Management and Monitoring

Jitka Kožaná

Quality Assurance Supervisor, JBoss by Red Hat

Advanced Java EE Lab @ FI MU

May 11 2015

Agenda

- Monitoring
  - JDK tools
  - System tools
  - WF8 specifics
- WF8 architecture
- WF8 Domain Model
- WF8 Management
  - CLI / Scripting + Java API + HTTP API
  - WebUI + RHQ / JON

Monitoring – motivation

You are using WildFly 8, so bright future lies ahead ...

Really?

We will learn how to do some basic investigation and JVM monitoring.

JDK tools - JAR level investigation

- List files in given jar archive
  - jar
  - unzip
- Disassemble the class file
  - javap

JDK tools – memory

- Memory map
  - jmap
    - Show heap, create heap dump
- Analyze heap dump
  - jhat
    - Parses a java heap dump, launches a webserver to browse the dump

JDK tools – stack trace and JVM stats

- Java stack traces of threads
  - jstack
  - stack traces of Java threads for a given Java process, core or remote server
  - for investigating thread locking issues
- JVM statistics monitoring
  - jstat

JDK tools – GUI

jconsole

- Heap and Non-Heap memory usage, CPU usage, VM summary
- Number of threads and classes, stack trace for each thread
- MBeans details

jvisualvm

- Nicer look & feel, based on NetBeans platform
- Heap (and PermGen) memory usage, CPU usage, VM summary
- Number of threads and classes, details for each thread, not stack trace
- Lightweight CPU and memory profiling + sampling

System information

- OS version
- Memory usage
- Disk space
- Processes
- Network – traffic and ports

WildFly8 specifics

JDR - JBoss Diagnostic Reporter

- $WF_HOME/bin/jdr.sh [.bat]
- JBoss specific tool for diagnostic
- add at least one user into ManagementRealm using bin/add-user.sh

jconsole

- $WF_HOME/bin/jconsole.sh [.bat]
- Jconsole with added WildFly management extension (JBoss Remoting + JSR 160)

Advanced tools

- your IDE debugger
- your IDE profiler
- JProfiler - http://www.ej-technologies.com/products/jprofiler/overview.html
- Java Decompiler - http://jd.benow.ca/
- TDA - Thread Dump Analyzer - http://java.net/projects/tda/
- MAT - Memory Analyzer - http://www.eclipse.org/mat/

- Wireshark - http://www.wireshark.org/

WildFly8: 2nd generation of JBoss AS7

- Why was AS7 rewritten from scratch?
- Legacy subsystems
- Boot time
- Memory footprint
- Bad modularity
- Administration options
- Not “good enough”

WildFly 8

- Builds on top of JBoss AS7
- Small and even #@*%ing faster
- No legacy stuff
- Better manageability
- Multi-node management
- Simplified configuration
- Modular, OSGi enabled

WildFly 8 Architecture

- **core**
- **extensions** to the core
- **clients** for management interface
  - CLI and web based management console

Core

- jboss-modules
  - is the first thing started
  - modular and concurrent classloading system
  - O(1) dependencies resolution
  - Module sees only its imports
- jboss-msc: modular service container
  - Everything is (interface based) service
  - Services are deployed on demand and in parallel
- Extensible management layer
  - Mediate access to service container
  - Provides configuration model for the AS

Operational modes

- Standalone mode
- Domain mode
- The only difference is in how severs are managed, not in the capabilities

Standalone

- Traditional JBoss single JVM server
- Managed individually: 1 configuration file
- No lifecycle management, just shutdown
- Development and embedded solutions

