DM-Verity

Attackers may not only be interested in attempting to modify bootloader or kernel images. There are many other software binaries and configuration files in storage which provide malware the property of persistence. Persistent malware is able to restart itself each time the system is rebooted. It does this by modifying programs or configurations on the system partition, which contains the system binaries, Android framework, and configuration files, that are started during boot. Once inserted into the boot path, the malware can survive system reboots. Additional problems can arise from tampering with system data and configurations, such as the granting of excessive privileges to vulnerable applications.

To prevent unauthorized modifications to the system partition, Knox integrates a customized implementation of Device Mapper Verity (DM-Verity), a Linux/Android kernel module that performs integrity checks on all data blocks contained in a block device (such as a partition). In stock Android, DM-Verity uses a hash tree to perform integrity checks of individual data blocks. The root of the hash tree is signed by an RSA key. Whenever a data block is read into memory, DM-Verity computes the hash of the block, and then uses it, along with the other hashes on the path to the root to compute the root hash. If this computed root hash matches the signed version, the block is considered good. Otherwise, unauthorized modification of the block is detected, and the access to the data block is restricted.

KNOX's implementation of DM-Verity differs from stock Android in supporting file-based firmware over-the-air (FOTA) software updates. This approach is easier to support with the existing infrastructure than the stock block-based approach.