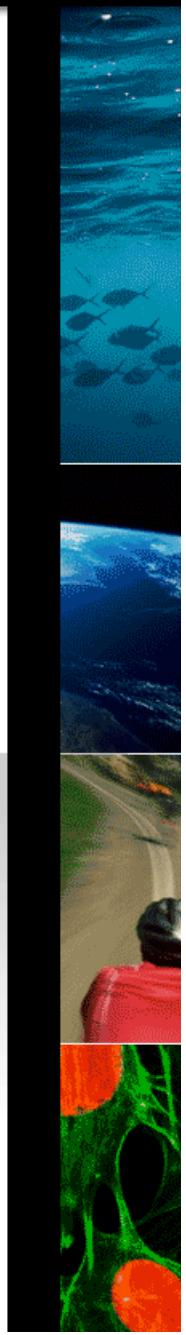
The Supercomputer Company

ALPS
Application Level Placement Scheduler

Michael Karo
mek@cray.com

CUG 2006



ALPS Design Goals (1)

- Scalability
  - Thousands of OS instances and applications
- Efficiency
  - Maximize resource utilization
  - Minimize overhead
- Predictibility
  - Consistent performance of applications
  - Guaranteed resource availability
- Adaptability
  - Mask architecture specific details
  - Exploit architecture specific capabilities

ALPS Design Goals (2)

- Extensibility
  - Adaptable to future architectures
  - Simplified integration with workload management systems
- Maintainability
  - Reduce complexity
  - Separate policy and mechanism
- Availability
  - Recover quickly with minimal impact
  - Minimize single points of failure

ALPS Operating Environment

- Hardware
  - Multiple node types
  - Multiple processor types
  - Processor and memory variations
  - Distributed shared memory
- Software
  - Multiple parallel programming paradigms
  - Multiple OS instances
  - **Supported on Compute Node Linux only**
  - Multiple workload managers
  - Administration and configuration tools
  - Resource and event monitoring

ALPS Core Services

- Launch and cleanup applications
- Binary executable distribution
- Monitor and report application status
- Application ID assignment
- Resource reservation management
- Signal propagation
- Standard input, output, and error management
- Resource availability monitoring
- Provide external access to application processes for debugging and performance analysis

ALPS Features: Gang Scheduling

- ALPS manages context switching
  - Consistent across entire application
  - Configurable interval
- Allows short and long running jobs to coexist
- Supports configurable CPU oversubscription factor
- No support for memory oversubscription

ALPS Features: Reservations

- Maintain resource availability for batch jobs
- Support interactive users
- Reservation states:
  - FILED **- Request registered**
  - CONFIRMED **- Resources locked**
  - CLAIMED **- Resources in use**

ALPS Features: BASIL

- **B**atch & **A**pplication **S**cheduler **I**nterface **L**ayer
- Extensible XML-RPC implementation
- Open interface specification
- No proprietary APIs or libraries
- Third party vendors manage integration
- Three primary functions:
  - Inventory
  - Reservation creation
  - Reservation cancellation
- BASIL programmer's guide

ALPS Features: Fanout Tree

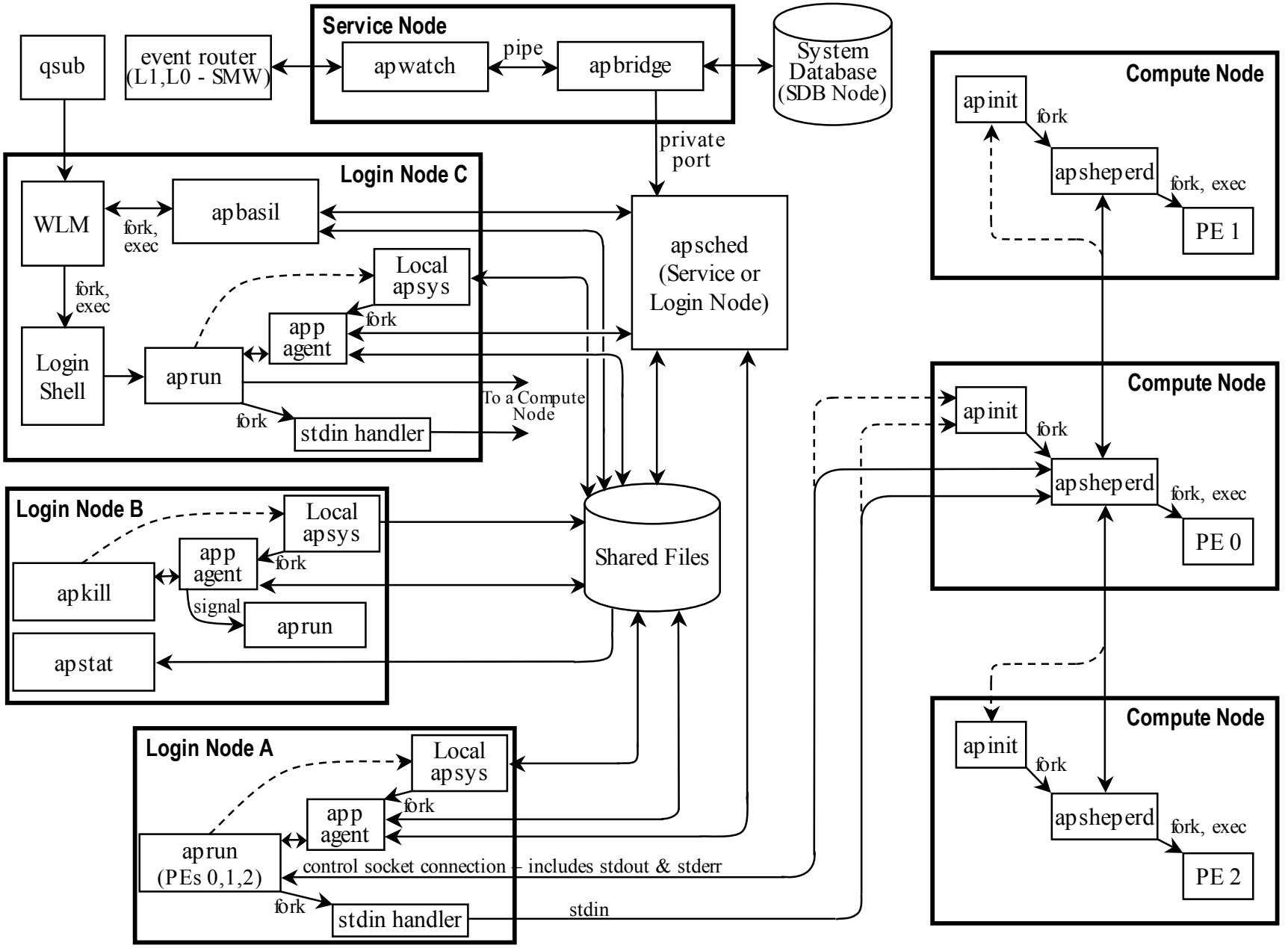
- Provides scalability
- Supports parallel operation
- Simulated broadcast on unicast network
- Configurable radix:

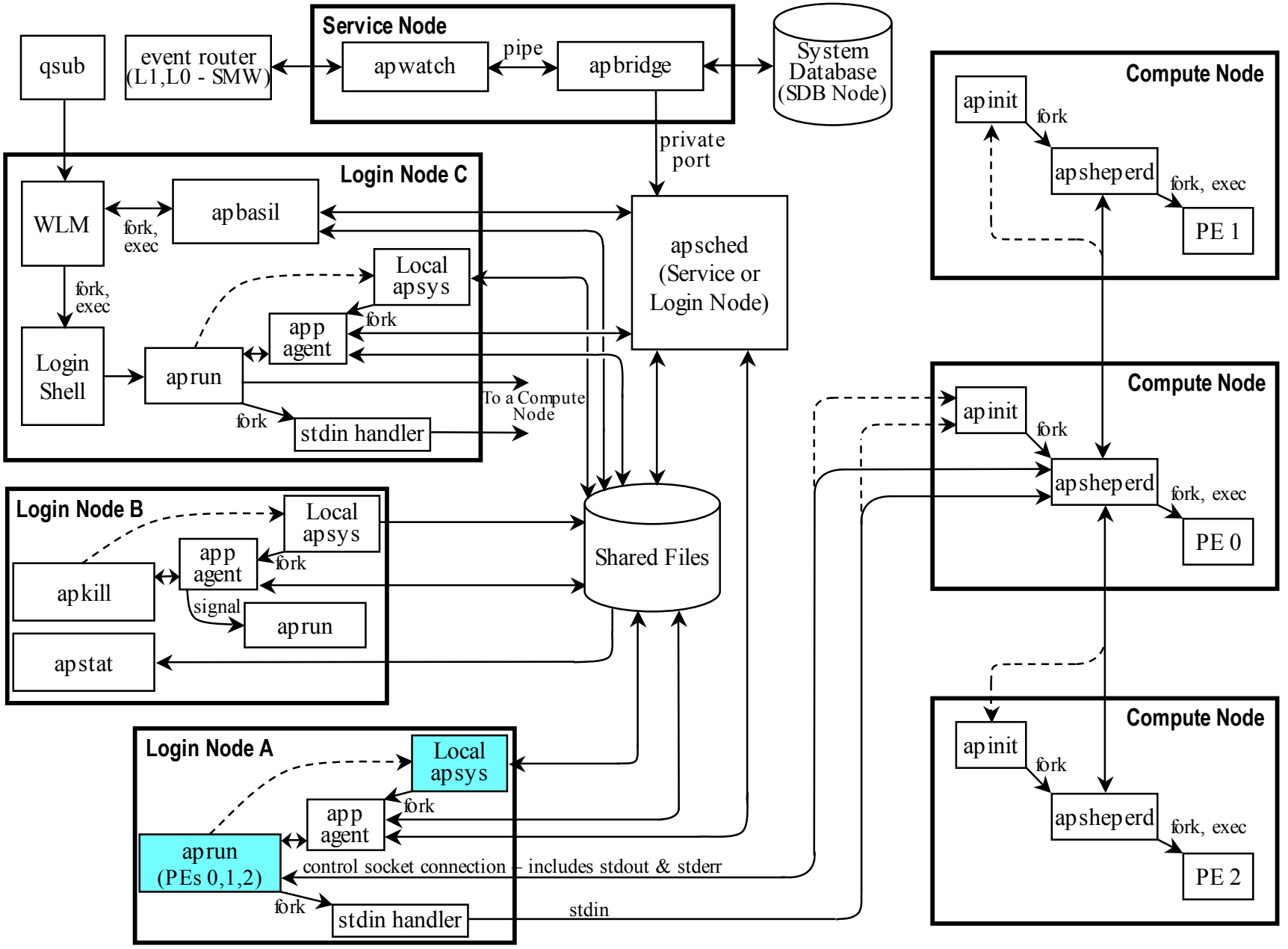
Tree Radix

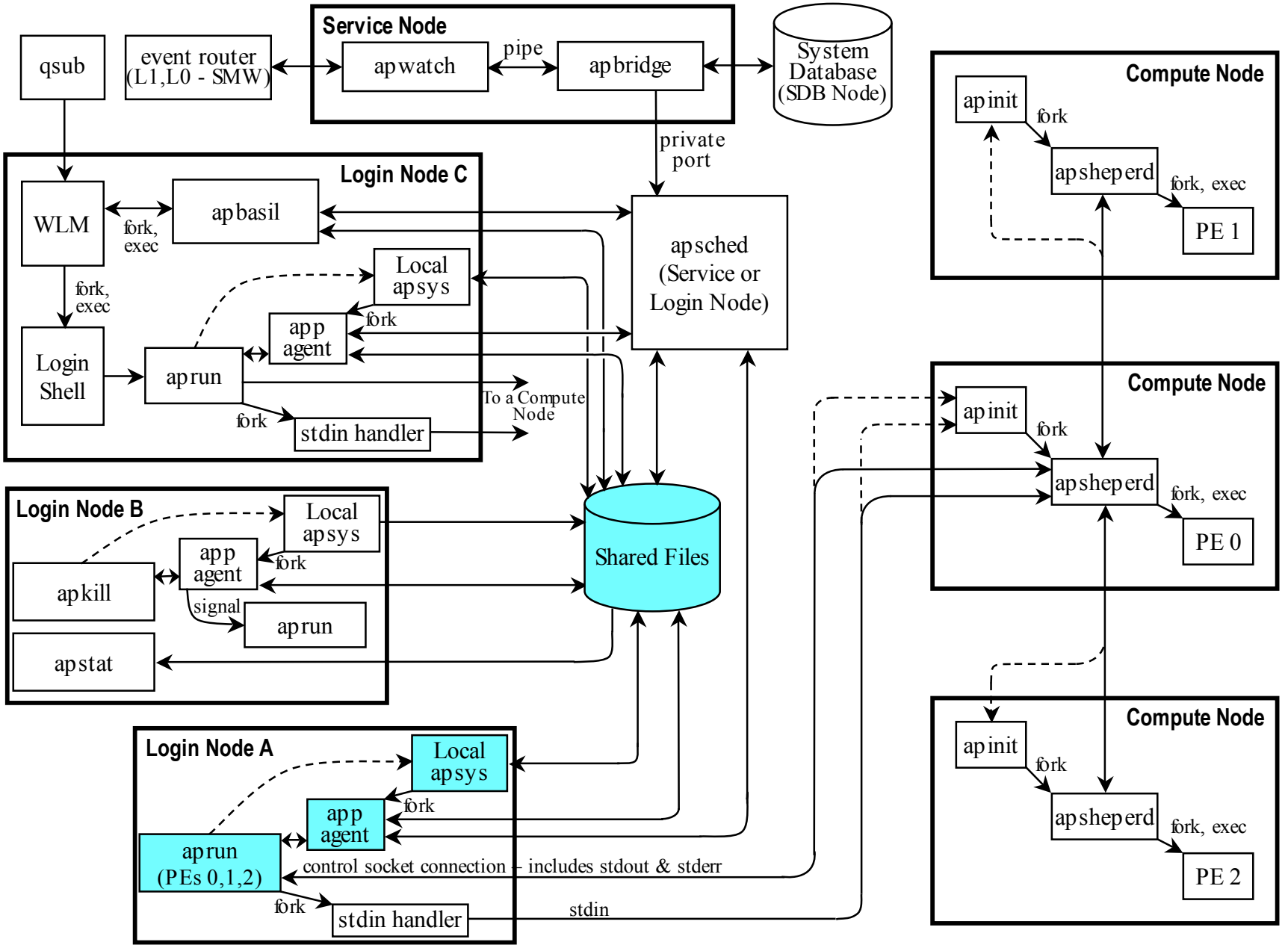
Tree Depth	2	4	8	16	32
1	1	1	1	1	1
2	3	5	9	17	33
3	7	21	73	273	1057
4	15	85	585	4369	33825
5	31	341	4681	69905	1082401

