Building Health Checks for AEM

Ana Vinatoru | Software Engineer

Bē
Craig Ward

Overview

1. Sling Health Check Framework
2. Operations Dashboard – Health Reports
3. Adding a Custom Check to the Dashboard
4. Demo

The Need for Health Checks

Large-scale Sling applications (bundles, content packages, configurations)

- How do we know everything works as designed?

Health Checks

- Define correct and expected behavior
- Assert against these expectations
  - **Configurations**
  - **Metrics**
- Simple result: OK / not OK

When to use

- Smoke test before production
- Revalidation after configuration changes
- Load balancer checks
  - **Instance is ready to process requests**
  - **Instance is under high load**

Sling Health Check Framework

Allows you to verify the health of a live Sling instance

- Open source, has been around since ~2013
- Code available on SVN and GitHub
- Can be used in any other Sling project, not just AEM

Lets you define what "system health" means to you

- Bundles running, workflow and replication queues OK
- Sling Metrics / JMX attribute values within bounds
- Security (default passwords changed)
- Content is there / is or isn't accessible
- System resources are within bounds (disk, CPU, memory)

A Simple Health Check

OSGI service, implements HealthCheck interface

```
interface HealthCheck {
   Result execute();
}

public class Result {
   public boolean isOk() {...} // aggregate result
   public Result.Status getStatus() {...} //OK, WARN, CRITICAL , HEALTH_CHECK_ERROR
   public Iterator<Entry> iterator() {...} //(parametrized) messages
}
```

Properties

- HealthCheck.NAME
- HealthCheck.TAGS – health check executor
- HealthCheck.MBEAN_NAME – JMX access
- HealthCheck.ASYNC_CRON_EXPRESSION (async, scheduled execution)

A Simple Health Check - @SlingHealthCheck Annotation

- *org.apache.sling.hc.annotations* bundle

```
import org.apache.sling.hc.annotations.SlingHealthCheck;
...
@SlingHealthCheck(
        name="My Check",
        mbeanName="myCheckMbean",
        description="Sample Check",
        tags={"sample"})
public class MyCheck implements HealthCheck {

@Override
    public Result execute() {
        // health check code
    }
}
```

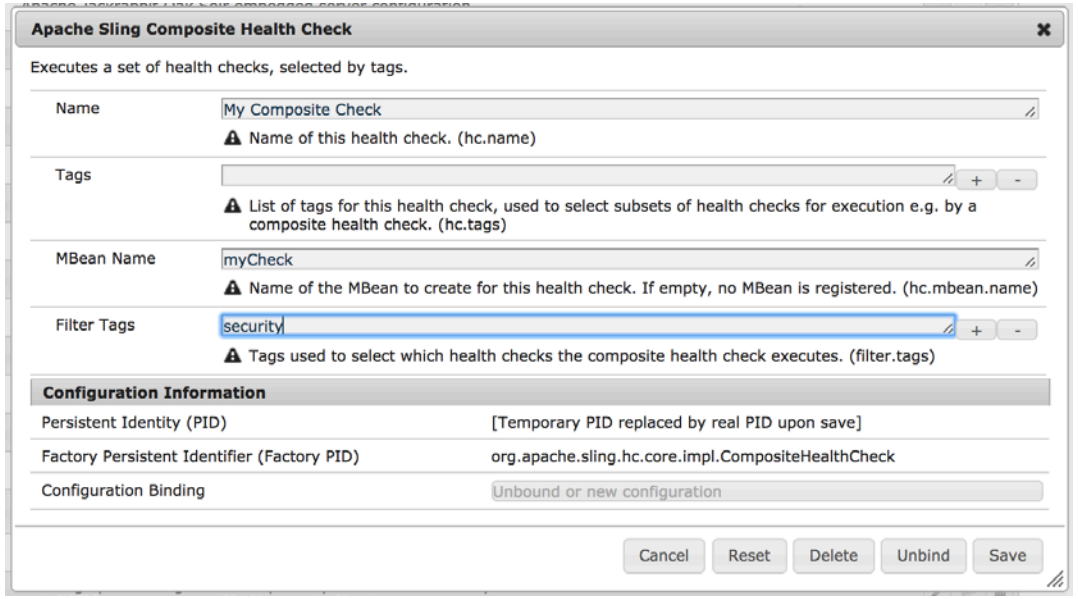
A Simple Health Check – OSGI service Annotations

