Mandrake Coordinator: Resource Management for Edge Clouds

Masters Presentation by Kyle Carson

Advisors: Chandra Krintz, Rich Wolski
Co-contributor: John Thomason
UCSB RACELab
03/12/19

Background & Motivation

- The usage of Internet of Things (IoT) devices is rapidly increasing
- Deployments of such devices need computational infrastructure to support them
- Traditional “cloud computing” infrastructure can be unsuitable due to connectivity, latency, and data transfer constraints
- Need a new infrastructure paradigm, “edge clouds” offer a solution to these issues

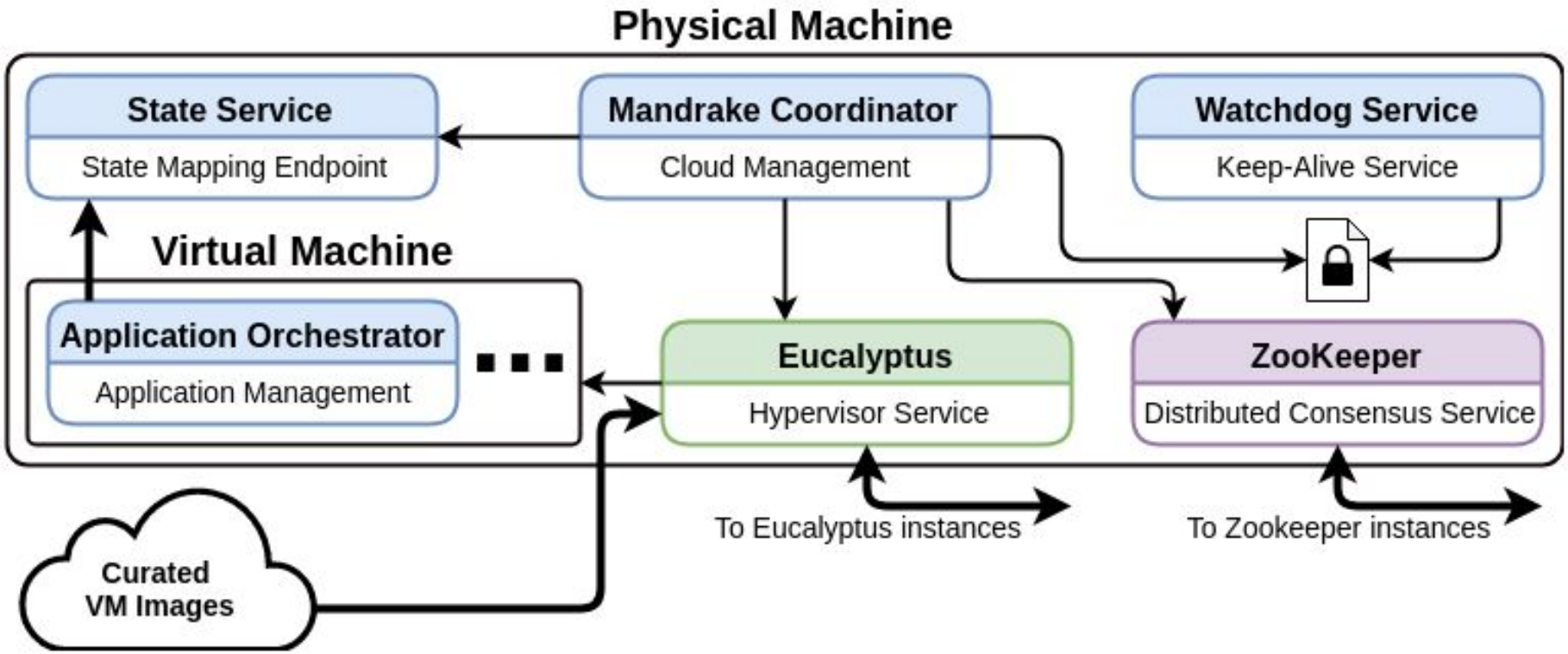
Edge Clouds

- Edge clouds are cloud computing infrastructure located at the "network edge"
- Enable real-time data processing with low latency response times by co-locating the infrastructure with the devices
- Additional issues including:
  - Hardware Failure
  - Network Partitioning
  - Delayed human maintenance

Mandrake

- Mandrake is a software system which provides automated infrastructure & application management
- Reprovisons virtual machines (VMs) in response to failures and recoveries
- Informs VM-level software services of the VM-to-Host mapping, breaking the traditional cloud model of abstraction
- VM-level services manage user application, enabling it to make more informed decisions

