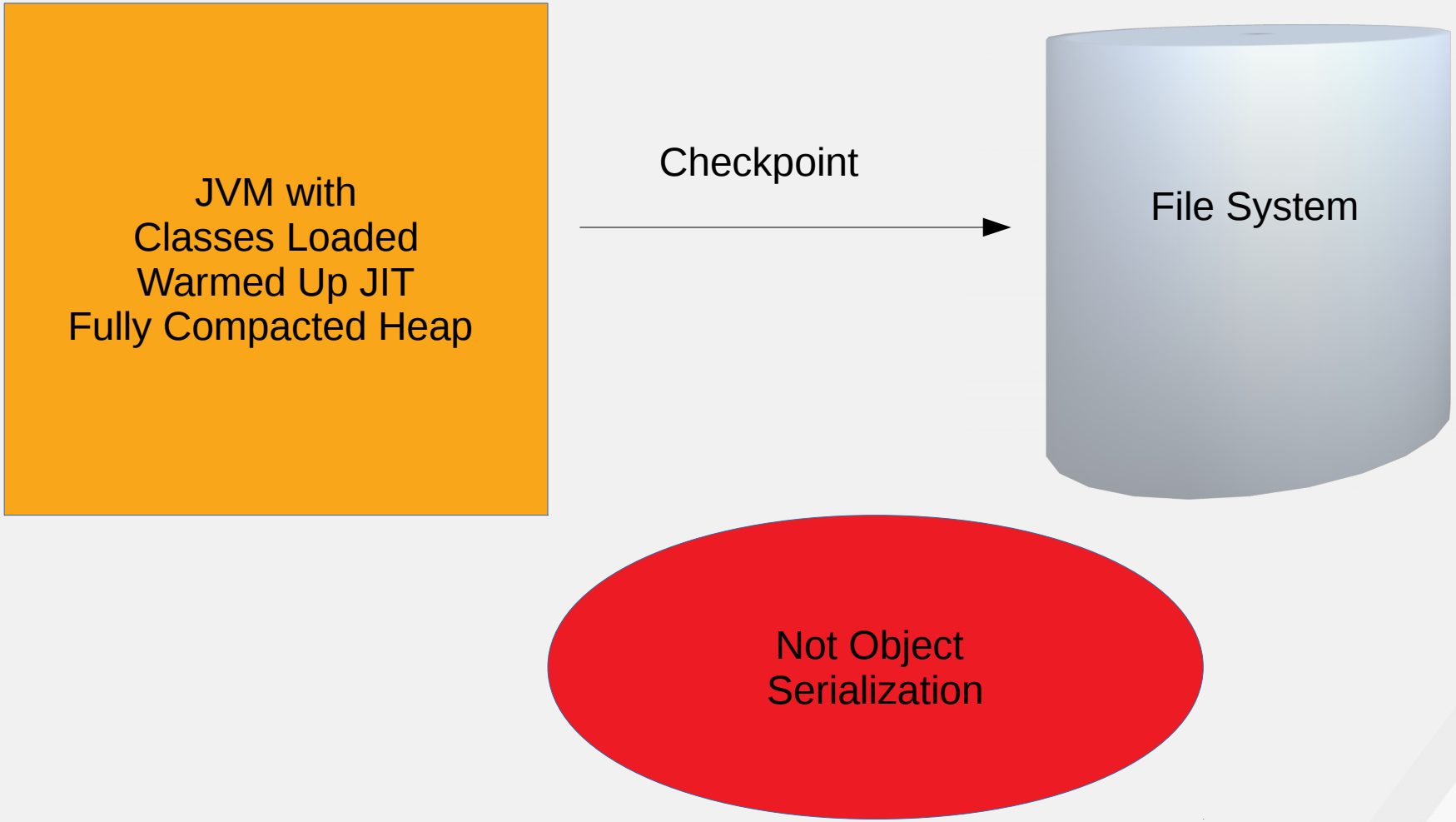
redhat.

Checkpoint Restore

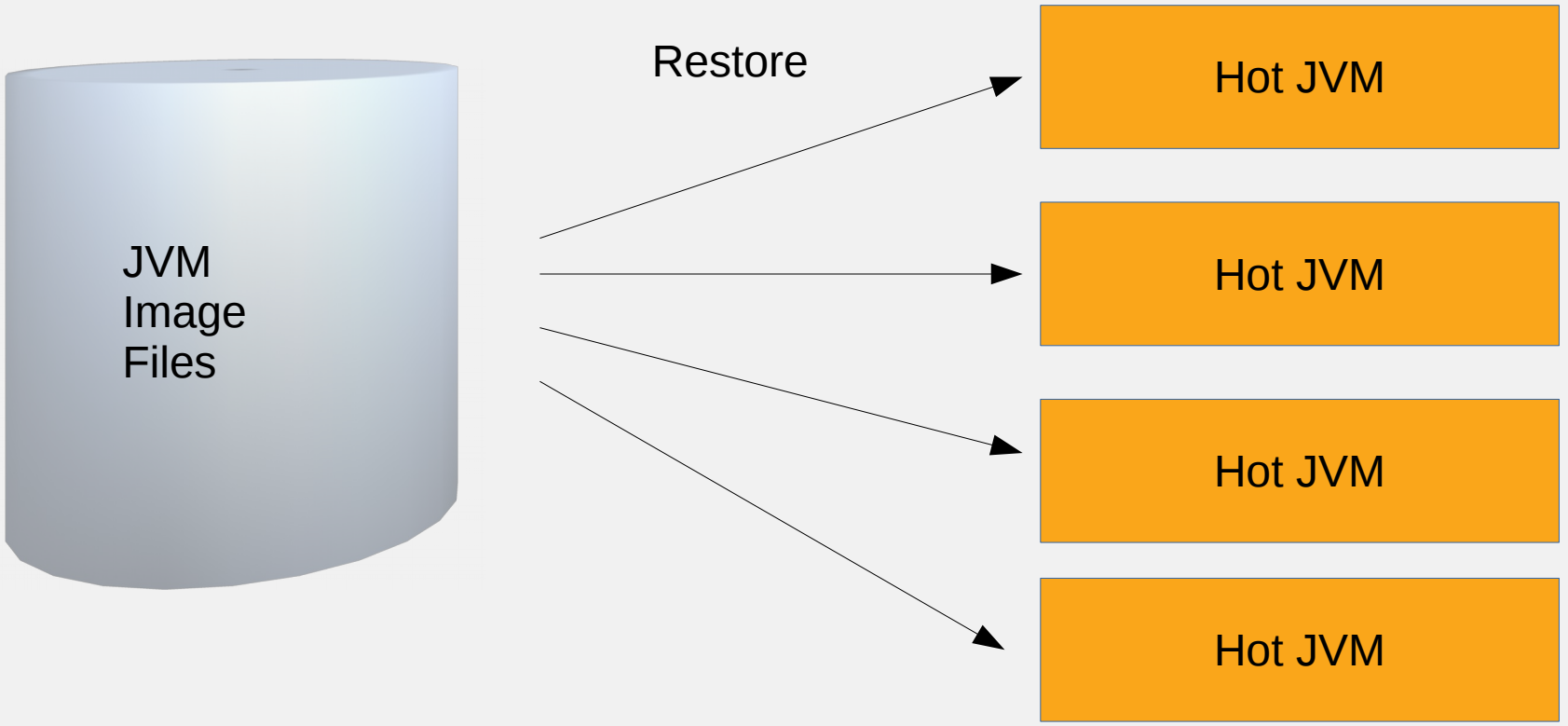
Fast Start-up For Java Applications

Christine H. Flood

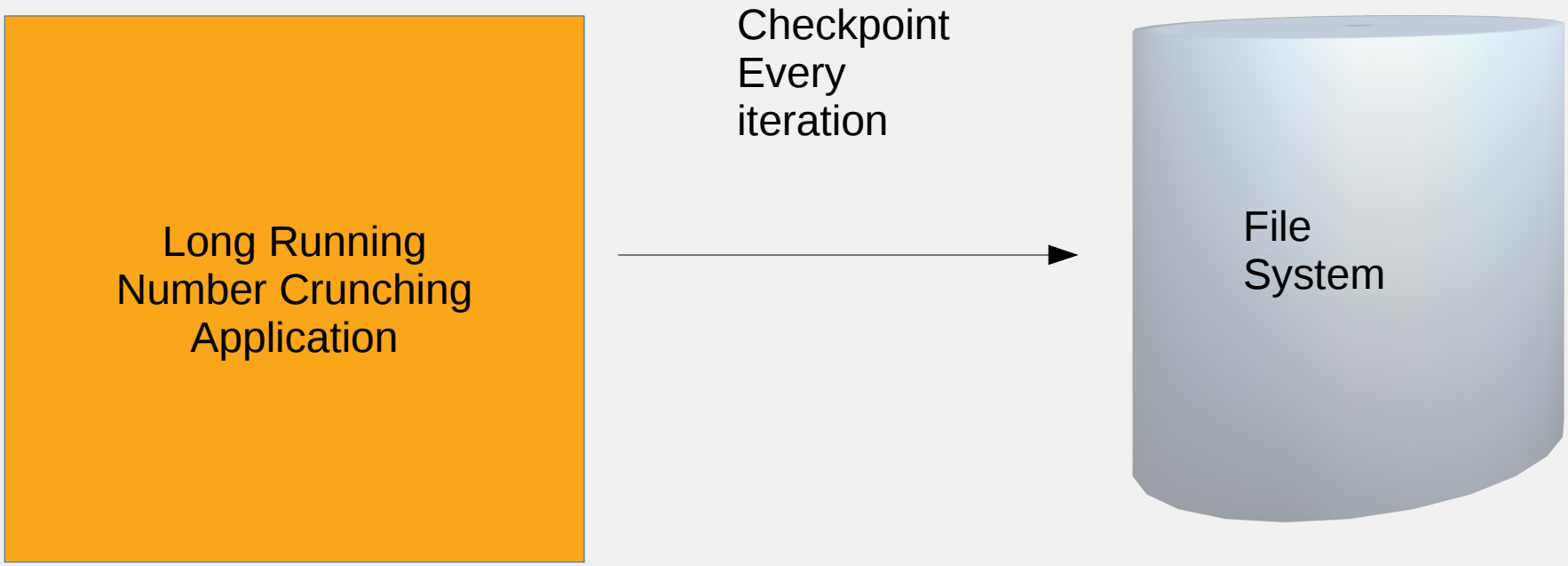
Checkpoint


JVM with
Classes Loaded
Warmed Up JIT
Fully Compacted Heap
Checkpoint
File System
Not Object
Serialization

