Understanding Database Containerization

Joe Carroll & Jeff Fried
Intersystems
Jeff Fried

Data Mgmt Nerd
Passionate About
• Databases
• Search
• Big Data
• Text Analytics
• Information Strategy

Joe Carroll

Operations Nerd
Passionate About
• Databases
• Containerization
• Provisioning
• System Management

Jeff.Fried@intersystems.com
@jefffried

Joe.Carroll@intersystems.com
@JoeCarroll3000
InterSystems IRIS Data Platform

A Unified Data Platform for Big Data + Fast Data Analytic Applications

Multi - Model
- Relational
- Document
- Multidimensional
- Object

Multi - Workload
- HTAP with Isolated Workload Processing
- Unique Approach to Big Data Analytics
- ACID Transactions + Real-Time Indexing

Open Analytics
- Analytic SQL
- Business Intelligence
- Natural Language Processing
- Predictive Model Runtime
- Connector Architecture

Interoperability
- Languages and Tools
- Data Integration
- Application Integration
- Composite Business Process Orchestration

Flexible Cloud | Hybrid | On Premises Deployment Options
containers and databases are incompatible
This session

The **What & Why**

Tension *between* Containers *and* Enterprise Data

Data Persistence

Data Security
The 2019 survey reveals that 87 percent of IT professionals surveyed are now running container technologies, with 90 percent of those running in production and 7 in 10 running at least 40 percent of their application portfolio in containers — an impressive increase from two years ago, when just 67 percent of teams were running container technologies in production. – Portworx study – May 21, 2019
Separate Systems

Development → Testing/QA → Production
Ideally
Reality
Reality
Why Docker Containers?

Docker allows our application to be -

• Immutable
• Portable
• Scriptable
Comparison between containerized applications and VMs

<table>
<thead>
<tr>
<th>CONTAINER BENEFITS</th>
<th>VIRTUAL MACHINE BENEFITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consistent Runtime Environment</td>
<td>✓</td>
</tr>
<tr>
<td>Application Sandboxing</td>
<td>✓</td>
</tr>
<tr>
<td>Small Size on Disk</td>
<td>✓</td>
</tr>
<tr>
<td>Low Overhead</td>
<td>✓</td>
</tr>
</tbody>
</table>
Containers tap into a huge ecosystem

Cloud Native Landscape

1 - 25 of 2,277,915 available images.

Couchbase
- Updated 23 minutes ago
- couchbase
- Couchbase Server is an NoSQL document database with a distributed architecture.

Redis
- Updated 24 minutes ago
- redis
- Redis is an open source key-value store that functions as a data structure server.

MongoDB
- Updated 24 minutes ago
- mongodb
- MongoDB document databases provide high availability and easy scalability.

NGINX
- Updated 24 minutes ago
- nginx
- Official build of NGINX.

Official Images
- 10M+ Downloads
- 5M+ Stars
Containerized systems can get BIG
Operational Benefits of Containerization

**Operations:**

Portability

Scalability

Simple/Fast Deployment

Developer Speed and Efficiency
Business Benefits:
Agility
Cost Savings
Customer Satisfaction
Terms

• **Image**
  - Standardized, Portable & Runnable software bundle
  - Application code and Dependencies
  - READ ONLY

• **Container**
  - Image that is Executed in Isolation & Resource Controlled
  - Running application
  - Linux cgroups & namespaces
Enterprise Data
Definition:

**Enterprise Data**

- Vital to *lives*, *wallets*, and/or the *business*
- If we lose it or compromise it, then we’re in the news, people get fired, and the boss goes to jail.
- Examples: Patient Records, Financial Data, Payroll, etc.
Needs

- Data Persistence
- Data Security
Tension?

• Disposable/Ephemeral
  • MicroServices, Temporary FileSystem
• Broad ecosystem of open tools
  • Easy for developers, nightmare for operations people
• Container daemon is privileged
Needs

• Data Persistence
• Data Security
Summary

1. Containers are Immutable, Portable, and Scriptable.

2. Enterprise Data Applications Are In Dire Need Of These Benefits

3. Containers are for Enterprise Database Applications
This session

The **What & Why**

Tension between Containers and Enterprise Data

Data Persistence

Data Security
Problems

• By default, data written to a container doesn’t persist when that container no longer exists.
• A container’s writable layer is tightly coupled to the host machine.
• Writing into a container’s writable layer requires a storage driver to manage the filesystem. This abstraction results in abysmal performance.
Container
Container

Data
Volume

Container native way of persisting files outside of a container

• “Bind Mount”
• “Managed volume”
Examples! (Databases + Volumes)

https://github.com/tjosephcarroll/DatabaseContainerExamples

docker-compose.yml

```yaml
version: "3.6"
services:
  iris:
    image: store/intersystems/iris:2019.1.0.511.0-community
    volumes:
      - data:/dur
    environment:
      - ISC_DATA_DIRECTORY=/dur
    ports:
      - "52773:52773"
      - "51773:51773"
  volumes:
    data:"
Not All Data Needs Enterprise Data Persistence
Summary

1. Critical Data Must Persist Outside The Container
2. Not All Data is Critical Data
3. Make Smart Design Decisions Regarding Where Data Goes From The Beginning
This session

The What & Why

Tension between Containers and Enterprise Data

Data Persistence

Data Security
Reminder - Mission Critical
Repository/Registry

A collection of Container Images stored in a centralized location accessible to the Container runtime engine.

- Create, Read, Update, Delete
- “Push”, “Pull”, “Delete”

Docker Hub and Store, but many commercial products
Trust, Scan, and Sign Your Images

- Build a registry of trusted images (build your own images)
- Scan your images for vulnerabilities
- Rotate your credentials
- There are many tools! This is easy! Do it please!
Secure Access To Your Data

• No database passwords in source.
• Defined at runtime.
• Encrypted at rest and in motion.
• Start with Secrets!
Examples! (Databases + Secrets)

https://github.com/tjosephcarroll/DatabaseContainerExamples
Secure Your Runtime Environment

• What user/group are your containers running as?
• Are the cgroups and namespaces what you want?
• Any container in your configuration can be the culprit.
Summary

1. Use Secrets To Protect Data Access
2. Runtime Security - Users, Namespaces, Cgroups
3. Scan, Sign, and Trust Your Images
containers and databases can work well together if you attend to data persistence and data security
What can you do with Databases and Containers?

What everyone wants:

”Elasticity”

- Virtualization, Commodity & Cloud
- Scale out / in
- Continuous Availability

What they don’t want to lose

”SQL”

- ACID (consistency)
- Existing SQL skills & code
- SQL database abstraction

“I want to elastically scale my SQL RDBMS to the cloud”
LET’S IMPLEMENT CLOUD COMPUTING SO I HAVE SOMETHING TO TALK ABOUT AT THE EXECUTIVE MEETING.

TELL THEM WE’RE EVALUATING IT. THAT WAY NEITHER OF US NEEDS TO DO ANY REAL WORK.

I LIKE IT WHEN YOU DO REAL WORK.

SORRY. I THOUGHT YOU WERE LEADING BY EXAMPLE.
Questions?

Contact:

jeff.fried@intersystems.com
joe.carroll@intersystems.com

www.InterSystems.com