Management and Monitoring

Jitka Kožaná

Quality Assurance Supervisor, JBoss by Red Hat

Advanced Java EE Lab @ ČVUT

Nov 28 2014

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 – process

- List of JVMs
  - jps -l [-m -v]
  - JDK specific
- Configuration information
  - jinfo
  - Java system properties + VM flags

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://java.decompiler.free.fr/
- 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

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
  - 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

Domain vs. standalone

Standalone

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

Domain

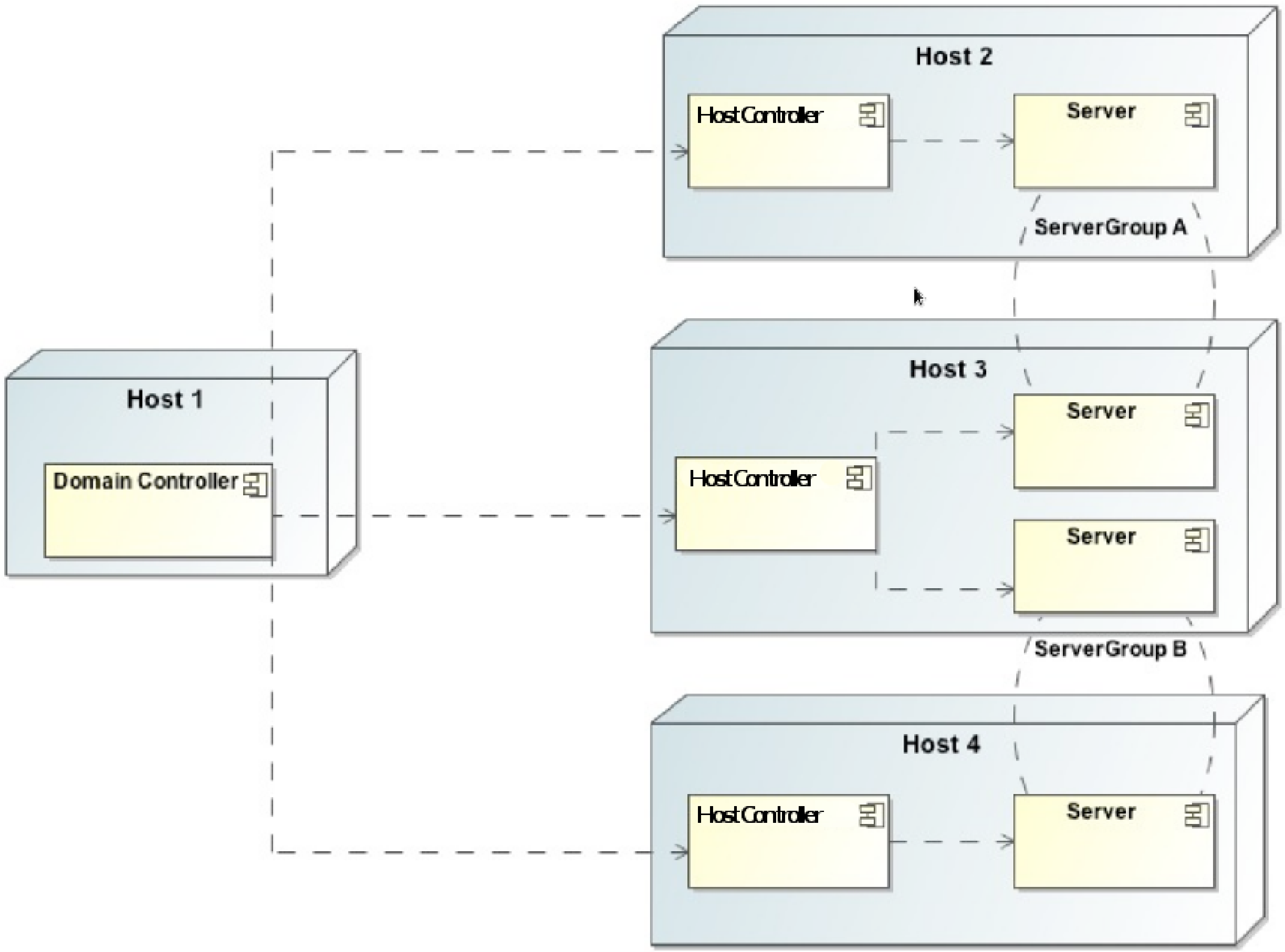
- Multi-JVM, multi-server model
- Lifecycle managed by Process Controller (PC)
- Management coordinated by Domain Controller (DC)
- Multiple server instances per host managed by Host Controller (HC)
- HC on master node is DC