- *org.osgi.service.component.annotations* bundle
- OSGI service component annotations are the norm
- Felix SCR annotations under maintenance mode
- Biggest change is how properties are defined (no more @Property annotation)

```
import org.osgi.service.component.annotations.Component;
...
@Component(service = HealthCheck.class,
        property = {
            HealthCheck.NAME + "=My Check",
            HealthCheck.TAGS + "=sample",
            HealthCheck.TAGS + "=test",
            HealthCheck.MBEAN_NAME + "=myCheckMbean"
        })

public class MyCheck implements HealthCheck {
    @Override
    public Result execute() {
    // health check code
    }
}
```

A Composite Health Check



- Aggregates results from multiple checks
- Selection by tags
- OSGI Configuration

  org.apache.sling.hc.core.impl.CompositeHealthCheck-mine

  filter.tags=["security"]

  hc.name=”myCompositeCheck"

  hc.tags=[""]

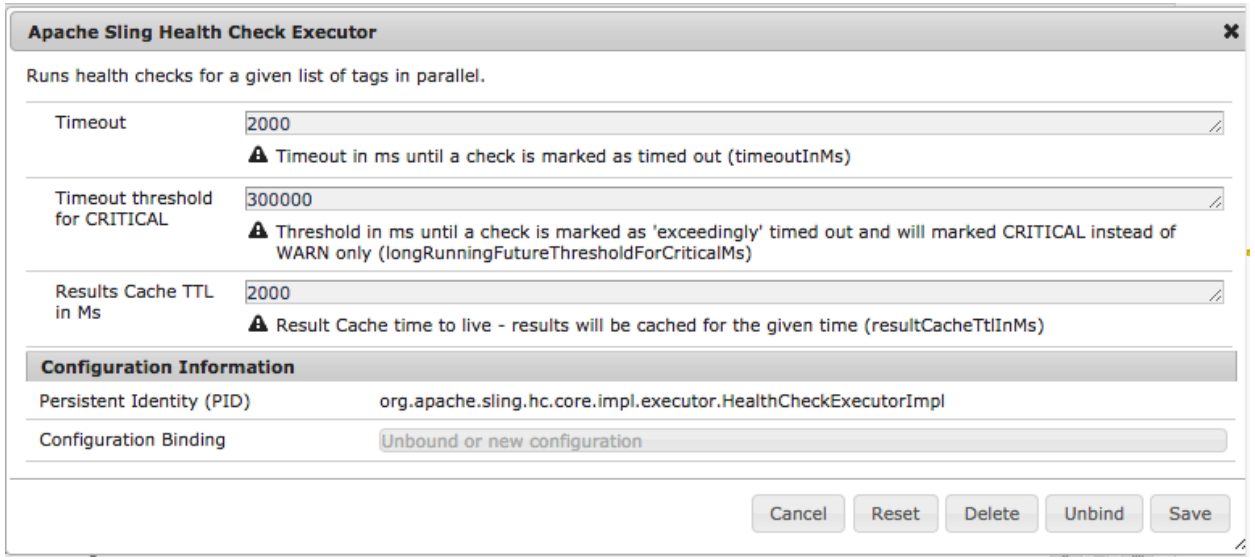
  hc.mbean.name=”myCompositeCheckMBean”

- When executed, it will aggregate the result of all the checks tagged with “security”
  - At least one check returns Critical => overall status is Critical
  - At least one check returns Warn => overall status is Warn
  - All checks return OK => overall status is OK

