AAI R2 Security/Vulnerability Threat Analysis

This template is intended to be used to document the outcome of the impact analysis related to the known vulnerability reported by Nexus-IQ (CLM tab in Jenkins). Nexus-IQ can identify the known vulnerabilities contained in the components use by onap components.

This table will be presented to TSC at Code Freeze milestone (M4) to the TSC.

It is recommended to **first update to the latest version** of the third party components available. In case the latest third party components still reports some vulnerabilities, you must provide an impact analysis as illustrated in the example below.

In the case where you have nested third party components (a third party component embedding another third party component) and there is **NO CVE** number for the upstream third party component (meaning the third party component you are embedding), it is recommended to open a vulnerability issue on the upstream third party component.

The following table is addressing 2 different scenarios:

- Confirmation of a vulnerability including an action
- False Positive

The information related to Repository, Group, Artifact, Version and Problem Code are extracted from the CLM report (see the below screenshot)

Repository	Group	Impact Analysis	Action
• aai/model-loader • aai/babel • aai/sparky-be • aai/data-router • aai/aai-resources • aai/aai-traversal • aai/event-client • aai/gizmo • aai/champ	com.fasterxml.jackson.core	False Positive. The exploit primarily is about enabling polymorphic type handling with the object mapper and writing class specifics into the JSON object. There are two ways of doing this: 1. ObjectMapper.enableDefaultTyping() 2. @JsonTypeInfo for marshalling / unmarshalling an object By default the ObjectMapper does not enableDefaultTyping, the code base is not using either approach, so the possibility of the exploit vector does not apply.	
aai/aai-common	com.fasterxml.jackson.core	False Positive. The exploit primarily is about enabling polymorphic type handling with the object mapper and writing class specifics into the JSON object. There are two ways of doing this: 1. ObjectMapper.enableDefaultTyping() 2. @JsonTypeInfo for marshalling / unmarshalling an object By default the ObjectMapper does not enableDefaultTyping, the code base is not using either approach, so the possibility of the exploit vector does not apply.	
• aai/aai-resouces • aai/aai-traversal • aai/champ	org.codehaus.jackson	False Positive. The exploit primarily is about enabling polymorphic type handling with the object mapper and writing class specifics into the JSON object. There are two ways of doing this: 1. ObjectMapper.enableDefaultTyping() 2. @JsonTypeInfo for marshalling / unmarshalling an object By default the ObjectMapper does not enableDefaultTyping, the resources code bases are not using either approach, so the possibility of the exploit vector does not apply.	~~AAI-900~~ - Security: CVE-2017-7525 jackson-mapper-asl 1.9.2 CLOSED

• aai /champ	org. codehaus.jackson	False Positive. The exploit primarily is about enabling polymorphic type handling with the object mapper and writing class specifics into the JSON object. There are two ways of doing this: 1. ObjectMapper.enableDefaultTyping() 2. @JsonTypeInfo for marshalling / unmarshalling an object By default the ObjectMapper does not enableDefaultTyping, the resources code bases are not using either approach, so the possibility of the exploit vector does not apply.	
aai/aai-common	org. codehaus.jackson	False Positive. The exploit primarily is about enabling polymorphic type handling with the object mapper and writing class specifics into the JSON object. There are two ways of doing this: 1. ObjectMapper.enableDefaultTyping() 2. @JsonTypeInfo for marshalling / unmarshalling an object By default the ObjectMapper does not enableDefaultTyping, the code base is not using either approach, so the possibility of the exploit vector does not apply.	
aai/search-data-service	com. fasterxml .jackson. core	False Positive. The exploit primarily is about enabling polymorphic type handling with the object mapper and writing class specifics into the JSON object. There are two ways of doing this: 1. ObjectMapper.enableDefaultTyping() 2. @JsonTypeInfo for marshalling / unmarshalling an object By default the ObjectMapper does not enableDefaultTyping, the search service is not using either approach, so the possibility of the exploit vector does not apply.	
aai/esr-server	com. fasterxml .jackson. core	False Positive Explanation: This vulnerability issue only exists if com.fasterxml.jackson.databind.ObjectMapper.setDefaultTyping() is called before it is used for deserialization. esr-server doesn't invoke this method, esr-server use new Gson().fromJson(String json, Obj.class) and new Gson().toJson(obj) to deserialization and serialization. https://github.com/FasterXML/jackson-docs/wiki/JacksonPolymorphicDeserialization In esr-server, Gson is used to deserialization and serialization: https://gerrit.onap.org/r/gitweb?p=aai/esr-server.git;a=blob;f=esr-mgr/src/main/java/org/onap/aai/esr/wrapper/EmsManagerWrapper.java;h=588baad96c7942e83e0670784bbf423505c7b194;hb=HEAD https://gerrit.onap.org/r/gitweb?p=aai/esr-server.git;a=blob;f=esr-mgr/src/main/java/org/onap/aai/esr/wrapper/ThirdpartySdncWrapper.java;h=874205920c156f12df0bc591638a24e3f5575c76;hb=HEAD https://gerrit.onap.org/r/gitweb?p=aai/esr-server.git;a=blob;f=esr-mgr/src/main/java/org/onap/aai/esr/wrapper/VimManagerWrapper.java;h=fe44536cecb3f9ae9eaa3d99ff7b2d52511e2d52;hb=HEAD https://gerrit.onap.org/r/gitweb?p=aai/esr-server.git;a=blob;f=esr-mgr/src/main/java/org/onap/aai/esr/wrapper/VnfmManagerWrapper.java;h=8c7c5d39ceadff5e17f9c6d26d5540be49ada070;hb=HEAD https://gerrit.onap.org/r/gitweb?p=aai/esr-server.git;a=blob;f=esr-mgr/src/main/java/org/onap/aai/esr/util/ExtsysUtil.java;h=3bd01772356055e9711705b8518d55f1678b5179;hb=HEAD	
• aai/aai-resources • aai/aai-traversal • aai/aai-common	org. apache. activemq	There is no newer version of the dependency to upgrade to. Issue is a false positive. This vulnerability is dependent on XalanXPathEvaluator.java using an insecure or absent document parser. AAI is not using this class.	

aai/champ	commons-httpclient	False positive. This is imported by hadoop which is used for hbase configs; in Beijing, AAI is configured with Janus on cassandra so it will not be accessing these classes. In Casablanca, Champ will serve as a multi-purpose data broker so we will look to upgrade the hadoop libraries to the most current versions.	