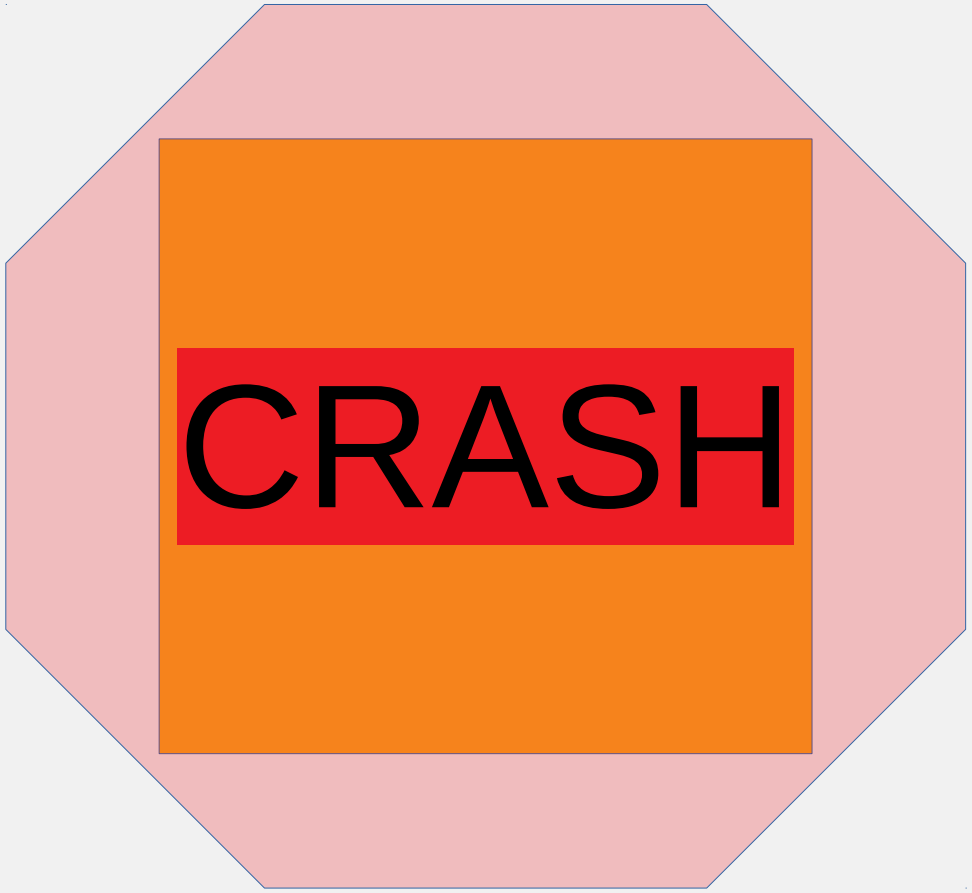
Restore



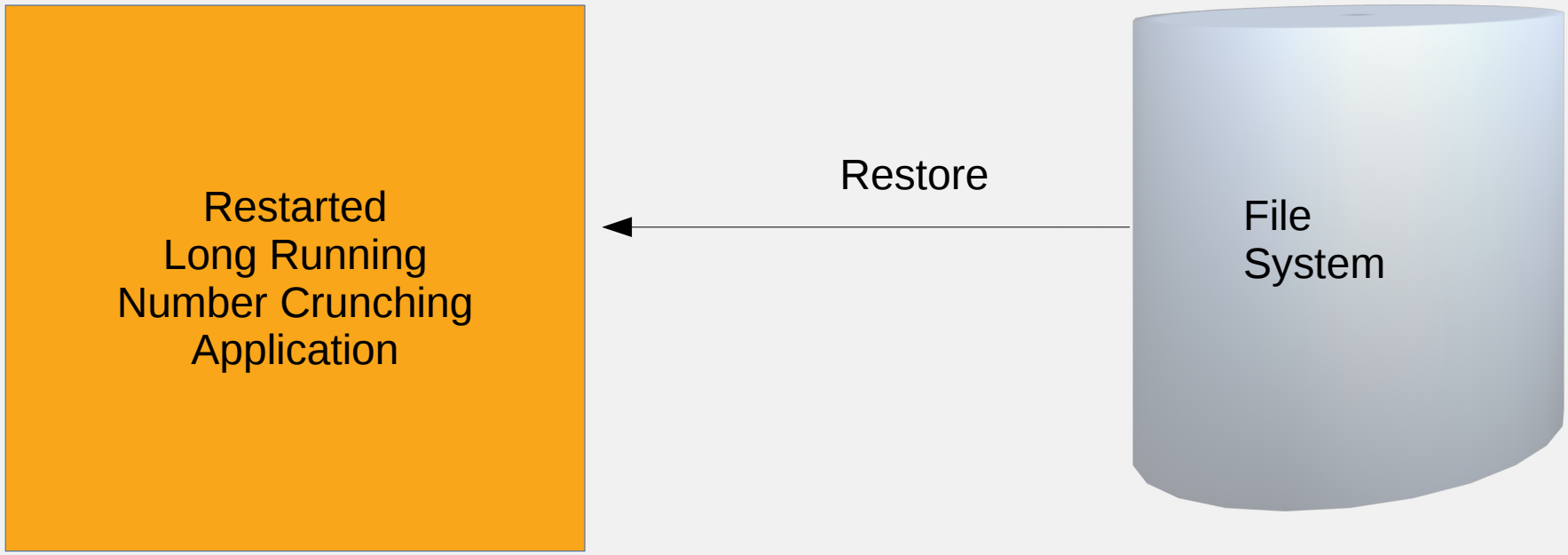
Another Use Case



Ruh Roh Raggy



Restore


Restarted
Long Running
Number Crunching
Application
Restore
File
System

Two More Use Cases

Buggy program that takes hours to reach error condition.

Time sensitive program where you want heap dumps.

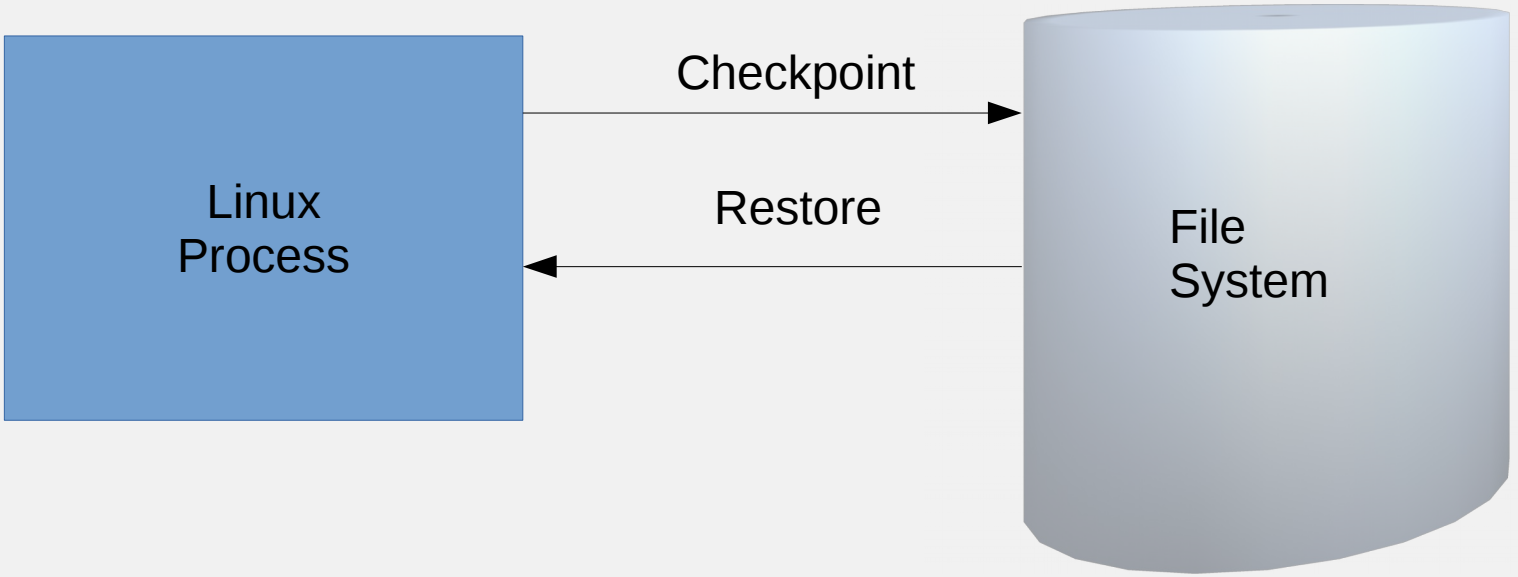
Two More Use Cases

Buggy program that takes hours to reach error condition.

