What if the FBI tried to crack an Android phone? We attacked one to find out

March 30 2016, by William Enck And Adwait Nadkarni, North Carolina State University

Credit: AI-generated image (disclaimer)

The Justice Department has managed to unlock an iPhone 5c used by the gunman Syed Rizwan Farook, who with his wife killed 14 people in San Bernardino, California, last December. The high-profile case has pitted federal law enforcement agencies against Apple, which fought a legal

order to work around its passcode security feature to give law enforcement access to the phone's data. The FBI said it relied on a third party to crack the phone's encrypted data, raising questions about iPhone security and whether federal agencies should disclose their method.

But what if the device had been running Android? Would the same technical and legal drama have played out?

We are Android users and researchers, and the first thing we did when the FBI-Apple dispute hit popular media was read Android's Full Disk Encryption documentation.

We attempted to replicate what the FBI had wanted to do on an Android phone and found some useful results. Beyond the fact the Android ecosystem involves more companies, we discovered some technical differences, including a way to remotely update and therefore unlock encryption keys, something the FBI was not able to do for the iPhone 5c on its own.

The easy ways in

Data encryption on smartphones involves a key that the phone creates by combining 1) a user's unlock code, if any (often a four- to six-digit passcode), and 2) a long, complicated number specific to the individual device being used. Attackers can try to crack either the key directly – which is very hard – or combinations of the passcode and device-specific number, which is hidden and roughly equally difficult to guess.

Decoding this strong encryption can be very difficult. But sometimes getting access to encrypted data from a phone doesn't involve any code-breaking at all. Here's how:

A custom app could be installed on a target phone to extract information.

In March 2011, Google remotely installed a program that cleaned up phones infected by malicious software. It is unclear if Android still allows this.Many applications use Android's Backup API. The information that is backed up, and thereby accessible from the backup site directly, depends on which applications are installed on the phone.If the target data are stored on a removable SD card, it may be unencrypted. Only the most recent versions of Android allow the user to encrypt an entire removable SD card; not all apps encrypt data stored on an SD card.Some phones have fingerprint readers, which can be unlocked with an image of the phone owner's fingerprint.Some people have modified their phones' operating systems to give them "root" privileges – access to the device's data beyond what is allowed during normal operations – and potentially weakening security.

But if these options are not available, code-breaking is the remaining way in. In what is called a "brute force" attack, a phone can be unlocked by trying every possible encryption key (i.e., all character combinations possible) until the right one is reached and the device (or data) unlocks.

Starting the attack

There are two types of brute-force attacks: offline and online. In some ways an offline attack is easier – by copying the data off the device and onto a more powerful computer, specialized software and other techniques can be used to try all different passcode combinations.

But offline attacks can also be much harder, because they require either trying every single possible encryption key, or figuring out the user's passcode *and* the device-specific key (the unique ID on Apple, and the hardware-bound key on newer versions of Android).

To try every potential solution to a fairly standard 128-bit AES key means trying all 100 undecillion (10^{38}) potential solutions – enough to

take a supercomputer more than a billion billion years.

Guessing the passcode could be relatively quick: for a six-digit PIN with only numbers, that's just a million options. If letters and special symbols like "$" and "#" are allowed, there would be more options, but still only in the hundreds of billions. However, guessing the device-specific key would likely be just as hard as guessing the encryption key.

Considering an online attack

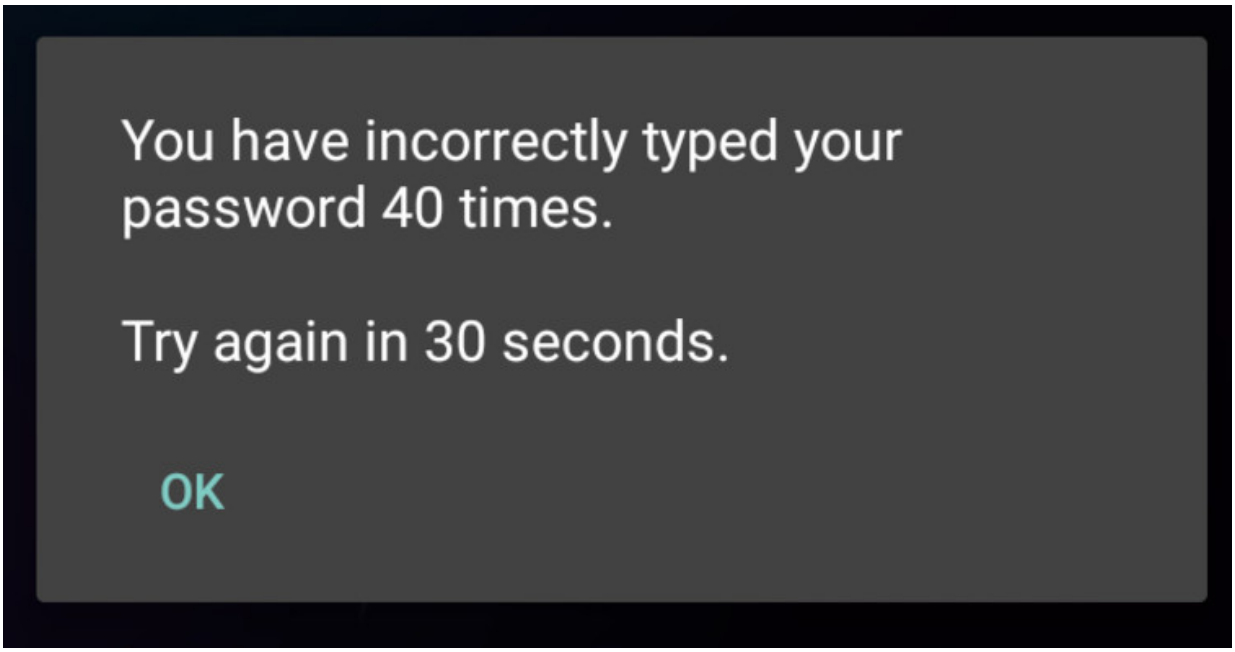
That leaves the online attack, which happens directly on the phone. With the device-specific key readily available to the operating system, this reduces the task to the much smaller burden of trying only all potential passcodes.

