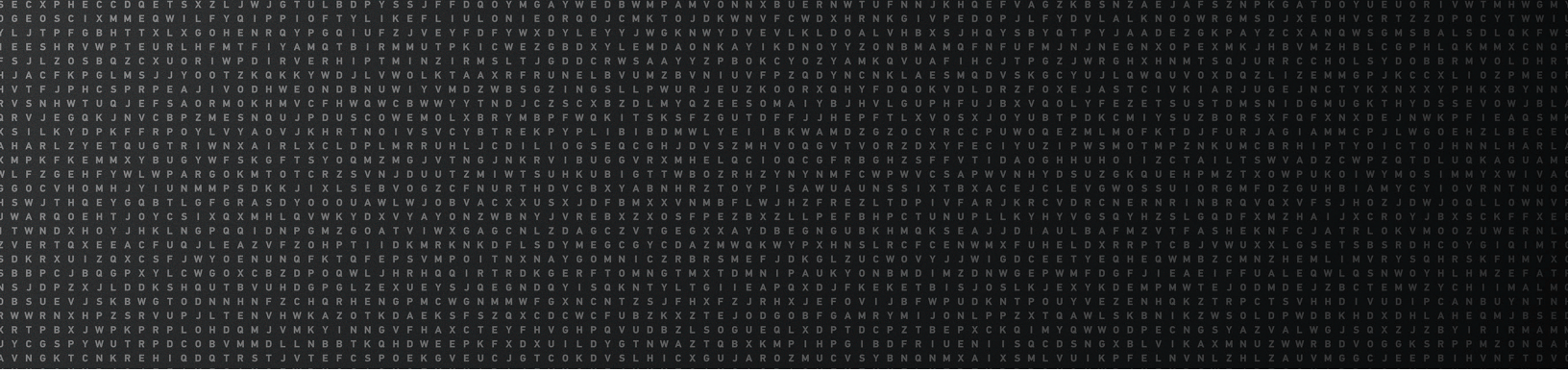
DEMYSTIFYING ANDROID MARSHMALLOW FORENSIC ANALYSIS: A GUIDE



MAGNET FORENSICS®

DEMYSTIFYING ANDROID MARSHMALLOW FORENSIC ANALYSIS: A GUIDE

FORENSIC WORKAROUNDS FOR ANDROID 6.0'S FULL DISK ENCRYPTION, PASSWORD STORAGE, ADOPTABLE STORAGE, AND OTHER SECURITY MEASURES

In a rapidly changing and uncertain world, consumers are pushing back with continued demand for secure devices that protect their data from prying hackers, data thieves, and even, in some cases, governments. Apple continues to lead the way, but as of the launch of Android 6.0 Marshmallow operating system, Google is catching up.

Marshmallow's newly introduced security changes add wrinkles to the forensic process by requiring extra steps, including:

- Full-disk encryption changes
- Application permissions
- Supported fingerprint API
- Factory reset protection (FRP)
- Gatekeeper password storage
- Adoptable storage encryption

For forensic examiners, the impact to previously reliable standby methods such as chip-off, JTAG, ISP, bootloaders, exploits, rooting, and APK injections may seem insurmountable. By tweaking your workflow and applying your natural curiosity, however, you may find that your examinations aren't as unattainable as they might seem.

"WAIT FOR IT..." ANTICIPATING AND ADAPTING TO ANDROID MARKETPLACE CHANGES

Android OS adoption rates are much slower than those seen for iOS. Unlike Apple, Google doesn't automatically push OS updates to its users; rather, carriers tend to control Android updates to allow for testing and approvals. This means that unless users have a Google NEXUS device, they either have to wait for an official carrier release, or they can opt to update for themselves, or they can upgrade their devices. As a result, it can take many months for a new Android OS to peak.

Marshmallow is no exception. As of September 2016—nearly a year following its October 2015 release—its uptake was still only around 19% distribution. However, Samsung, LG, Motorola, HTC, Sony, and other manufacturers began rolling it out to their devices in the last quarter of 2016. By January 2017, Android Community was reporting that Marshmallow's distribution had jumped to 30%—just three points behind previously dominant Lollipop.

And so, as Nougat begins to be released by carriers and manufacturers following a late 2016 launch date—and with only a 1% distribution—it's imperative to understand Marshmallow's impacts on mobile forensics. Not only does this enable you to work on a growing number of devices; it also helps you prepare for Nougat and the yet-to-be-released Android O.