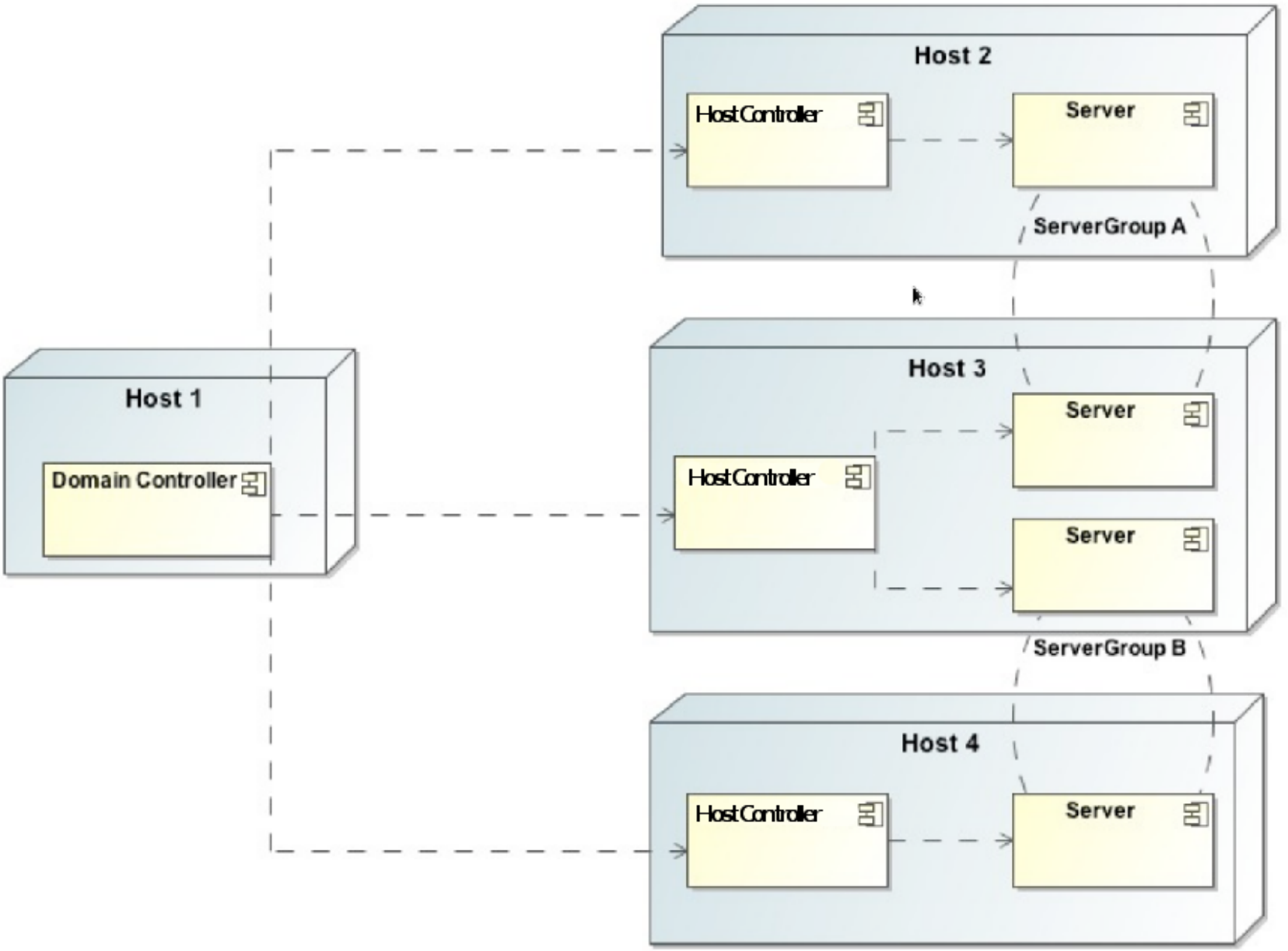
Domain

- Multi-JVM, multi-server model
- manage multiple servers via a single control point
  - configure a cluster, start/stop nodes in a cluster, deploy an application to all nodes in the domain,...
- end user configuration centralized in a few files
- schema files for all configurations
- everything in the configuration is exposed via management API

Domain mode: Processes

- **Process controller**
  - Managing & (re)starting processes
- **Host controller**
  - Responsible for configuration distribution across domain
  - One HC is selected as **Domain controller**, the rest are slaves
- **Server instance**

Domain model



Domain model - terms

- **server** - one AS instance
- **server group** - set of server instances that will be managed and configured as one
- **cluster** - server group with group communication services configured
- **module** - classloading space, grouping of classes in some jar(s)s
- **subsystem** - block of configuration, has its own namespace, basically some grouping of services
- **profile** - set of subsystems

Domain model - schema

- https://community.jboss.org/wiki/JBossASDomainSchema
- docs/schema/jboss-as-config_2_2.xsd in WF8 distribution
- docs/schema/*.xsd
- domain.xml, host.xml

```
<server-groups>
       <server-group name="main-server-group" profile="default">
           <jvm name="default">
               <heap size="64m" max-size="512m"/>
           </jvm>
           <socket-binding-group ref="standard-sockets"/>
       </server-group>
       <server-group name="other-server-group" profile="default">
           <jvm name="default">
               <heap size="64m" max-size="512m"/>
           </jvm>
           <socket-binding-group ref="standard-sockets"/>
       </server-group>
</server-groups>
```

```
<servers>
       <server name="server-one" group="main-server-group" auto-start="true">
           <jvm name="default"/>
       </server>
       <server name="server-two" group="main-server-group" auto-start="true">
           <jvm name="default">
               <heap size="64m" max-size="256m"/>
           </jvm>
           <socket-binding-group ref="standard-sockets" port-offset="150"/>
       </server>
       <server name="server-three" group="other-server-group" auto-start="false">
           <socket-binding-group ref="standard-sockets" port-offset="250"/>
       </server>
</servers>
```