However, the phone itself can be configured to resist online attacks. For example, the phone can insert a time delay between a failed passcode guess and allowing another attempt, or even delete the data after a certain number of failed attempts.

Apple's iOS has both of these capabilities, automatically introducing increasingly long delays after each failure, and, at a user's option, wiping the device after 10 passcode failures.

Attacking an Android phone

What happens when one tries to crack into a locked Android phone? Different manufacturers set up their Android devices differently; Nexus phones run Google's standard Android configuration. We used a Nexus 4 device running stock Android 5.1.1 and full disk encryption enabled.

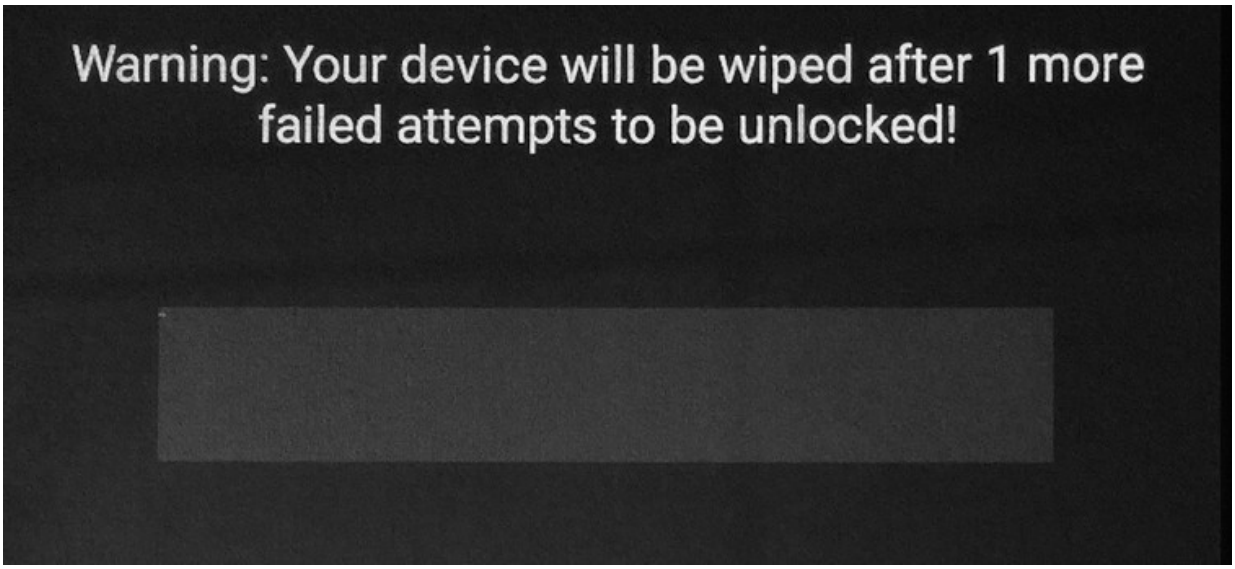


Android adds 30-second delays after every five failed attempts; snapshot of the 40th attempt. Credit: William Enck and Adwait Nadkarni, CC BY-ND

We started with a phone that was already running but had a locked screen. Android allows PINs, passwords and pattern-based locking, in which a user must connect a series of dots in the correct sequence to unlock the phone; we conducted this test with each type. We had manually assigned the actual passcode on the phone, but our unlocking attempts were randomly generated.

After five failed passcode attempts, Android imposed a 30-second delay before allowing another try. Unlike the iPhone, the delays did not get longer with subsequent failures; over 40 attempts, we encountered only a 30-second delay after every five failures. The phone kept count of how many successive attempts had failed, but did wipe the data. (Android phones from other manufacturers may insert increasing delays similar to iOS.)

