Suspicious pointer scaling to char

Name: Suspicious pointer scaling to char

Description: Implicit scaling of pointer arithmetic expressions can cause buffer overflow conditions.

ID: cpp/incorrect-pointer-scaling-char

Kind: problem

Severity: warning

Precision: low

Casting arbitrary pointers into char* and then accessing their contents should be done with care. The results may not be portable.

This query finds pointer arithmetic expressions where a pointer to char (or similar) is dereferenced even though the underlying value is of a type larger than char.

Recommendation

1. Whenever possible, use the array subscript operator rather than pointer arithmetic. For example, replace *(p+k) with p[k].
2. Cast to the correct type before using pointer arithmetic. For example, if the type of p is char* but it really points to an array of type double[] then use the syntax (double*)p + k to get a pointer to the k'th element of the array.

Example

```
1  char example1(int i) {
2    int intArray[5] = { 1, 2, 3, 4, 5 };
3    char *charPointer = (char *)intArray;
4    // BAD: the pointer arithmetic uses type char*, so the offset
5    // is not scaled by sizeof(int).
6    return *(charPointer + i);
7  }
8
9  int example2(int i) {
10   int intArray[10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
11   int *intPointer = intArray;
12   // GOOD: the offset is automatically scaled by sizeof(int).
13   return *(intPointer + i);
14 }
```

References

- Common Weakness Enumeration: CWE-468.