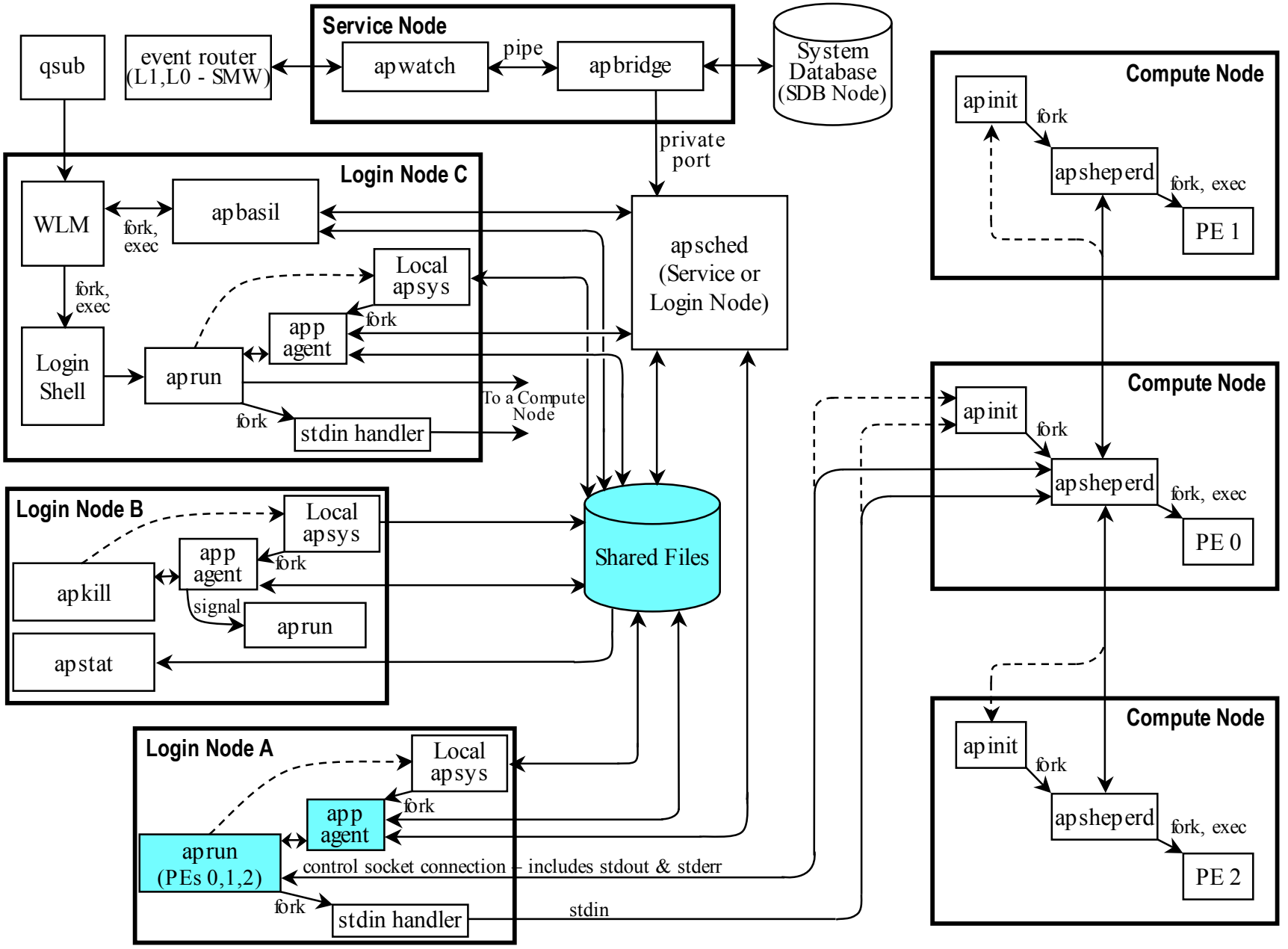
ALPS Components

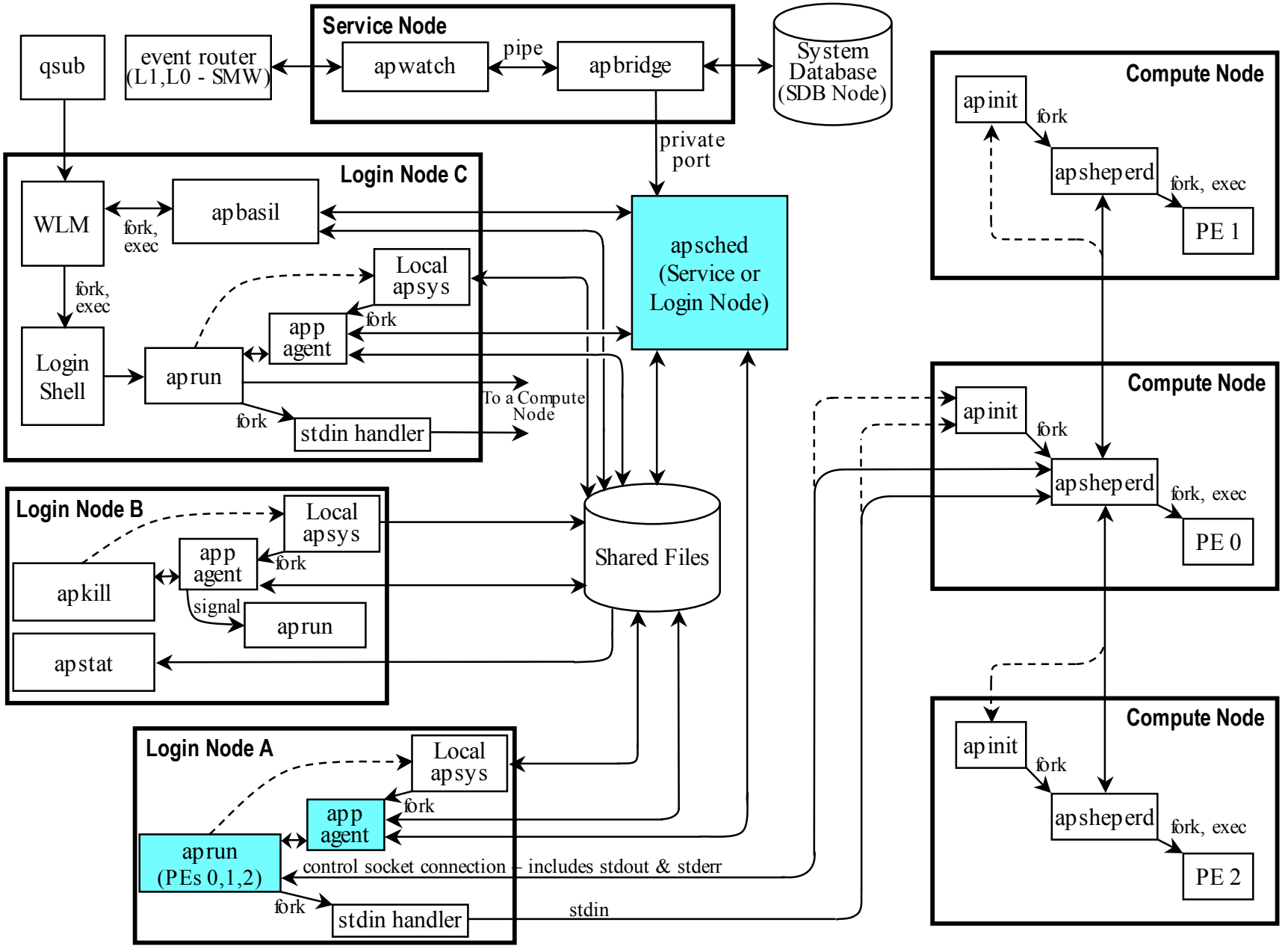
- Clients
  - aprun – Application submission
  - apstat – Application status
  - apkill – Signal delivery
  - apbasil – Workload manager interface
- Servers
  - apsys – Client interaction on login nodes
  - apinit – Process management on compute nodes
  - apsched – Reservations and placement
  - apbridge – System data collection
  - apwatch – Event monitoring

