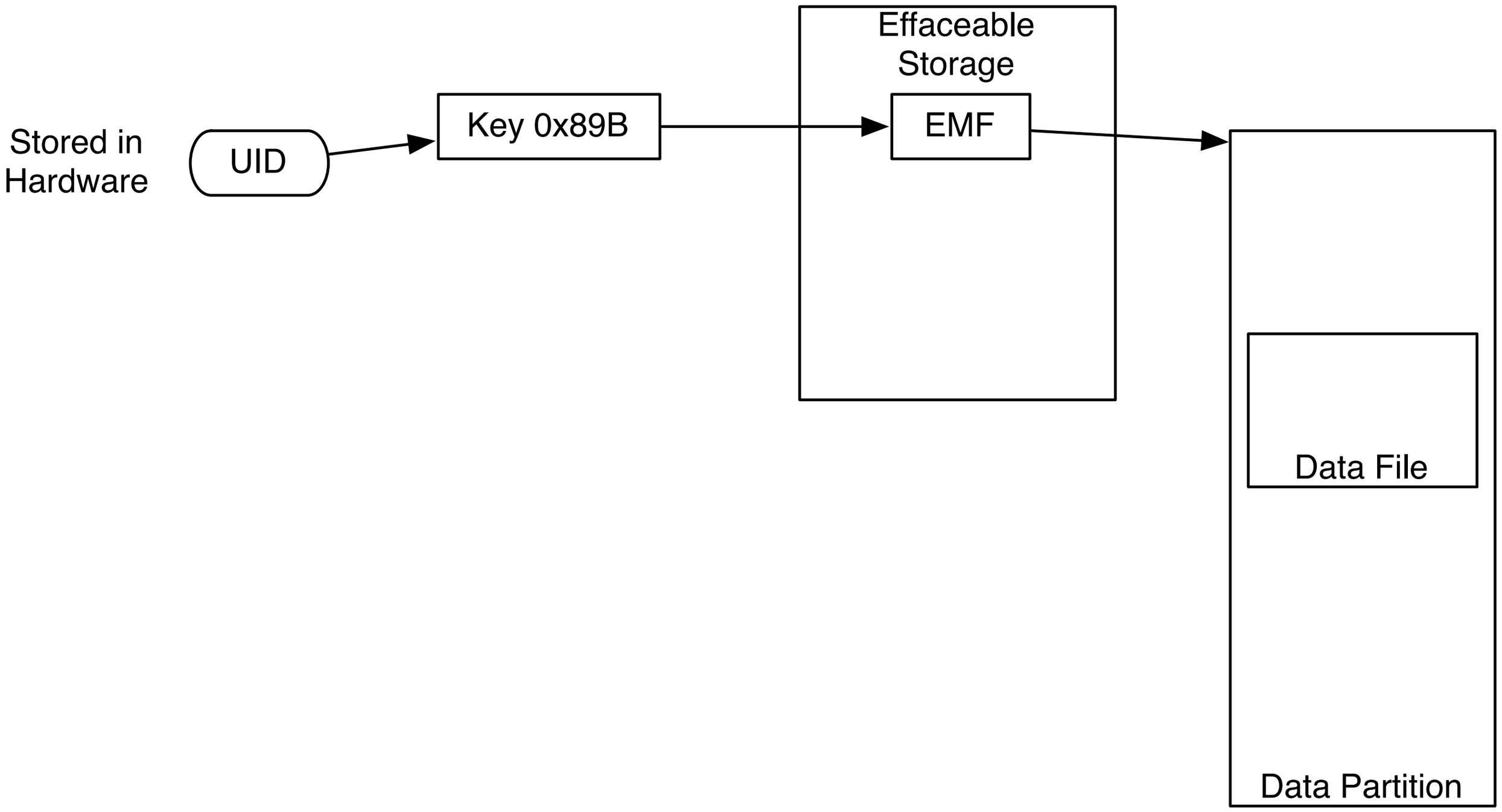


Full disk encryption

- iPhone 3GS / iOS 3
  - Dedicated AES processor
    - Located in DMA channel between CPU and Disk
- Generate a random key (EMF key)
- Encrypt EMF key using a hardware-derived key (0x89b)
- Store encrypted EMF key in special disk area