The only difference between domain and standalone is in how severs are managed, not in the capabilities

Domain model: key goals

- 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 model

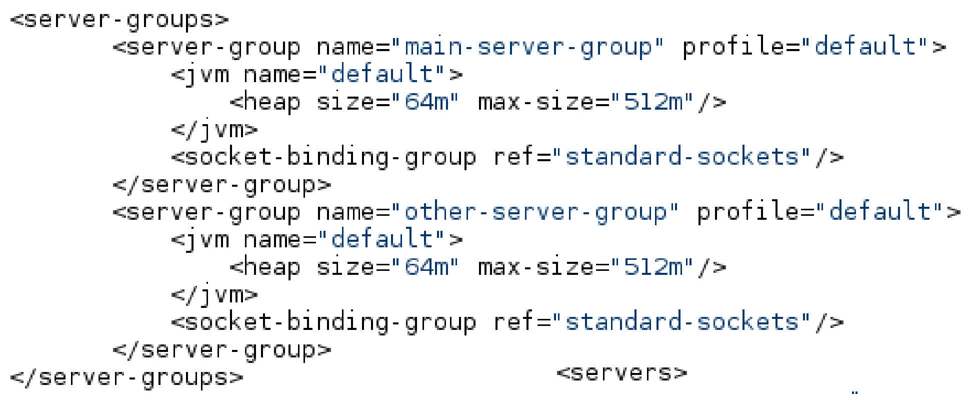


Domain model - terms

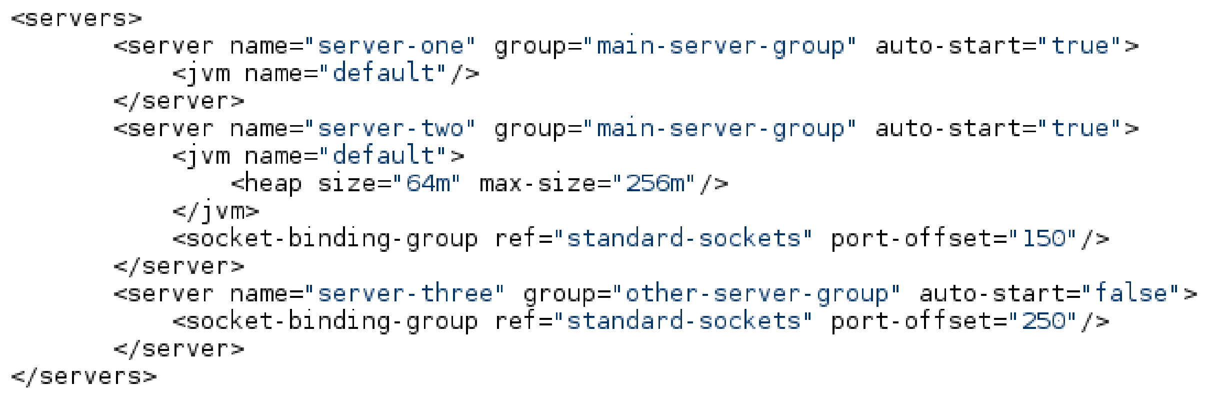
- **server** - one AS instance
- **server group** - set of server instances that will be managed and configured as one
- **cluster** - server group with group communication services configured
- **module** - classloading space, grouping of classes in some jar(s)s
- **subsystem** - block of configuration, 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_0.xsd in WF8 distribution
- docs/schema/*.xsd
- standalone.xml, 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

De-typed Management API

- https://community.jboss.org/wiki/FormatOfADetypedOperationRequest
- https://community.jboss.org/wiki/FormatOfADetypedOperationResponse
- https://docs.jboss.org/author/display/WFLY8/Description+of+the+Management+Model
- De-typed API: can construct any complex data types by using small number of basic Java classes (like java.lang.String, java.lang.Integer, etc)
- All parameter and return values in the API are expressed using these few types, making it easier to make it backward compatible
- Auto-conversion like dynamic languages