Time sensitive program where you want heap dumps.

I’m not proposing anything heroic.

CRIU already does the heavy lifting



CRIU

- What is it?
  - Checkpoint Restore In User Space
    - Linux utility which copies entire process state into files.
    - Files can be quickly restored on multiple different hosts.
    - Handles open files/sockets/almost all the gory details.
- https://criu.org/Main_Page
- Adrian Reber

CRIU Use Cases

- Process Migration
- Quick Process spin up
- Container Migration

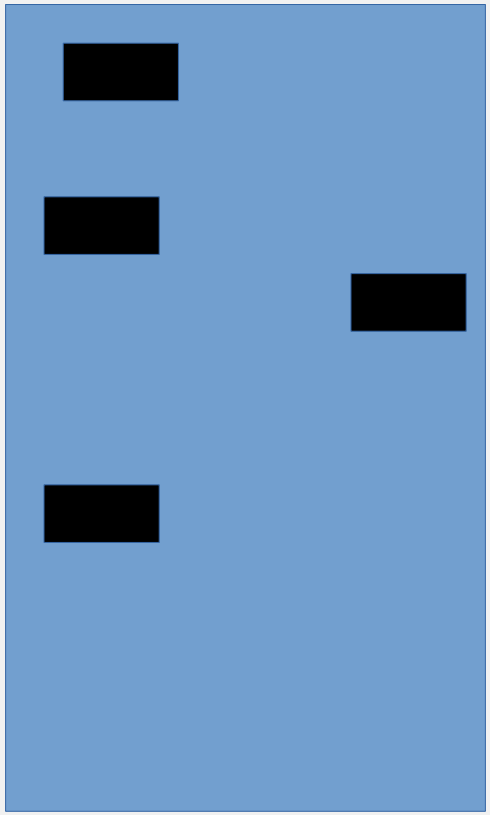
Inspiration: Lisp Machines

- Save World Command:
  - Save World (Complete or Incremental) pathname
  - Saves the current world. The system prompts for (Complete or Incremental) if Incremental Disk Save is enabled. Specify Complete to save the entire world, or Incremental(if enabled) to perform an Incremental Disk Save. The default is Complete.

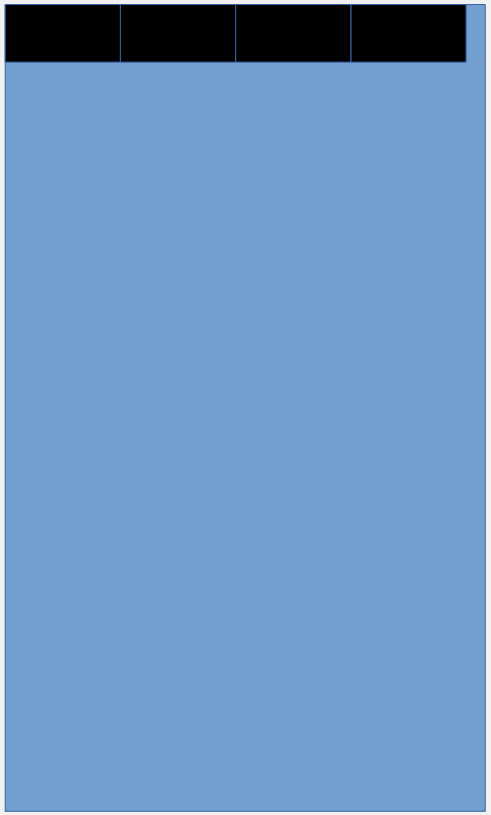
Some Java specific things we can do.

Fully Compact Heap and Release Unused Memory

64GB sparse heap

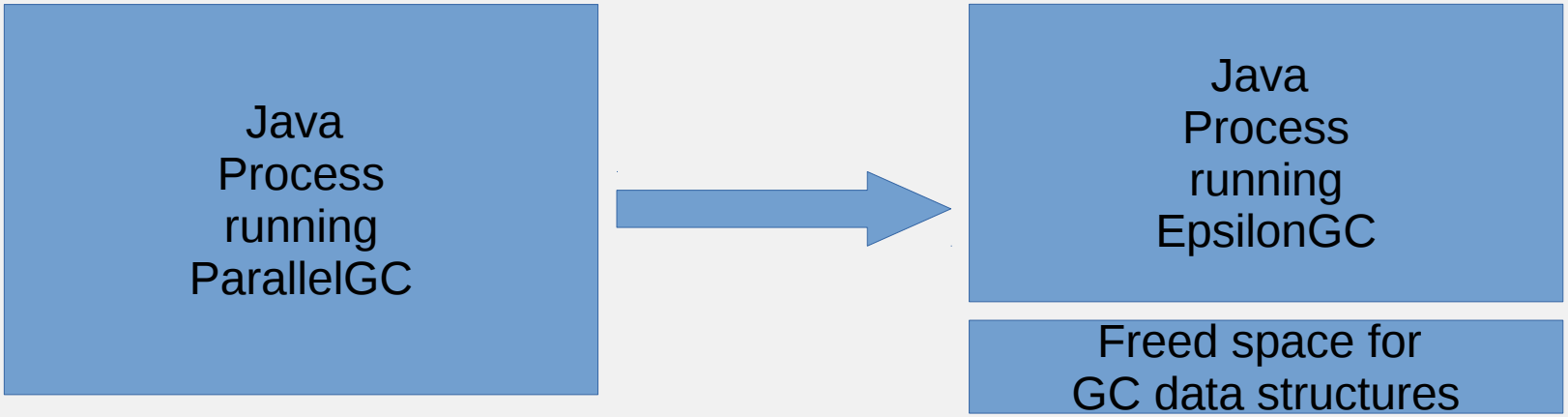


2GB dense heap



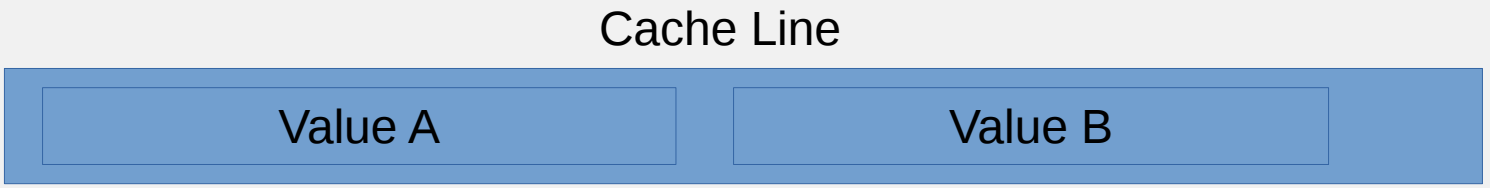
2GB heap with memory given back to the OS.