iOS 3 - FDE

NoVAHackers
Oct 13, 2014



Advantages

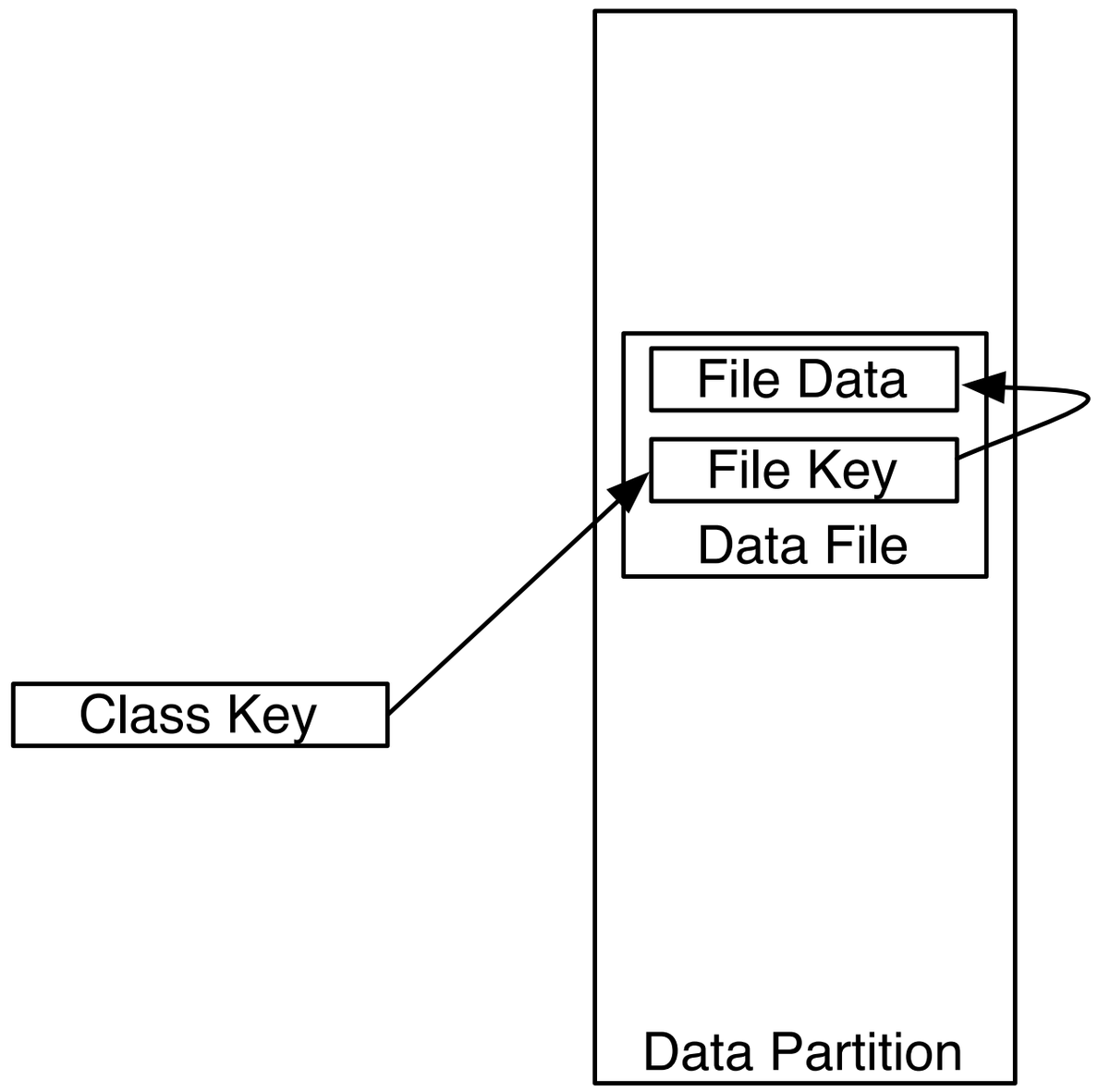
- Advantages
  - Fast wipe
  - Can’t access / modify data directly (without OS)
  - Can’t transfer chips to another device
- Limitations
  - Filesystem access grants access to everything
  - No additional protections when locked

