



Consulting Services

JobScheduler

Architecture Decision Template

Information for
Consulting Parties

- **Overview**
 - Components: JOC Cockpit / Web Service / Master / Agent
 - Security: JOC Cockpit / Web Service / Master / Agent
 - Platforms: JOC Cockpit / Web Service / Master / Agent
- **Agent Cluster**
 - Architecture: JobScheduler Agent Cluster
- **Master Passive Cluster**
 - Architecture: Primary JobScheduler Master
 - Architecture: Backup JobScheduler Master
- **Master Active Cluster**
 - Architecture: Active Cluster JobScheduler Master
 - Architecture: Active Cluster JobScheduler with failed instance
- **Master / Agent Cluster**
 - Architecture: Master/Agent Passive Cluster JobScheduler
 - Architecture: Master/Agent Active Cluster JobScheduler
- **Supervisor JobScheduler**
 - Architecture: Supervisor for Master Passive and Active Cluster
 - Architecture: Supervisor for Unclustered JobScheduler

Architecture Decision Template

Agent Cluster

- Agent Cluster
- Fixed Priority and Round-Robin Scheduling: Redundancy and automated fail-over

Passive Cluster

- Primary & Backup JobScheduler
- Redundancy and automated fail-over

Active Cluster

- Active Cluster JobScheduler
- Redundancy and load sharing

Master/
Agent Cluster

- Master/Agent Cluster JobScheduler
- Redundancy, load sharing, load distribution

Supervisor
JobScheduler

- Passive & Active Cluster Support,
- Master/Agent Cluster Support, Unclustered JobScheduler Support
- Central Configuration

Overview: Components

JOC Cockpit / Web Service

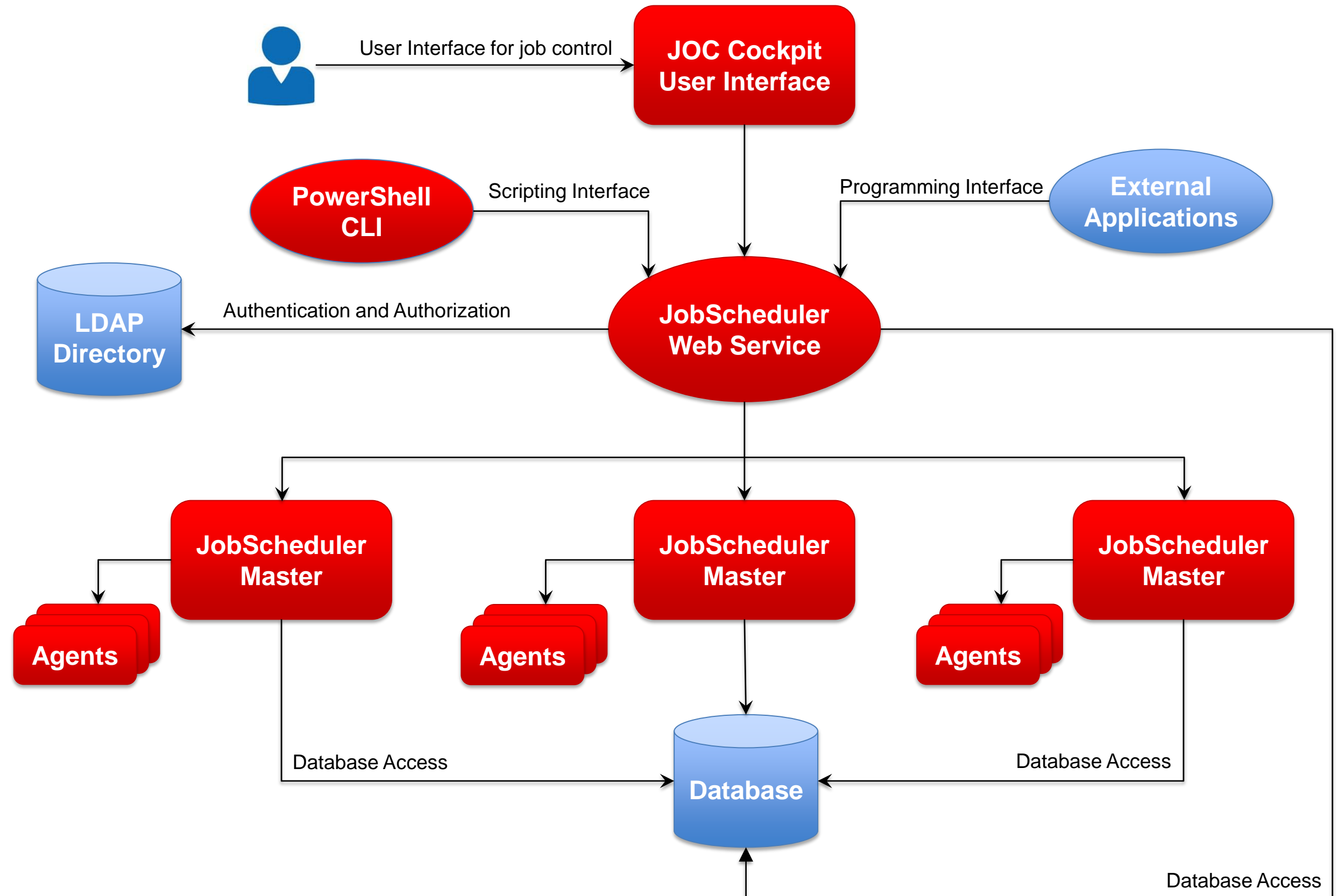
- The JOC Cockpit is a user interface for job control with browsers
- Users access the Master using a Web Service that performs authentication and authorization – optionally against an LDAP directory

Interfaces

- The PowerShell Command Line Interface and External Applications use the same Web Service for access to a JobScheduler Master
- Authorization is available for individual requests to the JobScheduler Master

Master / Agent

- The JobScheduler Master executes tasks and orchestrates Agents
- Agents are deployed on top of existing servers running the programs and scripts that should be scheduled