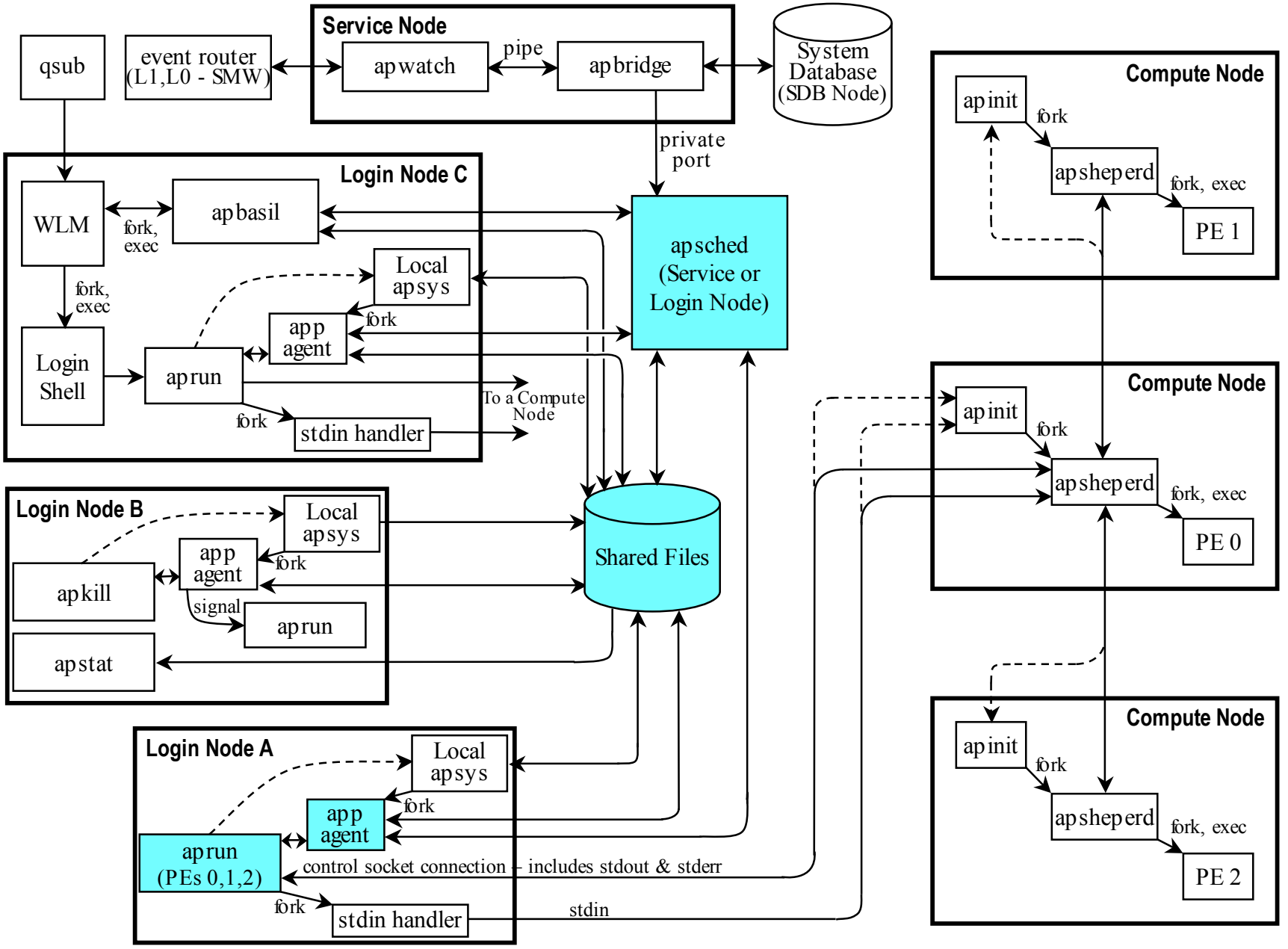


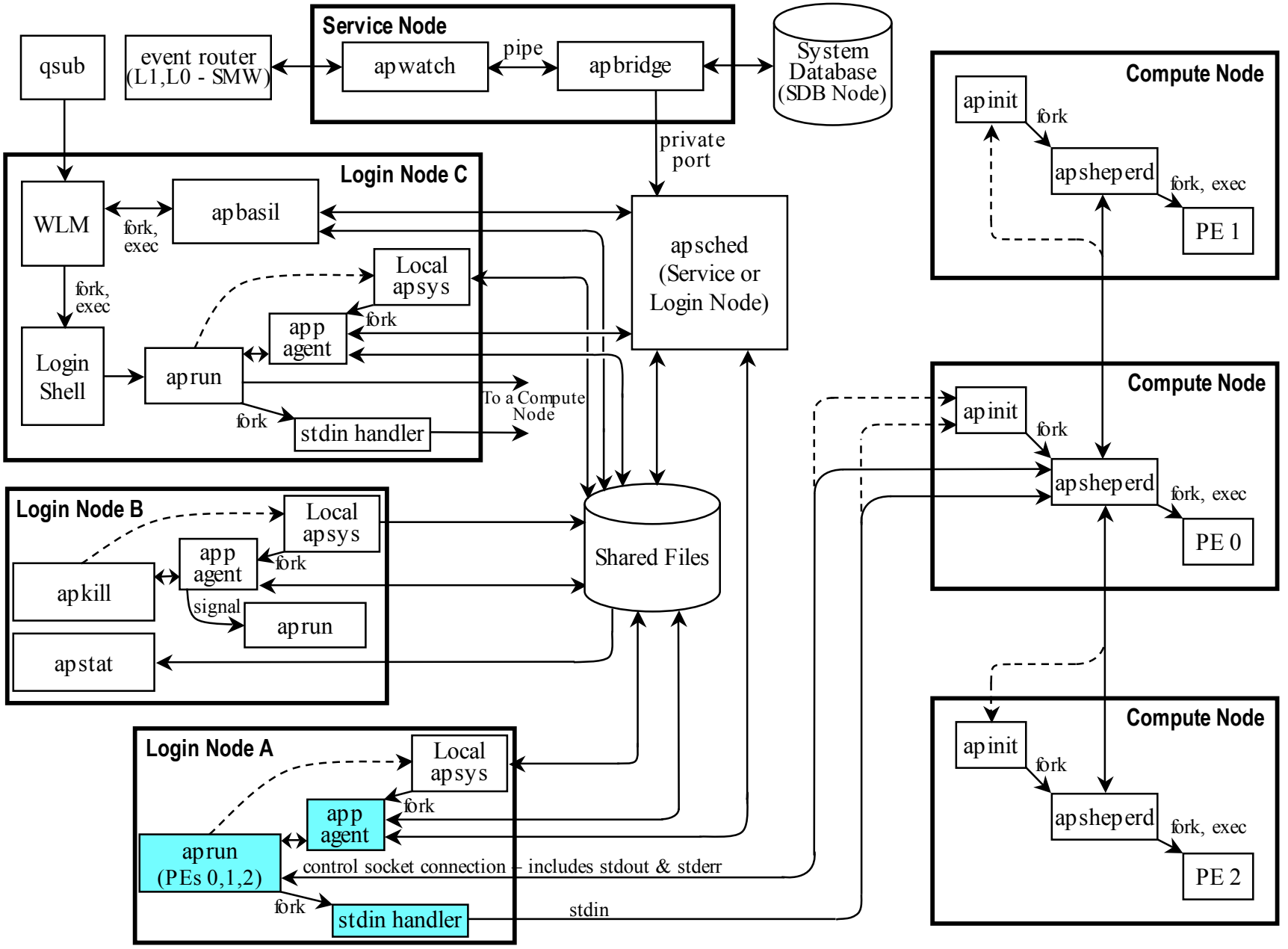


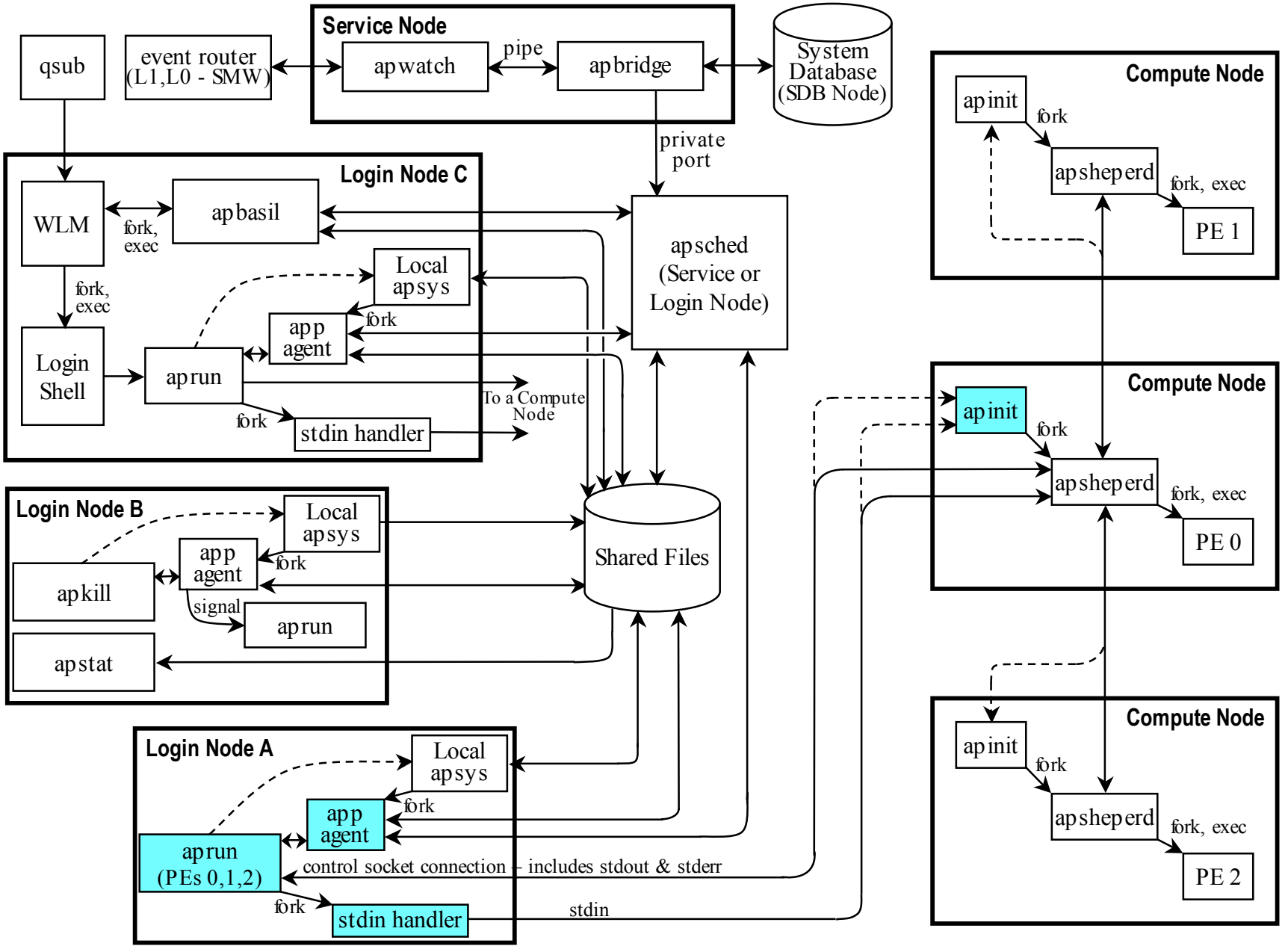


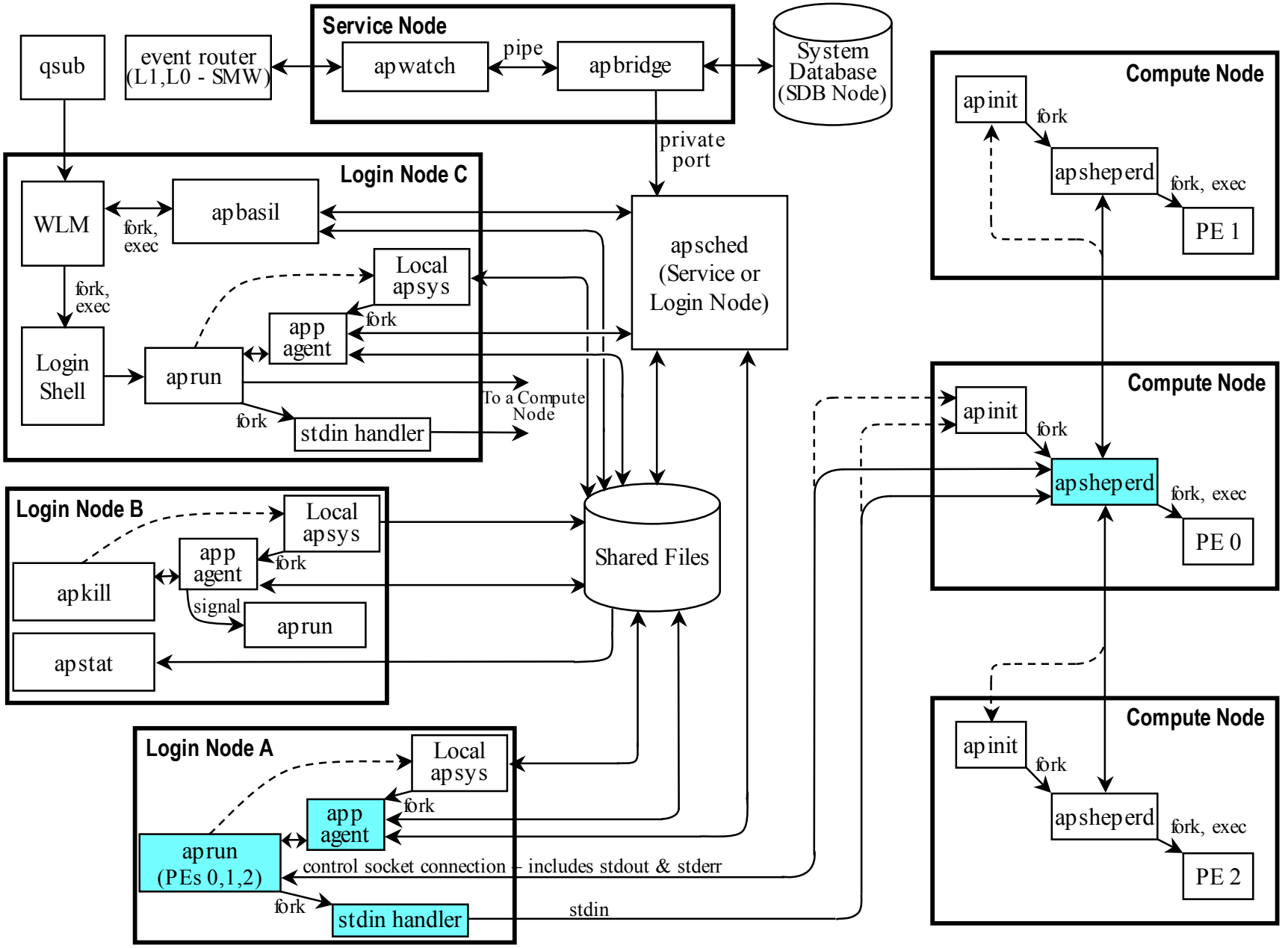


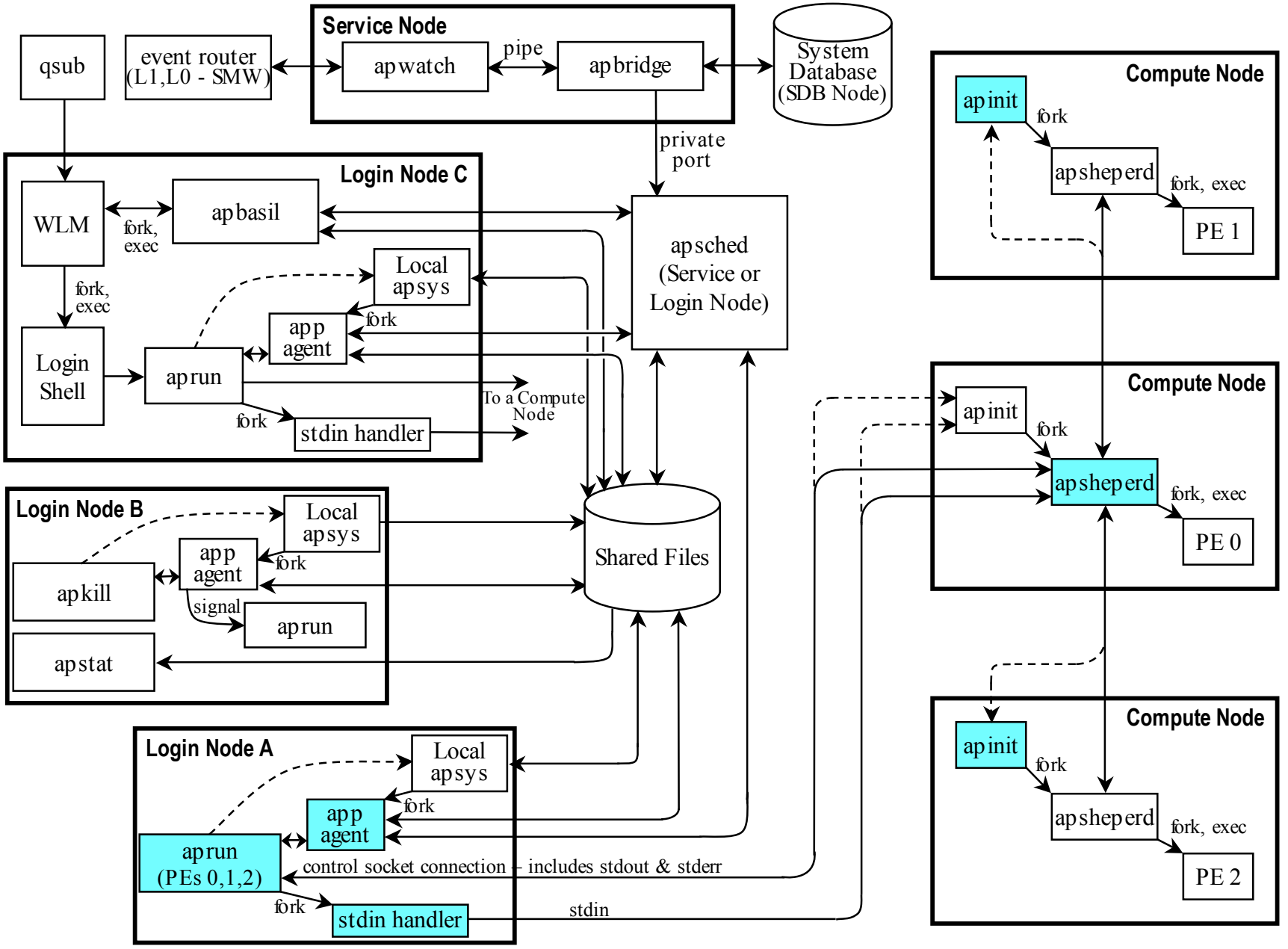