TRULY DEFAULT ENCRYPTION (AND HOW TO BYPASS IT)

Marshmallow's biggest and most disruptive impact by far on forensics is its built-in, default encryption. With lessons learned from Lollipop's "default" encryption problems, Google built Marshmallow to rely on hardware-accelerated, rather than software-based, encryption; furthermore, its APIs contain specifications that devices must meet in order to be encrypted:[1]

- If a device supports a lock screen, it must also support encryption.
- The /data (user data) and /sdcard (mounted storage) partitions—notably, the chief sources of forensic evidence—must be eligible for encryption.
- If the device's AES performance meets 50 megabits per second, then encryption must be an out-of-the-box experience. (Note: This is not enforced on cheaper phones running 32-bit systems on a chip [SoCs]; therefore, it's likely that many prepaid "burner" phones won't meet this specification.)

[1] Note: these requirements exclude Android Wear and Android TV.

Devices built to ship with Marshmallow, of course, are designed to meet all three requirements, and therefore are encrypted out of the box—no longer requiring users to take action to enable encryption, thereby helping to apply passive protection.

In fact, even Marshmallow devices with no password or pattern lock enabled can still present forensic challenges. Most of all, forensic methods like chipoff, JTAG, and ISP will still extract data from damaged devices or devices for which you don't have a password—but it will be unintelligible.

It's now more difficult to make an encrypted-out-of-the box device not run encryption. To do that, the user would have to modify the boot partition by uploading a custom boot partition to phone. In other words, just as the average user tends to leave encryption off when it is off by default, they also tend to leave encryption on when it is on by default.

Best practice: Be creative.

Whether you've been asked to perform a forensic examination by an investigator, or you're an investigator doing your own forensic exams, you may find your powers of interrogation and/or deduction more important than ever as encryption begins to gain more of a foothold in the mobile device space.

WHAT ABOUT UPGRADES TO MARSHMALLOW FROM LOLLIPOP?

A device that was originally shipped with Lollipop, but upgraded to Marshmallow, will make encryption optional in settings (as opposed to devices shipped with Marshmallow, which don't offer that option at all). Because encryption isn't turned on by default even after the upgrade to Marshmallow, the device user has to choose to turn it on. If they haven't done so, or if they've opted not to use a password or pattern lock, the forensic process can continue as normal; if they have enabled encryption, however, the Marshmallow forensics process outlined in this guide is needed.

THE FIRST STEP: THE BOOT PASSWORD

As of Lollipop, it became possible for users to configure an Android device to ask for a password upon booting the device—before loading any data or establishing connectivity. There are four types of boot passwords:

- Default password
- Set password
- Set PIN
- Gesture (pattern)

Upon setup, the device asks the user if they want to encrypt data using that same password. For those that don't, the device uses default_password. It generates a 128-bit key, which it then hashes with the default password and a salt value. This is signed with Trusted Execution Environment (TEE), which is part of Google APIs developed for this purpose.

If the user opts to change from the default_password to a boot PIN or pattern lock, the device re-encrypts the 128-bit key and stores the new information. This process is repeated every time the user changes the password.

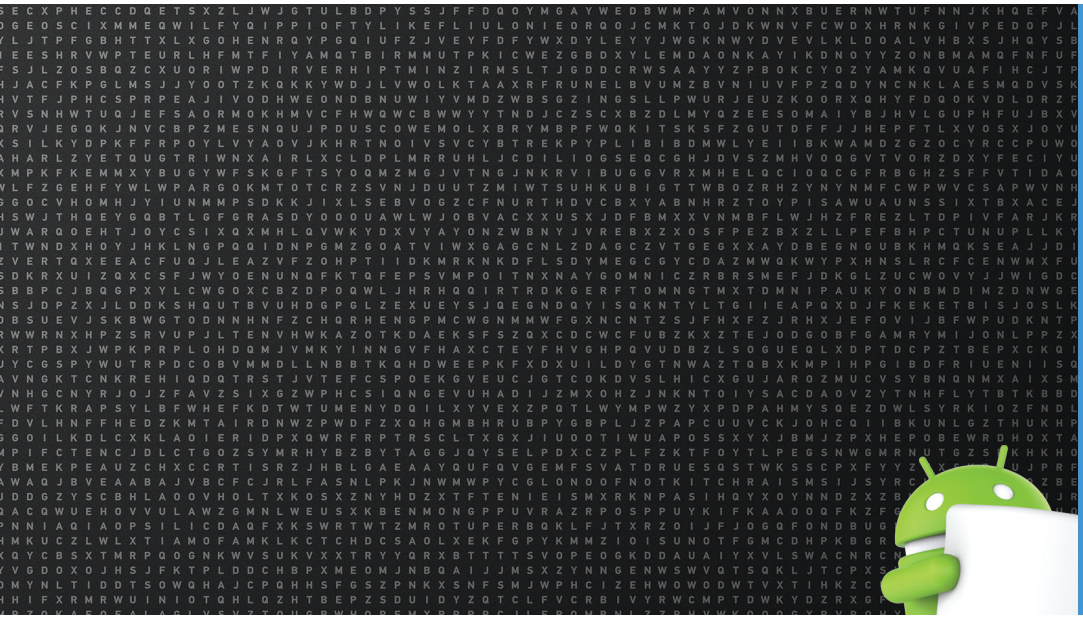
Sometimes, a user may opt to remove the screen lock PIN or change to a pattern screen lock—but not change the boot lock. In that event, the PIN will remain in place as part of the boot process, so that the boot lock and the screen lock are different. If the device dies or the user simply turns it off to encrypt their data, the examination may not be able to move forward if the user doesn't provide or doesn't recall the original boot lock.

