CCorel

C program \\
wo/buffer \\
overflows \\
\rightarrow \text{CCorel} \rightarrow \text{C program} \\
\wedge \% buffer \\
overflows

insert runtime checks

Common C Mistakes
- accessing outside an array \\
  \Rightarrow \text{bounds check pointer before each dereference.}

ex:

void alloc_and_call_f()
{
    char * p = malloc(...);
    \text{f}(p), \text{what to do bounds checking against?}
}

- possible solution for bounds checking:
  attach additional info \\
to each pointer

- malloc/free bugs \\
  \Rightarrow \text{No Free()}
  \Rightarrow \text{Garbage collector}
- casts & unions

char * p = (char *) 5;
\Rightarrow p \text{ should not be dereferencible}
\Rightarrow \text{casting int to ptr yields an underreferencible pointer.}
struct I
{
    int x;
}

struct P
{
    char *p;
}

struct *I *a;
struct P *p;

q = (struct P*) a;

⇒ Fix to above bar: Every word in memory has a bit that indicates if it's a pointer or an int.

SAFE (restricted C language)
- no pointer arithmetic
- no casts
- no unions
⇒ no bounds checks
⇒ no type checks
⇒ no bounds info

SEQ (another language)
- SAFE
  + pointer arithmetic
⇒ bounds checks/bounds info
⇒ don't need type checks
≤ (WILD, DYNAMIC)
**CCured:**
- partitions program into SAFE, SEQ, WILD parts.
- compile each part
- converts at boundaries

- Analysis is sound
  - code should be free of buffer overflows
- partition performed via type qualifier inference.

**Performance - Evaluation**

Overhead: 1.5x slowdown

SAFE: 90%
SEQ: 9%
WILD: 1%

**Problem**
- CCured vs. non CCured code.
  - convert pointers at interface between
    CCured/non CCured code.
  - may make pointers wild.
Privilege Separation
- Principle of Least Privilege

Better Solution:
Modularize Server

- Programs need privileges for 3 reasons
  - access secret data
  - perform special operations
  - change identity

Poscheck by Niels Provos
- OpenSSH