OpenMRS

CVE-2017-12796

Context

Deserialization of XML input into objects

Problem

Exploitation of this vulnerability is possible through a single HTTP POST request to the page at http://localhost/openmrs/admin/reports/reportSchemaXml.form Accessing this page through a browser without authenticating first will redirect the user to the login page (so far so good). Under the hood, however, the application is actually executing server-side code before the HTTP redirect response is generated (not so good). Through a Java debugger, with a few strategically placed breakpoints, it becomes apparent that a validation function is being called prior to any auth checks in the reportSchemaXml form controller. By itself, this is a relatively low-severity issue. The end result is still a HTTP 302 to the login page. The real problem here is revealed by stepping into the call to reportService.getReportSchema(rsx). Within this function, a deserialization call can clearly be observed. Furthermore, this deserialization call takes as input user-provided data from the original POST request. Again, this is a pre-auth POST request; no authentication checks have been run. An additional step into the deserialize() function shows that XStream is being used for deserialization instead of builtin Java deserialization fuctions. At this point it has been established that the application is deserializing arbitrary input from an unauthenticated user without any filtering. For a full explanation of why this is so bad, and why this will almost certainly lead to some kind of RCE vulnerability, check out this article by FoxGlove Security. The next step in the exploitation process is to craft a malicious Java object that, when passed to the XStream deserialize() function, will result in RCE.

Solution

Validation of the XML input before deserialization. This avoids that a plug-in injects OS commands.

Issue Tracking URL

- https://isears.github.io/jekyll/update/2017/10/21/openmrs-rce.html

Commit URL
