COMPARING MONITORING SOLUTIONS FOR CF AND LUCEE

Charlie Arehart
Independent Consultant/Server Troubleshooter
(CArehart.org, @carehart)
A DAY IN THE LIFE
WHO MIGHT YOU BE?

This Photo by Unknown Author is licensed under CC BY-NC-ND.
WHO AM I?
CHALLENGE: CHOOSING THE RIGHT TOOLS
COMMON PROBLEMS

- High CPU
- Hanging requests
- External calls hanging
- High memory
- DB calls hanging
- CF/Lucee Crashing
- And more
LESS COMMON PROBLEMS

- Problems when no traffic
- End-user time slow, not on server
- Unexpected traffic
- CF/Lucee crashing frequently
- Web server issues
- Web connector issues
- And more
PROBLEMS ON ANY RELATED SERVICES (DB, API CALLS, CACHE, ETC.)

CPU, MEMORY, DISK, NETWORK ISSUES ON BOX BUT NOT DUE TO CF/LUCEE

SPECIAL ISSUES WITH VMs
  - Issues on the VM host
  - Issues with other VMs within host

WHAT ABOUT MANY SERVER, VM’S, CLOUD? …
THE CHALLENGE OF SCALE
CF and Lucee are of course based on Java
  - And both run on Tomcat by default
Some people come to troubleshooting them with Java experience
  - Or find resources on the web proposing such approaches
Let’s consider first some of them…

SOME APPROACH CF/LUCEE TROUBLE WITH JAVA PERSPECTIVE
COMMON JAVA TROUBLESHOOTING APPROACHES

- Stack Traces
- Thread dumps
- Profiling
- Heap dumps
- JMX
- APM
- Logs
JAVA/TOMCAT TROUBLESHOOTING TOOLS

- Built-in JVM tools
- Tomcat Manager
- JVM Profilers
- And more
- 3rd party JVM tools
- JVM/Tomcat Logs
- APMs
CF/LUCEE TROUBLESHOOTING TOOLS

- Request Debug Output
- Lucee Admin Tools
- CF PMT/ Sv Mon
- And more
- CF/Lucee Logs
- CFSTAT, perfmon, getmeta ticdata etc.
- FusionReactor/ SeeFusion
STILL OTHER TROUBLESHOOTING TOOLS
Did you know of all these? Do you turn to them?

What can all these tools do? What can they not? Which should you use?
  - Simply no time to show them all
  - Each could deserve an hour

Focus will be on different monitoring/troubleshooting approaches
  - And which tools can (or cannot) do them
  - To help you decide which you can/should use

SO MANY TOOLS, SO LITTLE TIME
- Both are rudimentary, but better than nothing
- Logs generally don’t help much in a “monitoring” sense
  - Some may help with understanding what happened prior to a crash
- Debug output helpful for understanding a given request
  - Not enabled in production typically, so not helpful for such trouble
Most people naturally gravitate to these when stuff hits the fan:

- Task manager in Windows (or perhaps Process Explorer, other alternatives)
- Top, htop, ps commands in Linux (and other alternatives)
- Activity Monitor in MacOS

These to at least give high-level OS and process-level metrics

- Again better than nothing, and often useful
- Sometimes misleading, especially about memory “used by” CF/Lucee
  - Usually we need to see “inside of” CF/Lucee...
CF Admin offers little in terms of monitoring/troubleshooting
Lucee Admin Overview page shows heap, cpu graphs
  - Also counts of scope use
  - Counts of requests running/queued/threads, dsn connections
Even such basic metrics can be quite valuable
  - But what if you don’t run Lucee?...
CF TOOLS FOR BASIC METRICS

- CFSTAT - cmdline tool in cfusion/bin
- GetMetricData() CFML function (also in Lucee)
- Perfmon – CF’s integration with this Windows tool
- CF Metrics logs
  - CF10>: “metrics log” feature, enabled in CF admin
  - CF9<: jrun metrics, enabled in jrun.xml
The previous tools could at least better clarify IF there is a problem in CF/Lucee
- But they don’t help much to know really WHAT the problem is
- Or WHAT requests are running, for how long, from who, and why slow