Hot Swap GC algorithms

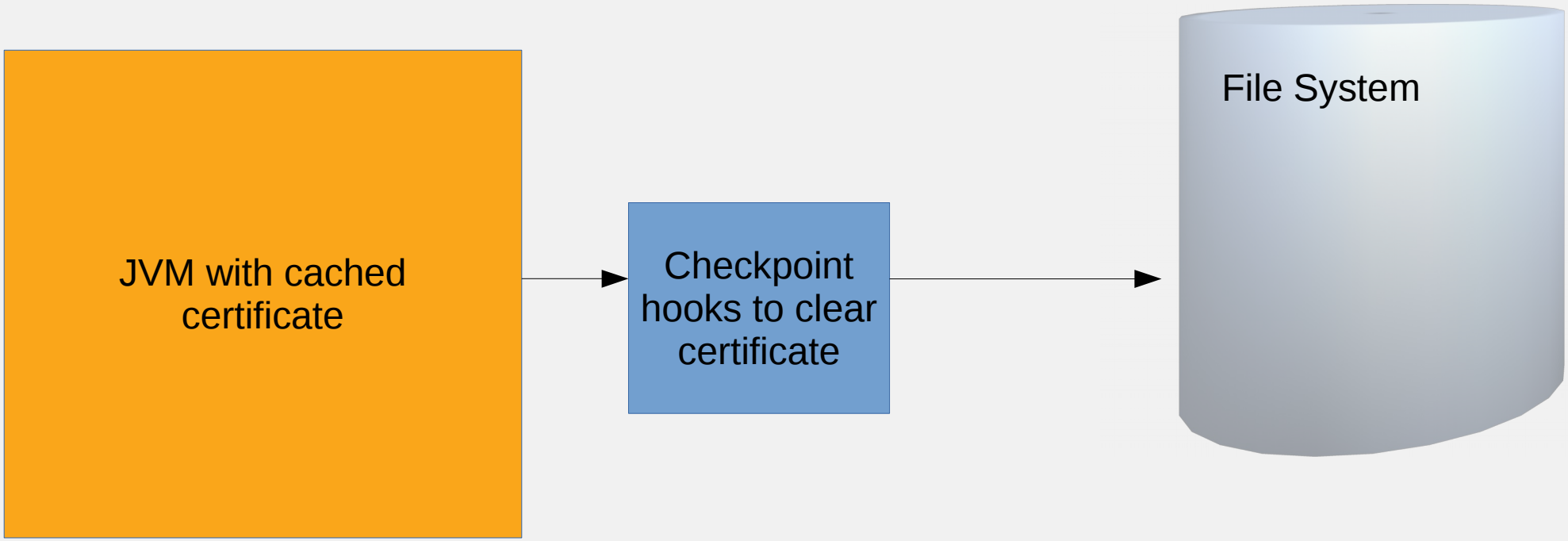


Optimize Heap Layout

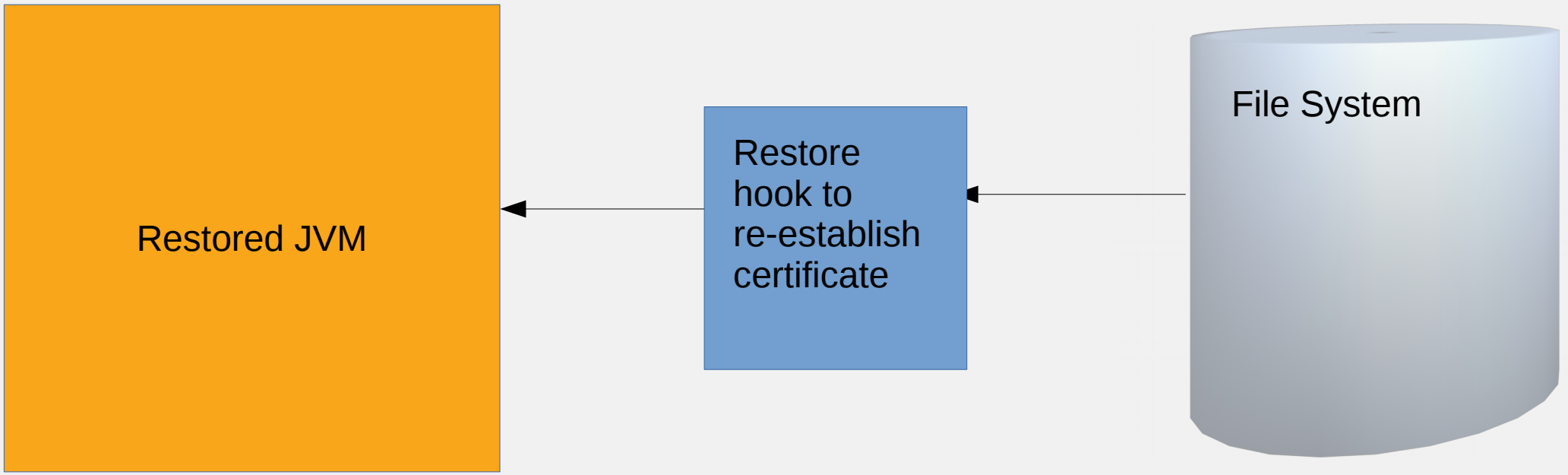
- Inspired by Remix: Online Detection and Repair of Cache Contention For the JVM Eizenberg, Hu, Pokam, and Devietti
- Monitor Hardware performance counters to detect things like false cache line sharing and then pad the data layout.



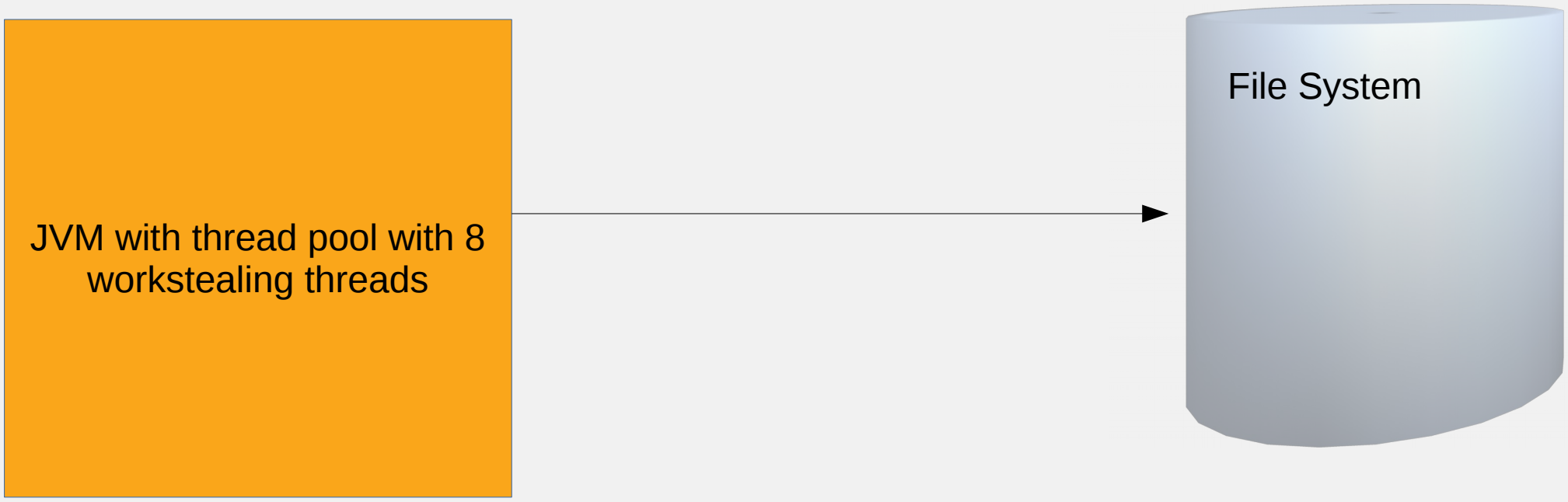
Provide Hooks to the Java Programmer



Provide Hooks to the Java Programmer



Fix Things Like Number of Processors



Fix Things Like Number of Processors

Specify Where Checkpointing Occurs

- Places where there's a small reasonable consistent state

Enough Motivation

How about an API?

Proposed Strawman API for Java

- Class World {
  - bool Check()
  - Runs criu check to ensure the proper kernel patches are there.
  - Runs check of JVM internal data structures to ensure they are all migratable.
  - bool Save()
  - Runs criu dump. Does not implicitly include OptimizeTheWorld in case users want an exact duplicate.
  - bool SaveIncremental()
  - Saves only those memory pages that have changed since the last dump.
  - bool SaveWithEpsilonGC()
    - GC Twice. Get rid of internal GC data structures (card table). Remove Barriers. Fire Up Epsilon GC.
  - …
-

Proposed Strawman API for Java

- Optimize()
- Runs Two Back to Back Full GCS to remove all floating garbage.
- Optimize Memory Layout
- Restore()
- Migrate(...)
  - Copies the running process to another host.
- AddCheckpointHook(...)
- AddRestoreHook(...)
- }

Current Status

Prototype:

- Uses JNI
- Implements CheckTheWorld
- SaveTheWorld
- Command line RestoreTheWorld.

Showing CRIU Works on Java

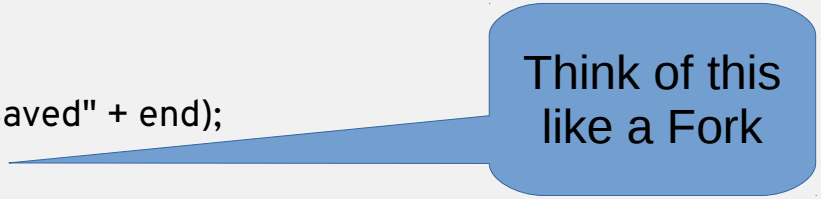
- setsid java TestRandom 10000 100000000000 < /dev/null &> test.log &
- ps -aux | grep java
- 13874 … java TestRandom 10000 100000000000
- sudo criu dump -t 13874 -o dump.log
- ps -aux | grep java
- …
- sudo criu restore -d -vvv -o restore.log && echo ok
- ps -aux | grep java
- chf 13874 … java TestRandom 10000 100000000000

Can restore multiple instances.

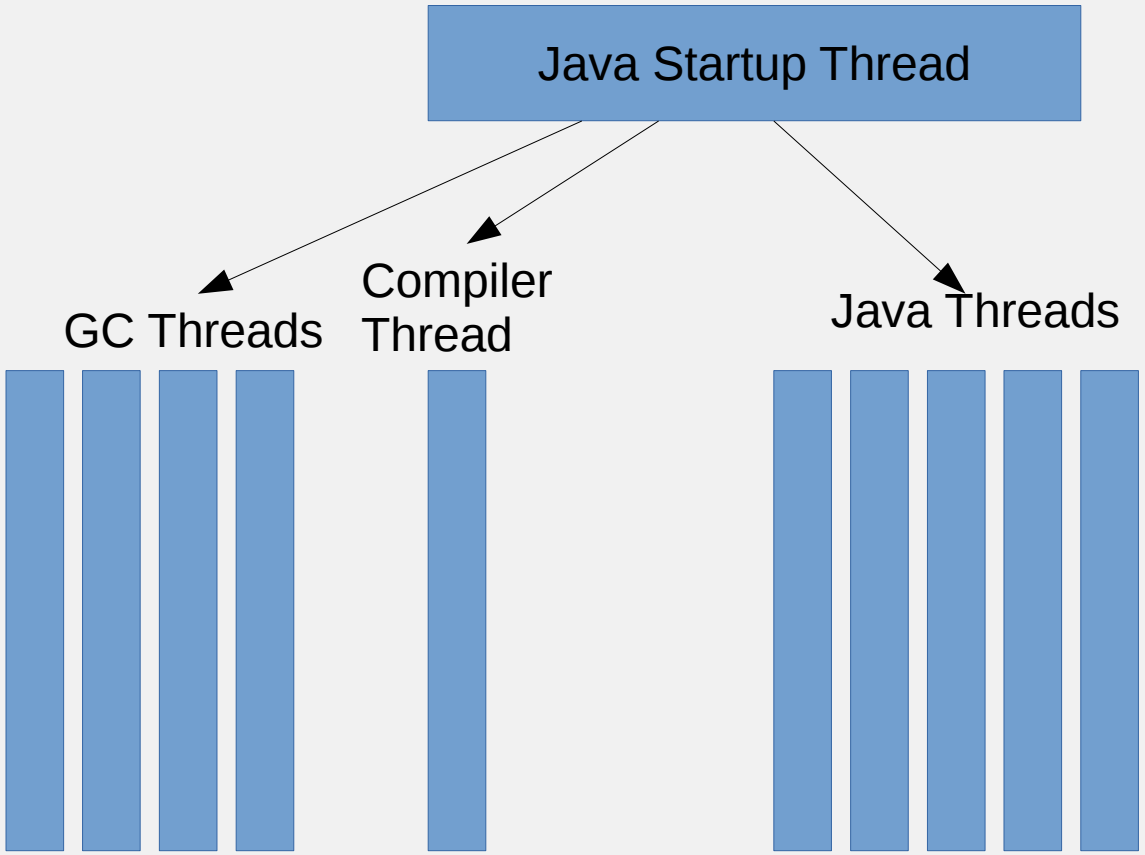
- sudo criu dump -t 13471 -o dump0a.log
- sudo criu restore -o restore0a.log
- <wait>
- sudo criu restore -o restore0a1.log