File-level encryption

- Data Protection API introduced in iOS 4
- Random file key created, used to encrypt
- File key is encrypted using a class key
- Encrypted file key stored with file metadata

iOS 4 - Data Protection API

NoVAHackers
Oct 13, 2014



Multiple classes

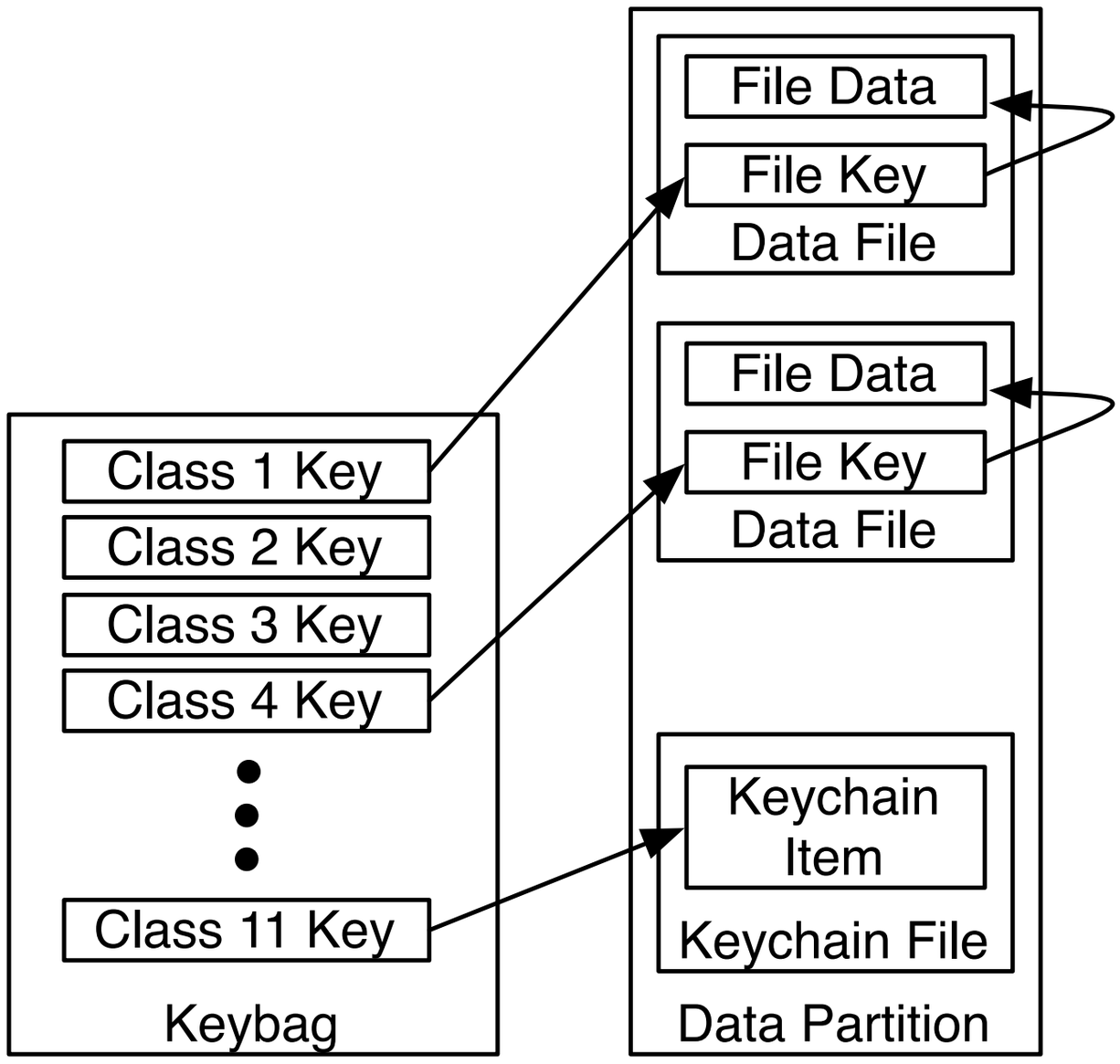
- Default class:
  - iOS 4 - 6 is “no protection”
  - iOS 7 - 8: Complete until First Authentication

- Most system apps through iOS 7 still used None

Protection Name	Description
`None`	**No additional encryption**
`CompleteUnlessOpen`	**Asymmetrical, for locking while writing**
`CompleteUntilFirstUserAuthentication`	**Encrypted after reboot, until 1st unlocked**
`Complete`	**Encrypted when device is locked**

