**Access Specifier Manipulation Development Mitigation SOP**

Access Specifier Manipulation vulnerabilities allow programmers to bypass access control checks through the AccessibleObject API by enabling them to allow a reflected object to bypass Java access controls. This could allow private fields to be modified or for private methods to be invoked, allowing behaviors that would normally be disabled. This could potentially allow Personally Identifiable Information (PII) to be released.

**Defense Against Access Specifiers**

Access specifiers should only be changed by a privileged class using arguments that an attacker cannot set. All occurrences should be examined carefully.

**Example**

void changePersistentStoreType(SessionConfigManager sessionConfigMgr) {

try {

Field persistentStoreTypeField =

getDeclaredField(SessionConfigManager.class,

“peristentStoreType”);

persistentStoreTypeField.setAccessible(true);

persistentStoreTypeField.set(sessionConfigMgr,

persistentStoreTypeOverride);

} catch (NoSuchFieldException | IllegalAccessException e) {

throw new RuntimeException(e);

}

}

**Explanation**

The method setAccessible changes persistentStoreTypeField, an access specifier. This allows the programmer to get around the proper access control checks provided by Java access specifiers, possibly allow PII to be exposed. Access specifiers can be used in a non-vulnerable way when used for Java reflection. For example, the code below shows reflection used to locate a getter for the enum type that will be used to map the events to methods.

**Recommendation**

private void resolveEnumGetter(MultiMethodObserverMapping<?> observerMapping) {

for (Method method :

observerMapping.getEventType().getMethods()) {

if (observerMapping.getEnumType().isAssignableFrom(

method.getReturnType() &&

method.getParameterTypes().length == 0)) {

method.setAccessible(true);

enumGetter = MethodRef.Factory.create(method);

break;

}

}

}

**Resources**

1. [HP Enterprise Security – Access Specifier Manipulation](https://vulncat.fortify.com/en/detail?id=desc.semantic.java.access_specifier_manipulation#Java%2fJSP)