Prototype calling from Java

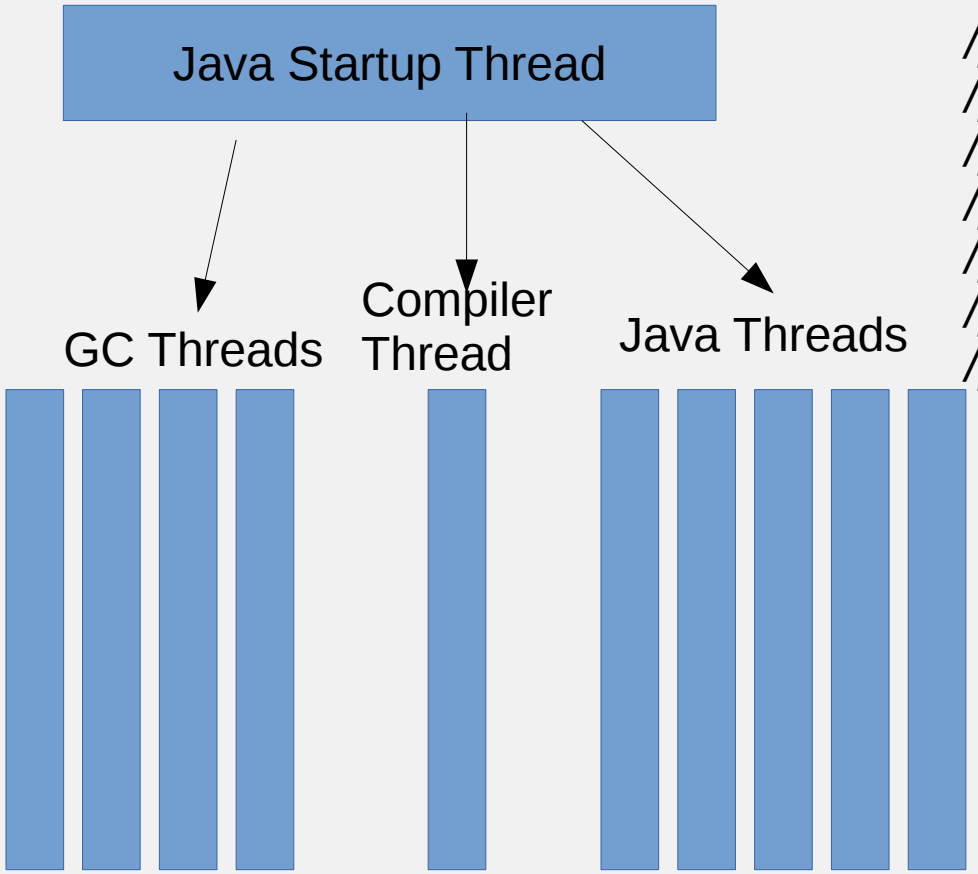
```
CheckpointRestore.CheckTheWorld();
CheckpointRestore.SaveTheWorld("/home/cflood/Velma");
long start = System.currentTimeMillis();

for (int step = 0; step < steps/2; step++) {
   doStep(MEG);
}

long end = System.currentTimeMillis();
System.out.println("Saving the world in directory /tmp/Saved" + end);
CheckpointRestore.SaveTheWorld("/tmp/Saved" + end);

long ms = ((end - start)) + 1;
checkTrees();
System.out.println("End Shaggy");
```



Java Process

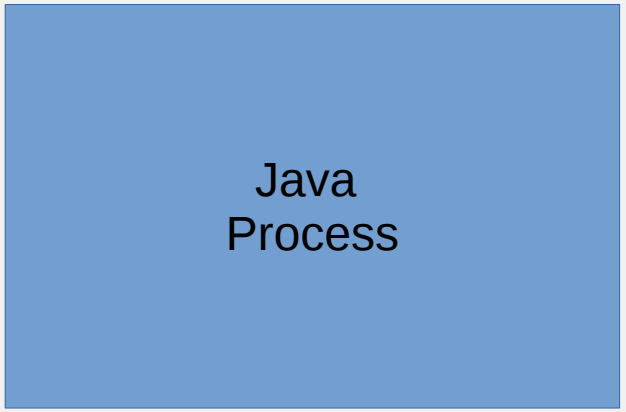


Java Process mirrored in /proc

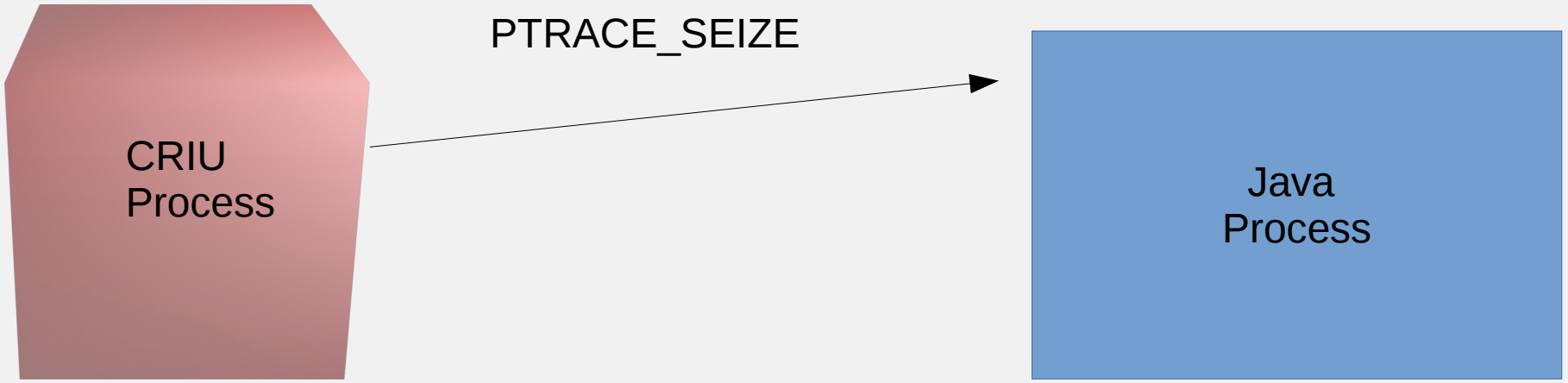


/proc/<java_pid>
/proc/<java_pid>/tasks/<gc_pid1>
/proc/<java_pid>/tasks/<gc_pid2>
/proc/<java_pid>/tasks/<gc_pid3>
/proc/<java_pid>/tasks/<gc_pid4>
/proc/<java_pid>/tasks/<compiler_pid1>
/proc/<java_pid>/tasks/<java_child_pid1>