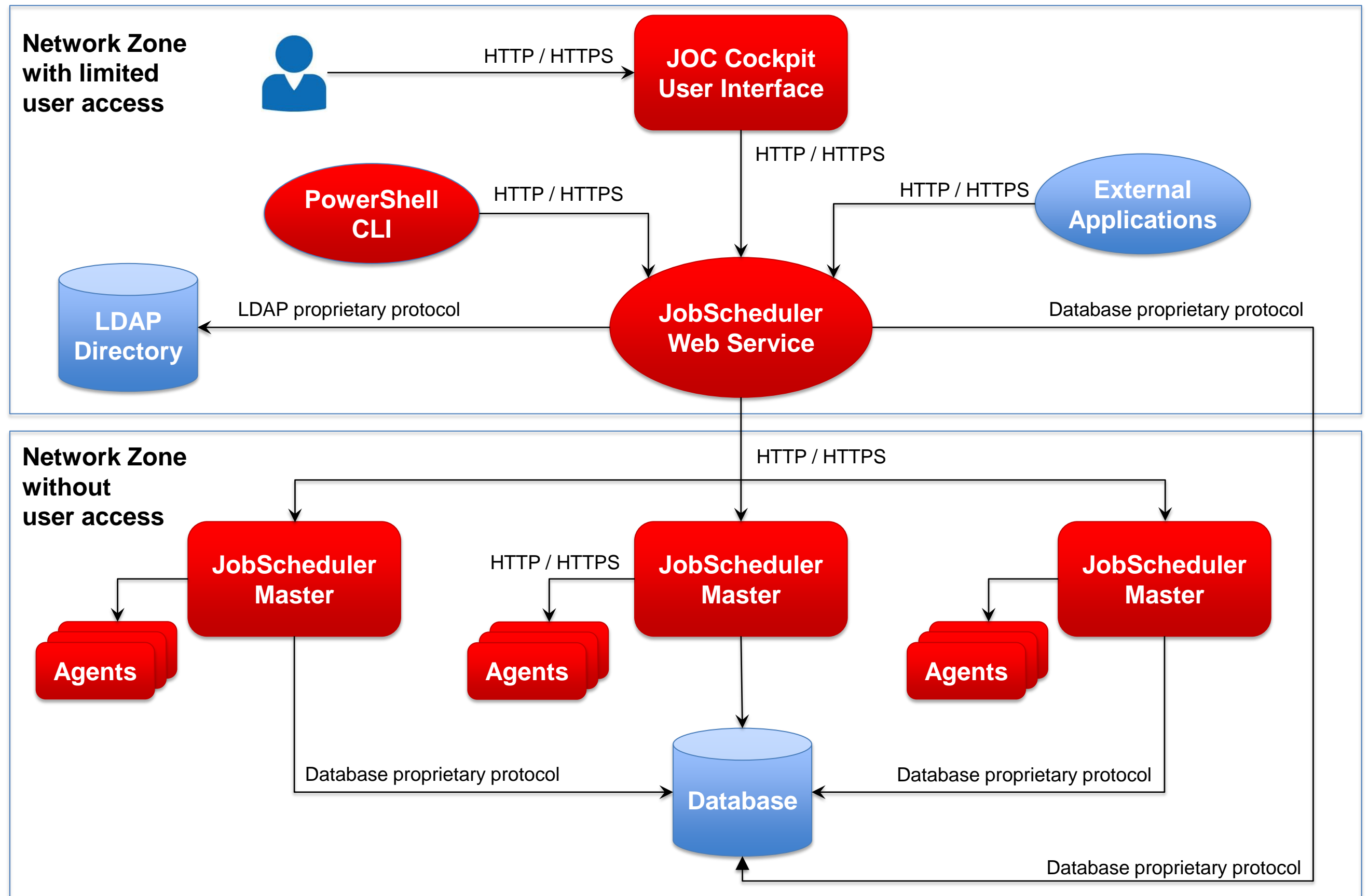
Overview: Security

Network Zone with restricted user access

- Users have limited access that requires authentication
- Any connection to a Master is authenticated by the Web Service that can be configured to use LDAP
- Use of HTTPS for connections can be enforced

Network Zone without user access

- Master and Agent instances are operated in this zone without direct user access
- The Master instances are accessed exclusively by the Web Service
- The Agent instances are exclusively accessed by Master instances



Overview: Supported Platforms

Cockpit / Web Service

- The JOC Cockpit and REST Web Service are available for Windows and Linux

Master / Agent

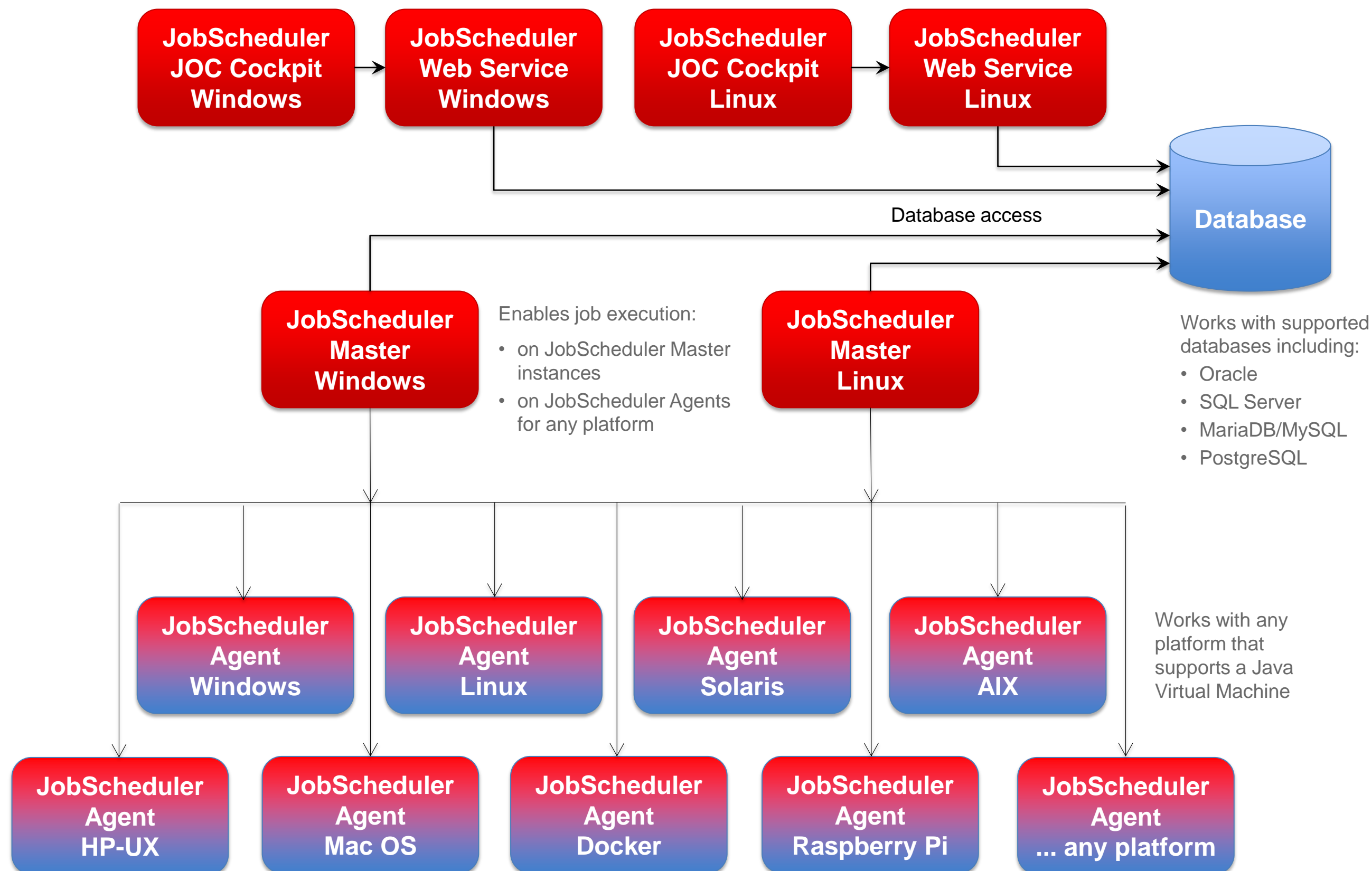
- JobScheduler Master is available for Windows and Linux
- JobScheduler Agents are available for any platform that supports a Java Virtual Machine

Database

- The JobScheduler REST Web Service and Master use a database on any platform

Jobs

- Jobs can be executed locally on the Master
- Jobs can be executed on any JobScheduler Agent



Master/Agent Platforms

- JobScheduler Master is available for Windows and Linux
- JobScheduler Agents are available for any platform that supports a Java Virtual Machine