SAMSUNG JUST HAS TO COMPLICATE EVERYTHING

Besides the encryption enabled with Marshmallow alone, be aware that some manufacturers, in particular Samsung, have followed Apple's lead in using additional chip-level protection. First, Samsung relies on a different encryption algorithm. This is turned on by default for the Galaxy S7, as well as any Galaxy S6 purchased with Marshmallow. Second, rather than using the industry standard eMMC chip, Samsung manufactures its devices using the universal flash storage (UFS) chip.

This isn't new standard procedure for Samsung. Before Gatekeeper's introduction, for instance, Samsung encrypted its handset PINs and passwords differently. Their software went through more iterations than were standard for other vendors, and they took pains to make their backup encryption stronger.

Other layers of encryption you may encounter in Samsung devices include KNOX software, which is designed for professional users and enables them to encrypt sensitive data on their mobile devices.



Using a custom tool like TWRP [2], ClockworkMod, or a proprietary forensic vendor recovery tool to recover unencrypted data from live devices? Bear in mind that as of TWRP version 3.0, you must either use the "no handset lock" option for bootup (in which case, default_password will be used), or provide the device's passcode.

Best practice:

When obtaining consent to search a Marshmallow-enabled device, be sure to specify both boot lock and screen lock. If the user doesn't use a boot lock, use default_password.

[2] Included in Magnet ACQUIRE as of v2.0.1.

THE NEXT STEP: THE SCREEN LOCK

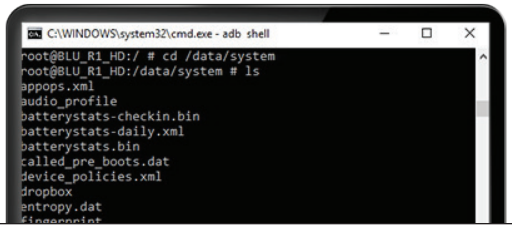
In addition to default encryption, Marshmallow has carried forward another feature introduced with Lollipop: Gatekeeper password storage. This presents another level of obfuscation, in that while PIN and pattern locks are still breakable, the process is not as easy because they no longer use simple hashes.

In previous Android versions, where the hash value of a PIN or pattern lock was stored on the device, it was easy for an examiner to run it against a kind of rainbow table [3], extract the pattern or password, and break the salt easily without much brute forcing.

In contrast, Gatekeeper password storage manages and verifies passwords with a hardware-backed secret key. In addition, it throttles consecutive failed verification attempts, as well as counting the total number of failed attempts. This means tools like MFC Dongle, SV Strike, or others that emulate a keyboard and mouse to brute-force passwords over many dozens of attempts will no longer work.

However, tools like Hashcat or John the Ripper can still brute-force passwords despite this extra level of obfuscation. That's because the keys remain stored in the device—albeit in a different location. Using the Android Debug Bridge (ADB) shell, you can see that instead of the locksettings.db (and associated logfile), password.key, and gesture.key which you may be used to seeing, you now see gatekeeper.password.key and gatekeeper.pattern.key. (Note: the key is stored in big endian for cracking purposes.)