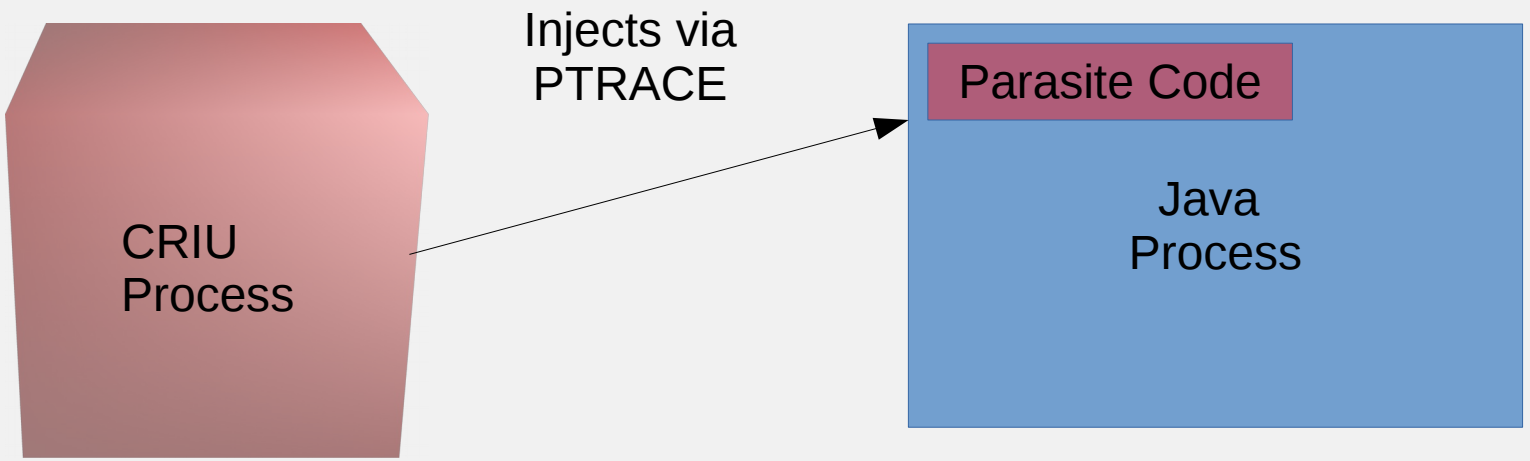
Simplified Java Process



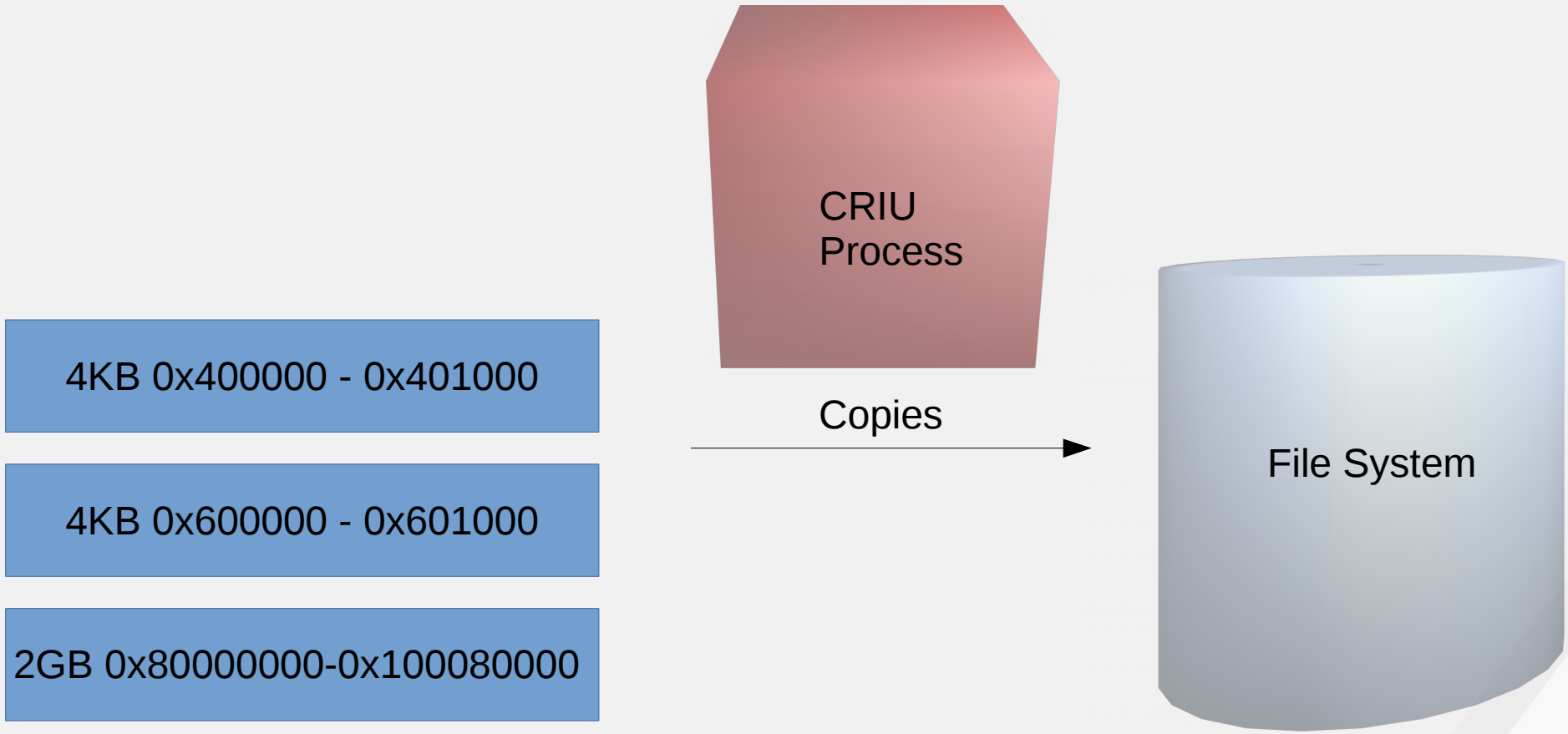
CRIU process



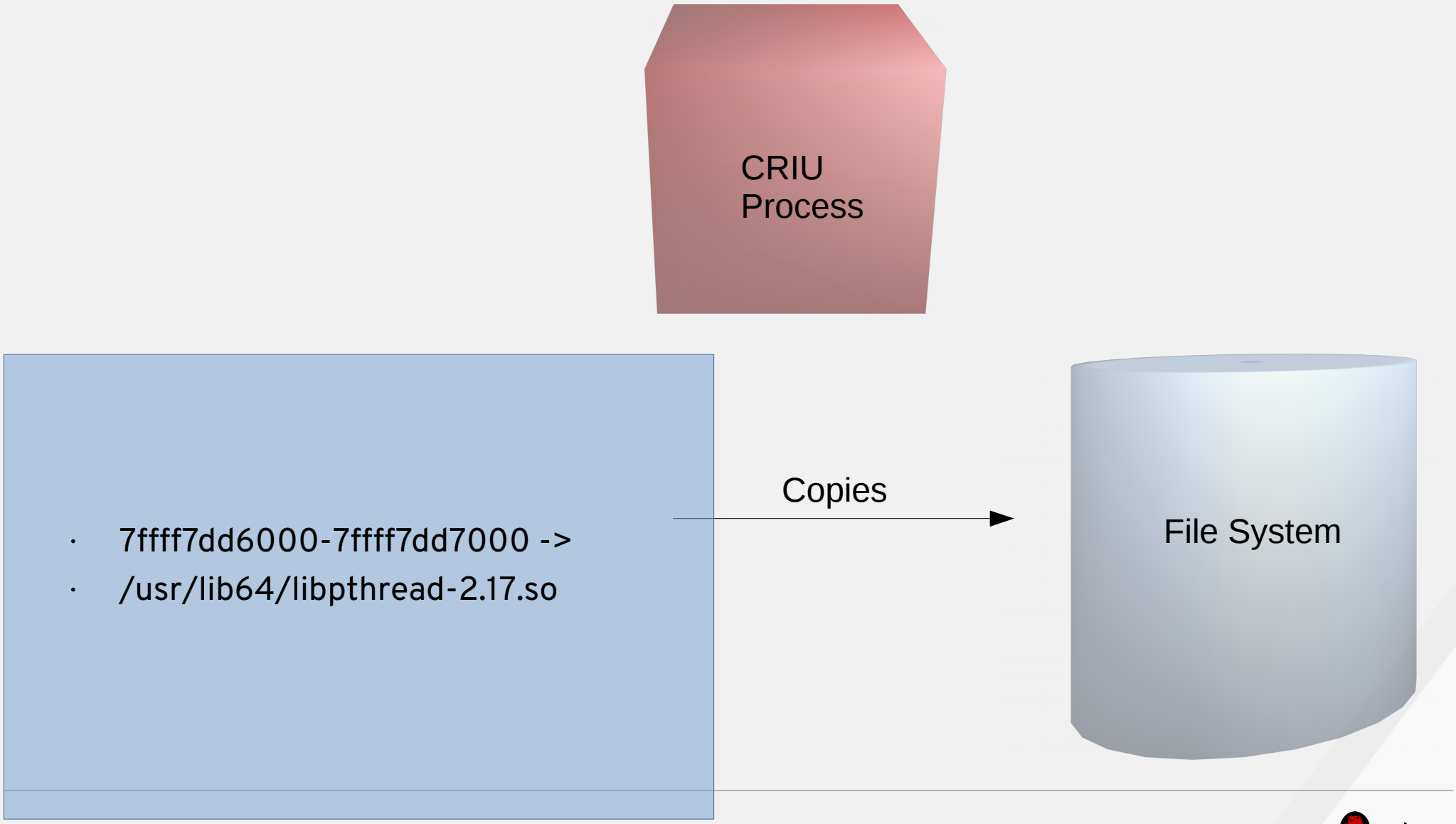
Parasite Code



Parasite code copies Virtual Memory specified in */proc/<java_pid>smaps*



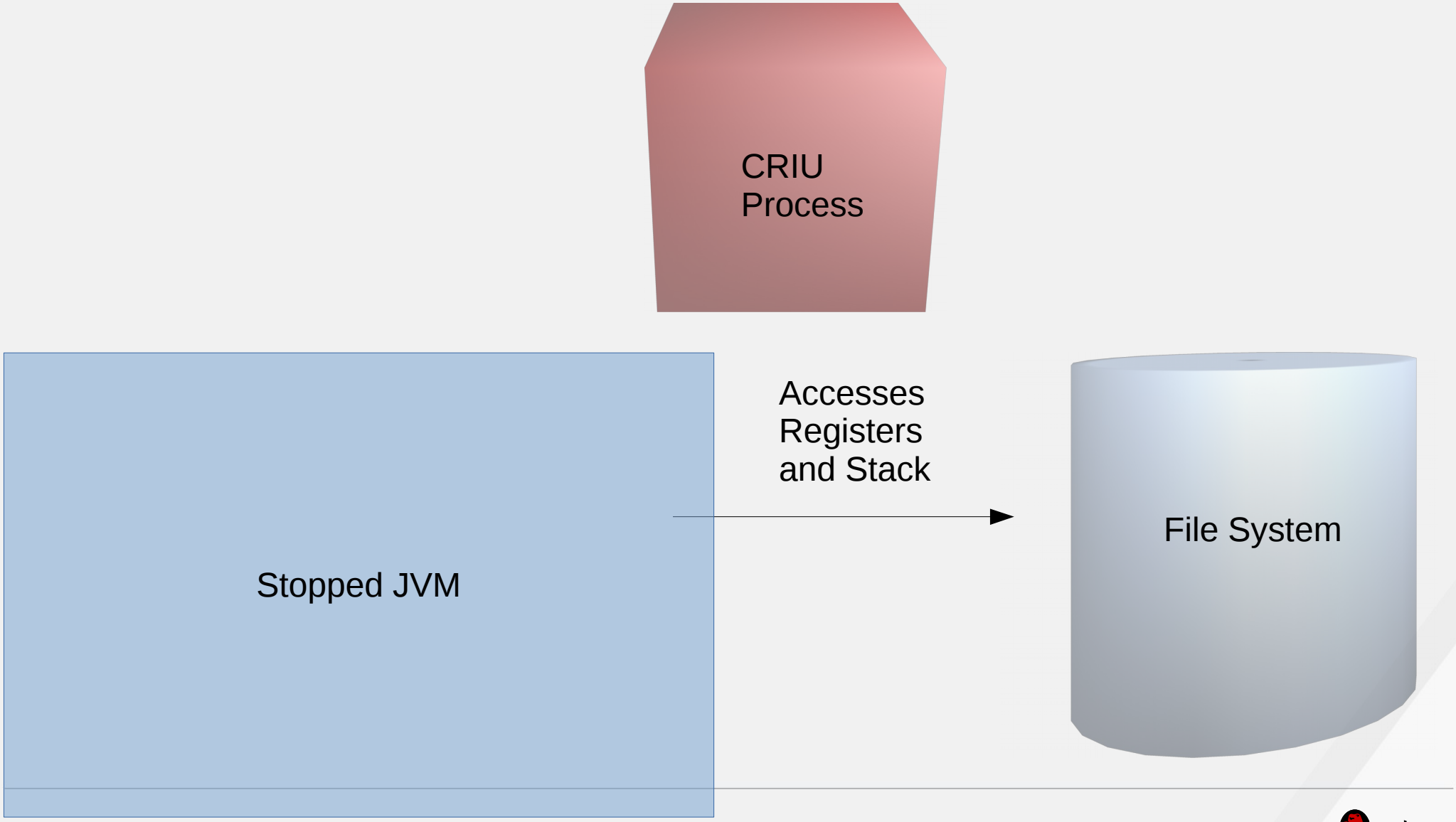
Parasite code copies Mapped Files from */proc/<java_pid>map_files*



And So On