Agent Cluster

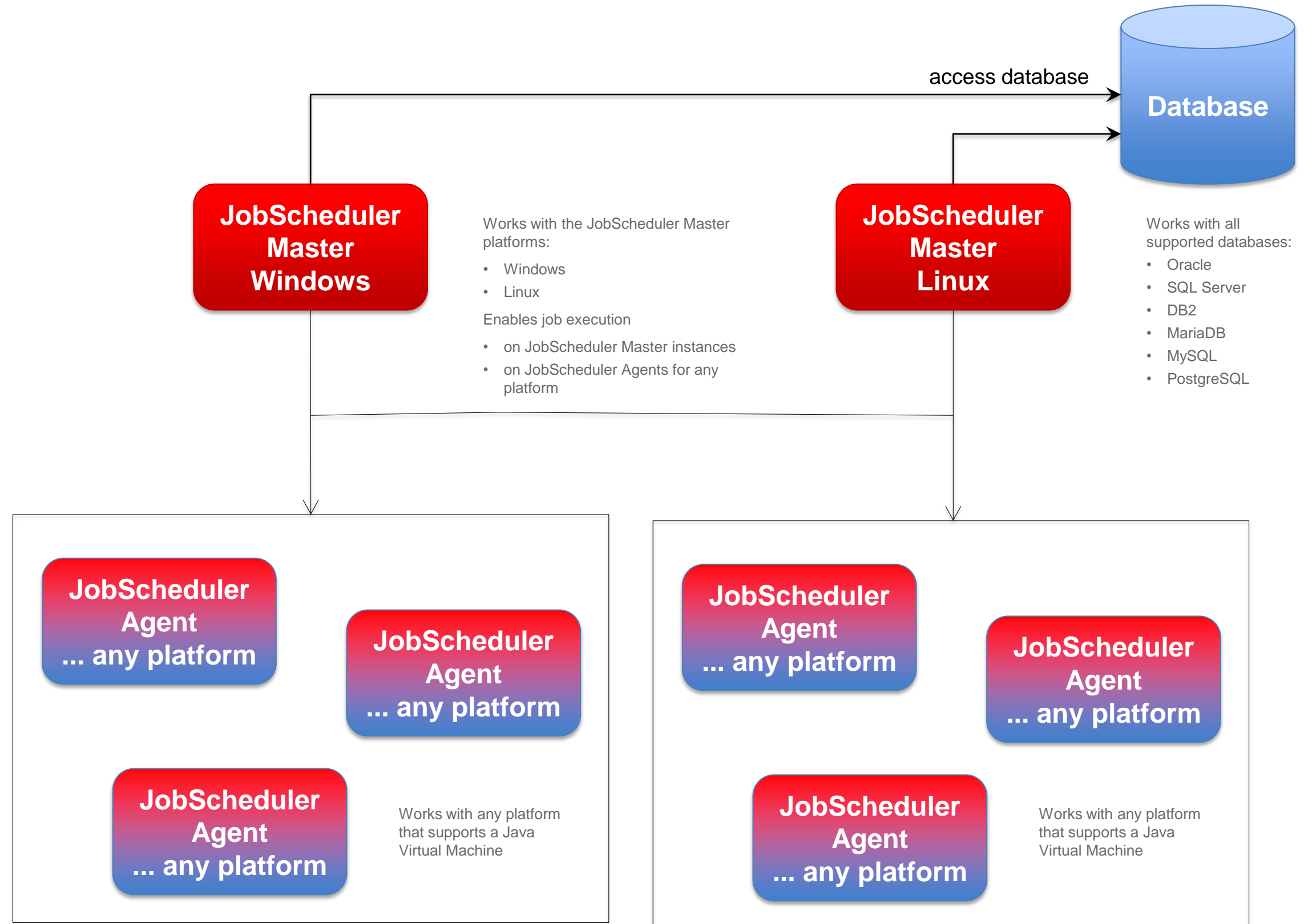
- Agents can be configured to work in a Cluster.

Fixed Priority Scheduling

- JobScheduler Master selects the first available Agent from a Cluster for job execution.
- Should an Agent not be available then the next available Agent is used.

Round-Robin Scheduling

- JobScheduler Master switches the Agent used for each job execution.
- Should an Agent not be available then the next available Agent is used.



Passive Cluster

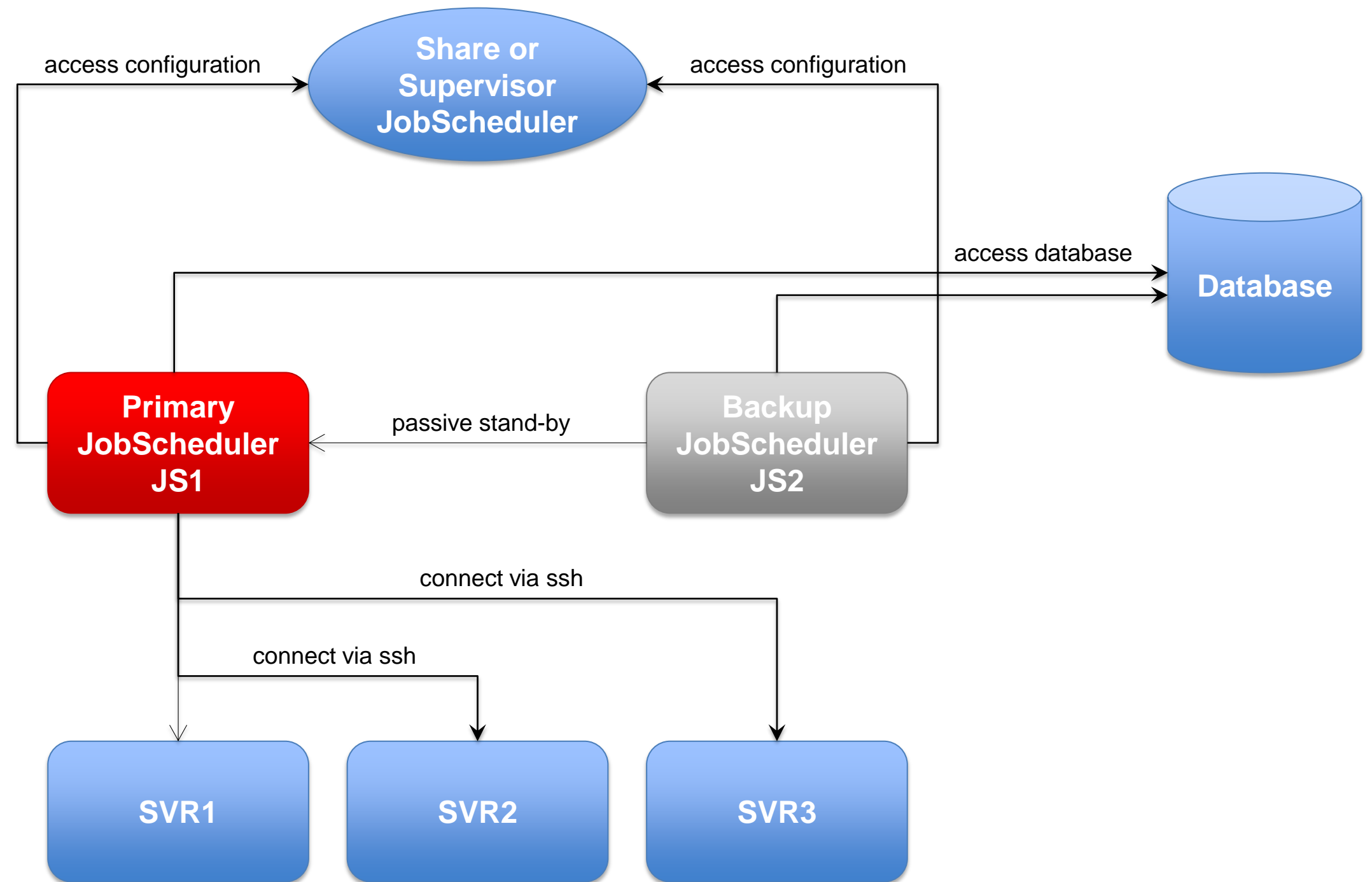
- Primary and Backup JobScheduler Master use the same database
- Primary JobScheduler Master is monitored by its failover instance
- Failover instance operates in stand-by mode
- All connections to servers use the SSH protocol