Executing Health Checks

Health Check Executor

- Configurable service
- Global timeouts
- Health Check result caching



Execution

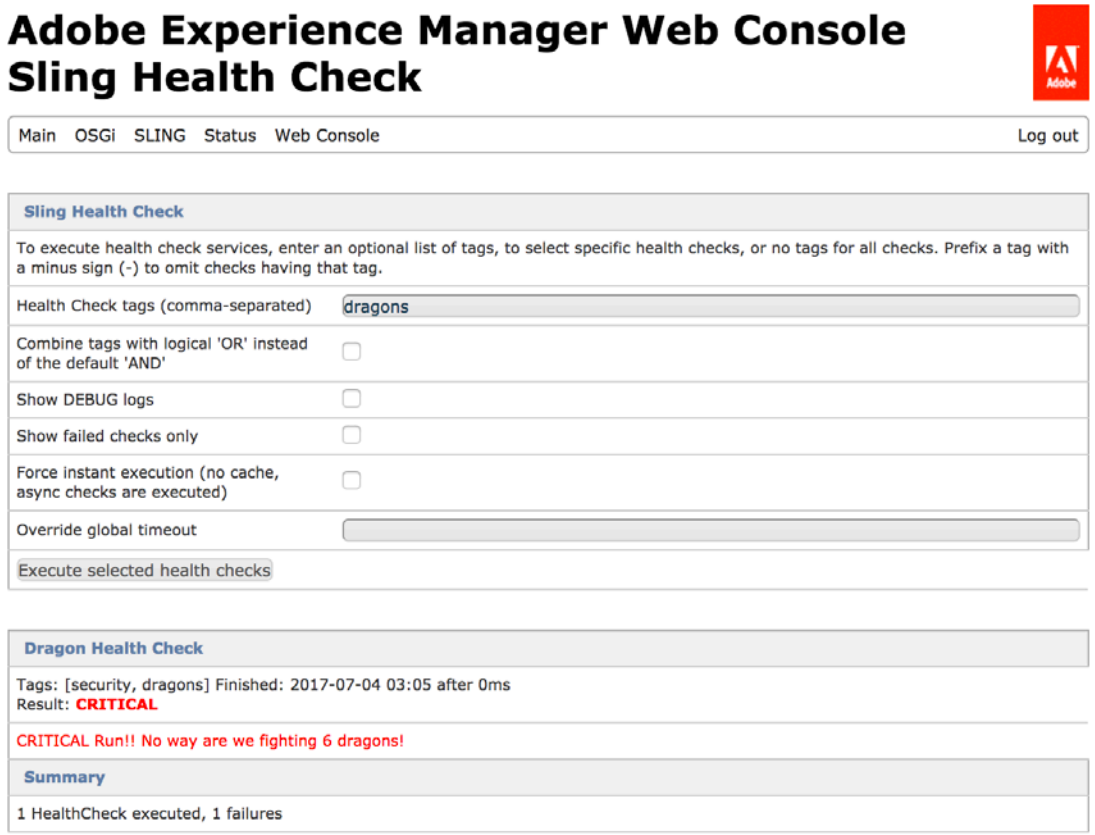
- Synchronous - default behavior
- Asynchronous
  - Cron expression in health check definition
  - Uses Sling Scheduler
  - Results are cached

Deciding which type of execution is more suitable depends on your use case

Executing Health Checks – Web Console

Health Check Web Console

- /system/console/healthcheck
- Filter by tags
- Debug messages
- Can show only failures
- Force instant execution