- File Descriptor Numbers from /proc/java_pid/fd
- Core Parameters from /proc/java_pid/stat

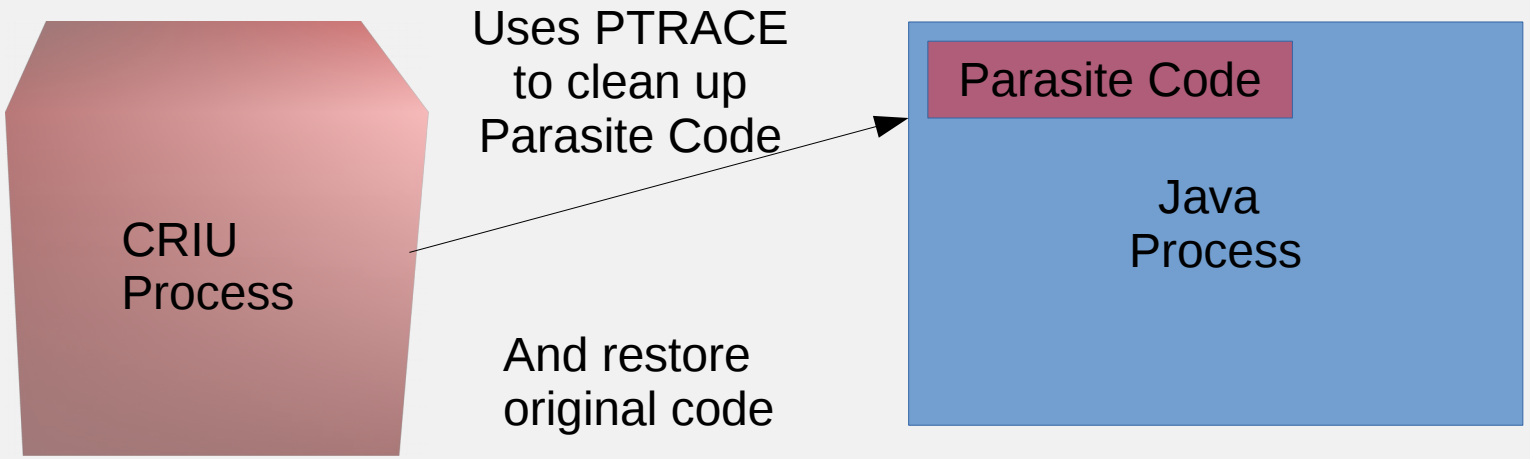
PTRACE_PEEKUSER



Parasite Code

- Parasite Code reads
  - Reads Credentials
  - Reads Contents of Memory

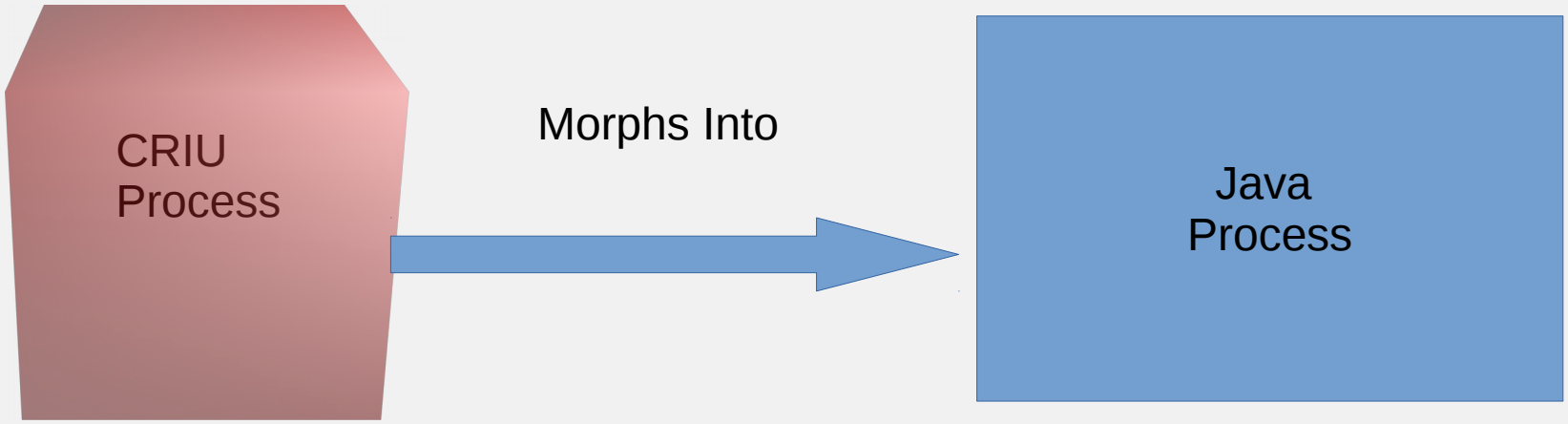
Cleanup


CRIU
Process
Uses PTRACE
to clean up
Parasite Code
And restore
original code
Parasite Code
Java
Process