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
```

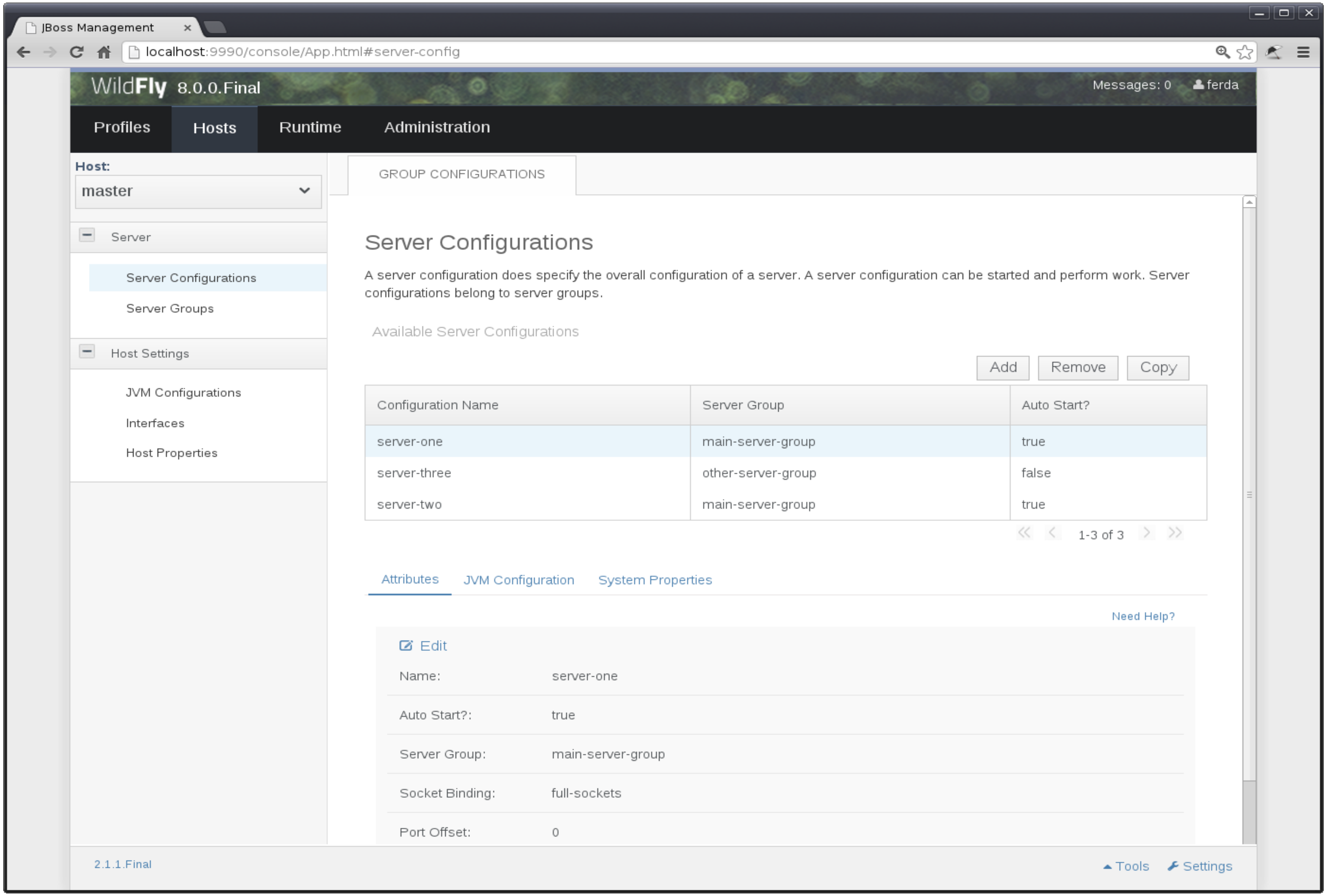
```
[domain@IP_ADDRESS:9999 /] /host=master/server-config=server-one/system-property=foo:add(value=bar)
[domain@IP_ADDRESS:9999 /] /host=master/server-config=server-one/system-property=foo:read-resource
[domain@IP_ADDRESS:9999 /] /host=master/server-config=server-one/system-property=foo:remove
```

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-linePublicAPI

Web console



Server Configurations

A server configuration does specify the overall configuration of a server. A server configuration can be started and perform work. Server configurations belong to server groups.

Available Server Configurations

Configuration Name	Server Group	Auto Start?
server-one	main-server-group	true
server-three	other-server-group	false
server-two	main-server-group	true

Name:	server-one
Auto Start?:	true
Server Group:	main-server-group
Socket Binding:	full-sockets
Port Offset:	0

Thank you for your attention.

Questions?