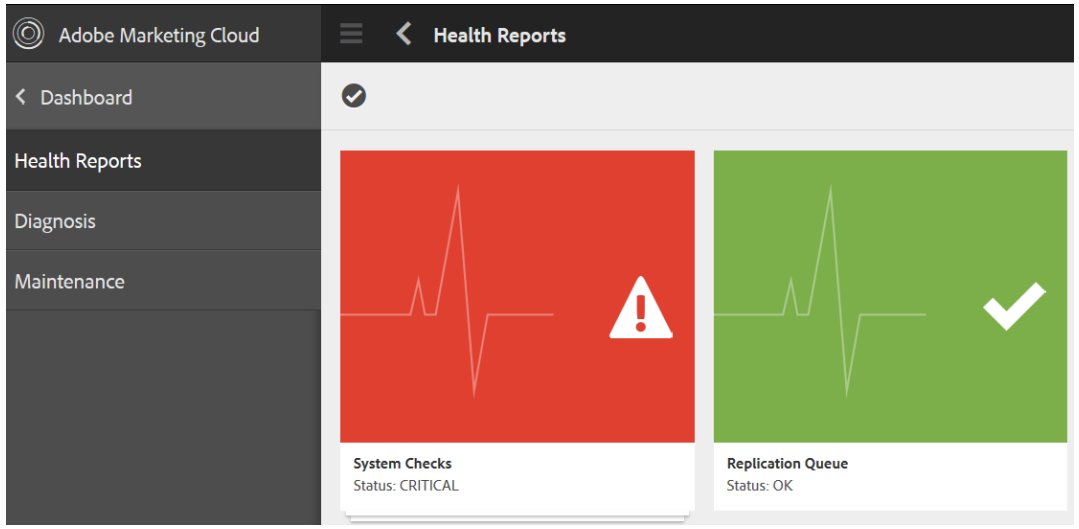
Executing Health Checks – JMX

- JMXResourceProvider: /system/sling/monitoring/mbeans/org/apache/sling/healthcheck/HealthCheck/myCheck.json
- Console: /system/console/jmx
  - MBean access
  - Properties exposed as attributes
- External access
  - JConsole
  - Any other tool

Executing Health Checks - Operations Dashboard

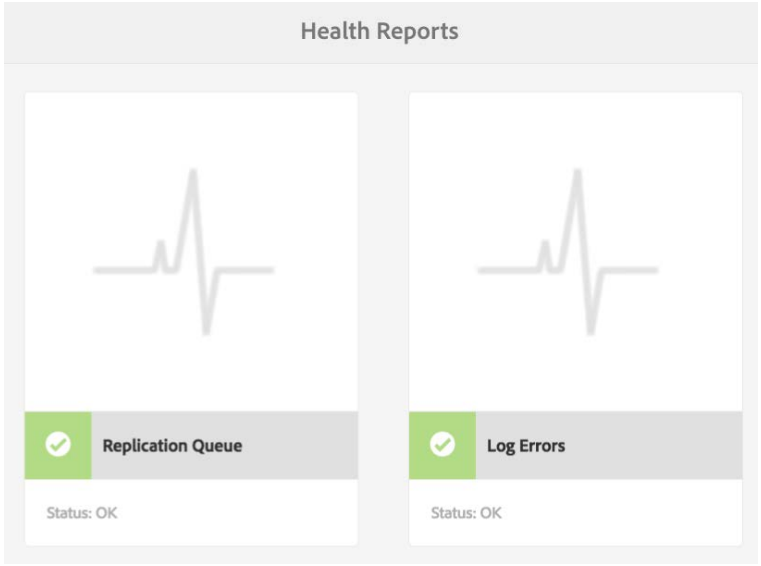
A set of tools for administrators and operators

- **Maintenance Tasks**
- **Health Reports**
- **Diagnosis Tools**
- **Monitoring (from 6.2)**



Timeline

- In AEM since 6.0
- Suffered a facelift in 6.2
- Configurations were moved in 6.3



Operations Dashboard – Health Reports

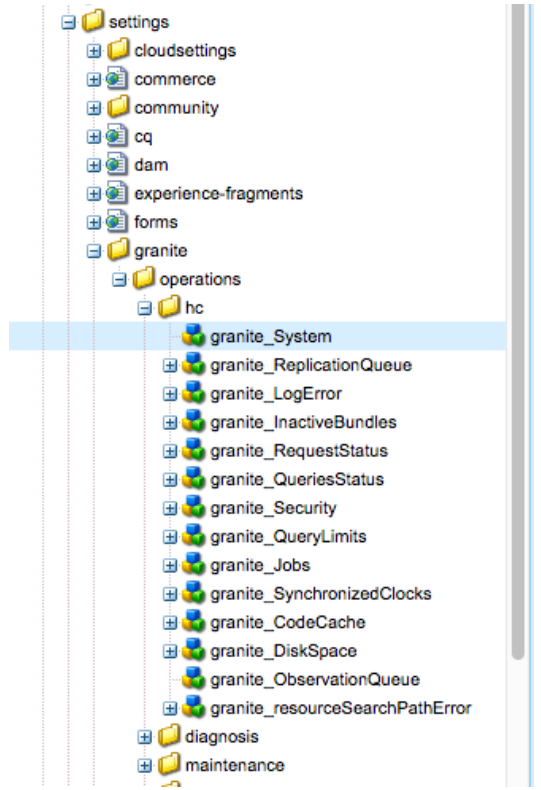
Main menu: Tools -> Operations -> Health Reports

One-click view of your system's health status

- AEM recommendations for security and performance
- Looks at MBean data, configurations, services
- Computes status for different product areas