These delays impose a significant time penalty on an attacker. Brute-forcing a six-digit PIN (one million combinations) could incur a worst-case delay of just more than 69 days. If the passcode were six characters, even using only lowercase letters, the worst-case delay would be more than 58 years.



Just one attempt remaining before the device wipes its data. Credit: William Enck and Adwait Nadkarni, CC BY-ND

When we repeated the attack on a phone that had been turned off and was just starting up, we were asked to reboot the device after 10 failed attempts. After 20 failed attempts and two reboots, Android started a countdown of the failed attempts that would trigger a device wipe. We continued our attack, and at the 30th attempt – as warned on the screen and in the Android documentation – the device performed a "factory reset," wiping all user data.

In contrast to offline attacks, there is a difference between Android and iOS for online brute force attacks. In iOS, both the lock screen and boot process can wipe the user data after a fixed number of failed attempts, but only if the user explicitly enables this. In Android, the boot process always wipes the user data after a fixed number of failed attempts. However, our Nexus 4 device did not allow us to set a limit for lock screen failures. That said, both Android and iOS have options for remote management, which, if enabled, can wipe data after a certain number of failed attempts.

Using special tools

The iPhone 5c in the San Bernardino case is owned by the employer of one of the shooters, and has mobile device management (MDM) software installed that lets the company track it and perform other functions on the phone by remote control. Such an MDM app is usually installed as a "Device Administrator" application on an Android phone, and set up using the "Apple Configurator" tool for iOS.

```
I/System.out(19010): TEST MDM: Resetting Password to :temp1234
D/VoldCmdListener(  169): cryptfs changepw password {}
D/QSEECOMAPI: (  169): QSEECom_start_app sb_length = 0x2000
D/QSEECOMAPI: (  169): App is already loaded QSEE and app id = 1
I/System.out(19010): TEST MDM: Password reset to :true
//...
D/QSEECOMAPI: (  169): QSEECom_shutdown_app
D/QSEECOMAPI: (  169): QSEECom_shutdown_app, app_id = 1
I/Cryptfs (  169): Using scrypt with keymaster for cryptfs KDF
D/QSEECOMAPI: (  169): QSEECom_start_app sb_length = 0x2000
D/QSEECOMAPI: (  169): App is already loaded QSEE and app id = 1
I/Cryptfs (  169): Signing safely-padded object
D/QSEECOMAPI: (  169): QSEECom_shutdown_app
```

Our test MDM successfully resets the password. Then, the scrypt key derivation function (KDF) is used to generate the new key encryption key (KEK). Credit: William Enck and Adwait Nadkarni, CC BY-ND

We built our own MDM application for our Android phone, and verified that the passcode can be reset without the user's explicit consent; this also updated the phone's encryption keys. We could then use the new passcode to unlock the phone from the lock screen and at boot time. (For this attack to work remotely, the phone must be on and have Internet connectivity, and the MDM application must already be programmed to reset the passcode on command from a remote MDM server.)

Figuring out where to get additional help

If an attacker needed help from a phone manufacturer or software company, Android presents a more diverse landscape.

Generally, operating system software is signed with a digital code that proves it is genuine, and which the phone requires before actually installing it. Only the company with the correct digital code can create an update to the operating system software – which might include a "back door" or other entry point for an attacker who had secured the company's assistance. For any iPhone, that's Apple. But many companies build and sell Android phones.

Google, the primary developer of the Android operating system, signs the updates for its flagship Nexus devices. Samsung signs for its devices. Cellular carriers (such as AT&T or Verizon) may also sign. And many users install a custom version of Android (such as Cyanogenmod). The company or companies that sign the software would be the ones the FBI needed to persuade – or compel – to write software allowing a way in.

Comparing iOS and Android

Overall, devices running the most recent versions of iOS and Android are comparably protected against offline attacks, when configured correctly by both the phone manufacturer and the end user. Older versions may be more vulnerable; one system could be cracked in less than 10 seconds. Additionally, configuration and software flaws by phone manufacturers may also compromise security of both Android and iOS devices.

But we found differences for online attacks, based on user and remote management configuration: Android has a more secure default for online attacks at start-up, but our Nexus 4 did not allow the user to set a maximum number of failed attempts from the lock screen (other devices may vary). Devices running iOS have both of these capabilities, but a user must enable them manually in advance.

Android security may also be weakened by remote control software, depending on the software used. Though the FBI was unable to gain access to the iPhone 5c by resetting the password this way, we were successful with a similar attack on our Android device.

This article was originally published on The Conversation. *Read the* original article.

Source: The Conversation

Citation: What if the FBI tried to crack an Android phone? We attacked one to find out (2016, March 30) retrieved 24 June 2024 from https://techxplore.com/news/2016-03-fbi-android.html