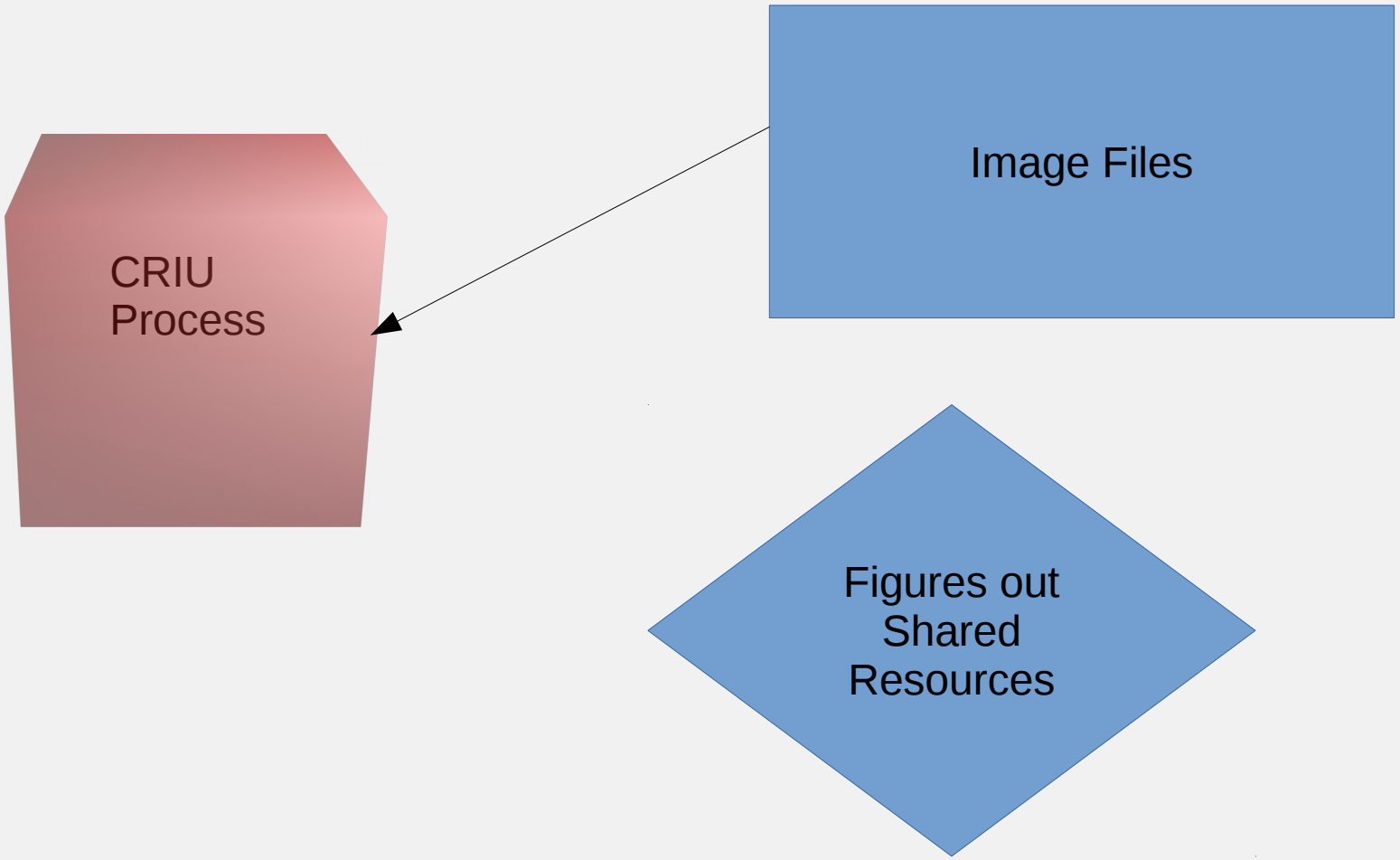
Cleanup

Detach

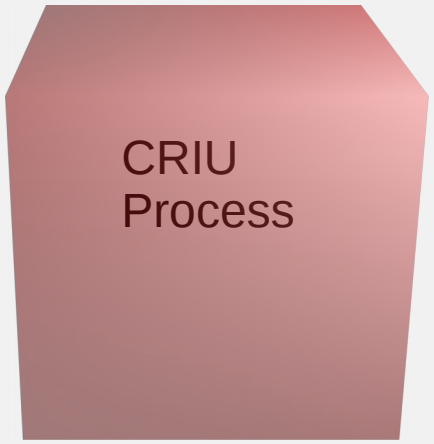
Restore

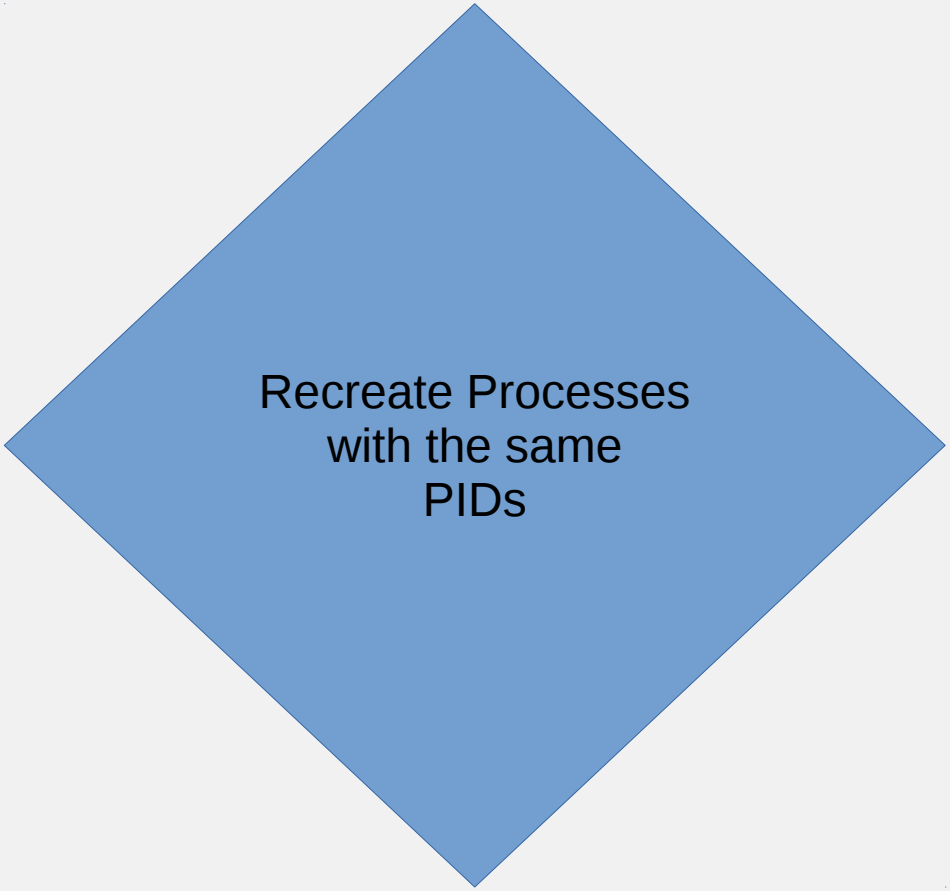


Resolve Shared Resources

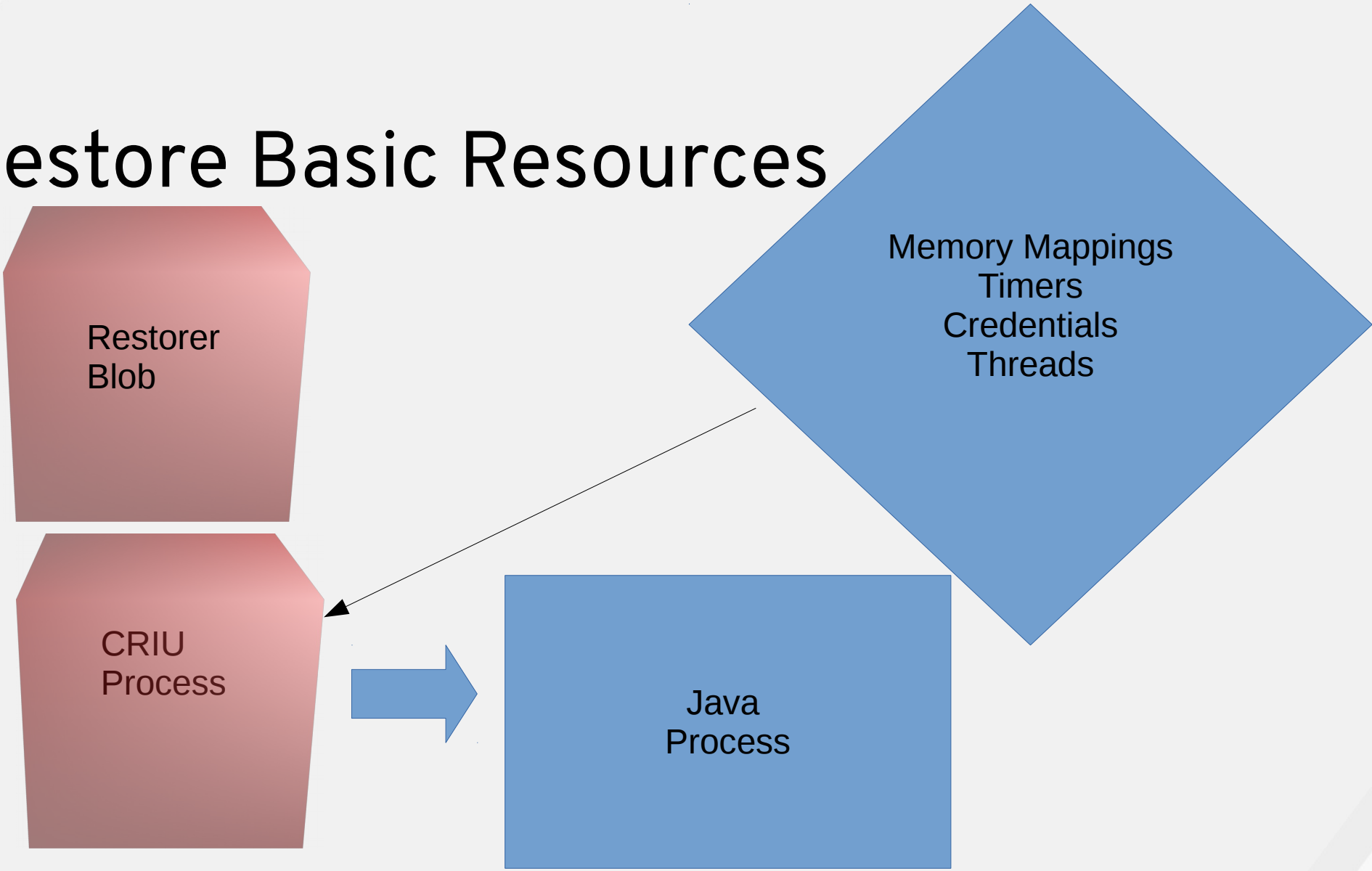


Fork The Process Tree





Restore Basic Resources



Gotchas

- Cached data must be dealt with.
  - Number of processors.
  - Size of Heap
  - Certificates
- Open Files must be present on restore host .
- Processes must have the same pids on checkpoint and restore
  - pid manager?

Contact Info

- chf@redhat.com