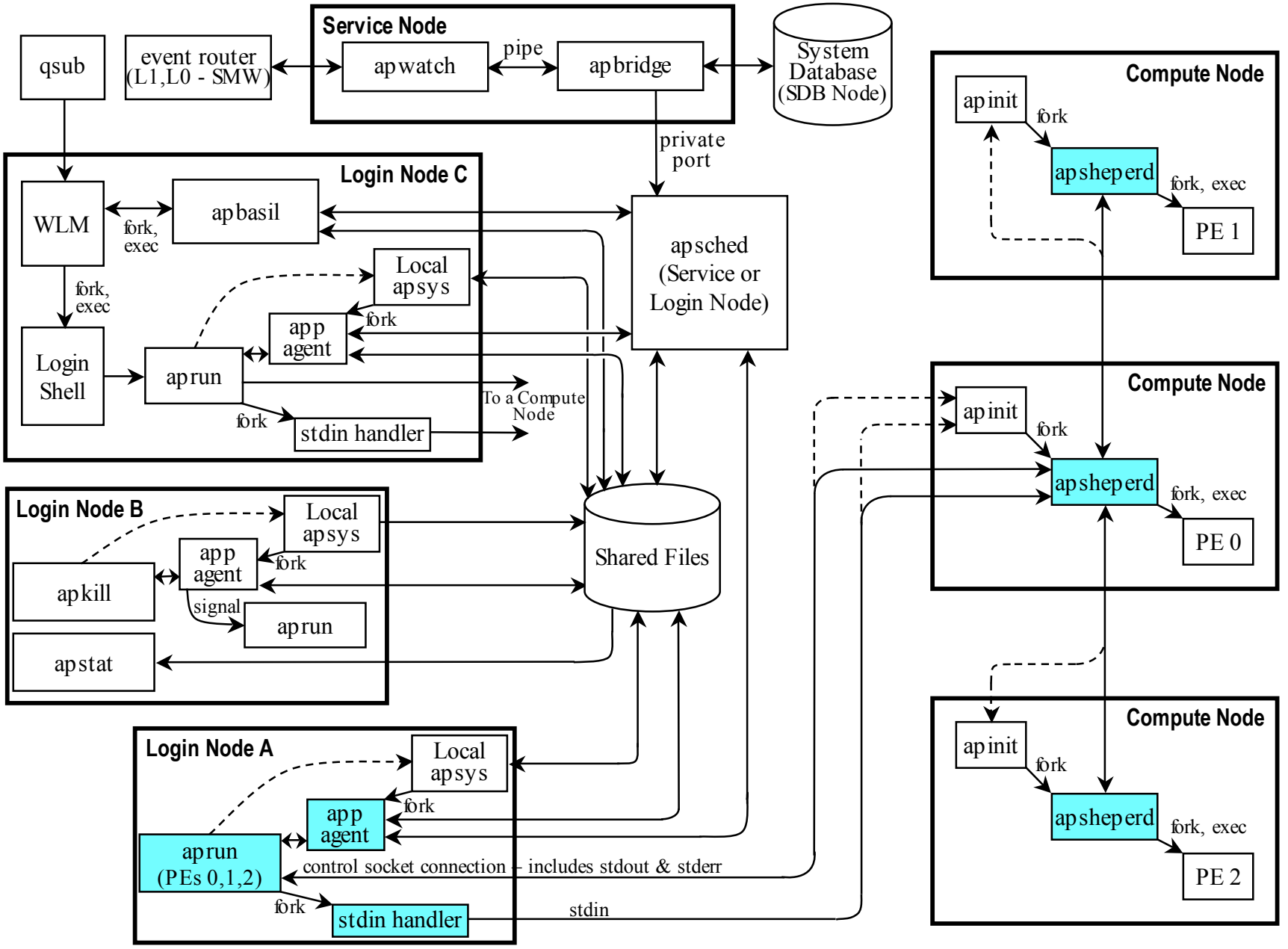


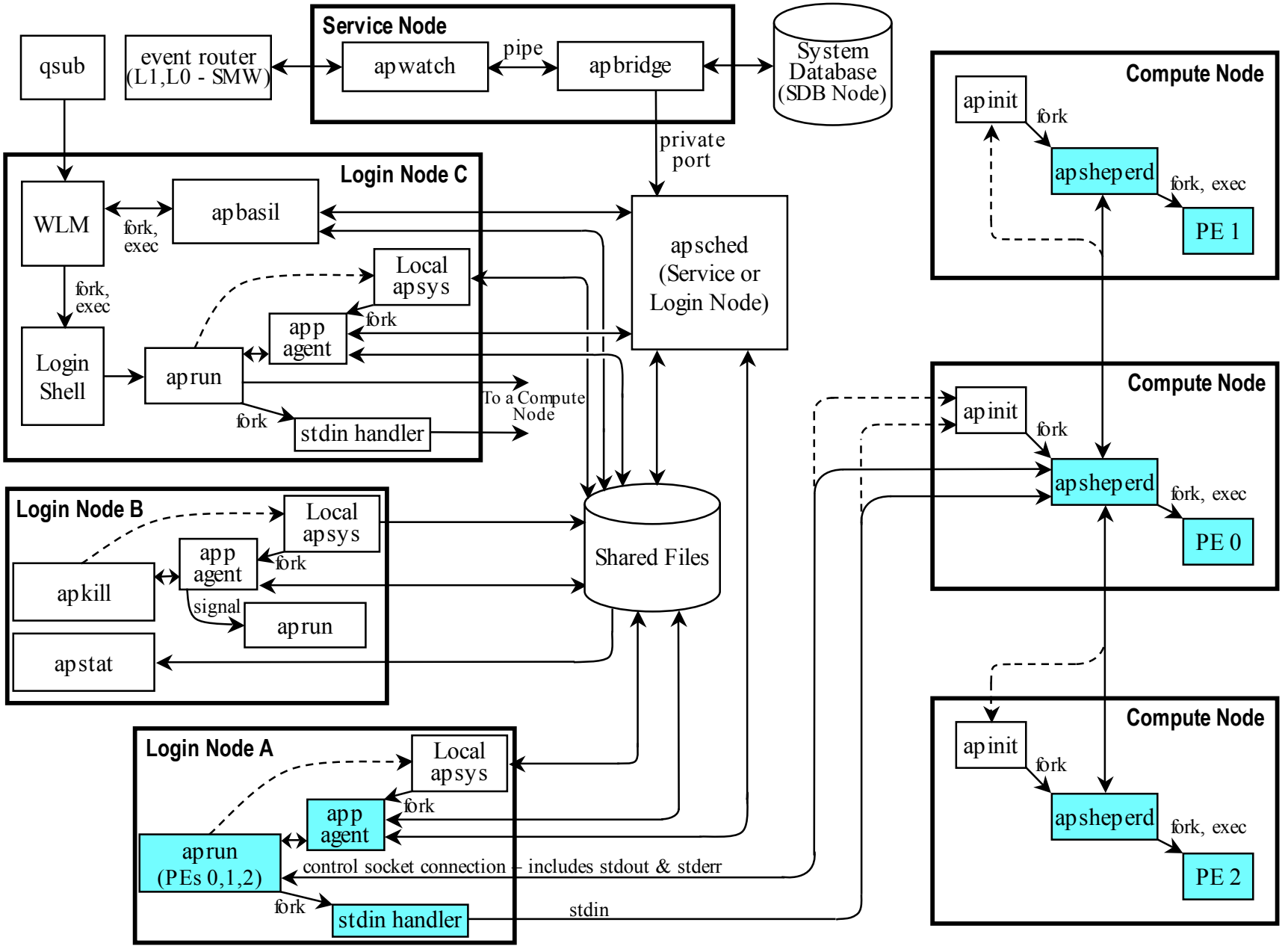


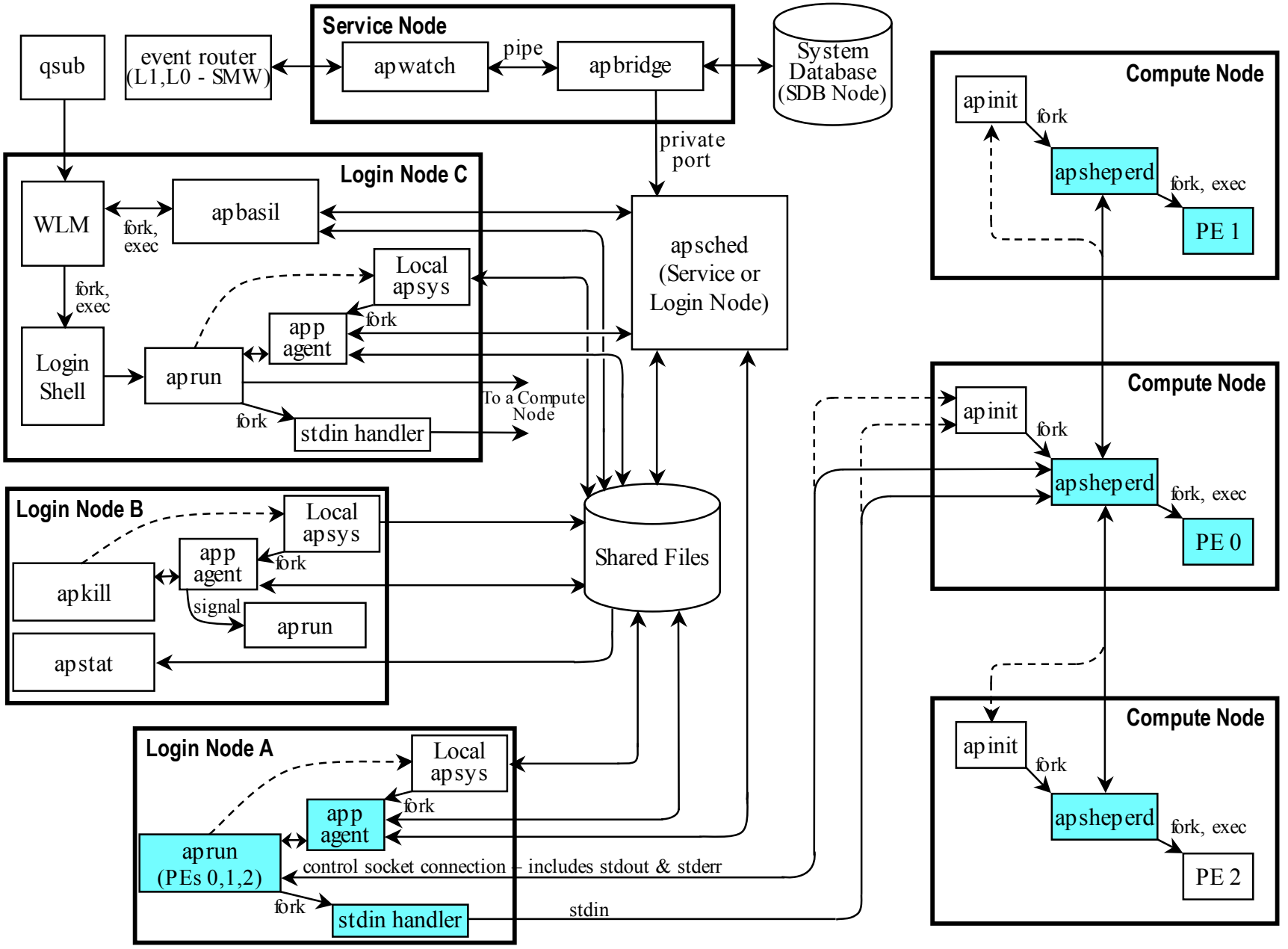


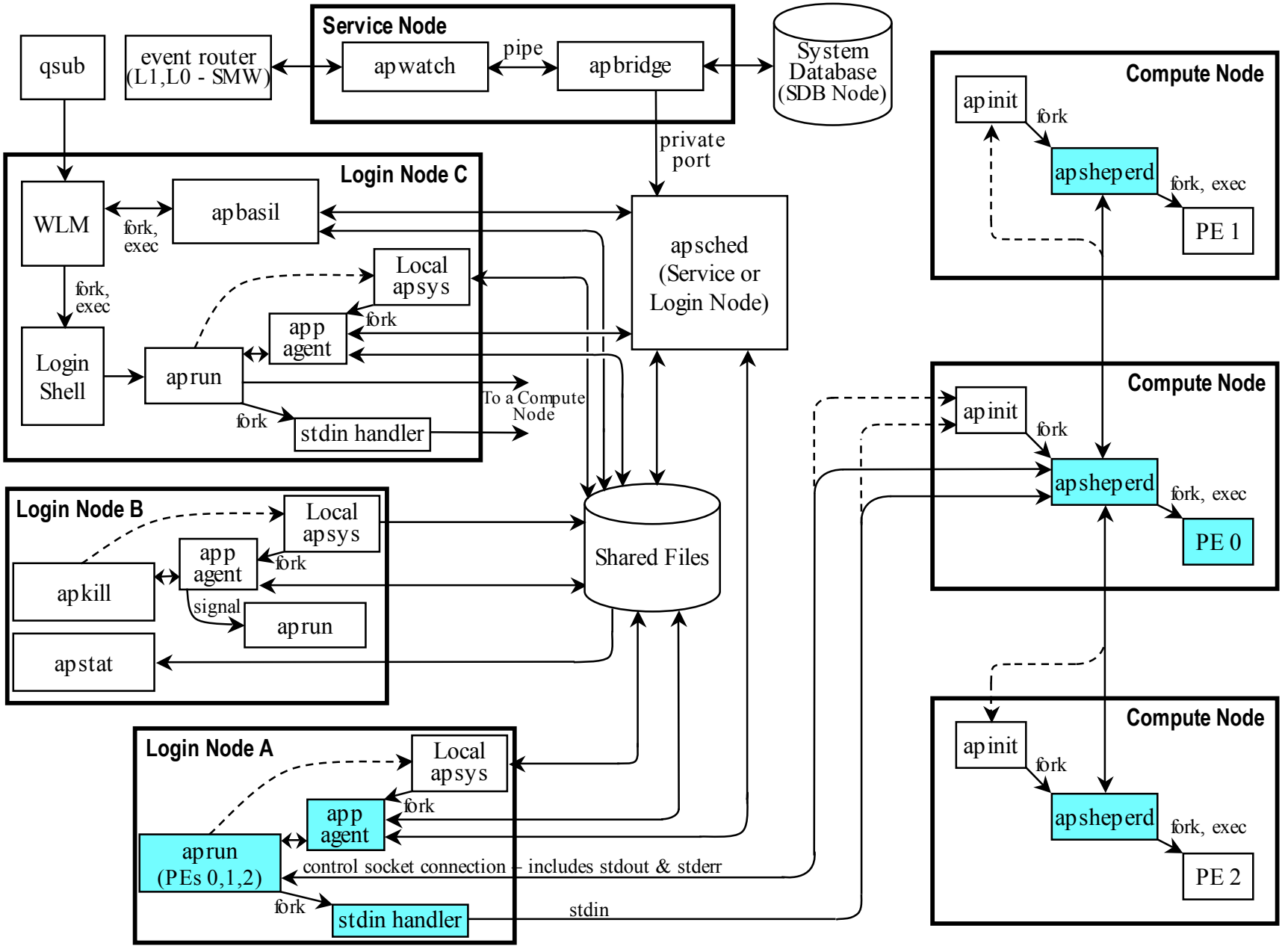


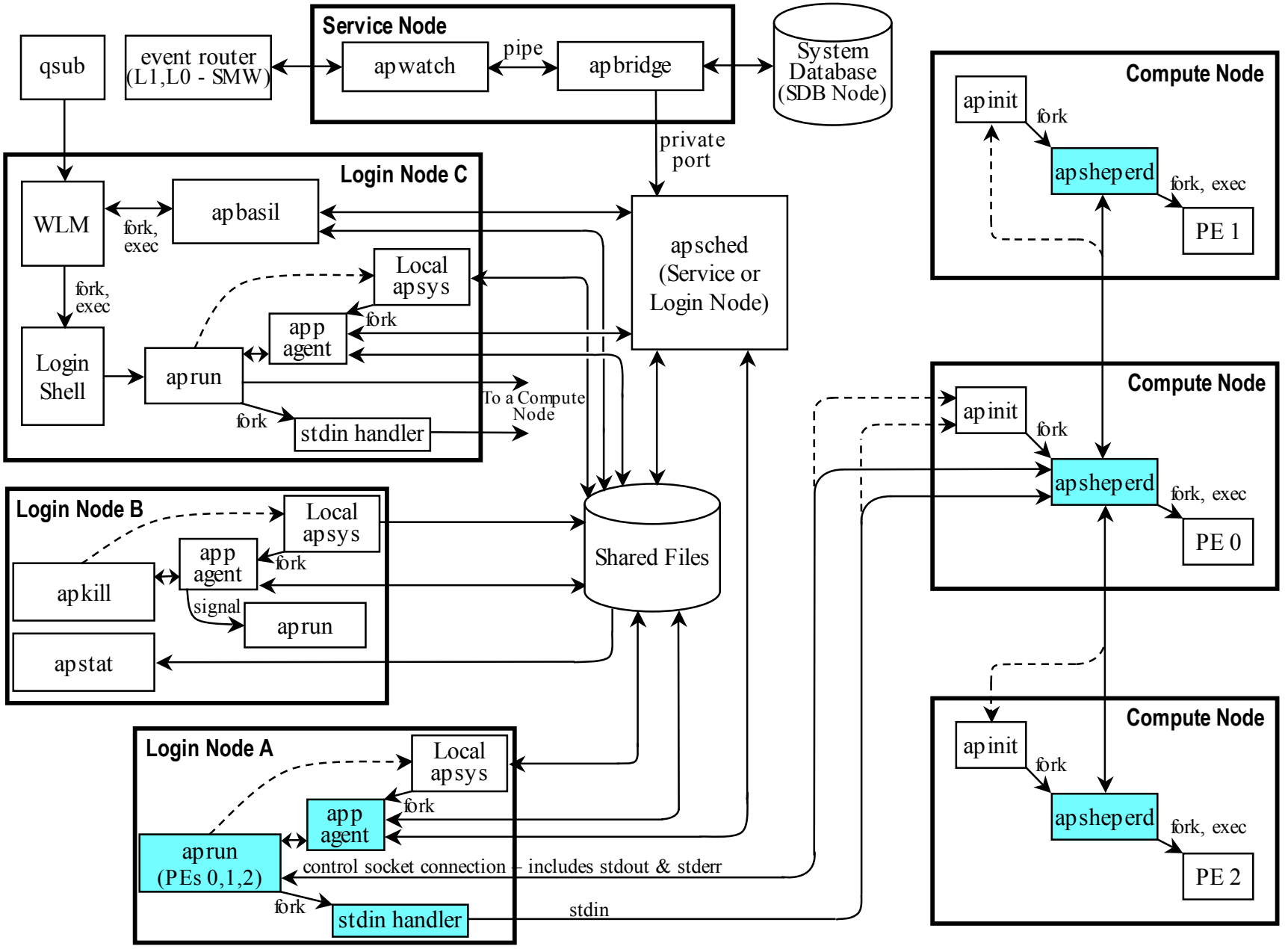


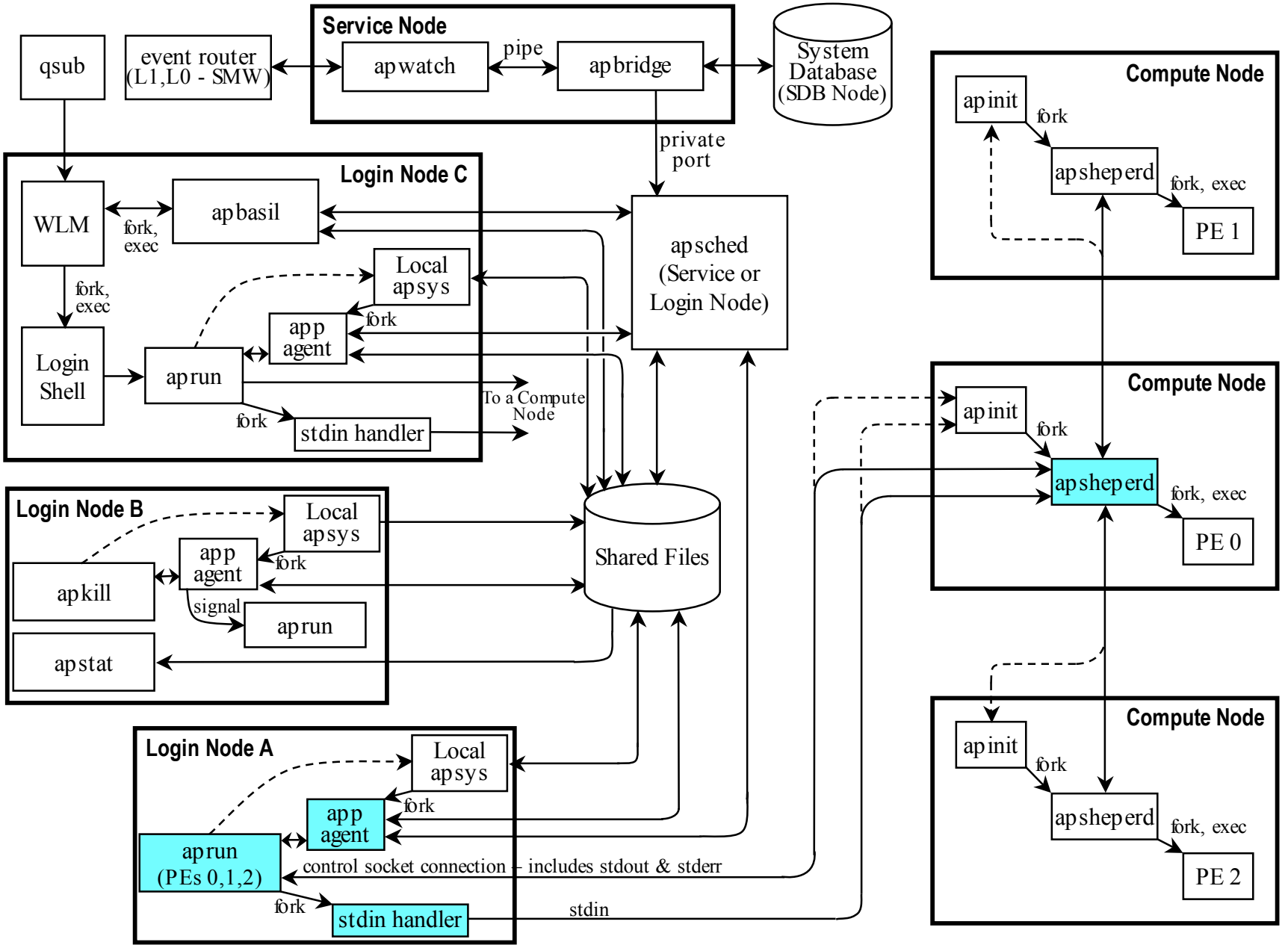


