Unchecked Return Value Best Practices

It is not uncommon for Java programmers to misunderstand read() and related methods that are part of many java.io classes. Most errors and unusual events in Java result in an exception being thrown. (This is one of the advantages that Java has over languages like C: Exceptions make it easier for programmers to think about what can go wrong.) But the stream and reader classes do not consider it unusual or exceptional if only a small amount of data becomes available. These classes simply add the small amount of data to the return buffer, and set the return value to the number of bytes or characters read. There is no guarantee that the amount of data returned is equal to the amount of data requested.

This behavior makes it important for programmers to examine the return value from read() and other IO methods to ensure that they receive the amount of data they expect.

If an attacker were to request a piece of data where a variable previously held values for another user, it is possible for information to leak about accounts or values the developer did not intend.

# Preventing Unchecked Return Value

Developers should ensure that the number of bytes or characters expected is the exact amount returned when building responses. Return errors when files or models do not match the expected size.

# Example

FileInputStream fis;

byte[] byteArray = new byte[1024];

for (Iterator i=users.iterator(); i.hasNext();) {

String userName = (String) i.next();

String pFileName = PFILE\_ROOT + "/" + userName;

FileInputStream fis = new FileInputStream(pFileName);

fis.read(byteArray); // the file is always 1k bytes

fis.close();

processPFile(userName, byteArray);

}

If an attacker is able to craft a user file with less than the expected 1k size, information from a previous user could be revealed.

# References

<https://cwe.mitre.org/data/definitions/252.html>

<http://www.hpenterprisesecurity.com/vulncat/en/vulncat/java/unchecked_return_value.html>