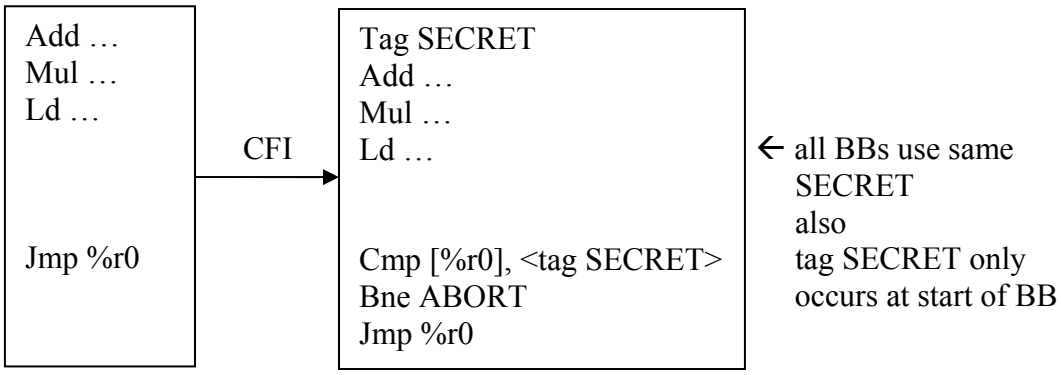
CSE 509 Class Note 3/27

Control Flow Integrity (CFI)

- Complete mediation for IRMs
- Key Idea : Ensure program always jumps to beginning of basic block.
- Basic block
  - Sequence of instruction containing no branches except last instruction
    - Put checks in same Basic block as action being checked

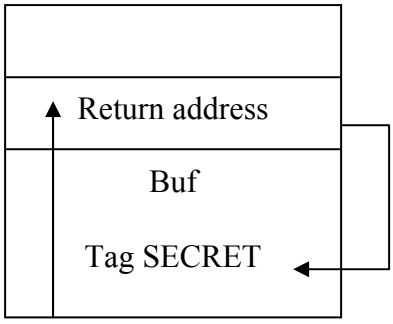
- Ensuring the basic block property

B.B.



- Why secret Tag parameters?
  - Buffer overflow/code injection

  Stack



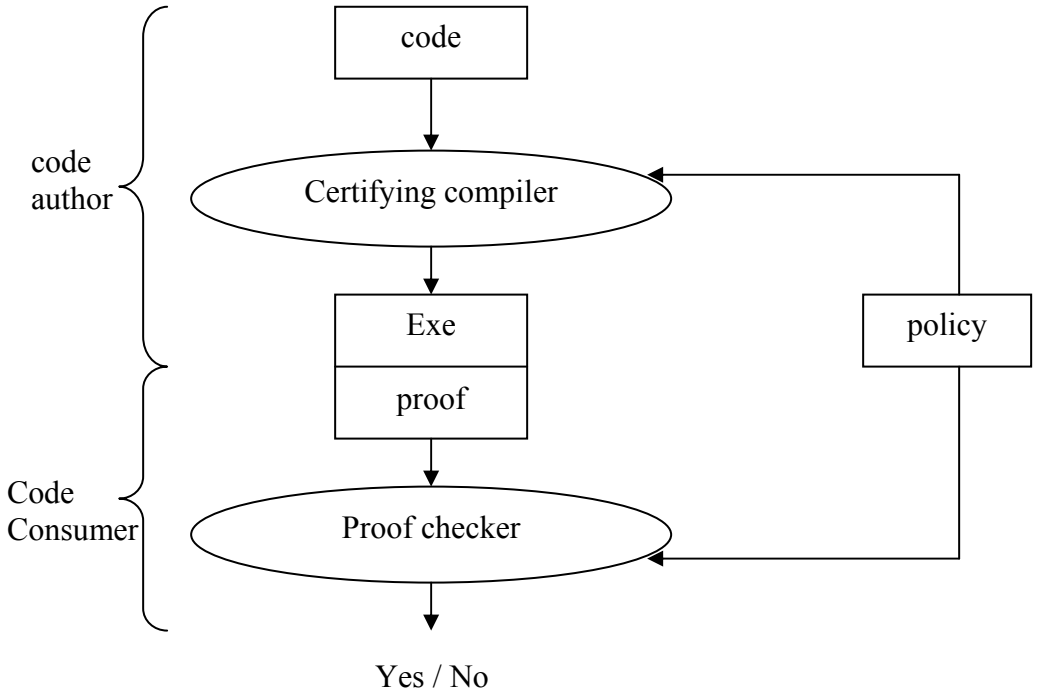
What if writeable pages are non-executable?

