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.

```
Add ...
Mul ...
Ld ...
Jmp %r0
```

```
Tag SECRET
Add ...
Mul ...
Ld ...
Cmp [%r0], <tag SECRET>
Bne ABORT
Jmp %r0
```

- 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
  o Only need to check once
  o Performance

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

Proving things about code

Hoare Triple: \(< P ; S ; Q >\)
\(P : \text{predicate, } S : \text{statement, } Q : \text{predicate}\)
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 \(<P';S;Q> P' \rightarrow P\)
i.e.) \( wp(x=x+2; x=7) \equiv x=5 \)
\( wp(x=x^2; x=9) \equiv (x=3 \text{ or } x=-3) \)

\( wp(x=e, Q) \equiv \left\lfloor \frac{e}{x} \right\rfloor * Q \)

i.e.) \( wp (x=x+2, x=7) \equiv \left\lfloor \frac{x+2}{x} \right\rfloor (x=7) \)
\( \equiv x+2=7 \)
\( \equiv x=5 \)
\( wp(x=x^2, x=9) \equiv x^2=9 \)

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

untrusted code : \( \pi \)
trusted system always call \( \pi \) with precondition \( P \) and wants \( \pi \) to establish postcondition \( Q \)

\[
P \rightarrow \text{wp}(\pi, Q)
\]

If this is true, code is OK.
- producer

```
Code : \( \pi \)
```

```
wp
```

```
P \rightarrow \text{wp}(\pi, Q)
```

```
Policy : P, Q
```

```
Theorem prover
```

```
< \pi, w>
```

code shipped to consumer
- consumer

\[
\pi \\
\rightarrow \\
wp \\
\rightarrow P, Q \\
\]

\[
P \rightarrow wp(\pi, Q) \\
\]

\[
\text{Proof checker} \\
\]

\[
\text{Valid / Invalid} \\
\]

\[
\text{Tail (l : \tau list)} \quad P = l : \tau list \\
\text{Return l \rightarrow next ;} \quad Q = \text{ret : \tau list} \\
\]