These files are still supplemented by device_policies.xml, which provides information about what type of password is in use, its length, and any special characters. In addition, the text and hex pattern are the same (albeit a little longer), giving password cracker tools what they need to function.



[3] Rainbow tables are a way to search a large amount of hashes quickly. Before Gatekeeper, they didn't apply much to passwords, which were salted, then run through an algorithm that gave a nonreversible SHA1 value. This way, you could find a matching value, because the tables are programmed to know that the value is given under certain condition.

WHAT IF THE DEVICE IS FINGERPRINT-PROTECTED?

For the first time, Marshmallow offers truly built-in, core fingerprint API support—not just support for manufacturer add-on fingerprint reader. These devices are shipped with the reader either on the front or on the back. Whether you can compel the user to provide their fingerprint is up to court systems in your jurisdiction; check with an attorney before moving forward.

Gatekeeper also erases old PINs that are no longer in use. For example, replacing a PIN with a pattern lock results in a gatekeeper.password.key file that is 0 bytes in size. While not immediately relevant to getting into the device, these files could once be used to divulge clues to passwords used for other devices.

Finally, whether tools like the Octoplus flasher box can still reset the screen lock depends on the device. If a device allows the Gatekeeper to be removed, then the box is usable; however, those devices that cannot operate without a user-specified lock risk wiping the device altogether.

Best practice:

Crack the screen lock using the encryption keys found in gatekeeper.password.key and gatekeeper.pattern.key.

ACQUIRING THE DATA FROM THE RIGHT PARTITION

Having gotten through the Marshmallow device's security, it might be tempting to think that all data within the /data directory is immediately available. However, that user data partition—whether /dev/block/platform/msm_sdcc.1/by-name/userdata or (on a physical chip) /dev/block/mmcblk0—is still encrypted, and a forensic analysis won't result in any usable evidence.

Instead, to be able to get a full physical extraction, you need to identify the container that holds the decrypted user data. **This is an extra step, but the most important one**: not every device is exactly the same, and what's the right container on one won't necessarily be the same on another.

Therefore, while many Marshmallow and upcoming Nougat devices are implementing Google's standard dm-0 decrypted partition (a reference to its decryption tool, dm-crypt), you should get in the habit of using the "mount" command in the ADB shell to find the proper container within /dev/block.

(That Android is an open-source operating system works to your advantage as an examiner, because it's easy for you to use simple command-line tools such as the ADB shell to understand variances in file system structures, files, and from there, how to exploit them.)

Point your forensic tool, such as Magnet AXIOM, at this emulated location, and a booted device (or one that is in recovery mode or in full boot process) will be able to be imaged as normal, providing all relevant user data.

Best practice:

While forensic tools adjust to address this change, treat it just as you would ADB imaging and recovery: make sure the correct data partition is mounted, that you apply the correct key, and then the image should proceed as expected.

NEW GRANULAR APP PERMISSIONS

Marshmallow offers an option long desired by Android users: the chance to disable individual permissions from installed apps. Rather than being forced to accept whatever the app wants upon installation, users can now turn on or off their choice of permissions.

While this isn't expected to change forensic examiners' methodology, including root access, it could help to explain why some data is missing from the app database. For example, photos and videos may not show up within Facebook Messenger if the user didn't grant permission for the app to access the device's camera roll.

For APK injections, meanwhile, a popup may want to make sure you know what you're doing, but the new permission structure shouldn't otherwise affect your methods.

WHAT FRP MEANS FOR RECOVERY MODE

Also new in Marshmallow is Factory Reset Protection (FRP), which, like the Find My iPhone Activation Lock, is designed to prevent theft of device or data. Although this feature didn't get a lot of media attention and isn't enabled on all Marshmallow devices—it's most likely to appear on Samsung devices, while LG devices instead rely on ODIN (download) mode—it can still cause issues for forensic examiners on two levels:

1. It can stop you from accessing the device via recovery mode.
2. Forensic tools do not disable it because this would change the evidence.

Advanced-level forensic examiners can use tools like flasher boxes, however, to remove FRP in order to perform a recovery-based extraction. Flasher boxes have long been a staple of digital forensic toolboxes, for instance when you need to downgrade firmware to get access to a device. They're a tool mainly for the highest stakes cases, when the need to retrieve data outweighs the need to preserve an evidence device.

