Memory Overflow

One of the common errors that Insure++ detects occurs when a program reads or writes beyond the bounds of a valid memory area. This type of problem normally generates a READ_OVERFLOW or WRITE_OVERFLOW error which describes the memory regions being accessed with their addresses and sizes as shown below.

```
[hello.c:15] **WRITE_OVERFLOW**
>>
strcat(str, argv[i]); Writing overflows memory: <argument 1>
bbbbbbbbbbbbbb
| 16 |4| wwwwwwwwwwwwwwww
Writing (w) : 0xbfffefb0 thru 0xbfffefc3 (20 bytes) To block (b) : 0xbfffefb0 thru 0xbfffefbf (16 bytes)
str, declared at hello.c, 11
Stack trace where the error occurred: strcat() (interface)
main() hello.c, 15 **Memory corrupted. Program may crash!!**
```

Overflow Diagrams

The textual information above describes the memory blocks involved in the overflow operation using their memory addresses and sizes.

To gain a more intuitive understanding of the nature of the problem, a text-based overflow diagram is also shown. This pattern attempts to demonstrate the nature and extent of the problem by representing the memory blocks involved pictorially.

```
bbbbbbbbbbbbbbbb
|     16       |2|
wwwwwwwwwwwwwwww
```

In this case, the row of b characters represents the available memory block, while the row of w's shows the range of memory addresses being written. The block being written is longer than the actual mem- ory block, which causes the error.

The numbers shown indicate the size, in bytes, of the various regions and match those of the textual error message.

The relative length and alignment of the rows of characters is intended to indicate the size and relative positioning of the memory blocks which cause the error. The above case shows both blocks beginning at the same position with the written block extending beyond the end of the memory region. If the region being written extended both before and after the available block, a diagram such as the follow- ing would have been displayed.

```
     bbbbbbbbbbbbbbbb
| 5  |      16       |2|
wwwwwwwwwwwwwwwwwwwwwwww
```

Completely disjointed memory blocks are indicated with a diagram in the following form:

```
                                                    bbbbbbbbbbbbbbbb
| 4 |            40             |      16      |
wwwww
```

Similar diagrams appear for both READ_OVERFLOW and WRITE_OVERFLOW errors. In the former case, the block being read is represented by a row of r characters instead of w's. Similarly, the memory regions involved in parameter size mismatch errors are indicated using a row of p characters for the parameter block. See PARM_BAD_RANGE for more information.