# Problem Description

CFI has long been an issue: examples include etc. Several means of defence have been designed to address some or all of the problematic results. These include:

**Prevention**

* Read XOR Write memory
* Encrypted instructions (allowing decryption based on a particular order of instructions following the application CFG)

**Detection**

* Stack canaries (detect ROP attempts, prevents CFI loss)
* Shadow stacks which aid in preventing return-oriented programming (detect ROP attempts, prevents CFI loss)
* Software-based CFI – Adabi et. al and all afterwards

**Attestation**

* Basic-block IDs hashed to create hash of control-flow path taken.
* Instructions intercepted by secondary hardware monitor to take flow measurements

**But what about audit?**

CFI prevention/detection has proven to lead to a high computational overhead, while attestation requires an always-on connection. What if we can store the control flow of an application on a hard disk, which can be retrieved later? This could be useful to prove to a third party that a particular device did what it should have done when the result wasn’t as expected, or if a particularly important operation has been carried out. It could also be useful for spot-checking inaccessible devices which have no internet connection – such as ICS components. Also see medical devices.

So, while we would not provide immediate detection or real-time attestation, the contents of the audit file can be compared with the CFG for an application at a later time. This could also be useful for reducing computing requirements as comparisons could be scheduled rather than being performed on an ad-hoc basis.