Vulnerability Advisory

Arbitrary File Read/Write in Kafka Connect Default Configurations

Created by Luca Carettoni
05/17/2022

Overview

This document summarizes a security vulnerability that was disclosed to the Apache Kafka Security Mailing List on November 23rd, 2021. Fixed released by Apache on May 17th, 2022.

All information contained in this document is believed to be accurate at the time of publishing, based on currently available data. The author reserves the right to update the document in case of any error or imprecision.

About Us

Doyensec is an independent security research and development company focused on vulnerability discovery and remediation. We work at the intersection of software development and offensive engineering, to help companies craft secure code.

Research is one of our founding principles and we invest heavily in it. By discovering new vulnerabilities and attack techniques, we constantly improve our capabilities and contribute to securing the applications we all use.

Arbitrary File Read/Write in Kafka Connect Default Configurations	
Vendor	Apache Kafka
Severity	High
Vulnerability Class	CWE-749: Exposed Dangerous Method or Function
Component	Kafka Connect < v6.x
Status	Closed
CVE	-
Credits	Andrea Brancaleoni and Luca Carettoni / Doyensec LLC

Summary

Kafka Connect is a tool for reliably streaming data between Apache Kafka® and other data systems. It makes it simple to quickly define connectors that move large data sets into and out of Kafka. Kafka Connect can ingest entire databases or collect metrics from all your application servers into Kafka topics, making the data available for stream processing with low latency. An export connector can deliver data from Kafka topics into secondary indexes like Elasticsearch or into batch systems such as Hadoop for offline analysis.

By default, Kafka Connect has the following Connector plugins available in the default config path, which are configurable via their respective `.properties` files:

```
org.apache.kafka.connect.file.FileStreamSinkConnector [Vulnerable]
org.apache.kafka.connect.file.FileStreamSourceConnector [Vulnerable]
org.apache.kafka.connect.mirror.MirrorCheckpointConnector
org.apache.kafka.connect.mirror.MirrorHeartbeatConnector
org.apache.kafka.connect.mirror.MirrorSourceConnector
```

Once Kafka Connect is initialized, with any supported connector property configuration in standalone or distributed mode, port 8083 serves the REST API to the Kafka Connect cluster. By default, **the API is accessible in an unauthenticated manner unless authentication is specifically configured**. A default instantiation of the Kafka Connect cluster would allow an attacker, with network access to the Connect server, to configure an environment with the `FileStreamSource` connector and then read files directly from

the Connect server, without further authentication, with the same user privileges and context as the Connect process.

Similarly, the `FileStreamSink` connector could also be leveraged to perform arbitrary file writes to the file system in its default unauthenticated state.

While this issue affects Kafka Connect, commercial products using such OSS are also affected (e.g. Enterprise Confluent 5.5.0 and below).

Proof-of-Concept

In the default configuration, Kafka Connect allows unauthenticated HTTP requests to two endpoints that are particularly crucial for security:

- `GET /connector-plugins`
  Lists installed connector plugins

- `POST /connectors`
  Instantiate a source or sink. While instantiating a sink, it's also possible to directly connect to a given source, triggering a read.

The specific payloads for the POST request have been omitted.

Impact

Since the `FileStreamSource` and `FileStreamSink` connectors are enabled by default, a standard Kafka Connect instantiation will be susceptible to this vulnerability, unless authentication is enabled.

As a result, an attacker can leverage the default available state of connectors to read data from non-Kafka components (e.g., the file system of the Connect server), that are otherwise not readable or cannot be written to directly. This could be used as a step to further compromise the underlying environment.

Remediation

Option 1: This vulnerability can be remediated by removing any default source and sink connectors that could lead to the arbitrary read/write vulnerability on unauthenticated

Kafka Connect installations.

As a short-term workaround for resolving this vulnerability in Kafka Connect within the Confluent Platform, the `FileStreamSource` and `FileStreamSink` connectors were removed off of the default classpath into a separate FileStream Connectors directory, so that they are not available to be invoked by Connect workers on a default installation.

Apache Kafka releases v3.2.0, v3.1.1, v3.0.2 addresses this issue by removing the inclusion of those two connectors by default in Connect deployments. The classes are still available with the packaged version of the project. However, users now have to take explicit extra steps to add these connectors to the paths that the Connect workers are using to load classes in order for them to be able to use these connectors. Kafka documentation now highlights that the file stream connectors are not meant to be included in production and that these connectors can read and write arbitrary files from the local filesystem.

More details in https://issues.apache.org/jira/browse/KAFKA-13748 and https://github.com/apache/kafka/pull/11908

Option 2: Alternatively, users of Kafka Connect are required to enforce authentication