SSH Connections*JITL Jobs*

- Requires a JVM per task
- Memory resources

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Passive Cluster

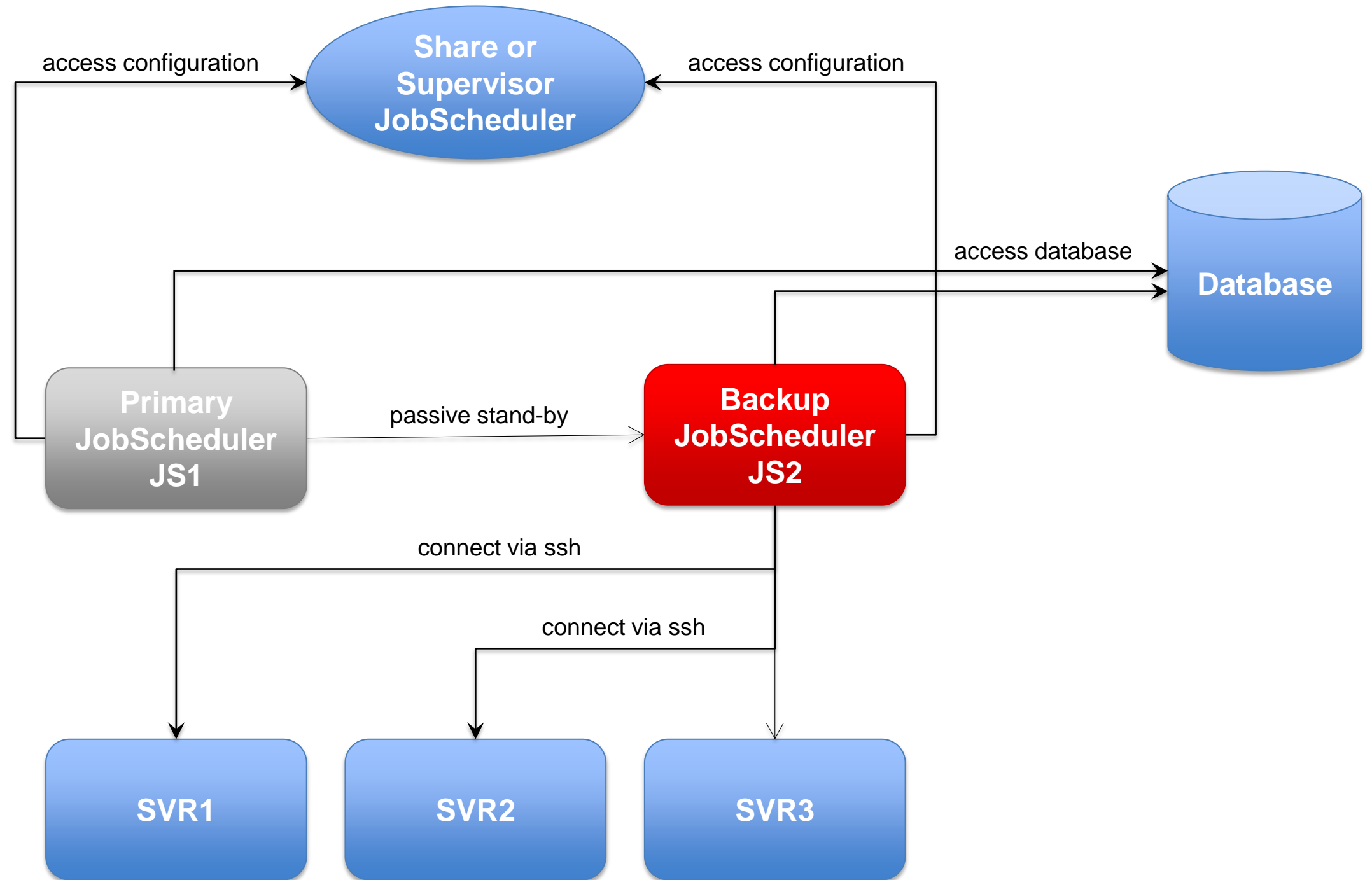
- Primary and Backup JobScheduler Master both use the same database
- Backup JobScheduler Master is active after failure of Primary instance
- Primary instance operates in stand-by mode
- All connections to servers use the SSH protocol

SSH Connections*JITL Jobs*

- Requires a JVM per task
- Memory resources

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Active Cluster

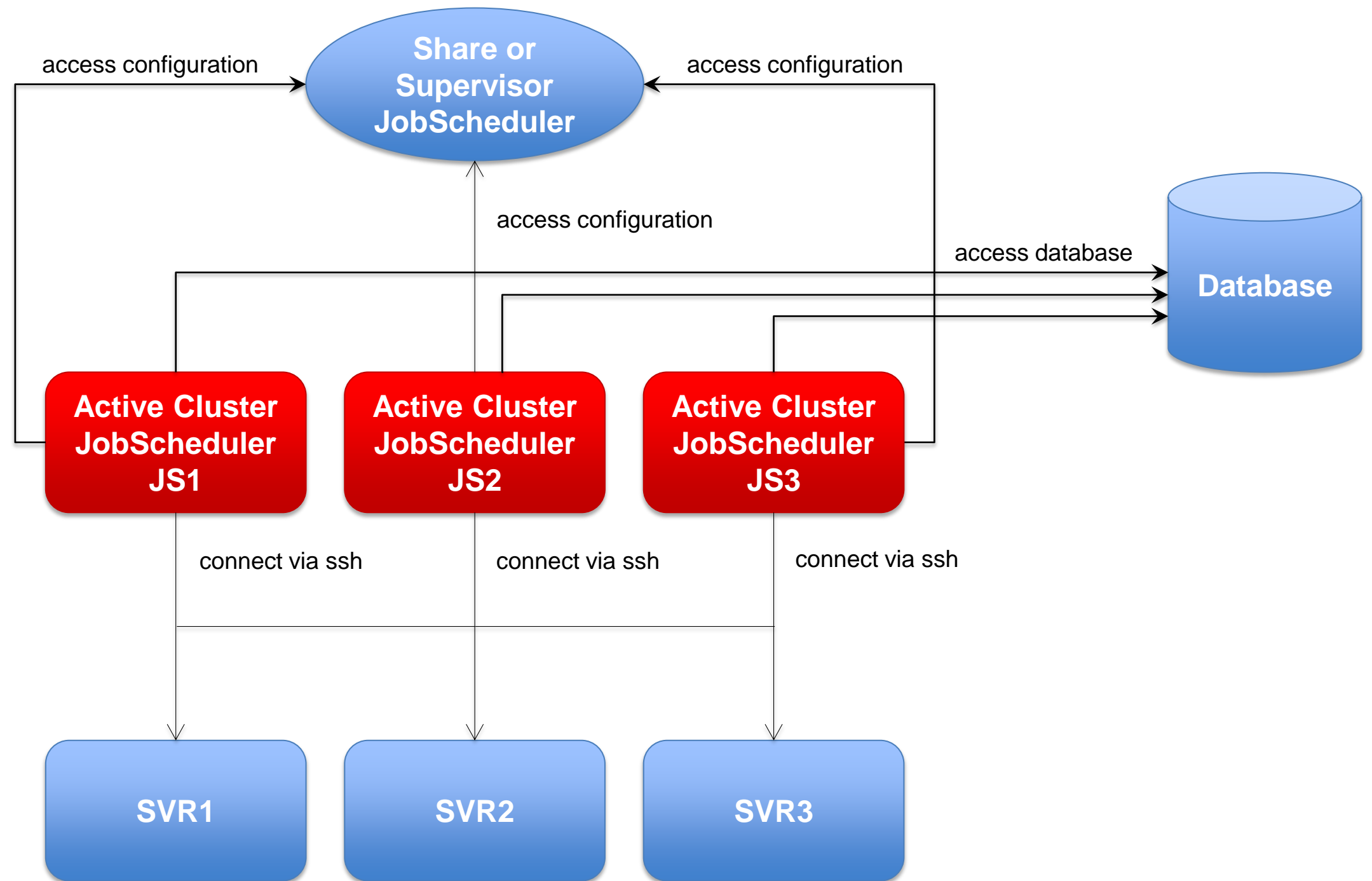
- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All connections to servers use the ssh protocol