Class keys in the keybag

NoVAHackers
Oct 13, 2014

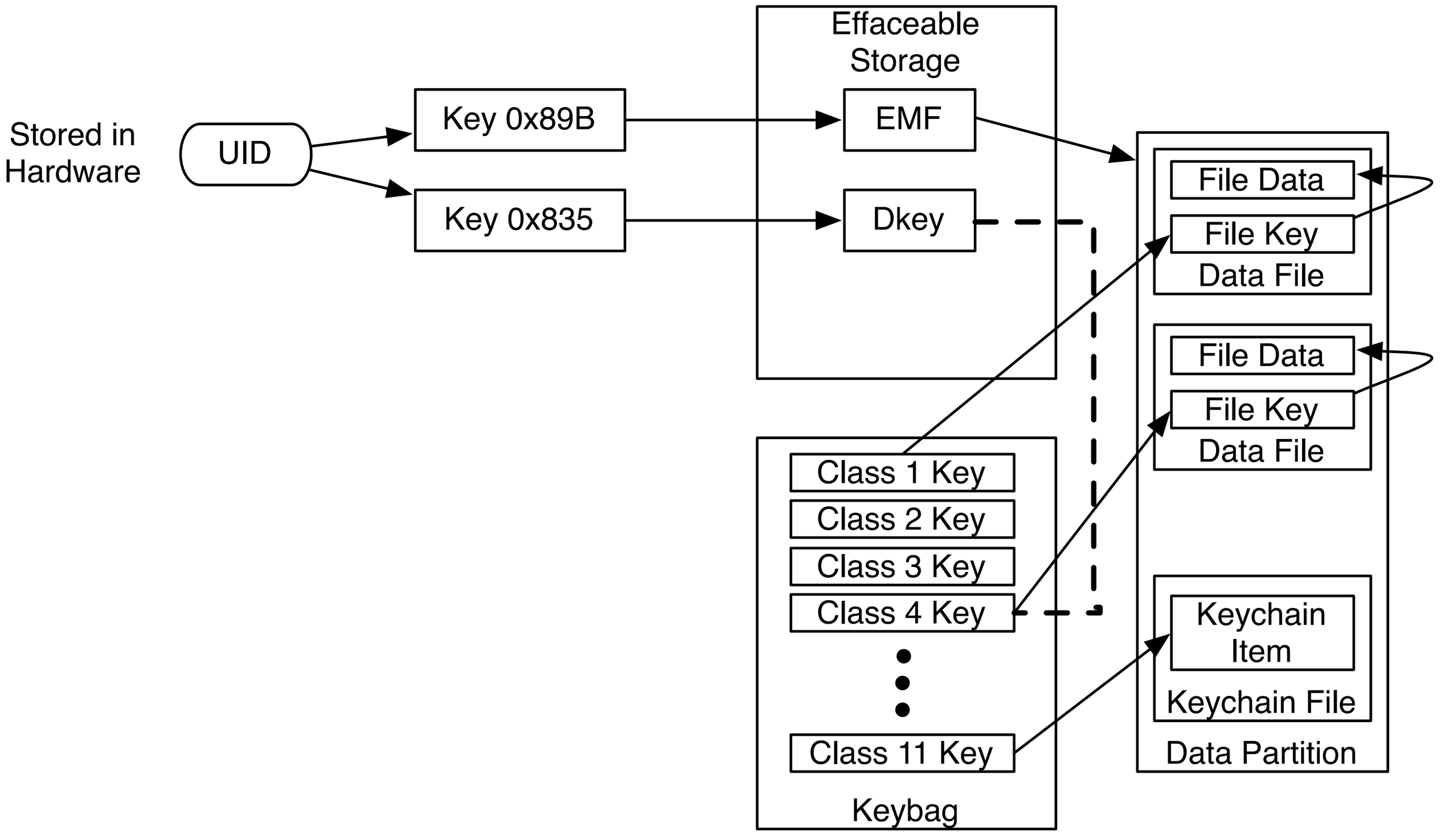


Data Protection: None

- Class 4 or D is File Protection “None” class
- Random Dkey generated
- Encrypted with key 0x835, derived from UID
- Encrypted key stored in effaceable storage