UI configuration == nodes in the repository

- Default config under /libs – should not be modified
- Overlay in /apps
  - Add new check
  - Overlay existing check



Operations Dashboard – Health Reports

Default view: cards

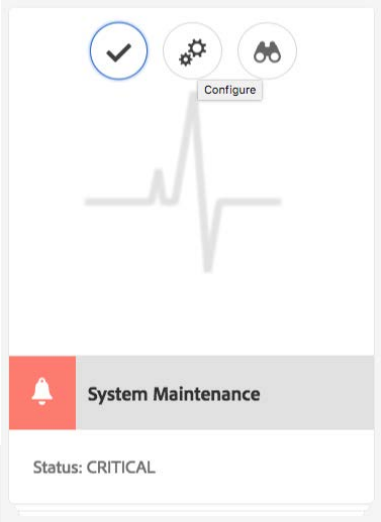
- Each check -> specific product area
- **OK**, **WARN** or **CRITICAL** color scheme
- Thresholds can be configured

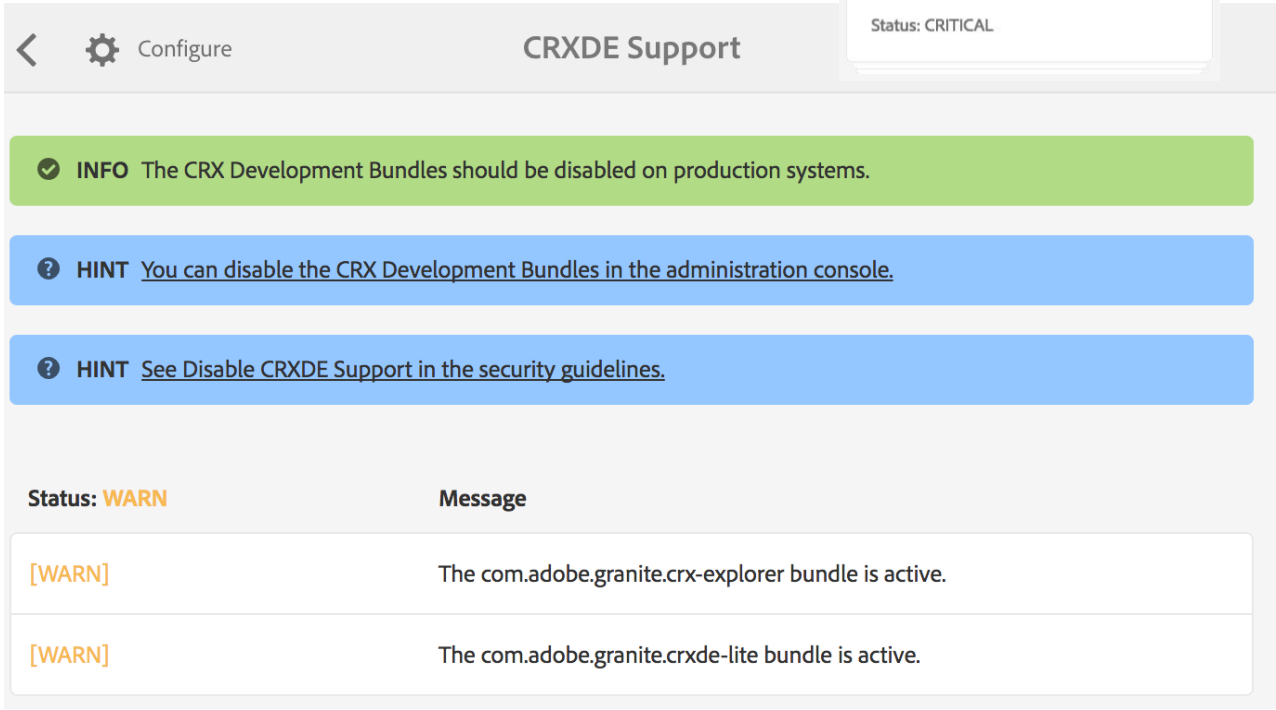
On click

- Composite Checks: composing checks
- Simple checks: detailed view

Detailed view

- Top:
  - Hints to help solve the issue
  - Links to other consoles and documentation
- Bottom
  - Detailed messages about execution