Best practice:

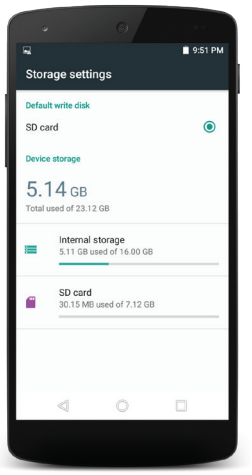
If you have to use these kinds of tools, start with a test device of the same make and model before using on your evidence device, and document your process carefully along with your results.

ADOPTABLE STORAGE

Created primarily for Android One low-end devices, adoptable storage adds support for a new type of "internal" storage: it turns micro SD or on the go (OTG) drives into internal storage areas not only with their own level of encryption, but also unusable on any other device.

By doing this, Marshmallow allows the drive to be seen as part of the mounted option in the device, treating two different storage options as one unit of internal memory. On the device, this displays as a combined "total used of XX GB" broken down between internal and adoptable storage, with data stored across the two without interruption.

A step beyond emulated storage on Android devices that don't have a micro SD card slot, this protects users from security flaws inherent in removable media, but also makes forensic examination more difficult.



So far, Android One has been pushed to markets in Africa, India, and Southeast Asia, but a January 2017 report indicated plans to release it in the United States as well. This runs counter to the historical trend of lower-end devices running older operating systems, by offering up-to-date, secure Marshmallow versions to people from much broader walks of life.

In other words, with adoptable storage, no longer can you simply remove the micro SD or USB drive from the device to image and carve data from it. To make it usable as internal storage, Marshmallow formats the drive in EXT4—which erases any previous data, including photos or apps—and encrypts it using a 128-bit AES key.

This key system ensures the card can't be used within another phone and, because it's encrypted, also ensures you can't read it using Windows EXT programs. Although it's mountable in Linux via live boot CD or virtual machine, forensic imaging will still result in an encrypted, unreadable drive.

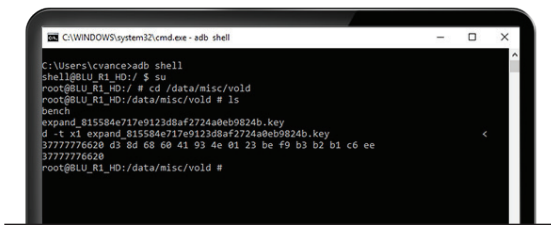
On the other hand, while this key prevents another device from communicating with the card, it also helps to decrypt it. Because the key structure is stored in unprotected plain text in the Android file system, you can use ADB access, an advanced bootloader, or other recovery method to root the device to see this area of the file system.

Within data/misc (the same folder where wpa_supplicant.conf and wifi profiles files are stored), another folder called /data/misc/vold stores volume D keys—including the adoptable storage 128 AES key—within the .key file. From there, you can now decrypt the adoptable storage.

Best practice:

To obtain the key in hex in raw form, expand the .key file using the Linux command od -t x1.

Similar to Android's full disk encryption, the Linux dm-crypt encryption method used here is open source. As a result, you can mount the SD card in a Linux environment, then use dm-crypt together with the extracted key from the evidence device to mount as a decrypted partition and image the internal space.



LEARNING TO IMPROVE YOUR FORENSIC SKILL SET

From a forensic standpoint, all is not lost with Marshmallow. Not only does the logical extraction process still work, enabling you to acquire SMS, call logs, and similar types of data; the ADB backup extraction process still works, too, because either way the device must be booted, whether or not the device has been rooted.

Marshmallow also offers you the opportunity to become a better forensic examiner. Not only can you use the ADB shell to explore its file system more readily; you can also expand your toolbox. When your tools fail, which they will because of one option or another being enabled on any given device or simply because the tool doesn't support the device, it's a new chance to find and learn a new tool.

The major hurdle is in obtaining the device's pattern or PIN lock, which may require investigators to use their powers of persuasion to encourage reluctant subjects to consent. Once that's done, imaging becomes a matter of ensuring your forensic tool is pointed at the appropriate user partition, covering all your device storage bases, and willingness to explore different avenues to get at the evidence you need.

FIND OUT HOW MAGNET AXIOM CAN HELP

If you'd like to learn more about Magnet AXIOM and how it can help find decrypted and obfuscated evidence you may be missing with other solutions, visit magnetaxiom.com. While you're there, you can learn more about the product, request an in-depth personal demo from an AXIOM expert, and request a free 30-day trial version.

Learn more at magnetforensics.com

For more information call us at 1-844-638-7884
or email sales@magnetforensics.com