For that, we needed better tools
- Preferably with a CF/Lucee focus, and there have been a few over the years, as we will see
- But many again instead fall into using Java tools. Let’s take a look at several

BASIC METRICS ARE THAT, BASIC
Manager available but **not enabled** by default
- And not easily enabled with CF
- Take care to consider documented **security concerns** with enabling it

Monitoring-oriented features:
- Shows server status, can see session counts, memory usage; trigger thread dumps

Tools provided in JDK only (not JRE)
  - Some only as of a certain JVM version (added/removed)
  - Often most easily used if JMX remoting enabled

jvisualvm / visualvm
  - Removed from JDK as of Java 9, now at https://visualvm.github.io

Java Mission Control / jmc
  - Was originally only available to Java licensees, then open sourced (still beta)
  - Contains Flight Recorder, JMX console, hprof dump analyzer
  - https://openjdk.java.net/projects/jmc/

JVM TOOLS
As well as older cmdline tools, in jdk/bin and lucee/jre/bin ...
On Windows, may need to “run as admin” for some to work
Jconsole
  Offers graphical UI about memory, threads, jmb, and more
And others that are “experimental/unsupported”
  Jstack: Obtains stack trace of a jvm thread
  Jmap: Print memory maps or heap memory details
  Jcmd: Lets you send any of dozens of diagnostic commands to JVM
Taking thread dumps
  - Again: VisualVM, Java Mission Control. Tomcat Manager (more ways below)
  - And within most APM tools (as well as FR, SF, CFSM/PMT)
  - jcmd, jstack, kill -3, windows (ctrl+break)

Analyzing them
  - http://fastthread.io/
  - TDA: https://github.com/irockel/tda

THREAD DUMP TOOLS
These monitor time spent within or across threads, by java method

Available in VisualVM and Java Mission Control, as well as:

- Some APMs (as well as FR, SF, CFSM/PMT)
- Yourkit: https://www.yourkit.com/
- Jprofiler: https://www.ej-technologies.com/
- Xrebel: https://xrebel.com/
- Netbeans: https://profiler.netbeans.org
Taking heap dumps
- VisualVM, Java Mission Control, Tomcat Manager
- jcmd, jmap, jmx, HeapDumpOnOutOfMemoryError
- And most APM tools (including FR, PMT)

Analyzing them
- Eclipse memory analyzer tool (https://eclipse.org/mat/)
- Jhat (removed since Java 8)
- https://heaphero.io
- https://go.nastel.com/autopilot-heap-detective
Forcing GCs: jvisualVM, jcmd, most APM tools (and FR, SF, CFSM/PMT)

Obtaining GC logs: -XX:+PrintGCDetails and related args

Tools to analyze GC logs

- http://gceasy.io/
- https://gcplot.com
- And others
jmx: java mgt extensions
  - Many java apps, including Tomcat, expose metrics via jmx
Tools to view them include: jvisualVM, Tomcat Manager
jmc, jconsole
Some APM tools (including FR)
https://jolokia.org/
NOW ON TO CF/LUCEE MONITORS
Came with CF8-2016, Enterprise only
- Launched from CF Admin (Monitoring menu)

Many ignored, dismissed, or reviled it
- But it did at least offer several basic metrics “for free” (no overhead)
- Or could monitor still more, with 3 “Start” buttons offered there
- Let’s take a look

Beware: “start memory tracking” could be dangerous, “profiling” less so
- Was also a Flash interface, which was a negative
- Replaced in CF2018 with …
New monitoring solution, available in CF2018, Std and Enterprise (and Dev)
  - Tool offers many, many monitoring features
  - See Adobe resources for more on all these, but let’s take quick look
Runs as service separate from CF, with datastore (elasticache) also separate
  - Most should consider running these on box apart from CF box
  - And should consider increasing their heap size
Beware also that there have been updates to the PMT
  - Must be applied manually. See my blog post on this
Previous two monitors work only in CF, next two work with CF or Lucee
  - FusionReactor, SeeFusion have been serving folks well for over a decade
  - FR is from Intergral, here in Germany, often a sponsor
    - And I’ve presented many times on it over the years
Both can show most of what we have seen so far, and more
  - And yes, like CFSM/PMT, both FR/SF are safe for/meant for use in production
Both products are commercial. See web sites for more
No time to fully demo FR or SF. Dom covered FR in his talk
  - See ~20 recorded FR webinars at fusion-reactor.com/webinars
But let’s take a quick look
We saw there are many things that FR does that SF and CFSM/PMT do not
  - There are also some things that each other does which FR does not
I’d love to do a comparison of all 4 tools, but that could be an entire talk!
  - I may do a blog post or series on that