Architecture



Mandrake Coordinator

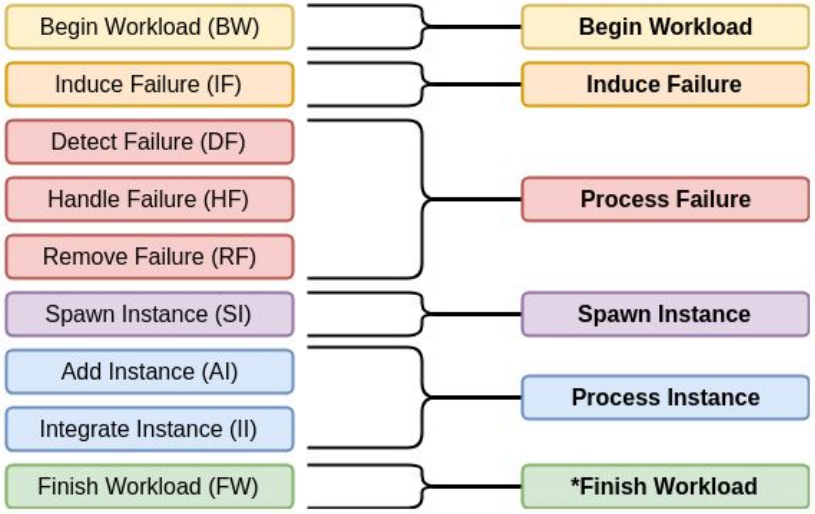
- The Mandrake Coordinator (MC) discovers information about the hosts/VMs and keeps track of their mapping
- Exposes this map through a set of APIs
- A user-defined set of configuration controls the number of VMs, the way in which they scale, the images to use, etc.
- Controls the allocation/deallocation of VMs in response to failures and recoveries
- Makes guarantees about the liveliness of the system, ensures resource allocation mechanisms will provide more reliability

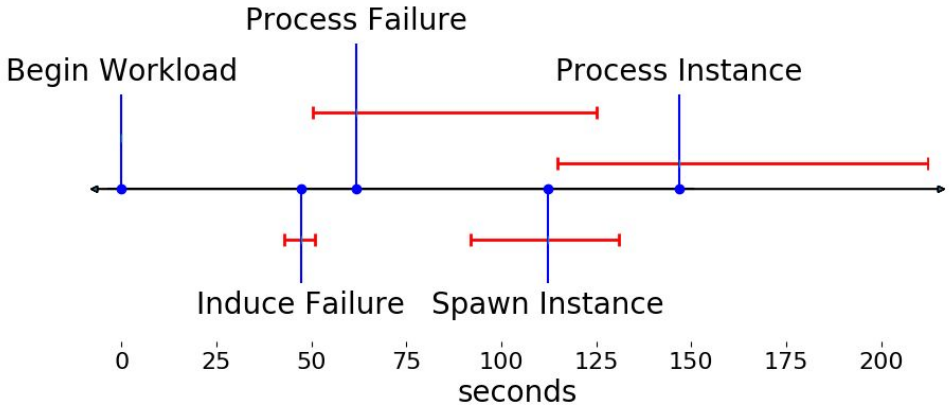
Experiment Setup

- 9 physical hosts with a target of 8 VMs, each with 2 CPU cores, 4GB of memory, and 100 GB of disk space
- Hadoop used as the case study user application
- HiBench used as the data processing benchmark
- Single failure simulated during each experiment run, 50 runs for each type of experiment

Mandrake Coordinator Timeline

- Outline of key events during an experiment run
- The events can be clustered based on relative timings to represent broader “phases”





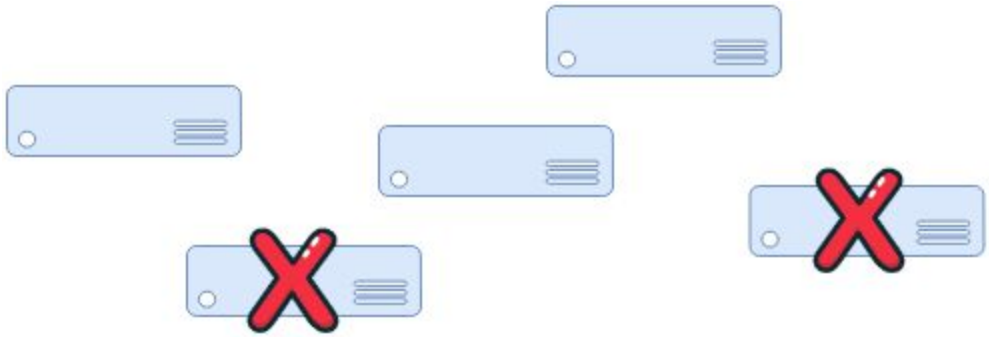
Results: Performance

- Mandrake Coordinator took an average of...
  - ~15 seconds to detect failures
  - ~6 seconds to handle a failure/update the mapping
  - ~65 seconds for end-to-end reconfiguration of the infrastructure
  - ~95 seconds for end-to-end reconfiguration of the entire system (infrastructure + application ingestion)

Phase	Event	Avg. Time (seconds)
Induce Failure	IF	0.1
Process Failure	DF	14.6
	HF	5.5
	RF	1.1
Spawn Instance	SI	43.6

Results: Failure Resilience

- Can sustain one or more failures every ~6 seconds and achieve a consistent mapping of the system at the infrastructure level
- Each failure “event” can be <50% of the machines, will reconfigure and continue to operate at degraded capacity



Takeaways & Conclusions

- It is possible to significantly enhance the reliability and availability of a system such as an edge cloud without ruining the performance
- Can even improve long-term performance by guaranteeing functionality (at degraded capacity) when the cloud is in a damaged state
- Merit to breaking the abstraction of the cloud when dealing with resource constrained environments
- Overall, Mandrake allows user applications to make much more informed decisions and preserve in the face of failures

Thank You!

Chandra Krintz

Rich Wolski

John Thomason

Markus Mock

...and everyone from RACELab, Port Hueneme Naval Base, the CS Dept. & UCSB