Default protection key

NoVAHackers
Oct 13, 2014

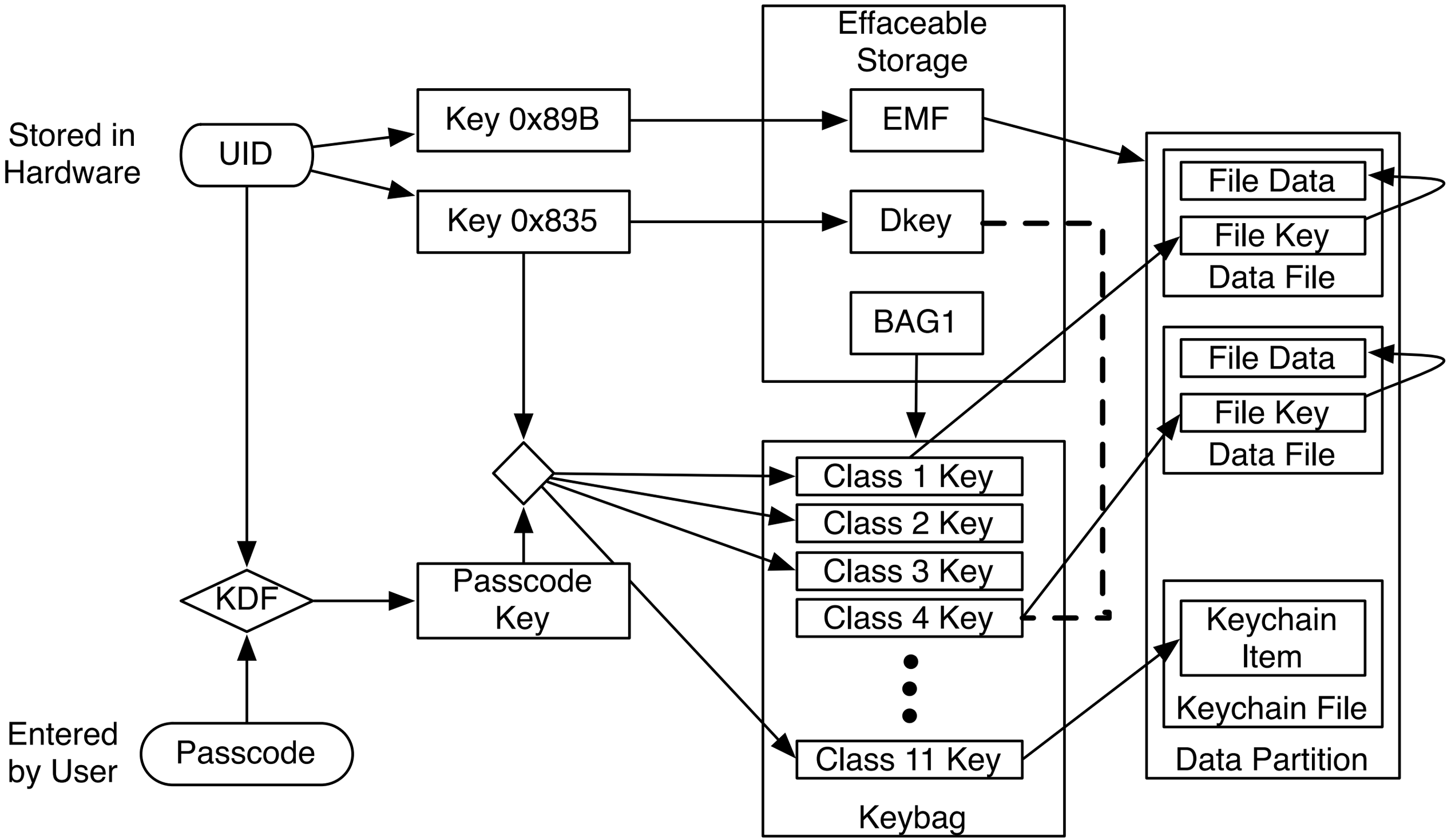


Class key protection

- Each class key is also wrapped or encrypted
  - Using the user's passcode key
- Entire keybag is encrypted
  - Using a bag key (stored in effaceable storage)
- When passcode is changed, old bag keys deleted

Passcode and keybag

NoVAHackers
Oct 13, 2014



Passcode KDF

- PBKDF2, using Passcode, Salt, UID, variable iterations
- Work factor depends on device
  - Constant time — approx. 80 mS / attempt
- A7 and A8 - 5 second delay
- Implemented in hardware (Secure Enclave)
- Depends on UID, which can’t be extracted

Brute forcing passcode

NoVAHackers
Oct 13, 2014

- Must be performed on the device
  - Signed external image
  - Using a bootrom vulnerability
- 80 mS per attempt
  - Now up to 5 sec, so multiply table by ~62
- Attempt escalation, auto-wipe are part of UI
  - When booted from external image, no limits