FUSIONREACTOR & SEEFSION (CONT.)
Lucee folks will ask plainly, how does FR compare to SF?
- They do have some overlap
- They do differ in pricing models (again, see their sites)
  - Note that FR does offer a Developer edition at much lower cost than Std edition
  - But put plainly, FR does offer many advantages...

**BUT WHICH SHOULD I CHOOSE?**
FR can also be used with any Java application/server
Offers error history, requests by memory/by app/by status code, and more
Tracks calls to external services (and DBs without need of wrapping)
Offers profiling of individual slow requests, automatically
Offers heap analysis, tracks GC’s, memory spaces, class loading, JMX, more
Can track thread CPU time, profile CPU across all threads on demand, etc.
Offers feature to handle ephemeral instances, automatically monitor them
Offers FR cloud, which adds even more capabilities

FR ADVANTAGES OVER SF
WHAT ABOUT APM’S?
OF COURSE, MANY AVAILABLE JAVA APMS

- Appdynamics
- Dynatrace
- NewRelic
- Datadog
- Stackify (prefix and retrace)
- Java Melody
- Plumbr
- Perfino
- Sematext
- Moskito.org
- Hawkular.org
- And FR and others
Again, no time to demo them

Beware many have a Java focus, but may still be useful with CF/Lucee
  - Some folks even run such APMs alongside also FR/SF/CFSM/PMT

But many only know of APMs, miss out on the CF focus of the others

That said, some do offer to track transactions across multiple services

Also, some apm’s actually offer more than “java apm” features
  - To include options for system monitoring, db monitoring, etc.
Some APM's also track queries within requests, like FR/SF/CFSM/PMT do
  - And may tracking external server call time (like FR, PMT)
Some APM’s offer alerting on trouble (like FR/SF/CFSM/PMT)
Some monitor from central repository off-server (like FR Cloud, PMT)
Some APM's offer daily/weekly/monthly reports (like FR)
Some APM's track sessions build up, perhaps due to spiders/bots (like FR, CFSM, PMT)
  - But Java ones will track only if JEE sessions are used (in CF, Lucee)
Some track end user response time (network time/browser render time)
  - May offer js code to add to your app (like FR)
Some offer error tracking (like FR, CFSM, PMT)
  - More on error tracking in a moment
- Interactive step debugging
  - Possible with CF using CFBuilder
  - Possible with CF and Lucee using FR and older FusionDebug
- Code coverage
  - Possible with CF and Lucee using FR, also commandbox extension
- Error tracking
  - FR offers a powerful new event snapshot feature
  - And there are still other error handling tools ….
ERROR TRACKING TOOLS

- CFML-based, open source:
  - BugLogHQ
  - Hoth
  - Irongate

- Generic, free/commercial
  - Airbrake
  - Bugsnag
  - Raygun
  - Sentry

Charlie Arehart (CArehart.org, @carehart)
Finally, important also to monitor other parts of environment

- Database/SQL Monitoring Tools
- Event Log Monitoring
- SAN or NAS Monitoring Tools
- System Monitoring Tools
- VM/Virtualization Monitoring/Mgt Tools
- Web Server Analytics Tools
- Web Server Request Monitoring Tools
- Web Site Uptime Monitoring Tools
We've seen many problems, approaches, tools. Some traditional for Java:

- Heap dumps, thread dumps GC logs. profiling, jmx
- But often need to see inside CF/CFML processing, which they/APMs don’t
  - For CF/Lucee, tools like CFSM/PMT/FR/SF are uniquely suited
  - Some better than others
- Even most basic tools (task mgr/top, cfstat/perfmon, etc.) better than nothing
- Get and use diagnostics. Don’t just “restart”! 😊
- If I can answer any questions about my talk, please contact me:
  - charlie@carehart.org
  - @carehart (twitter, linkedin, github, facebook, etc.)