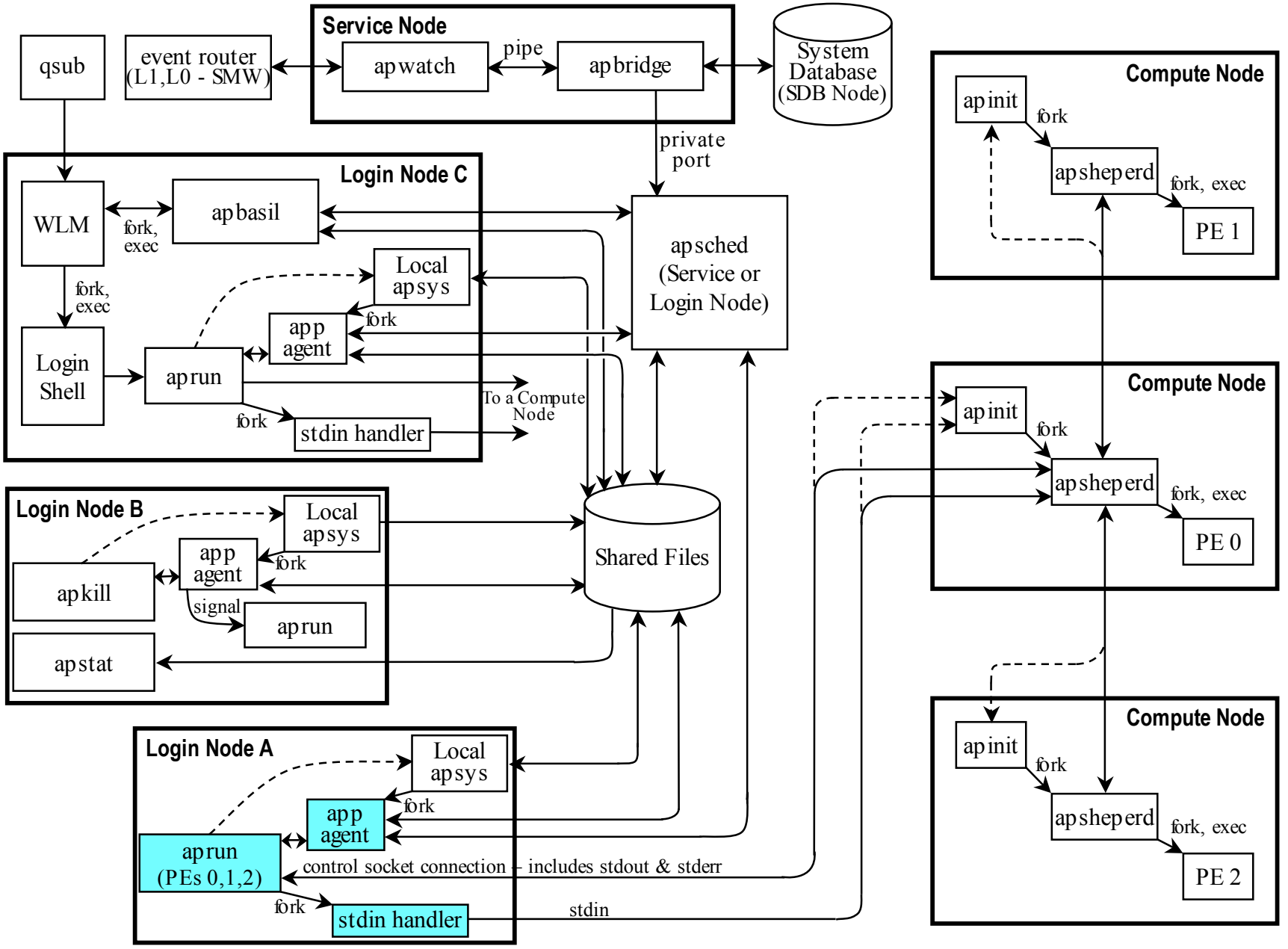


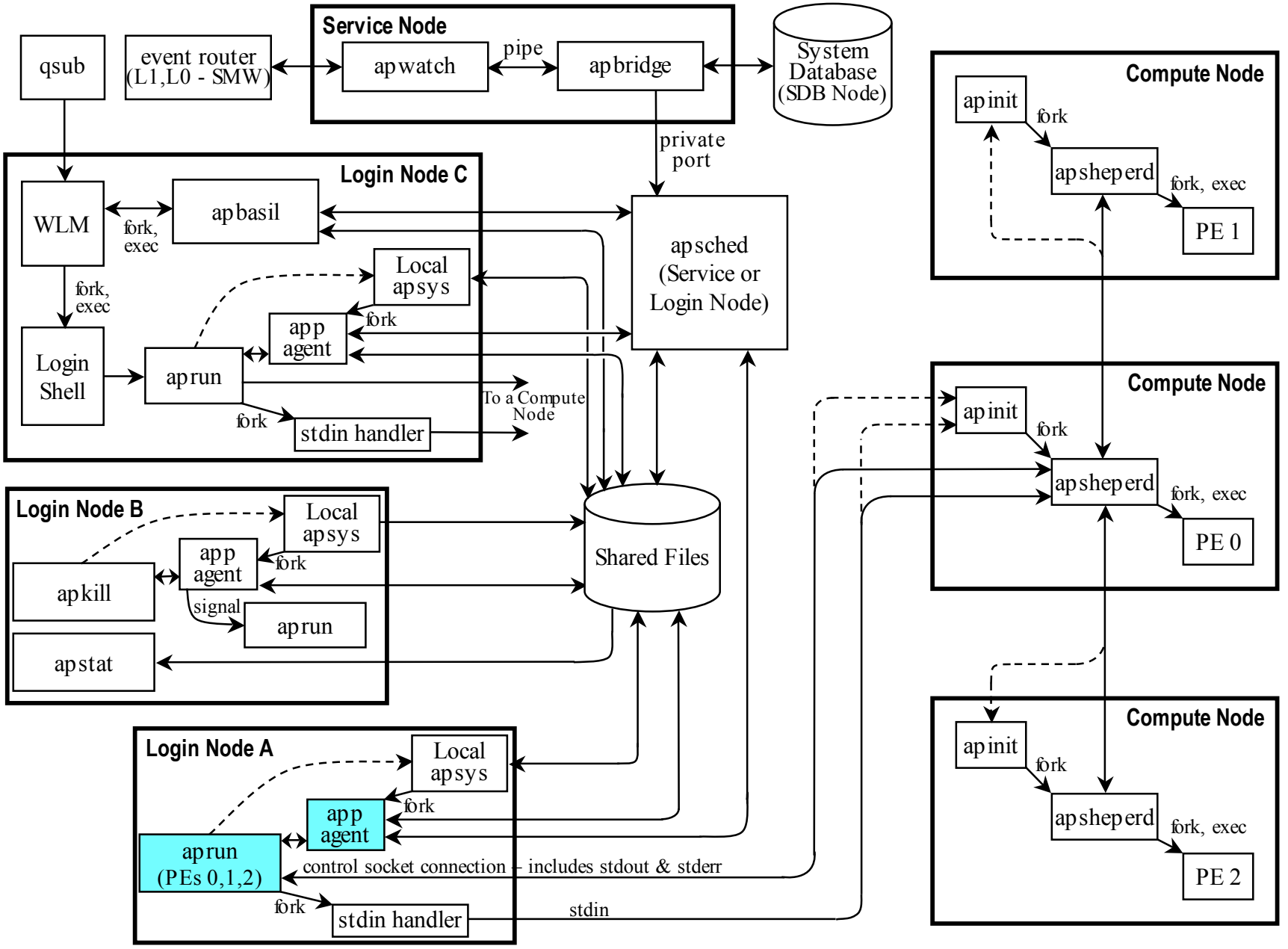


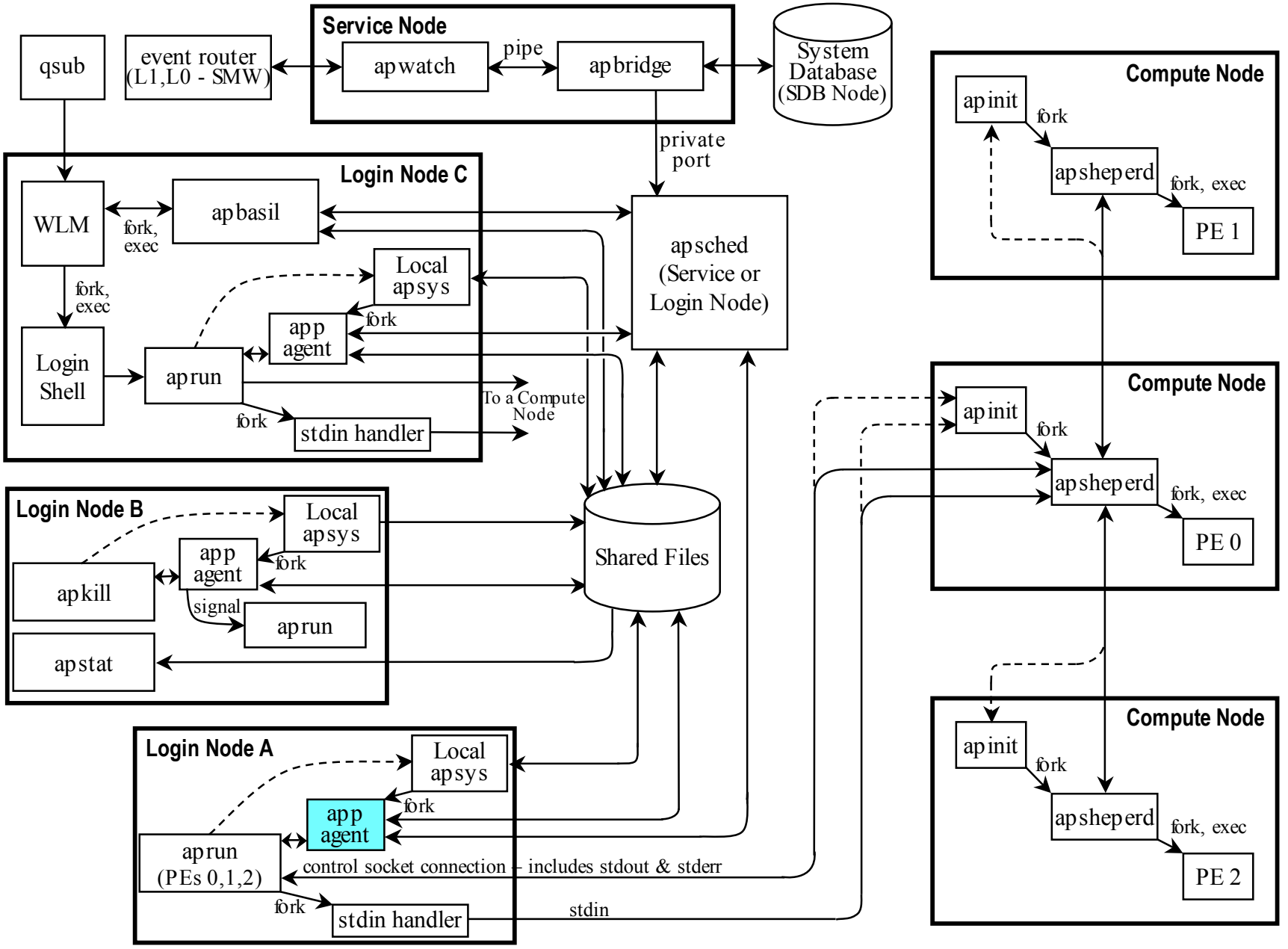


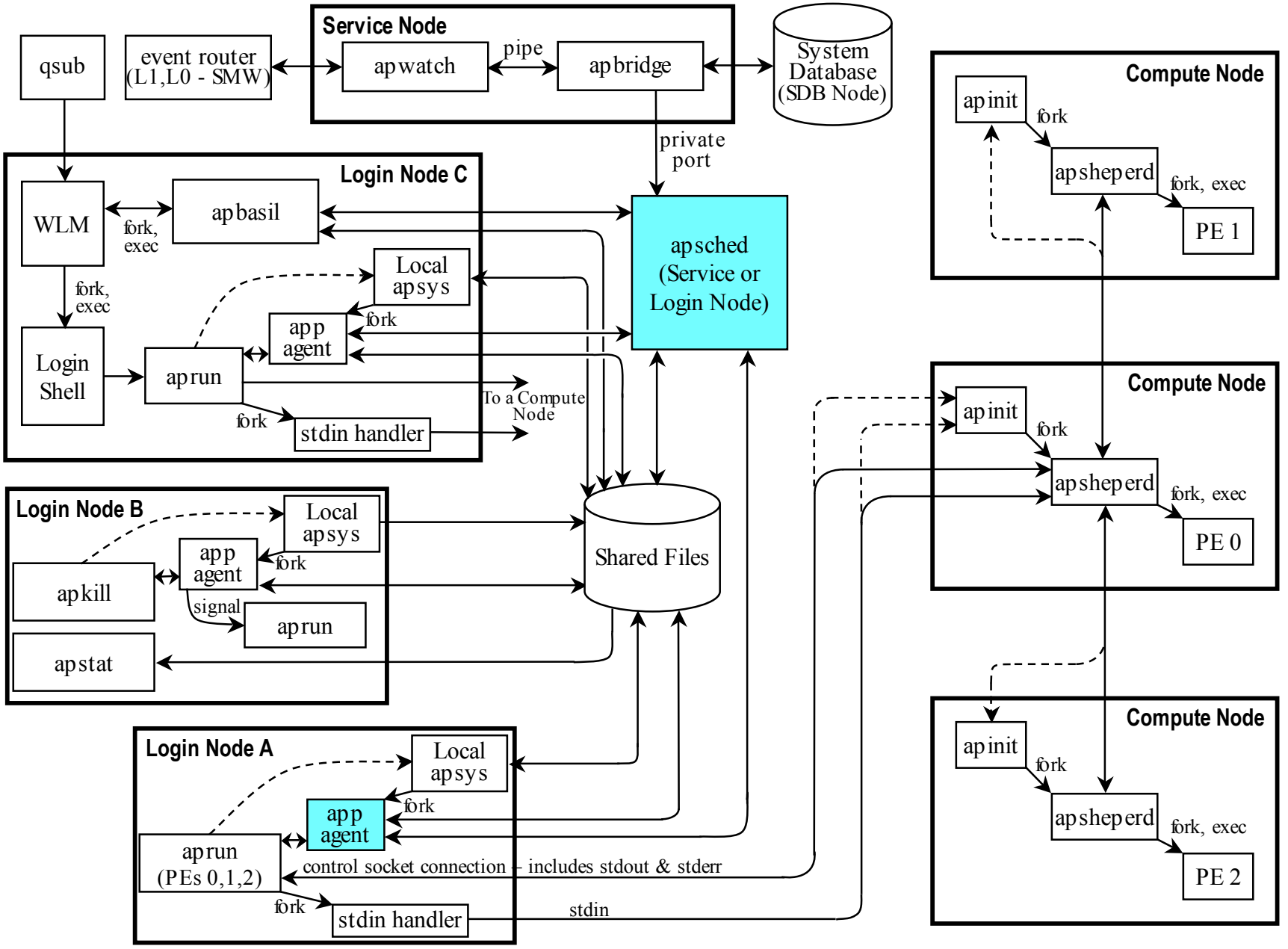


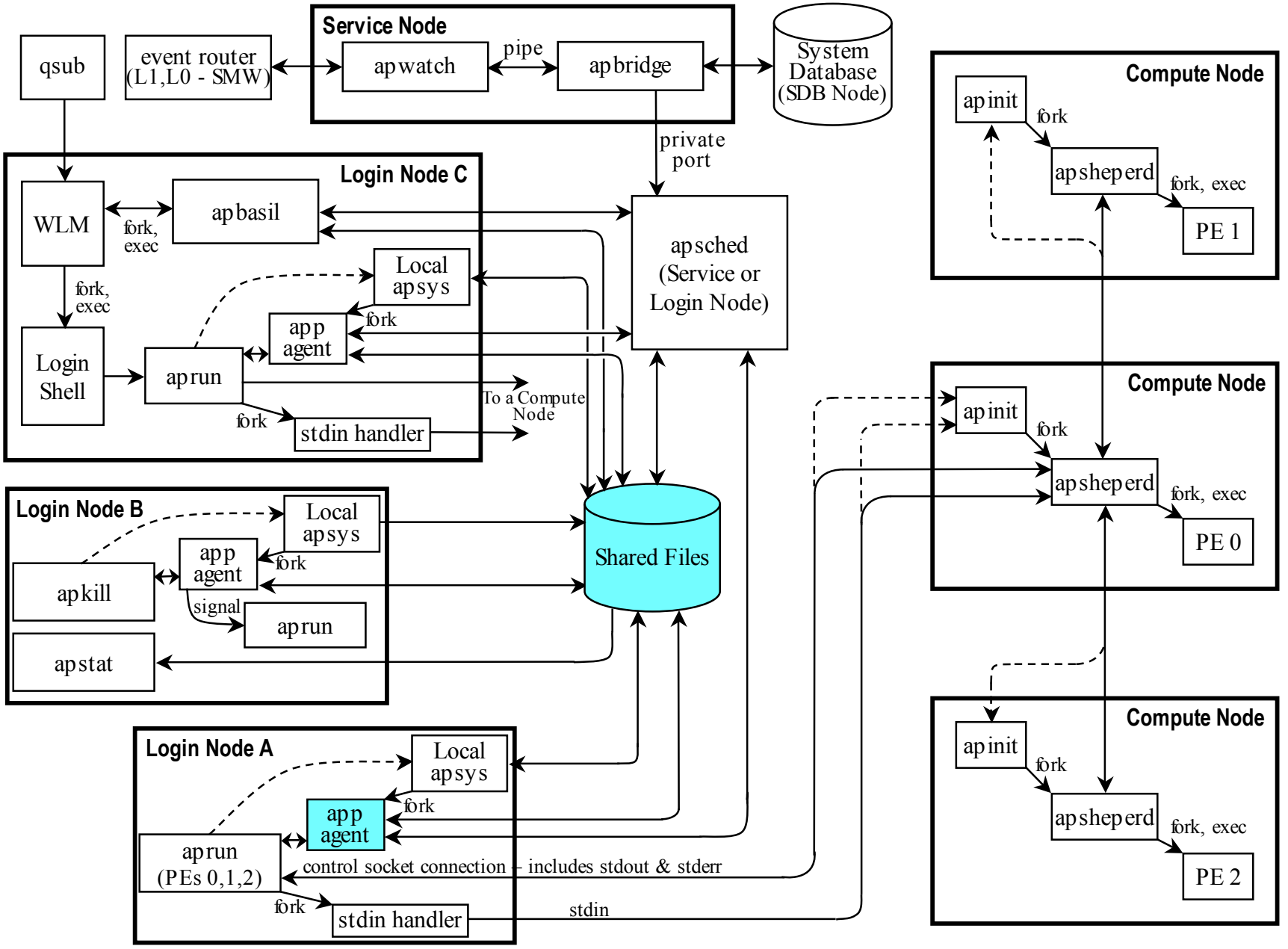