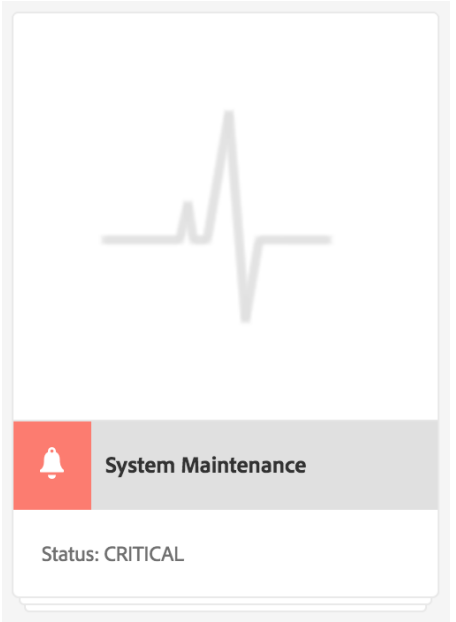




Operations Dashboard - Composite Checks

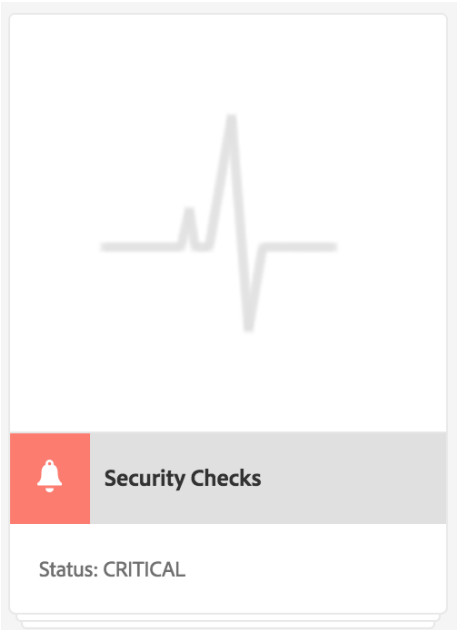
System Maintenance

- Maintenance task -> health check
- Critical or Warning
  - Task execution failed
  - Not configured (AuditLog Purge and Workflow Purge)
  - The tasks never had a chance to run (improved in 6.4)
- Resolution: investigate task failure
  - Error logs, MBeans



Security Checks

- Based on our security checklist (~20)
- Critical or Warning
  - Admin password wasn't changed
  - CRXDE enabled on production
  - Dispatcher settings too permissive
  - HTML library manager is not configured properly (minification)
  - Different filters are not properly configured
- Resolution: change the default configurations



Adding a New Check to the Operations Dashboard

1. Evaluate if it's indeed a case for a health check
   - A lot of processing => maybe something else is more suitable (external tool, scheduled job)
   - Result shouldn't fluctuate with isolated incidents
   - Synchronous vs. asynchronous

2. Decide which tags are suitable
   - The "security" tag is automatically picked up by OOTB composite check
   - You can add your own tags to the OOTB checks, using the Configuration Manager

3. Overall performance isn't affected by the check
   - Consume precomputed data (don't perform a large query)
   - Should not have side-effects

4. Reevaluate your thresholds as you gain knowledge

Adding a New Check to the Operations Dashboard

5. Implement the Java code and install it

6. Create a new configuration node
   - 6.0 -> 6.2 - /apps/granite/operations/config/hc
   - 6.3+ -> /apps/settings/granite/operations/hc
     - jcr:primaryType = nt:unstructured
     - sling:resourceType = granite/operations/components/mbean
     - resource = /system/sling/monitoring/mbeans/org/apache/sling/healthcheck/HealthCheck/myCheckMbean

7. IMPORTANT for 6.3+:
   - Context-aware configuration manager
   - Add the sling:configCollectionInherit = true to the /apps/settings/granite/operations/hc node
   - Otherwise, your configuration will replace the default AEM checks

Demo

Conclusions

- Sling Framework:
  - **Easy to use**
  - **Easy to extend**
- Operations Dashboard can be extended with your custom checks

Further reading

- Operations Dashboard Documentation
- Sling Health Check Framework
- AdaptTo() 2013 Slides
- "AEM: What to Monitor"
- OSGI Component Annotations


Adobe