**Code Correctness: Comparison of Boxed Primitive Types Development Mitigation SOP**

Code correctness vulnerabilities occur when an Object API is not used properly or as intended. Code correctness vulnerabilities can occur from comparing box primitives using equality operators instead of their equals() method because it can result in unexpected behavior.

**Defense Against Code Correctness: Comparison of Boxed Primitive Types**

The boxed primitives’ equals() method should be used instead of equality operators.

**Examples**

**General Example**

…

Integer mask0 = 100;

Integer mask1 = 100;

…

if( file0.readWriteAllPerms ) { mask0 = 777; }

if( file1.readWriteAllPerms ) { mask1 = 777; }

…

if( mask0 == mask1 ) {

// assume file0 and file1 have same permissions

…

}

…

**Explanation**

The code above uses *Integer* boxed primitives to compare two int values. If both values are 100, then mask0 == mask1 will return *true*, however because both values are equals to 777, that comparison will return false because those values are not within the range of cached values for those boxed primitives. To fix this, the boxed primitives’ equals() method should be used to compare the values.

**Example**

if(claimId != that.claimId) {

return false;

}

**Explanation**

The above example uses the != operator to compare 2 Integers.

**Recommendation**

Object comparisons, such as Integers, should use the equals() method.

**Resources**

1. [HP Enterprise Security – Code Correctness: Comparison of Boxed Primitive Types](https://vulncat.fortify.com/en/detail?id=desc.structural.java.code_correctness_comparison_of_boxed_primitive_types#Java%2fJSP)