**Code Correctness: Class Does Not Implement equals()**

The equals() method is called on an object that does not implement equals().

Explanation

When comparing objects, developers usually want to compare properties of objects. However, calling equals() on a class (or any super class/interface) that does not explicitly implement equals() results in a call to the equals() method inherited from java.lang.Object. Instead of comparing object member fields or other properties, Object.equals() compares two object instances to see if they are the same. Although there are legitimate uses of Object.equals(), it is often an indication of buggy code.

**Example**

public class AccountGroup

{

private int gid;

public int getGid()

{

return gid;

}

public void setGid(int newGid)

{

gid = newGid;

}

}

...

public class CompareGroup

{

public boolean compareGroups(AccountGroup group1, AccountGroup group2)

{

return group1.equals(group2); //equals() is not implemented in AccountGroup

}

}

**Recommendations**

Verify that the use of Object.equals() is really the method you intend to call. If not, implement an equals() method or use a different method for comparing objects.

**Example**

The following code adds an equals() method to the example from the Explanation section.

public class AccountGroup

{

private int gid;

public int getGid()

{

return gid;

}

public void setGid(int newGid)

{

gid = newGid;

}

public boolean equals(Object o)

{

if (!(o instanceof AccountGroup))

return false;

AccountGroup other = (AccountGroup) o;

return (gid == other.getGid());

}

}

...

public class CompareGroup

{

public static boolean compareGroups(AccountGroup group1, AccountGroup group2)

{

return group1.equals(group2);

}

}

**References**

* http://www.hpenterprisesecurity.com/vulncat/en/vulncat/dotnet/code\_correctness\_class\_does\_not\_implement\_equals.html