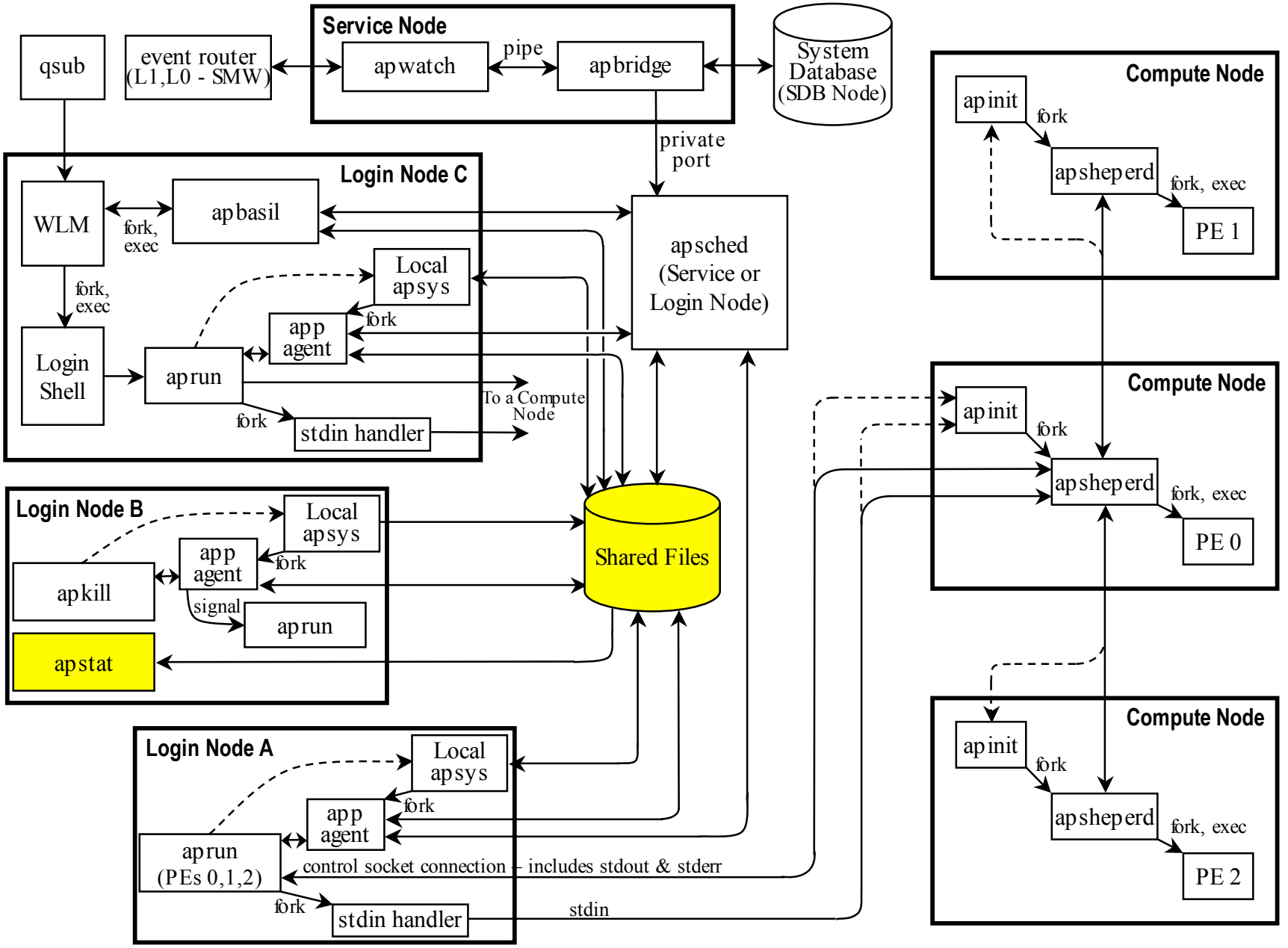


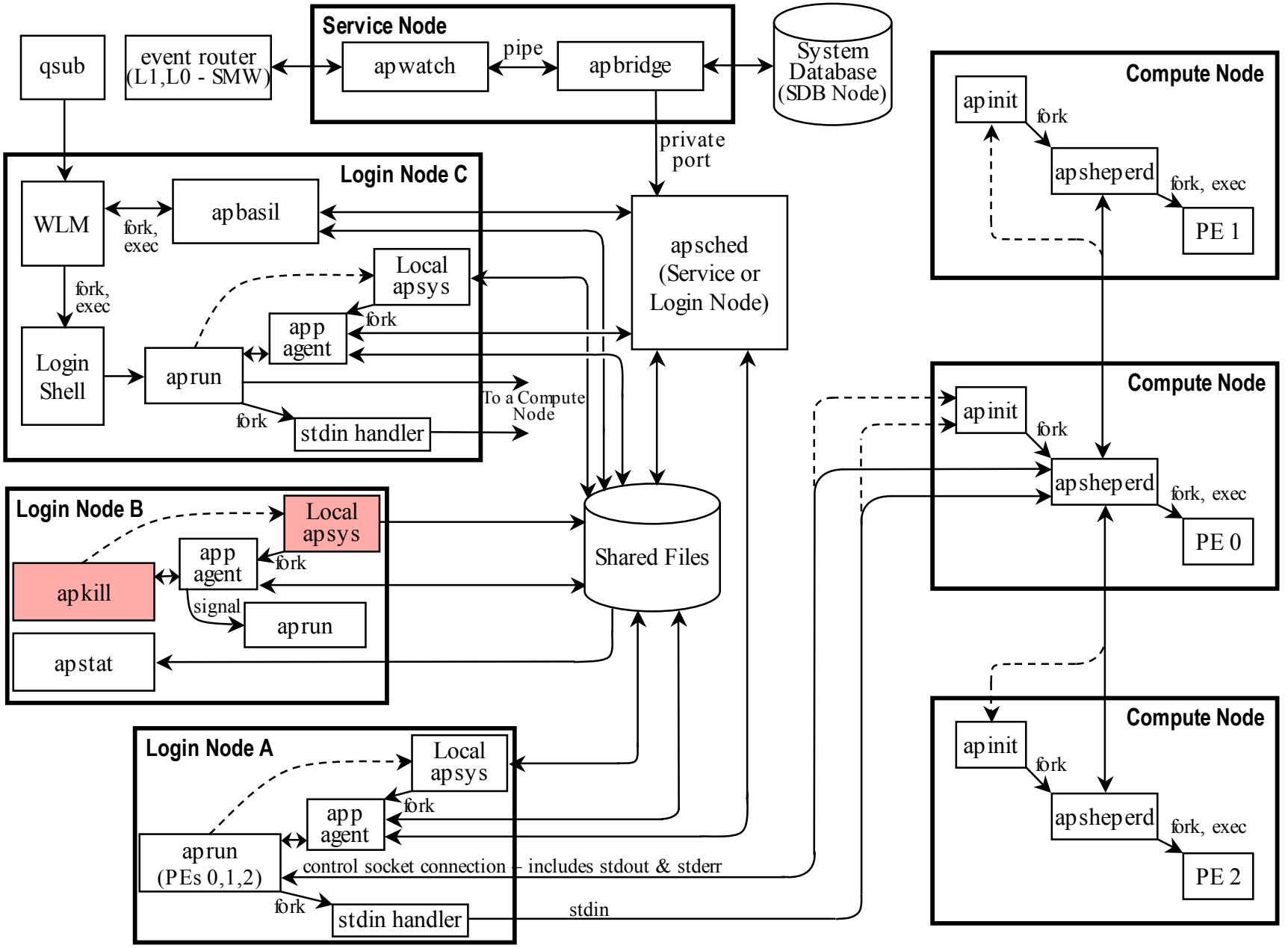


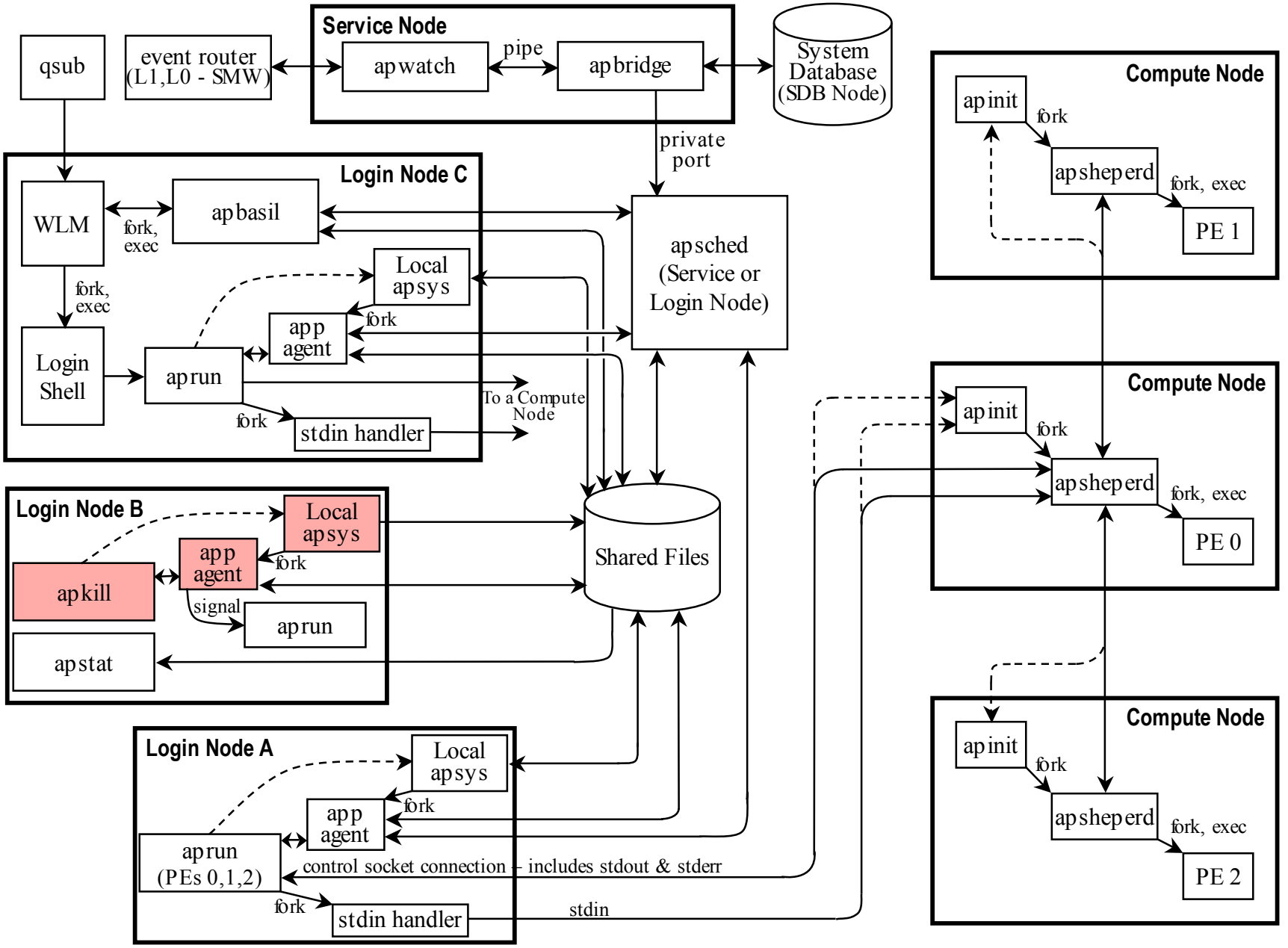


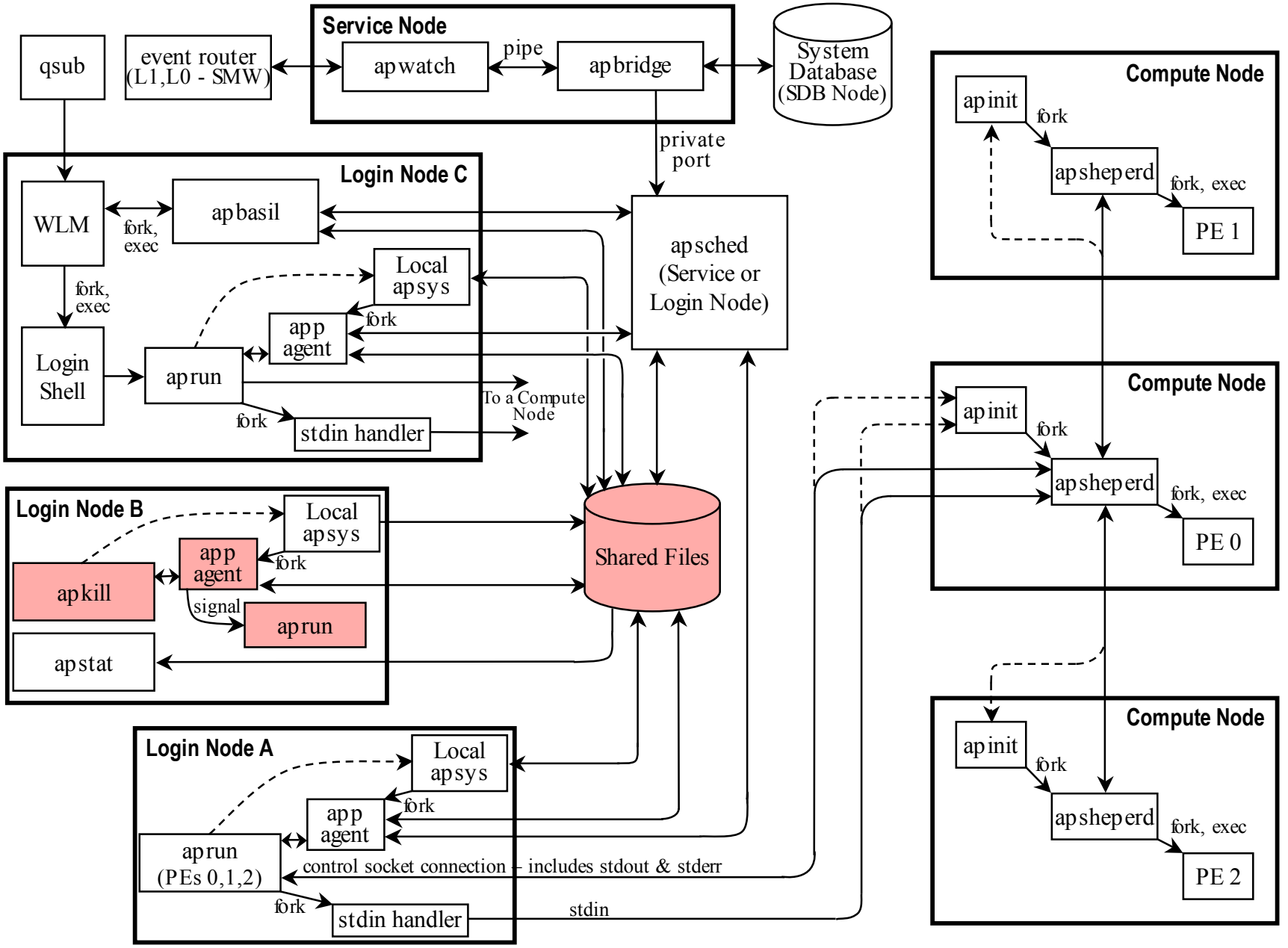


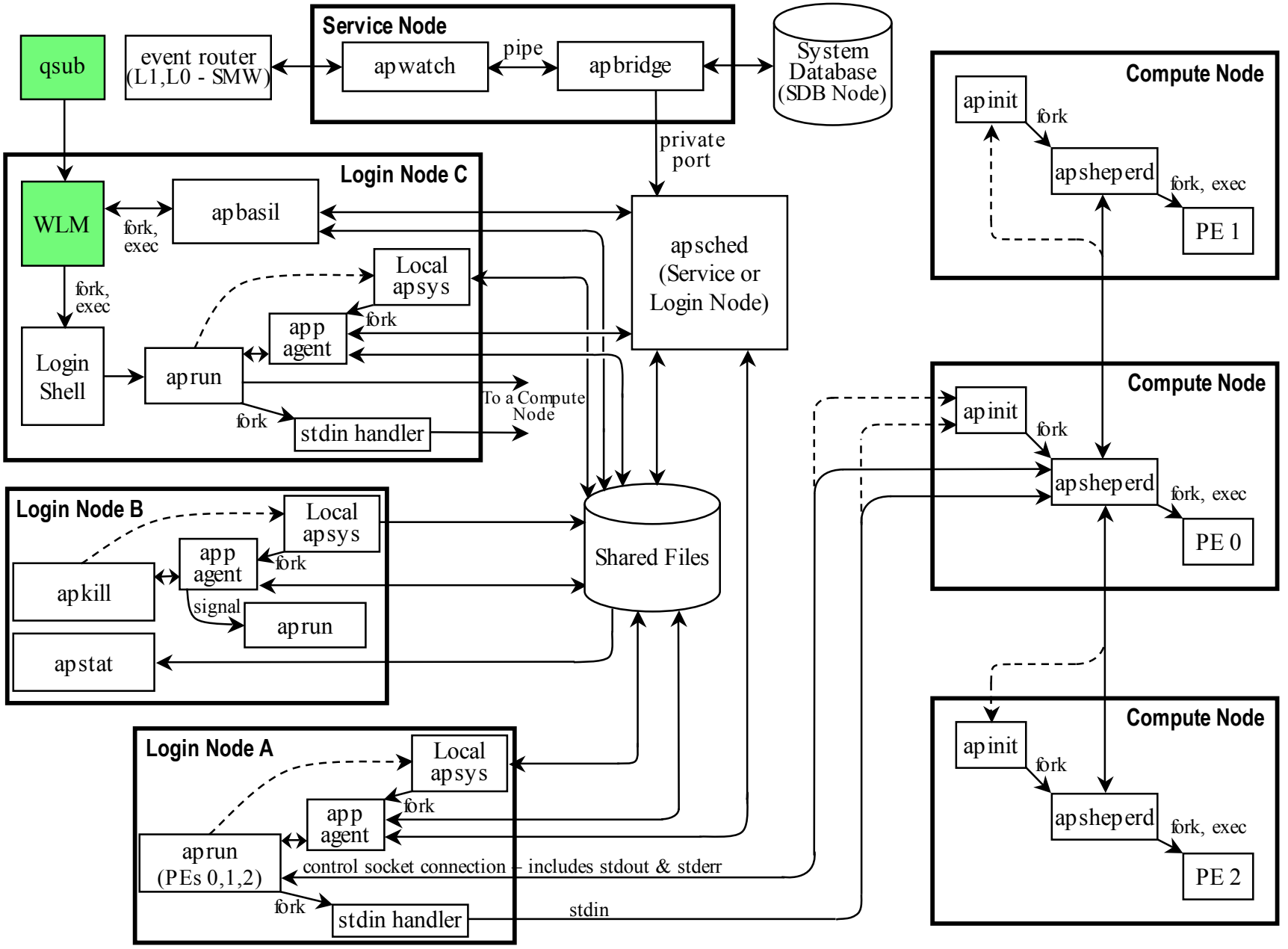


