TaintDroid: an extension to the Android mobile-platform that tracks the flow of privacy sensitive data through third-party applications.

\tip

taint flow \_ data dependency(explicitly)

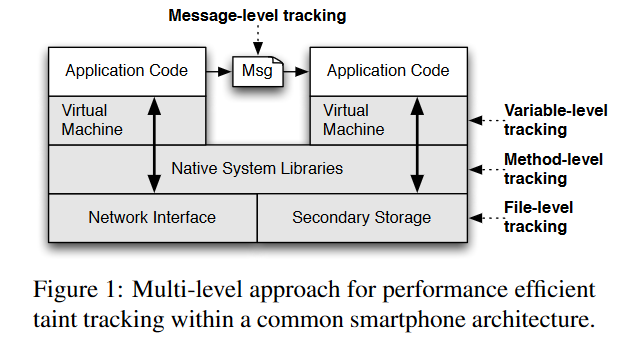
\_ control flow dependency(implicitly)

taint tracking tradeoff: accuracy && efficiency

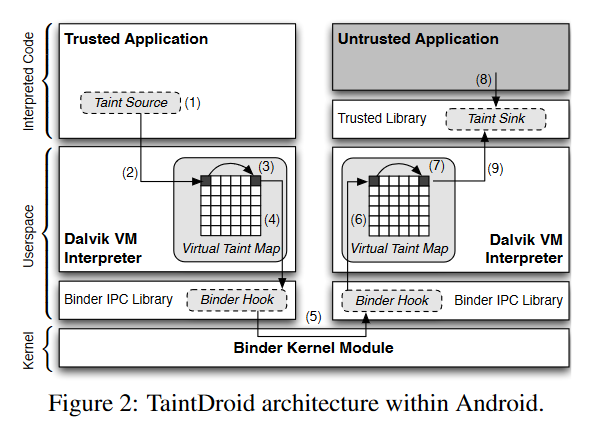
e.g. mark some long arrays containing some taint information, when A read from it, even though A may not read the parts that are tainted, A will be tainted, there might be false positives. Not accurate but more efficient.

four levels of tracking:

* variable-level tracking(Interpreter)
* message-level(IPC)
* method-level(if the params are tainted, the return values are tainted)
* file-level(secondary storage) -> file system extended attributes.



## traindroid



The taint tags are stored in memory and shadow memory.

32-bit bitvector for each variable to encode the taint tag.

the variables that needed to be tracked: method local variables, method arguments, class static fields, class instance fields, arrays.

The propagation of taints is defined by a logic(propagation rules)

the native code follows the same propagation rules plus two necessary postconditions

The sources of the privacy information:

* low-bandwidth sensors
* high-bandwidth sensors
* information databases
* device identifiers
* network taint sink