Non-interference

Non-interference analysis is a **flow analysis** that ensure the *absence* of **dangerous flow** between high and low level entities

Dangerous flow means flow from **High** security to **Low** security level

Non-interference

Some examples: Let “H : High” and “L : Low” variables

✓ “H := L”

✗ “L := H”

} **Explicit Flow**

Implicit Flow

✗ “if H = 1 then L := 1 else L := 0”

Suppose H is either 1 or 0,

H is implicitly copied into L

Non-interference

Approach using a **Type systems** and a notion of **soundness** for the system that can be viewed as a form of non-interference

Soundness is established by proving that all *well-typed* programs have this **non-interference** property

Non-interference

Formally, **soundness** states that:

1. $\lambda \vdash c : \rho$,
2. $\mu \vdash c \Longrightarrow \mu'$,
3. $\nu \vdash c \Longrightarrow \nu'$,
4. $\mathrm{dom}(\mu) = \mathrm{dom}(\nu) = \mathrm{dom}(\lambda)$, and
5. $\forall$ l such that $\lambda(l) \leq \tau$, $\nu(l) = \mu(l)$

Implies that $\forall$ l such that $\lambda(l) \leq \tau$, $\nu'(l) = \mu'(l)$

Non-interference

*“A program, has the **non-interference** property if and only if any sequence of low inputs will produce the same low outputs, regardless of what the high level inputs are.“*

Wikipedia