**Thread Dump**

**FastThread**:This tool is used to analyze thread dump in a graphical manner.It shows cpu consumed threads,dead lock threads and same stack threads

**How to get thread dump**:actuator,jstack and program way

**Jstack**

jps -l

o/p

753 target/ServerApp-1.0-SNAPSHOT.jar

754 jdk.jcmd/sun.tools.jps.Jps

Pid of java is given to get thread dump in text format

jcmd 753 Thread.print > threadDump.txt

we can have jstack browser to view thread dump but we can see only in client side

We can upload thread dump in **FastThread** by uploading threaddump file or copy paste.

**ThreadMxBean :**By using this interface we can write program to get thread dump

**Actuator**

**Yml file property**

management.endpoints.web.exposure.include=threaddump

start springboot application

curl 'http://localhost:8080/actuator/threaddump'

By default thread dump is in json but there are no tools to analyze thread dump for json object so below command is used to get thread dump in text format

curl 'http://localhost:8080/actuator/threaddump' -i -X GET -H 'Accept: text/plain'

**How to handle deadlock threads**

Use synchronization as required only.using too much can also lead to deadlock situation

Use lock priotization

Don’t use unnecessary locks use concurenthashmap instead of synchronized hashmap

Use lock free datastructure if possible

Avoid Nested Locks

**How to handle stackoverflow error**

The simplest solution is to carefully inspect the stack trace and detect the repeating pattern of line numbers

Check whether recursion is implemented properly or not if implemented properly then increase stack size

**Cyclic Relationships between classes** Class A1 is instantiating Class A2, and the constructor of Class A2 is again instantiating Class A1

-Xss to increase stack .Ideal stack size is 1mb