→ Then tag parameters does not need to be secret

Proof Carrying Code

- untrusted plugins to trusted system
- Goal : untrusted plugin obeys interface to trusted code
- Offline Reference Monitor
    - Only need to check once
    - Performance

- example policy
    - untrusted function returns linked list of ints
    - untrusted function doesn't modify memory
    - untrusted function terminates in x stpes
    - untrusted function preserves types of memory locations



Proving things about code

Hoare Triple : < P ; S ; Q>
P : predicate, S: statement, Q : predicate
Means "If P is true before S executes, then Q will be true afterwards."
i.e.) <x=5 ; x=x+2 ; x=7>

- Weakest precondition
  Wp(S,Q) = P
  such that <P;S;Q> and all P'
  such that <P';S;Q> P'→ P

i.e.) wp(x=x+2 ; x=7) ≡ x=5
wp(x=x^2 ; x=9) ≡ (x=3 or x=-3)

wp(x=e, Q) ≡ $\lceil e/x \rceil$ * Q

i.e.) wp (x=x+2, x=7) ≡ $\lceil (x+2)/x \rceil$ (x=7)
≡ x+2=7
≡ x=5

wp(x=x^2, x=9) ≡ x^2=9

wp(S1;S2,Q) = wp(S1, wp(S2,Q))

p=wp(S2,Q)

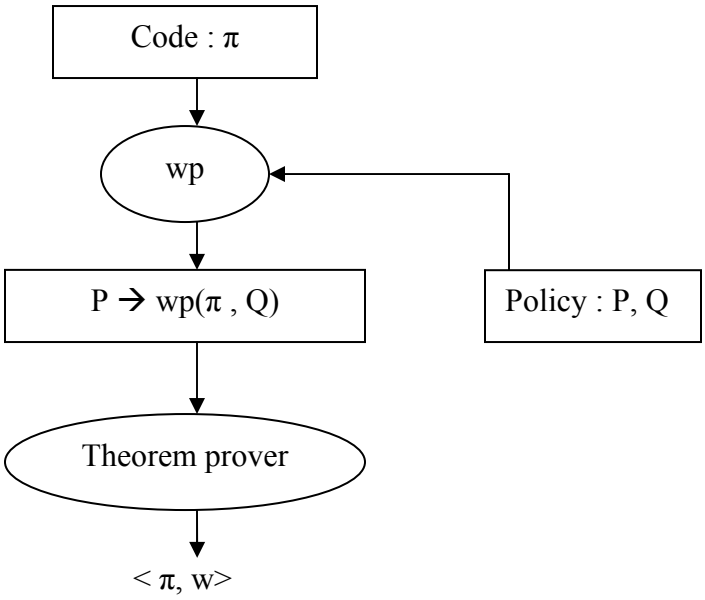
wp(S1,p)

untrusted code : π
trusted system always call π with precondition P and wants π to establish postcondition Q

P → wp(π , Q)

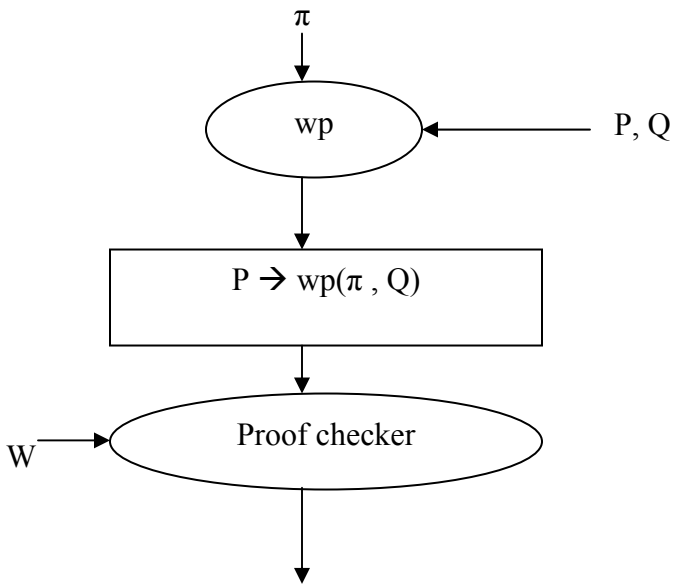
If this is true, code is OK.

- producer



code shipped to consumer

- consumer



Tail (l : τ list) P = l : τ list
Return l→ next ; Q = ret : τ list