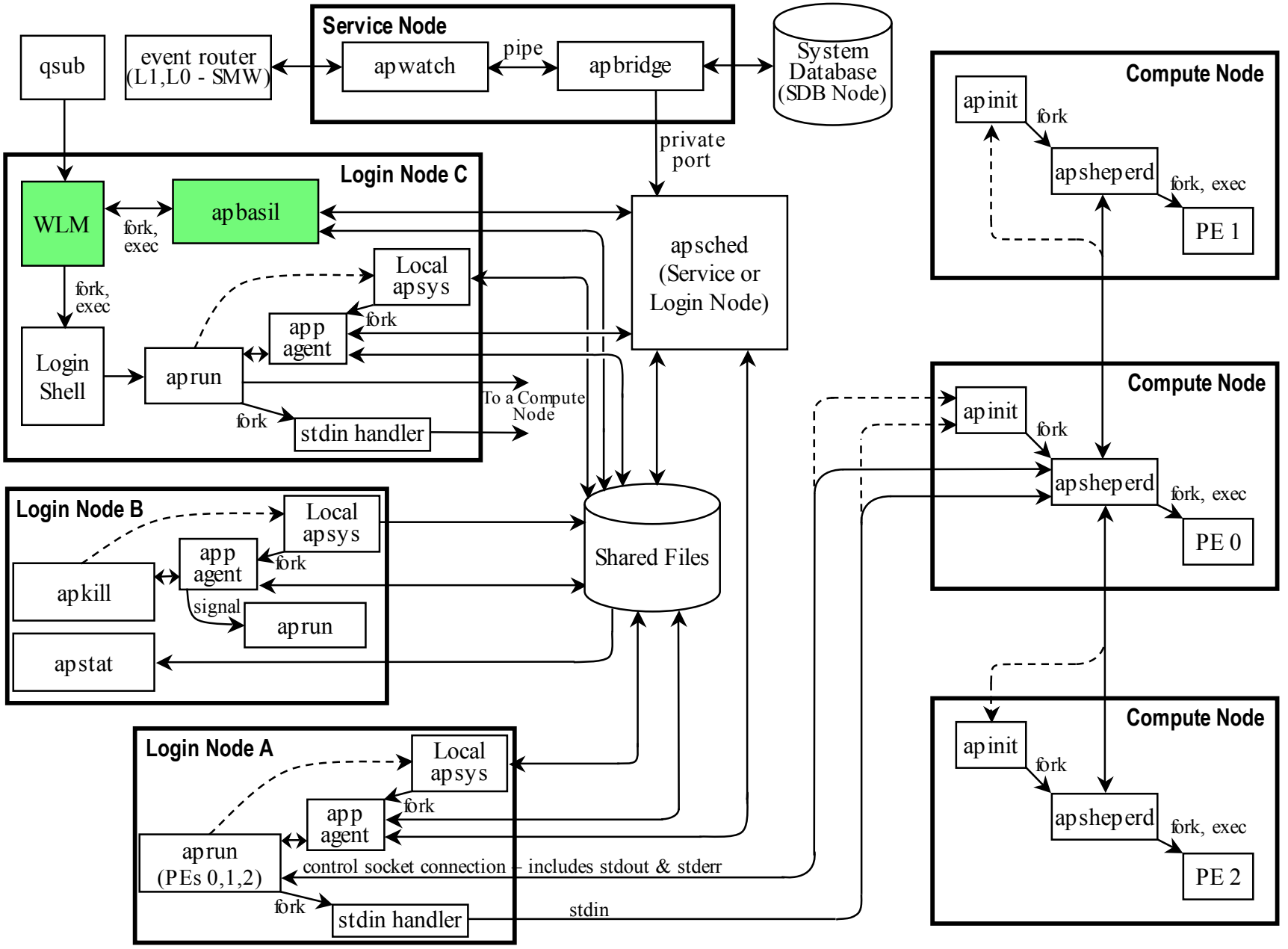


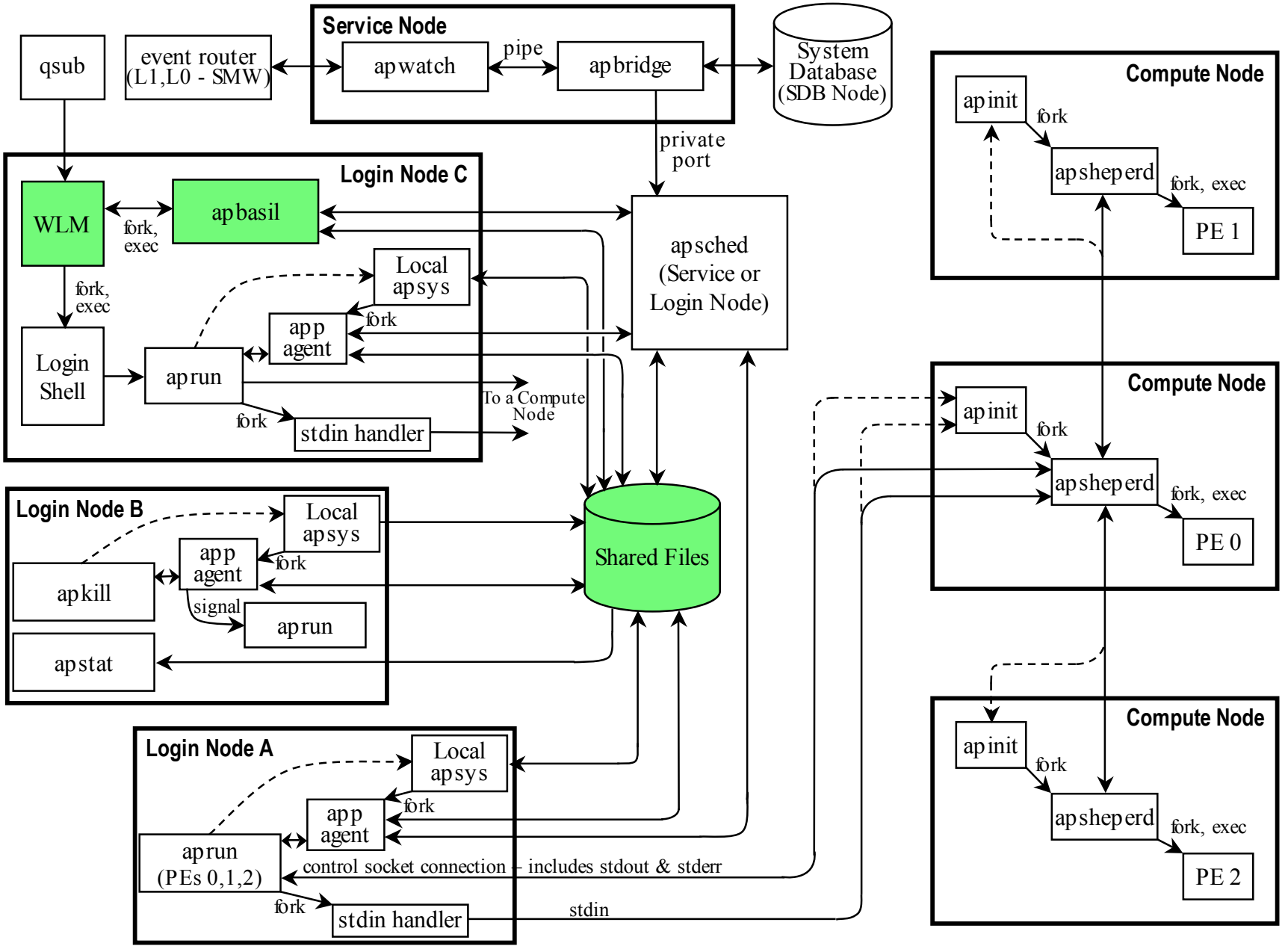


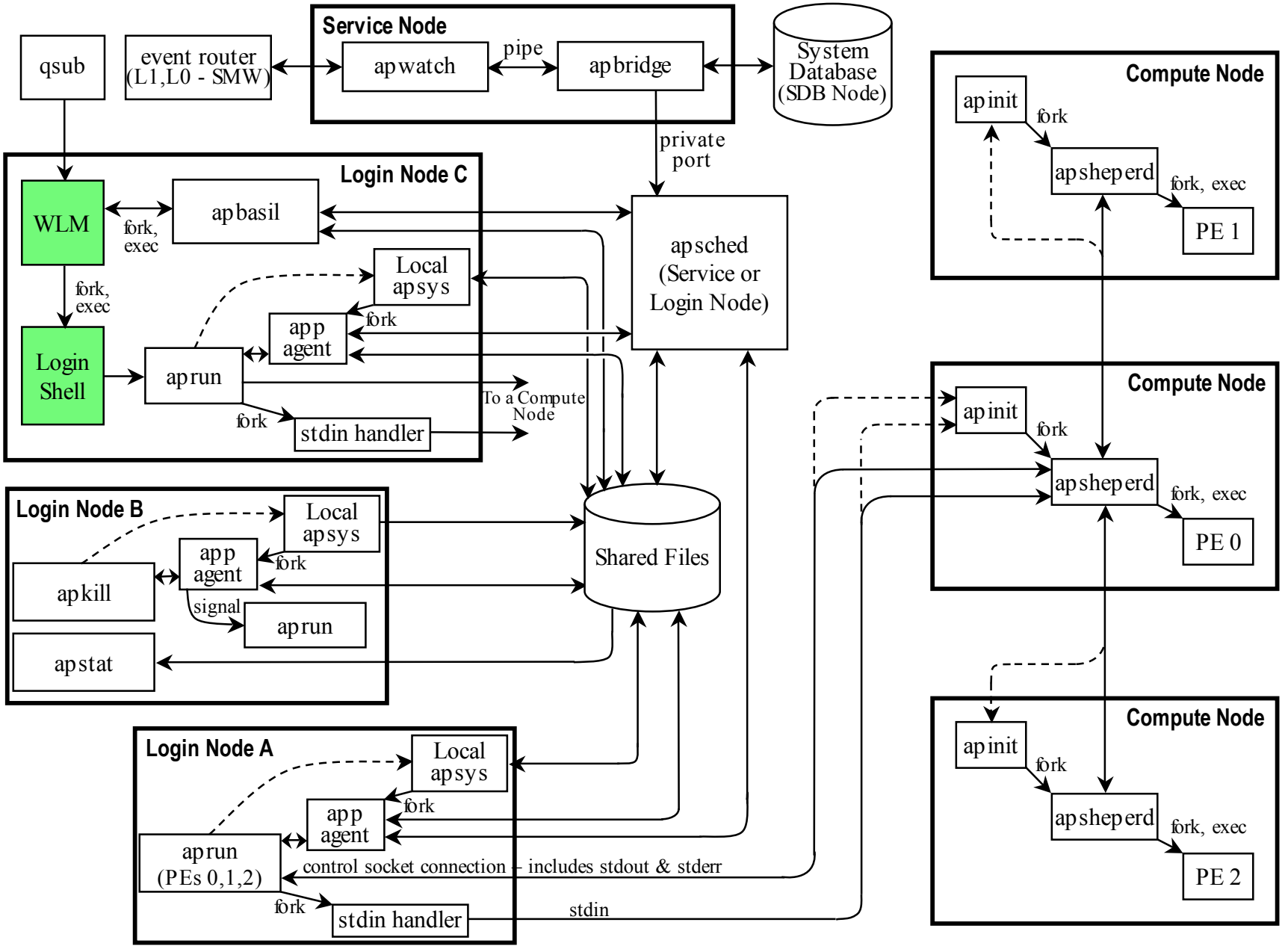


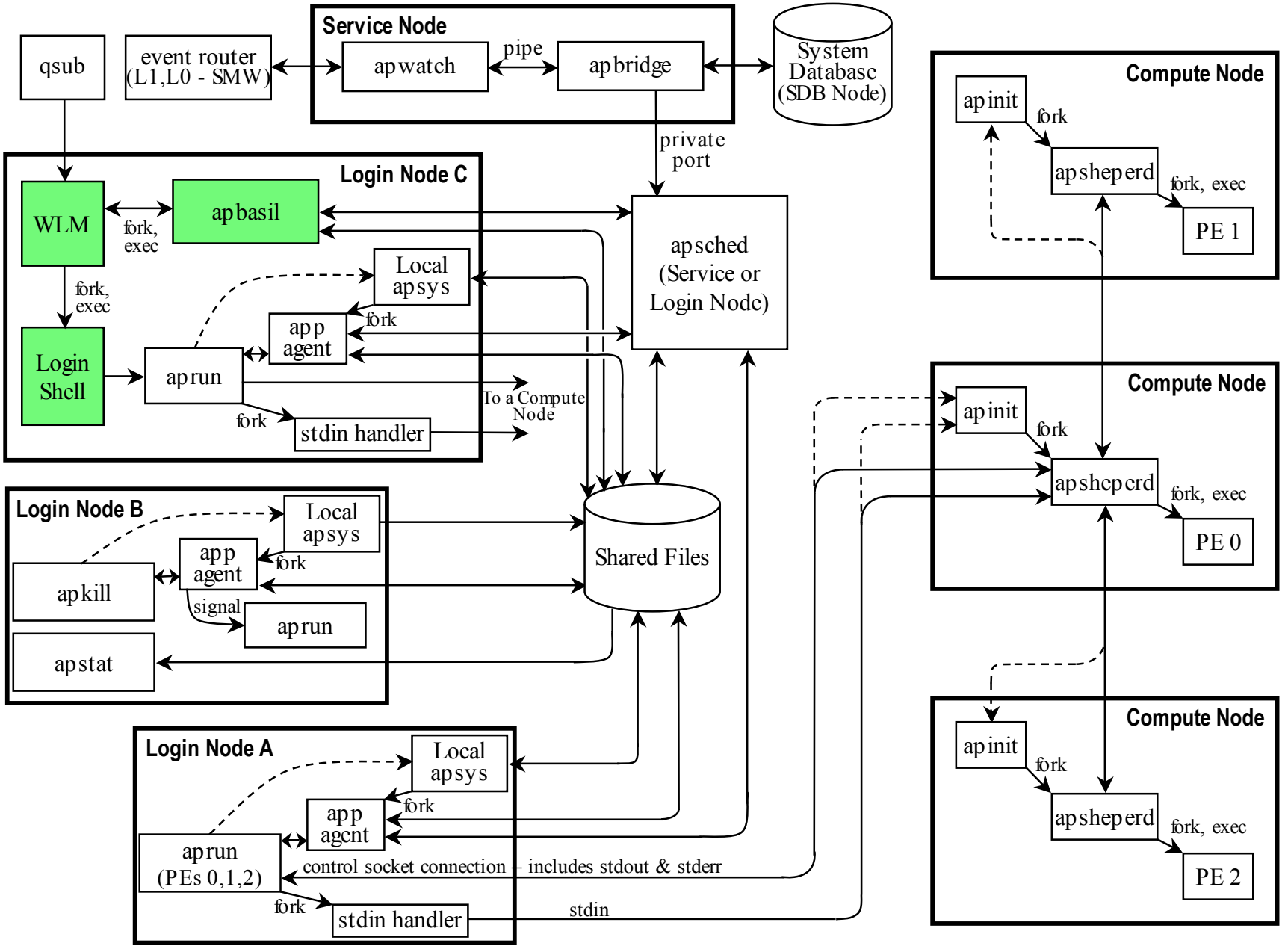


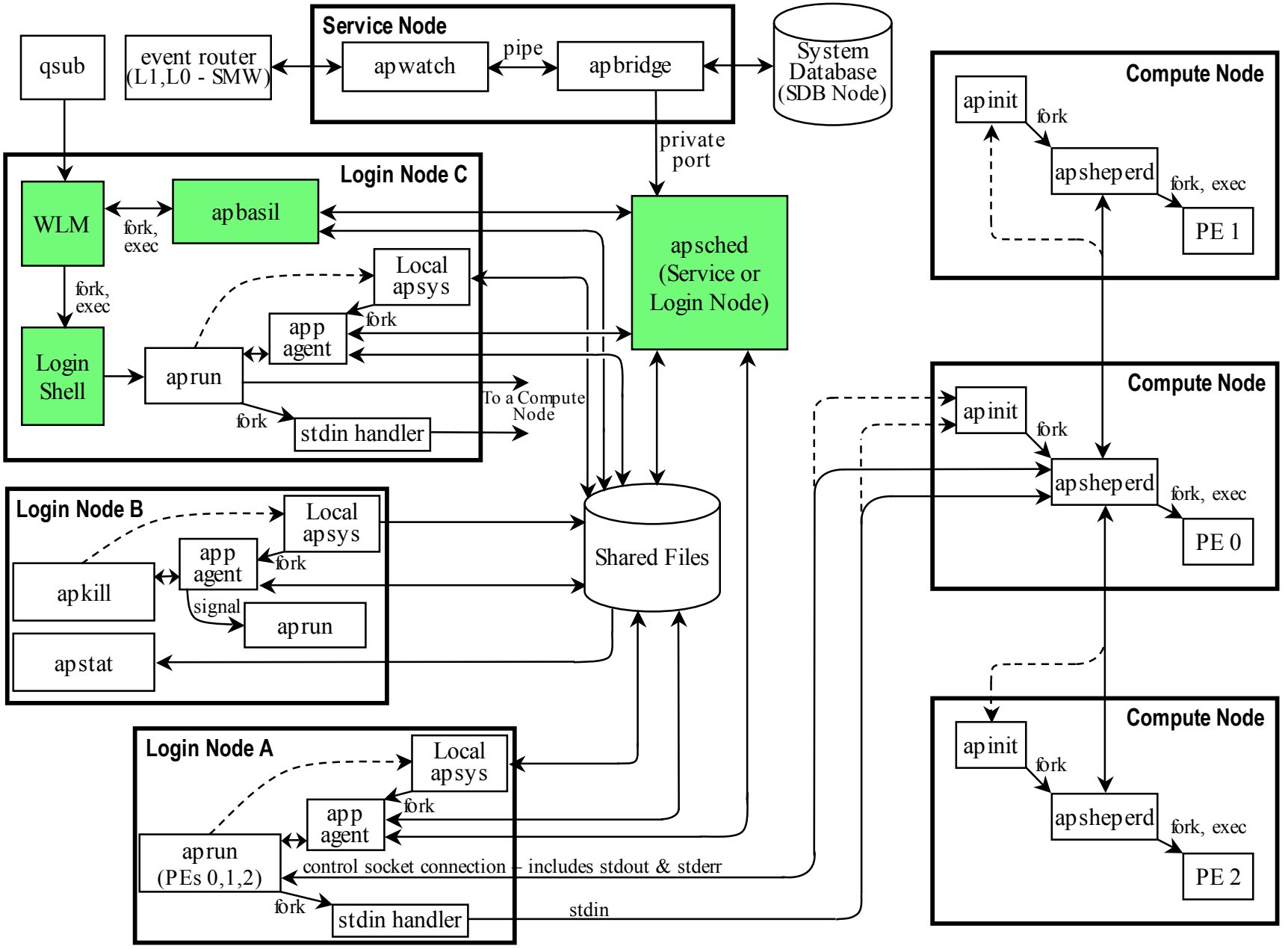


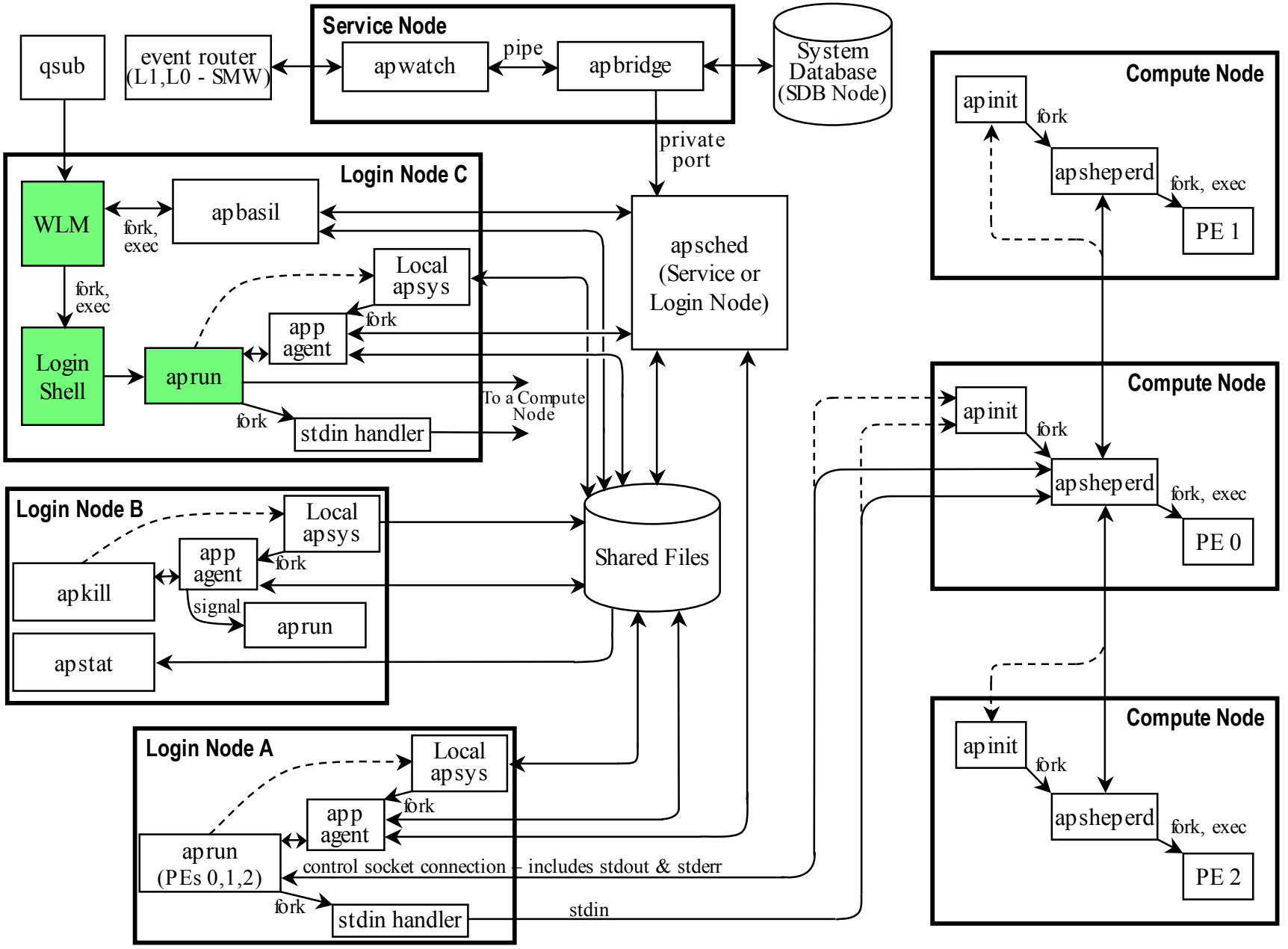