Size	Time
4-digit numeric	15 min
6-digit numeric	22 hours
6-char lowercase	286 days
6-char mixed case	50 years

Locking…

- FileProtectionComplete key removed from RAM
- All Complete protection files now unreadable
- [I once found an edge case where this doesn't happen…]

Changing passcode…

- The system keybag is duplicated
- Class keys wrapped using new passcode key (encrypted with 0x835 key, wrapped with passcode)
- New BAG key created and stored in effaceable storage
  - Old BAG key thrown away
- New keybag encrypted with BAG key

Rebooting…

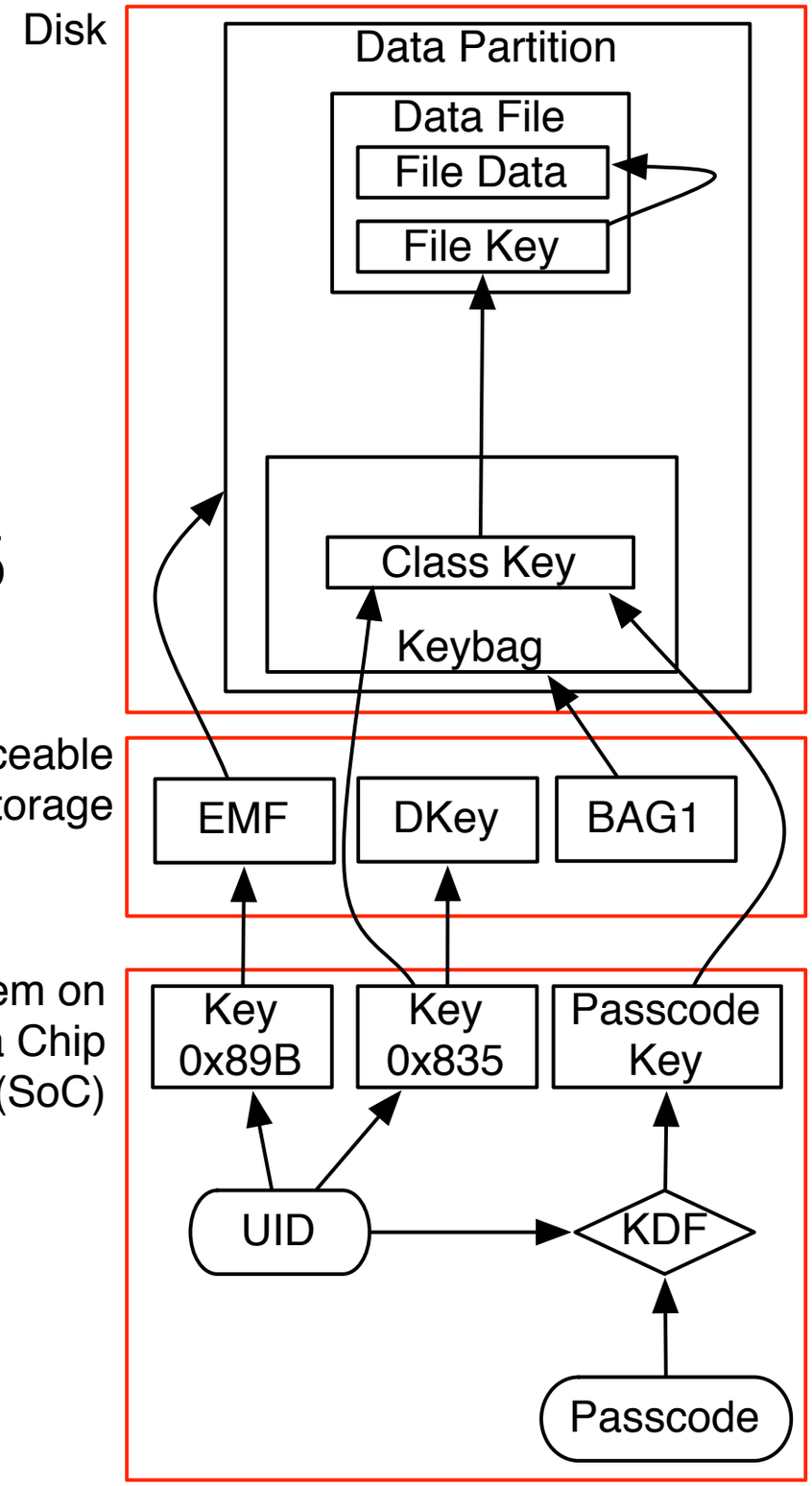
- File Protection Complete key lost
- Complete until First Authentication key also lost
- Only “File Protection: None” files are readable
  - And then only by the OS on the device
    - Because FDE