SSH Connections*JITL Jobs*

- Requires a JVM per task
- Memory resouces

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Active Cluster

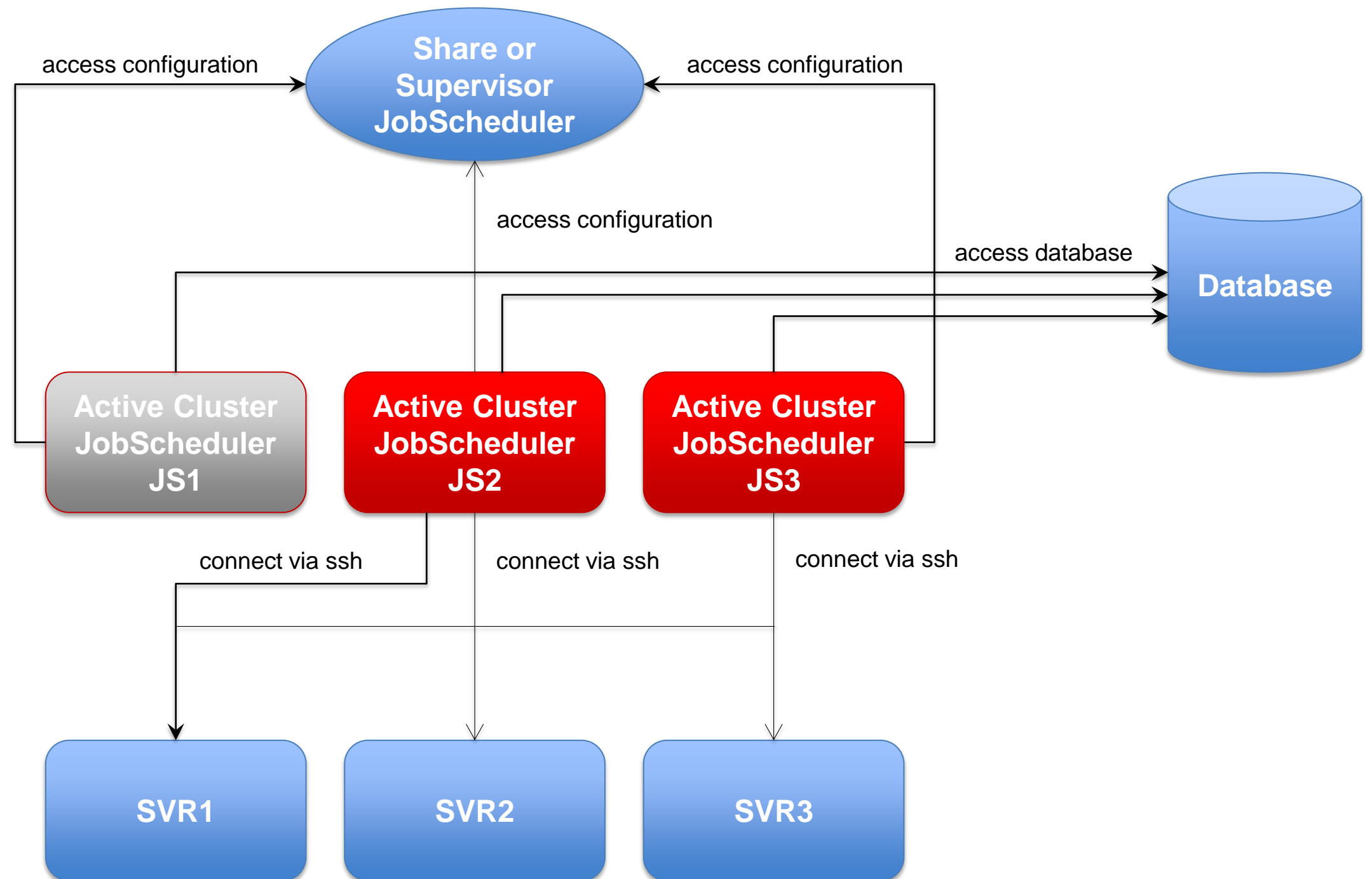
- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All connections to servers use the ssh protocol