Management

- The problem: management model too large and complex
- The requirements for the API:
  - Simple, powerful, stable
  - As few compile time and runtime dependencies as possible
  - Backward compatibility

- WF8 uses de-typed management API and a small library:

  jboss-dmr.jar

DMR – dynamic model representation

- https://github.com/jbossas/jboss-dmr
- https://docs.jboss.org/author/display/WFLY8/Detyped+management+and+the+jboss-dmr+library
- All management operations operate with/on DMR
- Compatibility is stressed
- Convertible from/to JSON

Java API

- Native management interface uses an open protocol based on the JBoss Remoting library
- The management protocol is an open protocol, so a completely custom client could be developed without using prepared libraries (e.g. using Python or some other language)

- Maven artifact org.wildfly:wildfly-controller-client

- https://docs.jboss.org/author/display/WFLY8/The+native+management+API

Java API

```
ModelControllerClient client = ModelControllerClient.Factory.
        create(InetAddress.getByName("localhost"), 9999);

ModelNode op = new ModelNode();
op.get("operation").set("read-resource");
op.get("recursive").set(true);
op.get("include-runtime").set(true);
op.get("recursive-depth").set(10);

ModelNode returnVal = client.execute(op);
System.out.println(returnVal.get("result").toString());
client.close();
```


HTTP API

- http://localhost:9990/management
- Sometimes called REST API
- HTTP request in JSON like format
- The default operation is read-resource
- add user into ManagementRealm using bin/add-user.sh

- https://docs.jboss.org/author/display/WFLY8/The+HTTP+management+API
- https://community.jboss.org/wiki/HTTPJSON-likeAPI

CLI

- Command line management tool for the WF8 server
- Command bin/jboss-cli.sh or bin/jboss-cli.bat
- Interactive mode
- Non-interactive mode
- Batch mode
- GUI mode
- Operations based on model

CLI

```
$ ./bin/jboss-cli.sh --connect controller=IP_ADDRESS
[standalone@IP_ADDRESS:9999 /] /system-property=foo:add(value=bar)
[standalone@IP_ADDRESS:9999 /] /system-property=foo:read-resource
{
    "outcome" => "success",
    "result" => {"value" => "bar"}
}
[standalone@IP_ADDRESS:9999 /] /system-property=foo:remove
{"outcome" => "success"}
```

```
[domain@IP_ADDRESS:9999 /] /system-property=foo:add(value=bar)
[domain@IP_ADDRESS:9999 /] /system-property=foo:read-resource
[domain@IP_ADDRESS:9999 /] /system-property=foo:remove
```

```
[domain@IP_ADDRESS:9999 /] /host=master/system-property=foo:add(value=bar)
[domain@IP_ADDRESS:9999 /] /host=master/system-property=foo:read-resource
[domain@IP_ADDRESS:9999 /] /host=master/system-property=foo:remove
```

