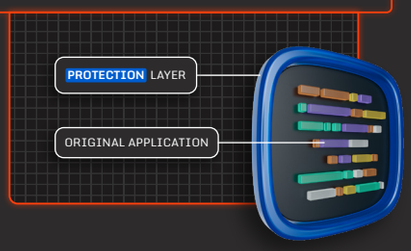
Flutter Mobile App Protection: DOs & DON'Ts

GUARDSQUARE
Mobile application protection

DON'T

Simple protection solutions are easily defeated.

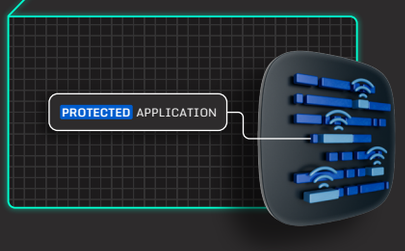
Protecting a Flutter mobile application by relying only on the built-in name obfuscation, or by wrapping the app, may deter the casual attacker, but often prove to be no match for experienced reverse engineers and modders.



DO

Compiler-based solutions are much more effective at deterring reverse engineers and modders.

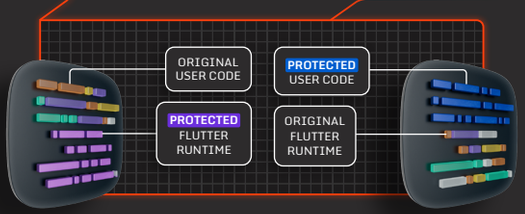
Compiler-based protection solutions modify the app's code itself and provide protection against static and dynamic analysis.



DON'T

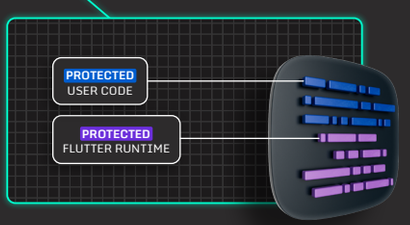
Unprotected or easily detected links between the code and the Flutter engine will result in an easily modified and controlled app.

Replacing the Flutter engine with a tool like Reflutter, is a well-known attack technique enabling reverse engineers to control the execution of the protected application through a modified engine.



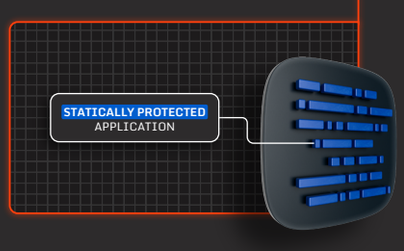
DO

Protecting the interface between the code and Flutter engine is just as critical as protecting the source code to deter Flutter engine replacement attacks.



DON'T

Using only **simple techniques** like renaming classes and methods, which **can be easily defeated, leaves your app vulnerable.**



DO

Adding more **complex obfuscation techniques,** along with **RASP & runtime monitoring,** will provide the **strongest mobile application protection.**