Diagnosing Minecraft Server Lag Using Spark

Lag can ruin the player experience on your Minecraft server. Whether it’s a consistent delay or sudden spikes, diagnosing the cause is crucial for a smooth gaming environment. **Spark**, a performance profiling plugin, provides powerful tools to identify and resolve lag issues. This guide explains how to use Spark to diagnose lag spikes.

Understanding Lag Spikes

Lag spikes occur when one or a few game ticks take significantly longer than average to execute. They might:

- Happen frequently (e.g., once every 20 ticks).
- Occur rarely (e.g., once every minute).
- Be tied to specific player actions or server events.

Lag spikes can be challenging to diagnose because typical profiling averages out the data, masking the spikes. However, Spark offers tools like `/spark tickmonitor` and `/spark profiler` to pinpoint these issues effectively.

Installing Spark

If you're running PaperMC for **1.21** or later, Spark is **already built in**, and you do not need to install it separately. For all other server types or older versions, you'll need to install Spark manually.

Installing Spark on Non-PaperMC Servers

1. **Download Spark**
   - Visit the Spark Website.
   - Download the latest version of Spark for your server type (e.g., `.jar` file for Paper, Spigot, or Bukkit).
2. **Place Spark in Your Server's Plugins Folder**
   - Upload the downloaded `spark-x.x.x.jar` file to the `/plugins` folder of your server directory.
3. **Restart the Server**
   - Restart your Minecraft server to load the Spark plugin.
4. **Verify the Installation**
   - Once the server is running, use the `/spark` command in the server console or chat to verify the installation.
     - Run:

       ```
       /spark
       ```

       You should see Spark's command list and version details.

Use /spark tickmonitor to Detect Lag Spikes

What Does `/spark tickmonitor` Do?

This command tracks tick execution time and highlights spikes by comparing individual ticks against the server’s average tick duration.

How to Use It

1. **Enable Monitoring**
   Run:

   ```
   /spark tickmonitor
   ```

   By default, Spark will report ticks that take 100% longer than the average.
2. **Set a Custom Threshold**
   You can use an absolute tick duration threshold. For example:

   ```
   /spark tickmonitor --threshold-tick 50
   ```

   This will report any tick exceeding 50 milliseconds.
3. **Observe Gameplay**
   - Wait for a lag spike to occur.
   - Note the in-game effects during the spike and correlate them with the monitoring output in chat.
4. **Adjust Sensitivity if Needed**
   If no spikes are detected, lower the threshold:

   ```
   /spark tickmonitor --threshold-tick 70
   ```

Example

If a WorldEdit action creates a spike, you might see ticks reported with over 1000% increases in duration.

Profile Laggy Ticks Using /spark profiler

What Does `/spark profiler` Do?

The `/spark profiler` command captures detailed performance data. With the `--only-ticks-over` option, you can filter the profile to include only laggy ticks.

How to Use It

1. **Choose a Threshold**
   Use the tick duration from Step 1 as a reference. For example, if laggy ticks exceed 150 milliseconds:

   ```
   /spark profiler --only-ticks-over 150
   ```

2. **Run the Profiler**
   - The profiler will start recording, but only for ticks exceeding the threshold.
   - Let it run long enough to capture lag events.
3. **Stop the Profiler**
   Use:

   ```
   /spark profiler stop
   ```

4. **Inspect the Results**
   Open the profiling report and analyze the data. The laggy ticks will stand out, making it easier to identify the cause.

Example

In the profiler output, you might see a high percentage of time spent on a plugin, mod, or player action. This helps isolate the problem.

Common Causes of Lag Spikes

1. **Heavy Plugin Tasks**
   - Plugins like WorldEdit, if used on large areas, can cause spikes.
   - Identify such plugins in the profiler and adjust their usage or settings.
2. **Inefficient Mods or Data Packs**
   - Look for mods causing high execution times.
   - Update or replace inefficient mods.
3. **Resource Overload**
   - Large numbers of entities or block updates can cause spikes.
   - Use tools like `/kill @e[type=!player]` carefully to manage entities.

Conclusion

Using Spark's tools, you can efficiently diagnose and address lag spikes on your Minecraft server. The combination of `/spark tickmonitor` for detection and `/spark profiler` for detailed analysis ensures you can pinpoint and resolve performance issues quickly.

For further assistance or advanced optimization, consult the Spark documentation.

Additional Help

If you encounter any issues or need further assistance, open a support ticket via the Billing Area. Our support team is ready to assist you.