SSH Connections*JITL Jobs*

- Requires a JVM per task
- Memory resources

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system

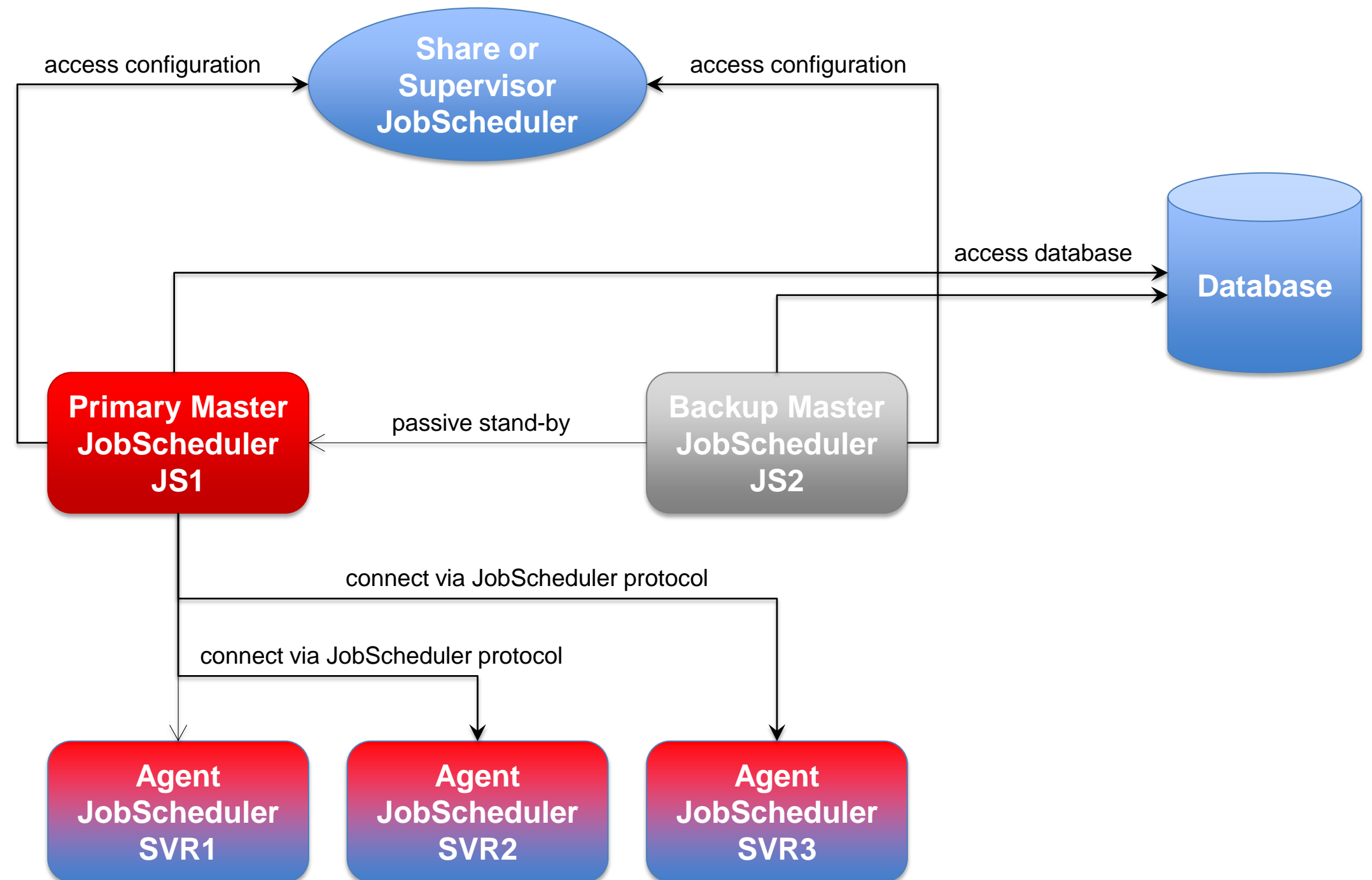


Master/Agent Passive Cluster

- Primary and Backup JobScheduler use the same database
- Primary JobScheduler is monitored by its Backup instance
- Backup instance operates in stand-by mode
- All Cluster instances use Agents to execute jobs on remote servers
- Connections to servers use the internal protocol

Job Execution

- Jobs are executed locally per JobScheduler Agent.
- No central resources required for job execution
- Pre-/post-processing
- Use of JITL Jobs or script files with parameter substitution

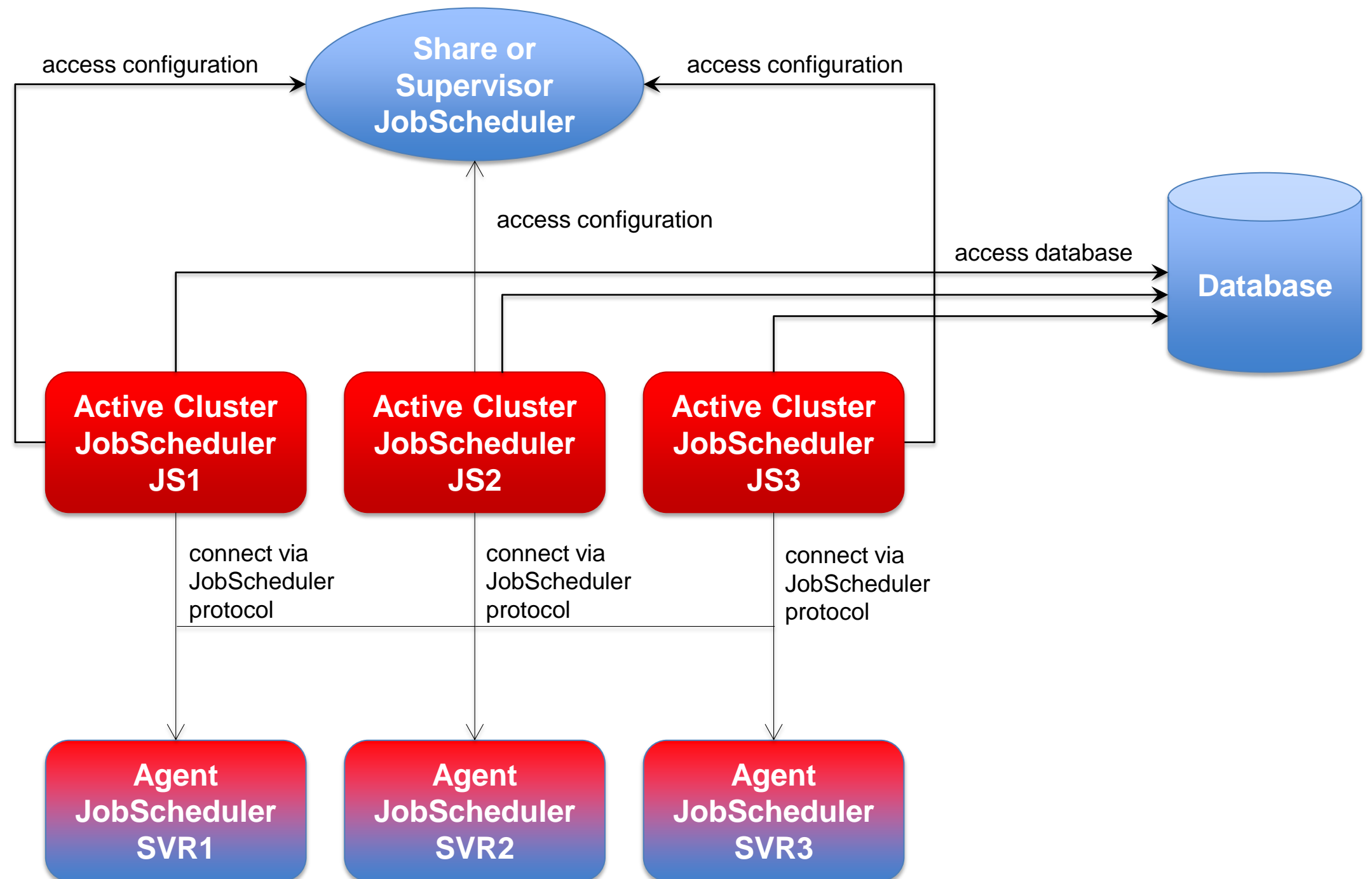


Master/Agent Active Cluster

- JobScheduler Cluster members use the same database
- JobSchedulers Cluster members share the workload of jobs
- All Instances operate in active mode
- All Cluster instances use Agents to execute jobs on remote servers

Job Execution

- Jobs are executed locally per JobScheduler Agent.
- No central resources required for job execution
- Pre-/post-processing
- Use of JITL Jobs or script files with parameter substitution