Wiping device…

- Effaceable storage is wiped, destroying:
  - DKey: All “File protection: none” files are unreadable
  - Bag key: All other class keys are unreadable
  - EMF key: Can’t decrypt the filesystem anyway

Dem bones…

- File is encrypted with a File Key
- File Key encrypted with Class Key
- Class Key encrypted with Passcode, 0x835
- Passcode Key derived from UID
- Keybag encrypted with Bag Key
- Entire disk encrypted with EMF Key
- EMF key encrypted using UID



Bypassing encryption!

NoVAHackers
Oct 13, 2014

- Hardware: AES processor probably inside SoC
- Software: No dice, must boot and get a shell
- Boot: Jailbreak, or boot trusted external image
  - Only Apple can do this
  - Oh, and hackers (iPhone 4 / iPad 1 and earlier)

Apple can get to FS

- Anything with “FileProtectionNone” is readable
- Any other files: Nope
    - Protection Complete: obviously encrypted
    - Complete Until First Unlock: we just rebooted
- What uses “None”?
    - Any apps not updated for iOS 7+
    - Most system apps (up to iOS 7)
    - Preferences, etc.

Lock screen bypasses

NoVAHackers
Oct 13, 2014

- Really just bugs in “Phone App”
- Jumping from one window (lock screen) to others (contact list)
- Even the one bypass (iOS 5, 2011) that got to springboard couldn’t go anywhere due to crypto

Apple and warrants

- Again, didn't have a magic key (else why a backlog?)
- Could attach a trusted image
  - Then read anything that's File Protection: None
- Could maybe brute force passcode
  - Don't know if they offered this as a service
  - Not feasible for strong passcodes

So what changed?

- iOS 7 defaults:
  - 3rd party apps: Complete Until First Unlock
  - System apps: None (except Mail)
- Now System Apps default to Until First Unlock
  - Files unreadable after a reboot

See for yourself

- iOS 7 phone:
  - Reboot, Call from landline
  - See full contact information (name, picture, etc.)
- iOS 8 phone:
  - Reboot, call from landline, just see phone number
  - Unlock, lock again call again
  - Now you see everything

Why does this matter?

- Disk-level forensics require filesystem access
  - (ignoring USB forensics, more shortly)
- Access requires booting from trusted image
- Booting from a trusted image requires:
  - Reboot.
- Therefore “complete until first auth” keys are lost.

What can Apple do now?

- Boot from trusted image
- Extract anything that's not encrypted
  - Data from older applications
  - App preferences (generally unencrypted…may be required)
  - Anything explicitly left unencrypted by developer
  - Odds and ends
- Should be technically able to brute force weak passcodes

Other forensic magic?

- Forensic Magic:
  - I don’t know. They won’t let us play. (also, $$$$)
  - Still subject to encryption controls
- Stuff we understand:
  - Trusted machines can get everything
  - That’s why they’re “Trusted”
  - Log into your co-workers desktop, steal pairing record

Other avenues for police

- Warrant for trusted machines synced to device
- Warrant for iCloud based data
- Warrants for EVERYTHING ELSE stored online
- Court order to unlock phone

Unanswered questions

- Can Apple brute force passcodes?
  - Would they?
  - Could they be ordered to?
  - Has this happened already?
- Is the crypto processor located within the SoC?

Disabling brute force

- Is KDF permanently burned into silicon
  - Or is it part of Secure Enclave firmware
    - Can it be upgraded or replaced?
- Is 5-second delay permanent, or replaceable?
- Have they added a brute force counter in hardware?

Best bet

- Use a strong passcode
- Limit the number of computers which are “Trusted”
- If you’re being arrested, power down the phone

References

- Apple “iOS Security” paper
- “iPhone data protection in depth” (Sogeti, HITB Amsterdam 2011)
- “Evolution of iOS Data Protection and iPhone Forensics: from iPhone OS to iOS 5”, (Elcomsoft, Black Hat Abu Dhabi 2011)
- All noted in recent posts on my blog