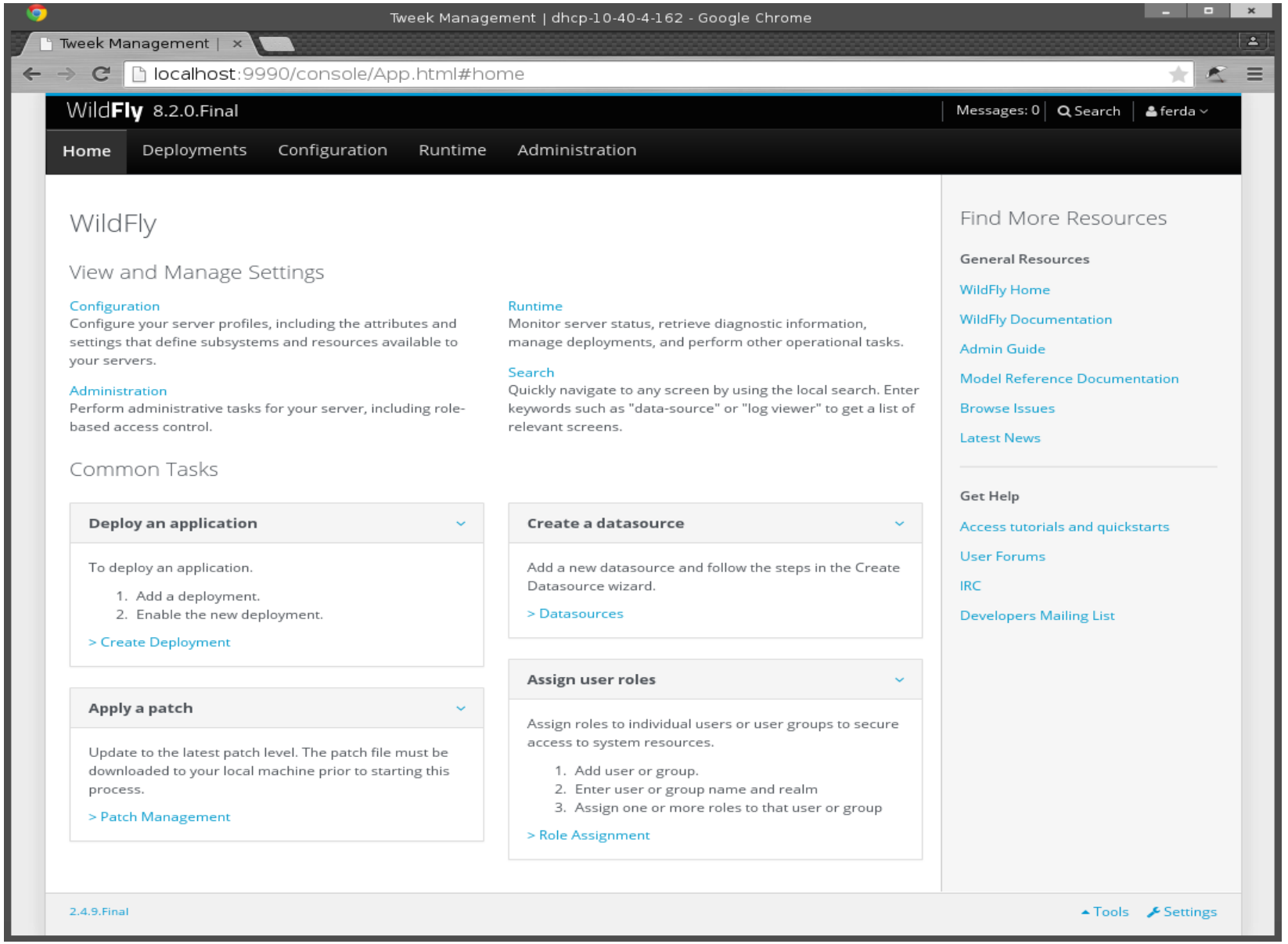
```
[domain@IP_ADDRESS:9999 /] /host=master/server-config=server-one/system-property=foo:add(value=bar)
[domain@IP_ADDRESS:9999 /] /host=master/server-config=server-one/system-property=foo:read-resource
[domain@IP_ADDRESS:9999 /] /host=master/server-config=server-one/system-property=foo:remove
```


CLI

- https://community.jboss.org/wiki/CommandLineInterface
- https://community.jboss.org/wiki/GenericTypeCLICommands
- https://community.jboss.org/wiki/CLICompoundValueFormat
- https://community.jboss.org/wiki/CLINon-interactiveMode
- https://community.jboss.org/wiki/CLIBatchMode
- https://docs.jboss.org/author/display/WFLY8/CLI+Recipes

- https://community.jboss.org/wiki/JBossAS7Command-linePublic API

Web console


Tweek Management | dhcp-10-40-4-162 - Google Chrome
Tweek Management
localhost:9990/console/App.html#home
WildFly 8.2.0.Final
Messages: 0
Search
ferda
Home
Deployments
Configuration
Runtime
Administration
WildFly
View and Manage Settings
Configuration
Configure your server profiles, including the attributes and settings that define subsystems and resources available to your servers.
Administration
Perform administrative tasks for your server, including role-based access control.
Runtime
Monitor server status, retrieve diagnostic information, manage deployments, and perform other operational tasks.
Search
Quickly navigate to any screen by using the local search. Enter keywords such as "data-source" or "log viewer" to get a list of relevant screens.
Common Tasks
Deploy an application
To deploy an application.
1. Add a deployment.
2. Enable the new deployment.
> Create Deployment
Create a datasource
Add a new datasource and follow the steps in the Create Datasource wizard.
> Datasources
Apply a patch
Update to the latest patch level. The patch file must be downloaded to your local machine prior to starting this process.
> Patch Management
Assign user roles
Assign roles to individual users or user groups to secure access to system resources.
1. Add user or group.
2. Enter user or group name and realm
3. Assign one or more roles to that user or group
> Role Assignment
Find More Resources
General Resources
WildFly Home
WildFly Documentation
Admin Guide
Model Reference Documentation
Browse Issues
Latest News
Get Help
Access tutorials and quickstarts
User Forums
IRC
Developers Mailing List
2.4.9.Final
Tools
Settings

Thank you for your attention.

Questions?