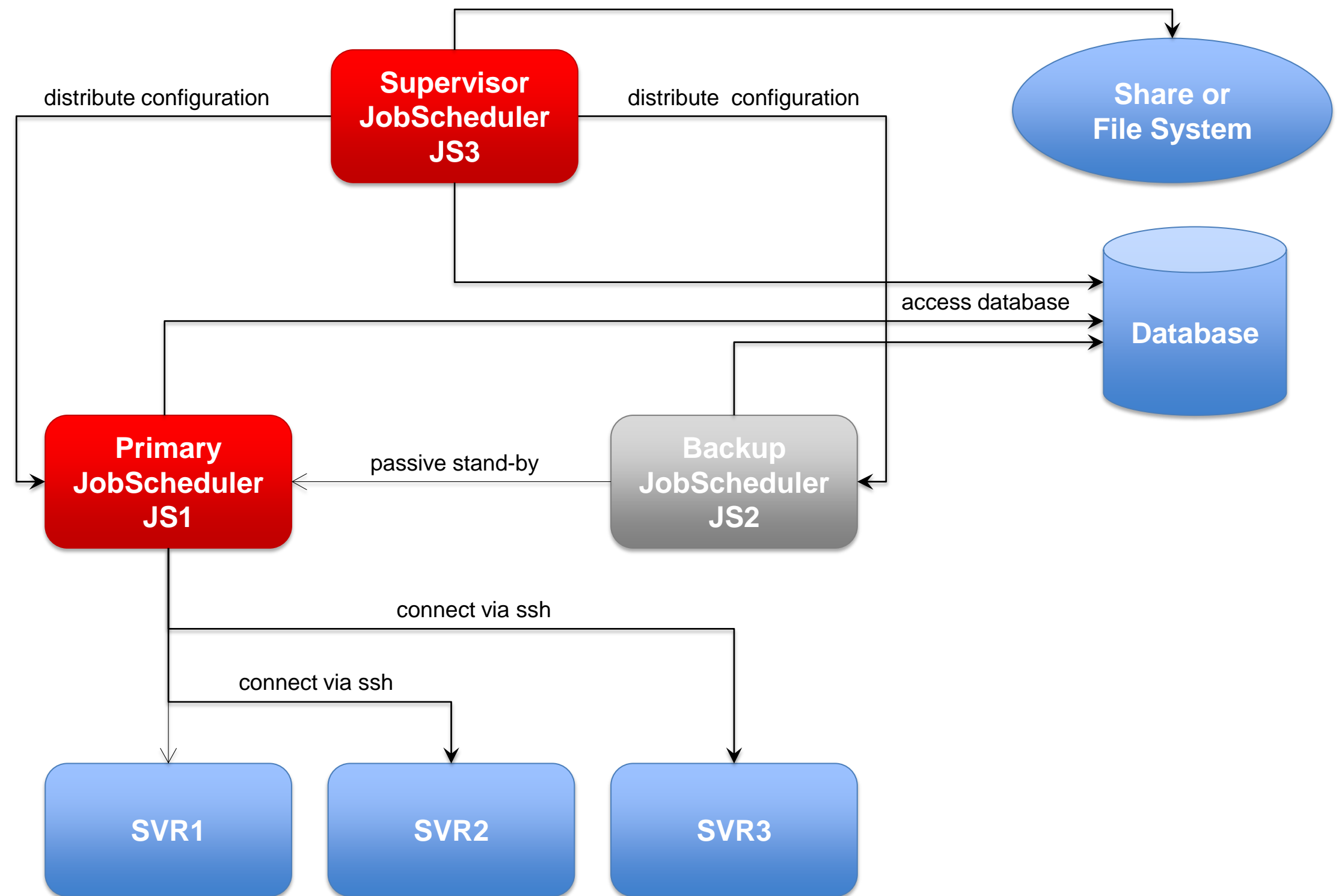


Passive Cluster

- Primary and Backup JobScheduler use the same database
- Primary JobScheduler is monitored by its Backup instance
- Backup instance operates in stand-by mode
- All connections to servers use the ssh protocol

Supervisor JobScheduler

- Distribute configuration to Primary and Backup JobScheduler instances

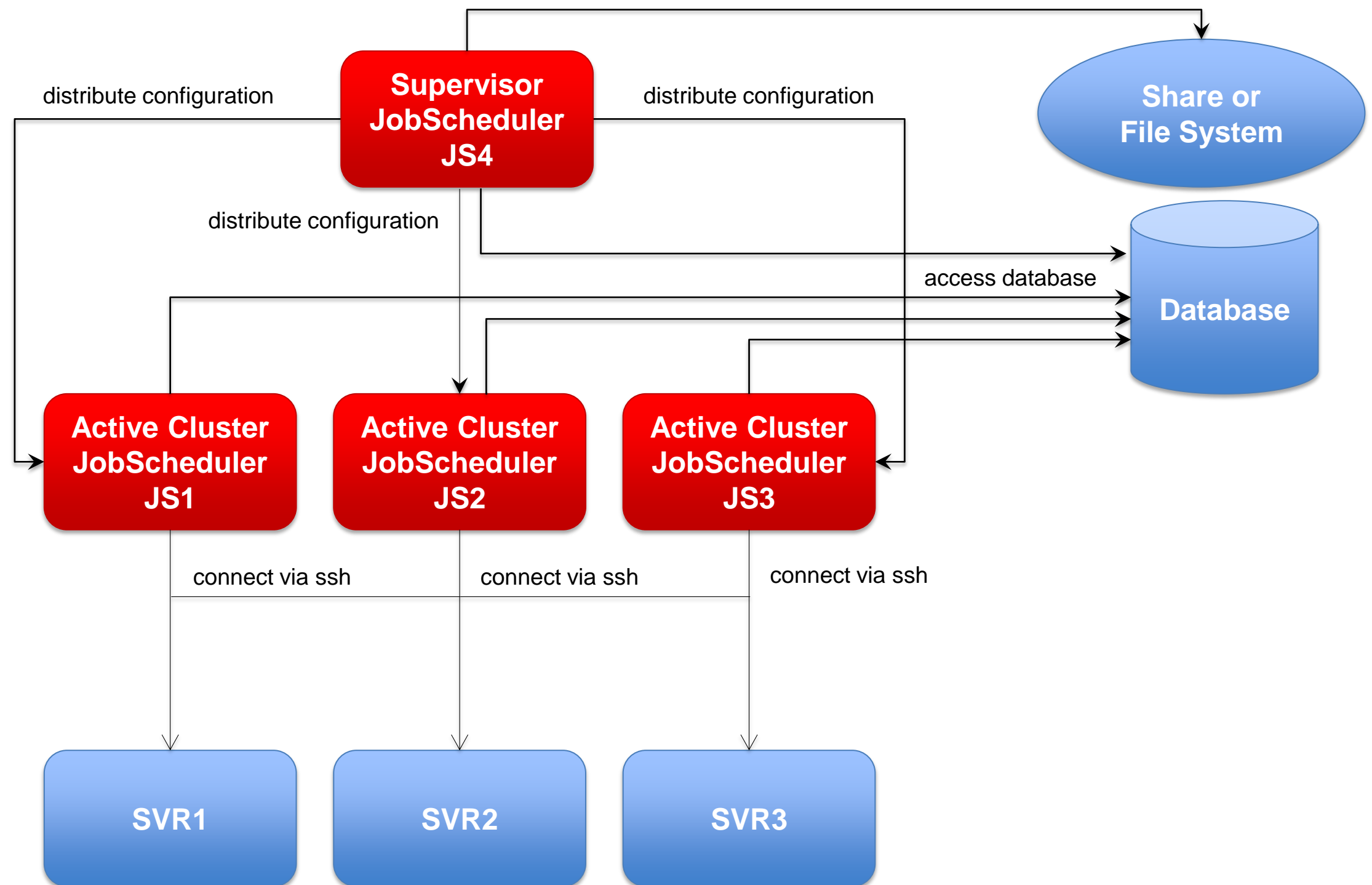


**Active Workload
JobScheduler Cluster**

- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All connections to servers use the ssh protocol

**Supervisor
JobScheduler**

- Distribute configuration to Cluster JobScheduler instances

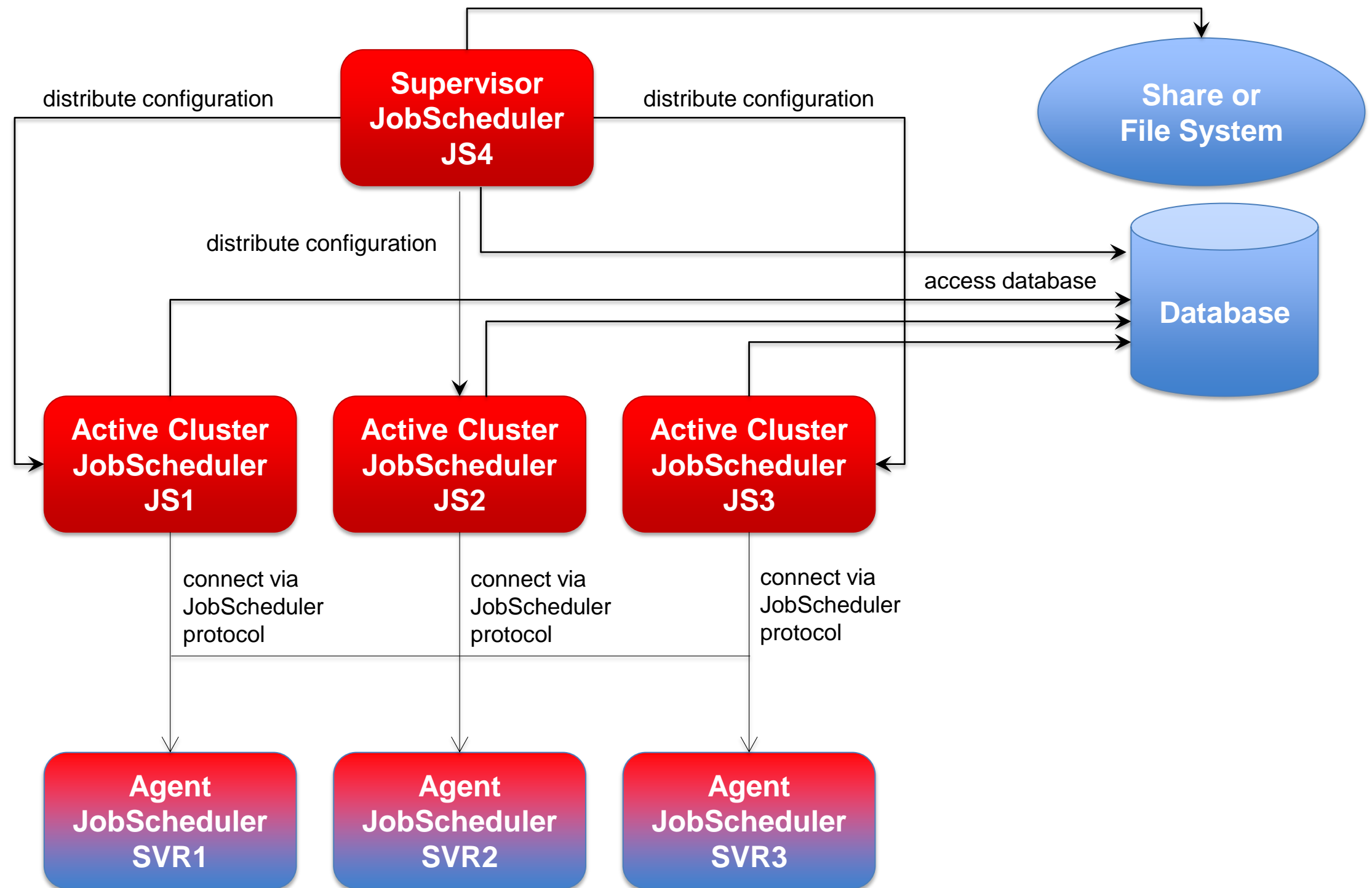


Master/Agent Active Cluster

- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All Cluster instances use Agents to execute jobs on remote servers

Supervisor JobScheduler

- Distribute configuration to Cluster JobScheduler instances

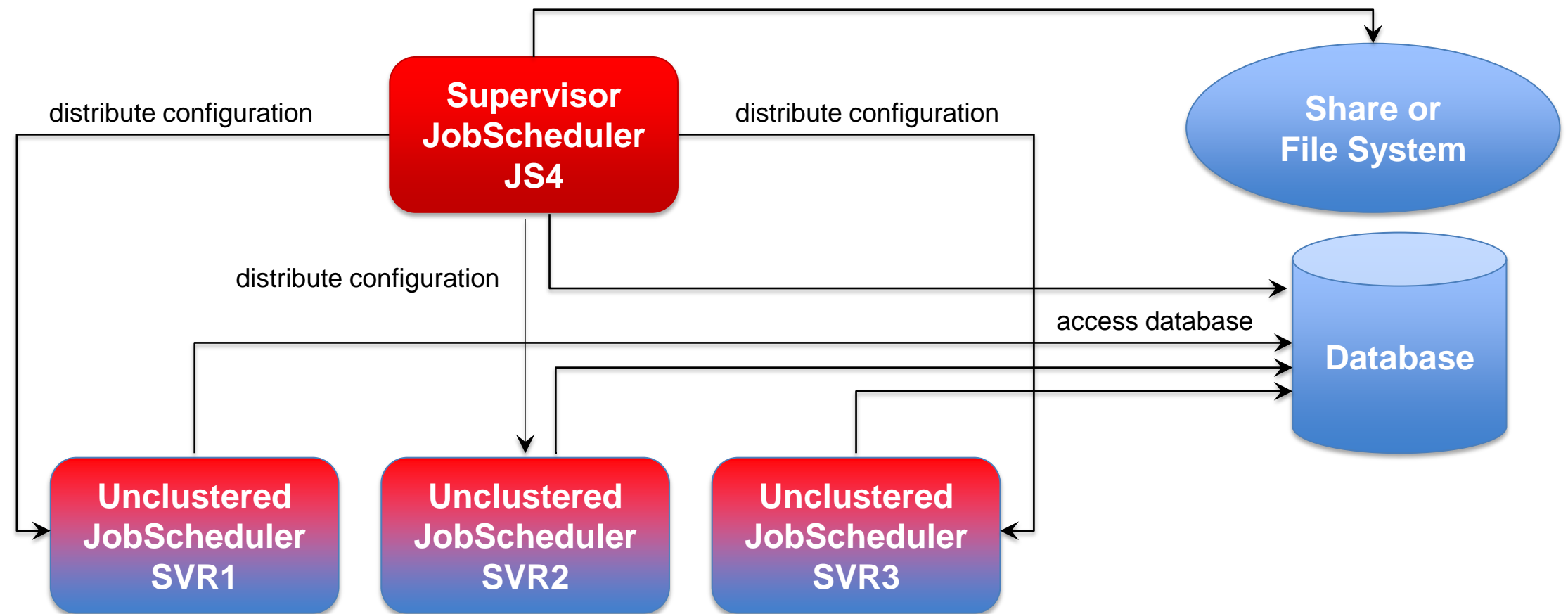


Unclustered JobSchedulers

- JobSchedulers use the same database
- JobSchedulers operate independently from each other
- All Instances operate in active mode

Supervisor JobScheduler

- Distribute configuration to JobScheduler instances





Questions?
Comments?
Feedback?

Software- und
Organisations-
Service GmbH

Giesebrechtstr. 15
D-10629 Berlin

info@sos-berlin.com
<http://www.sos-berlin.com>