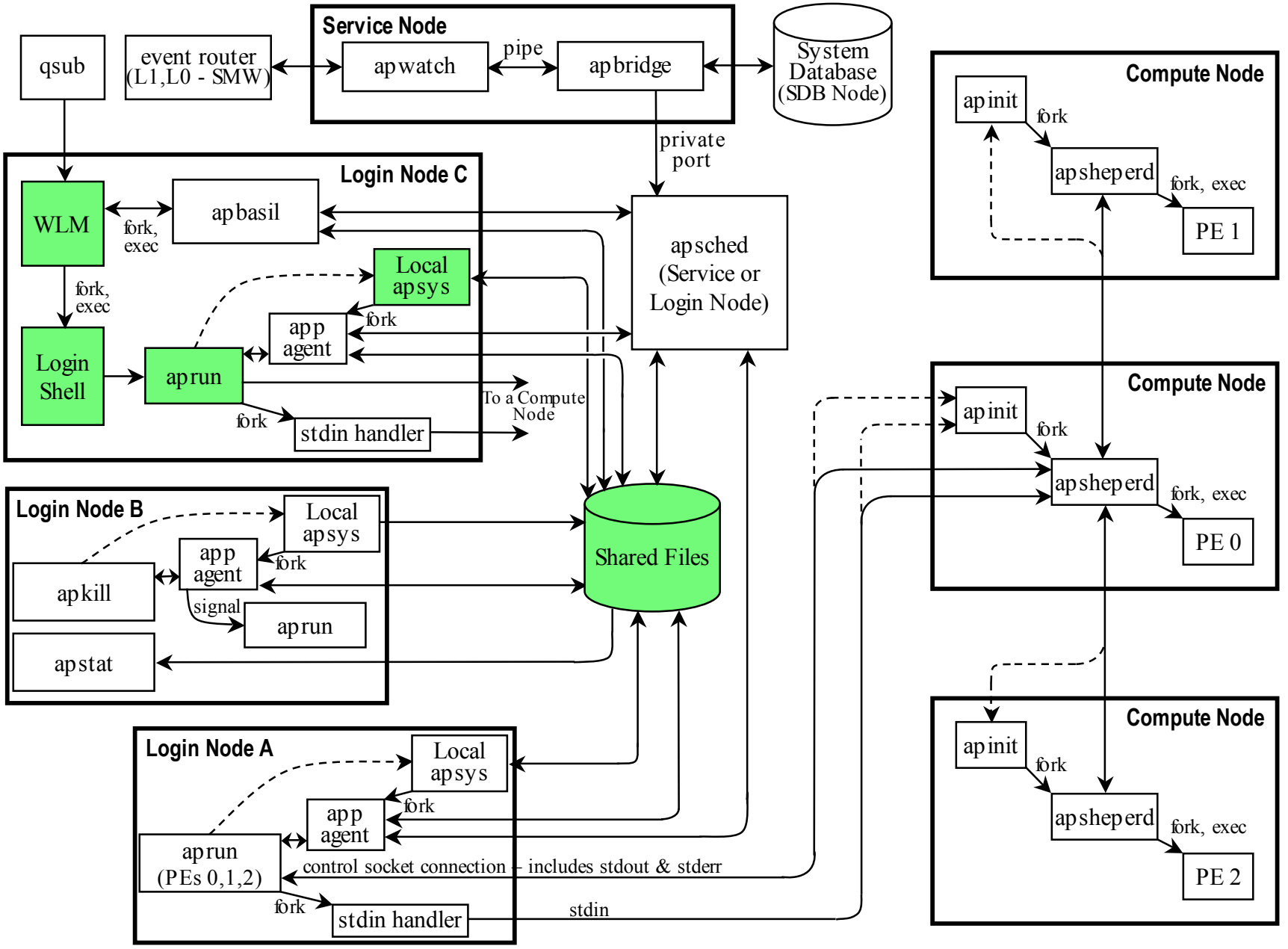


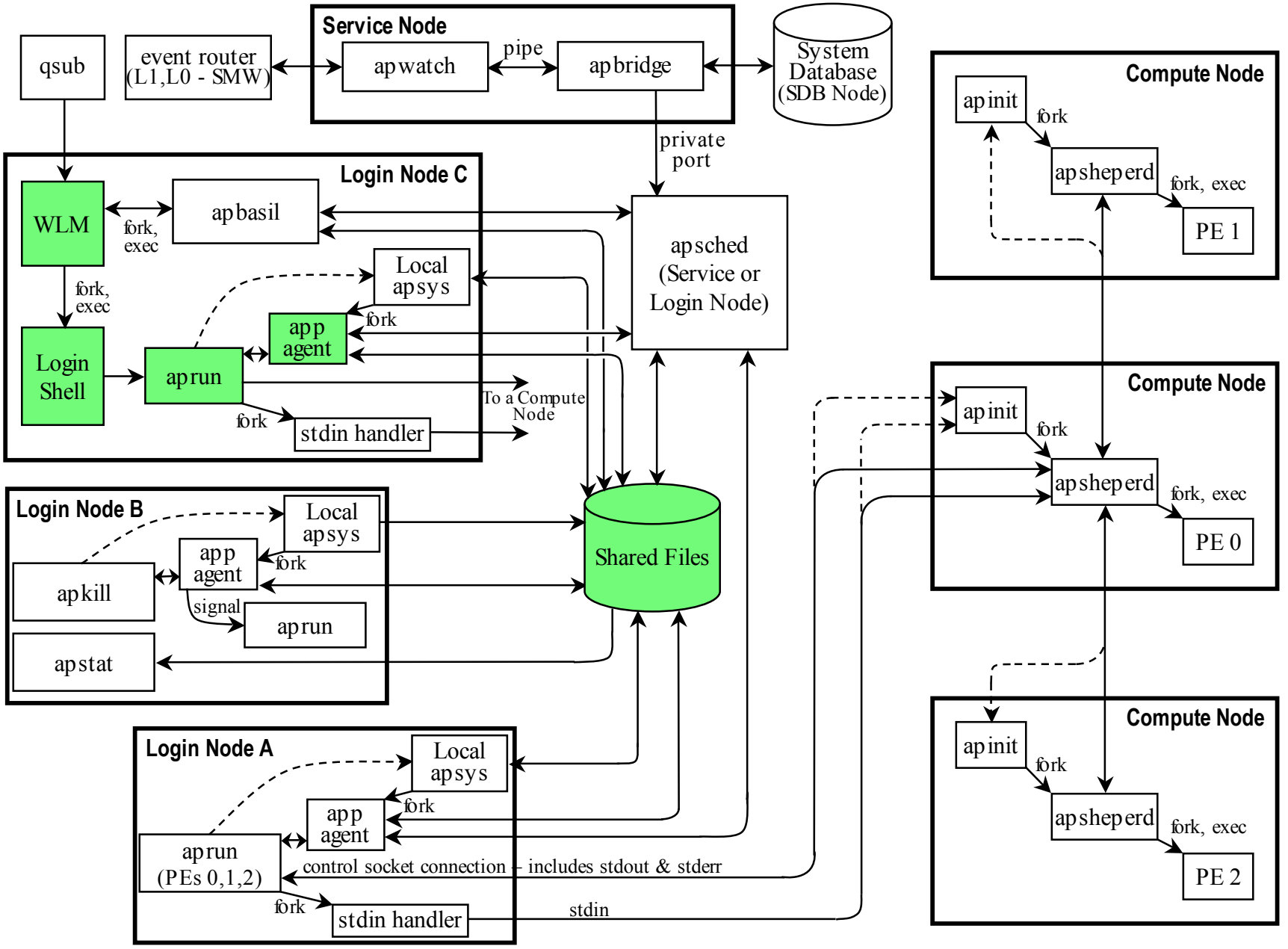


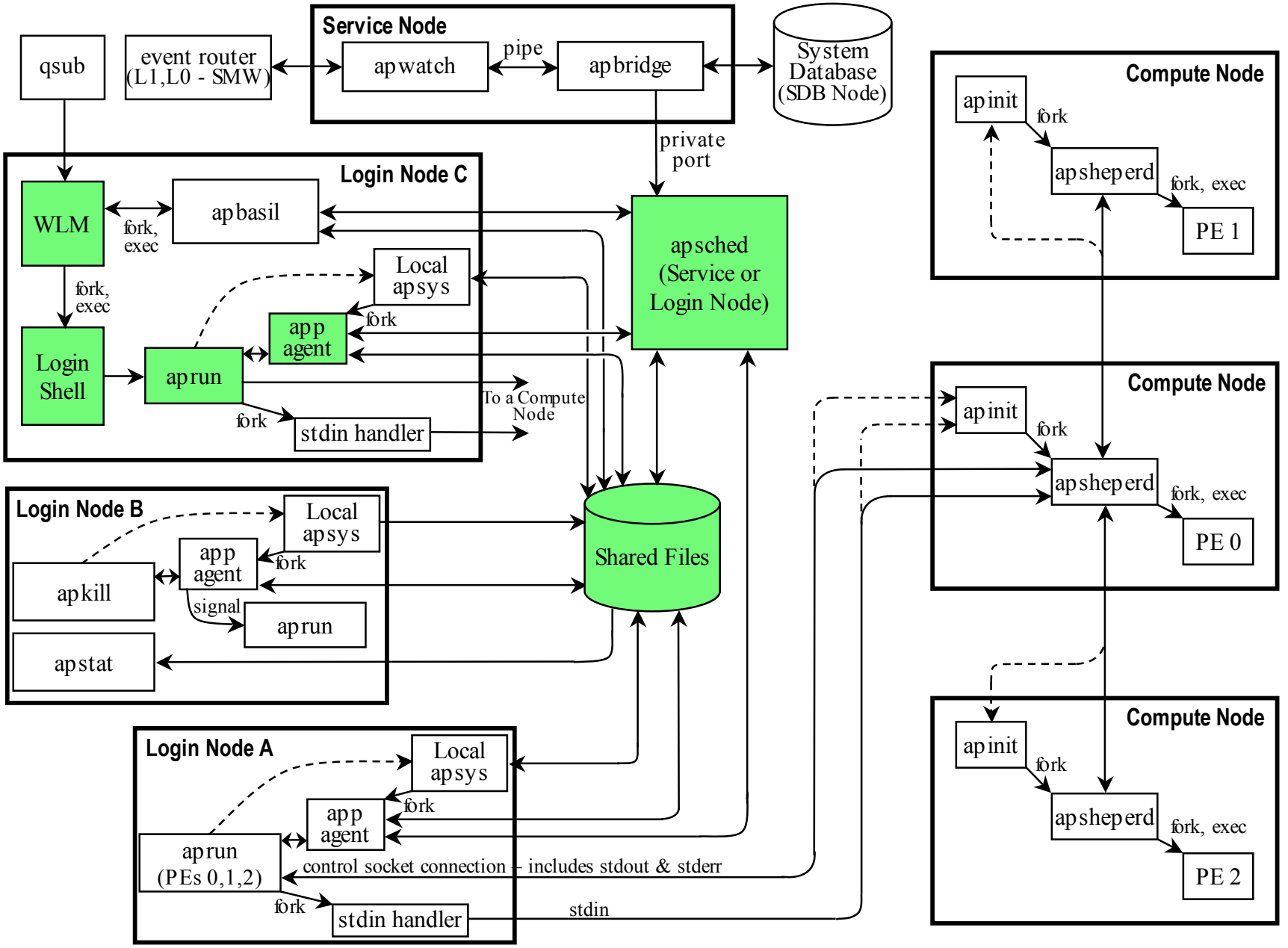


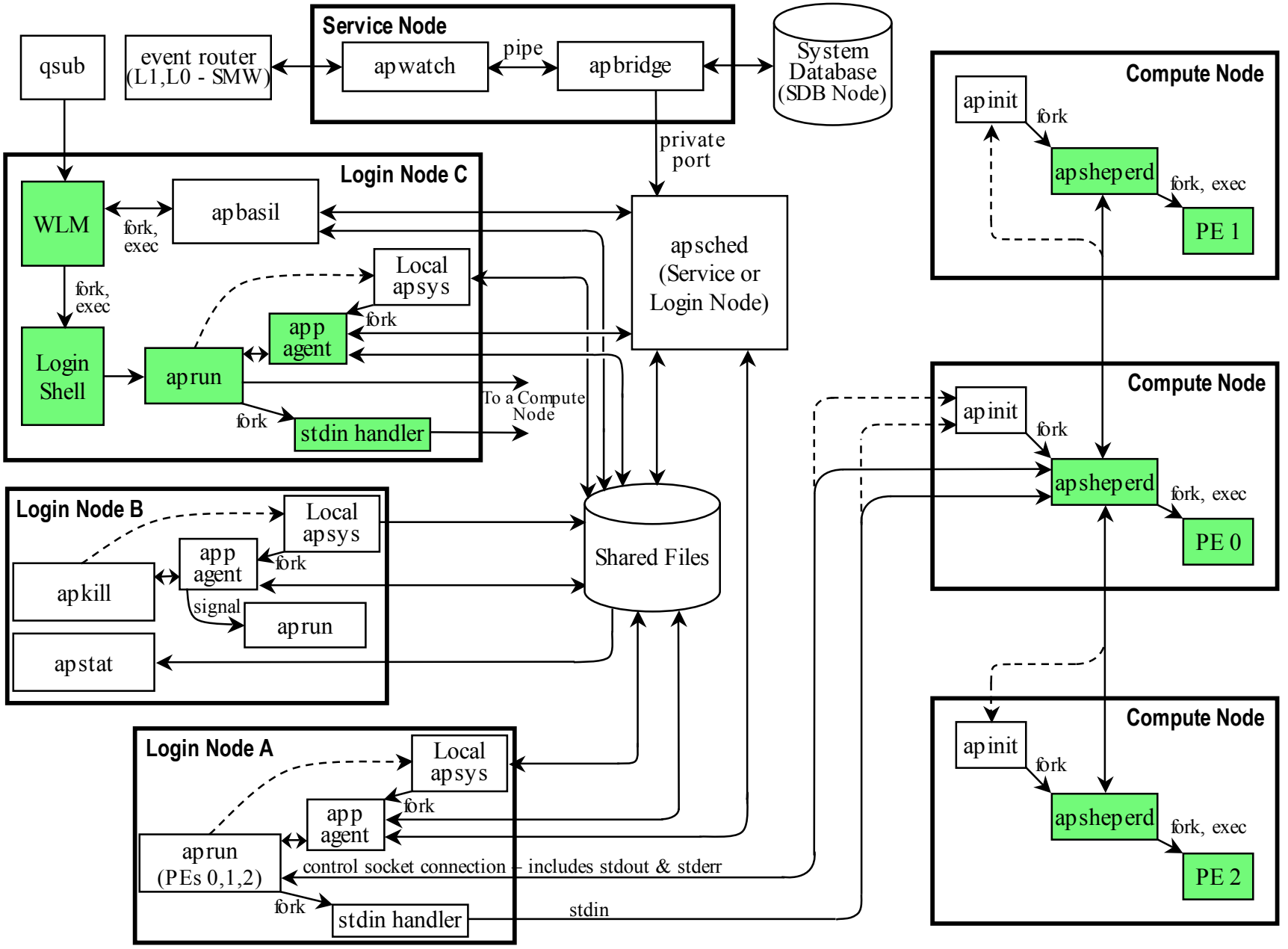


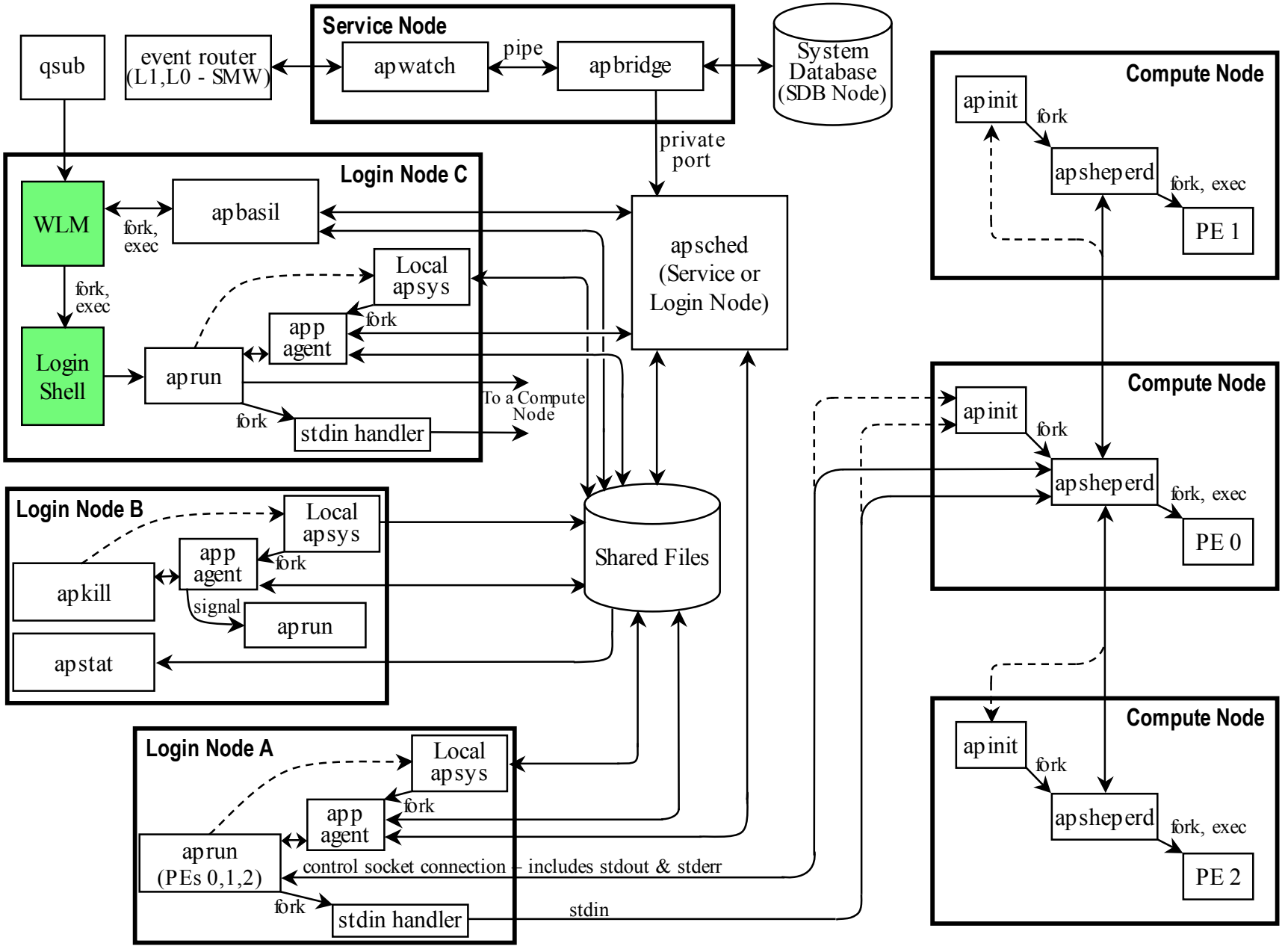


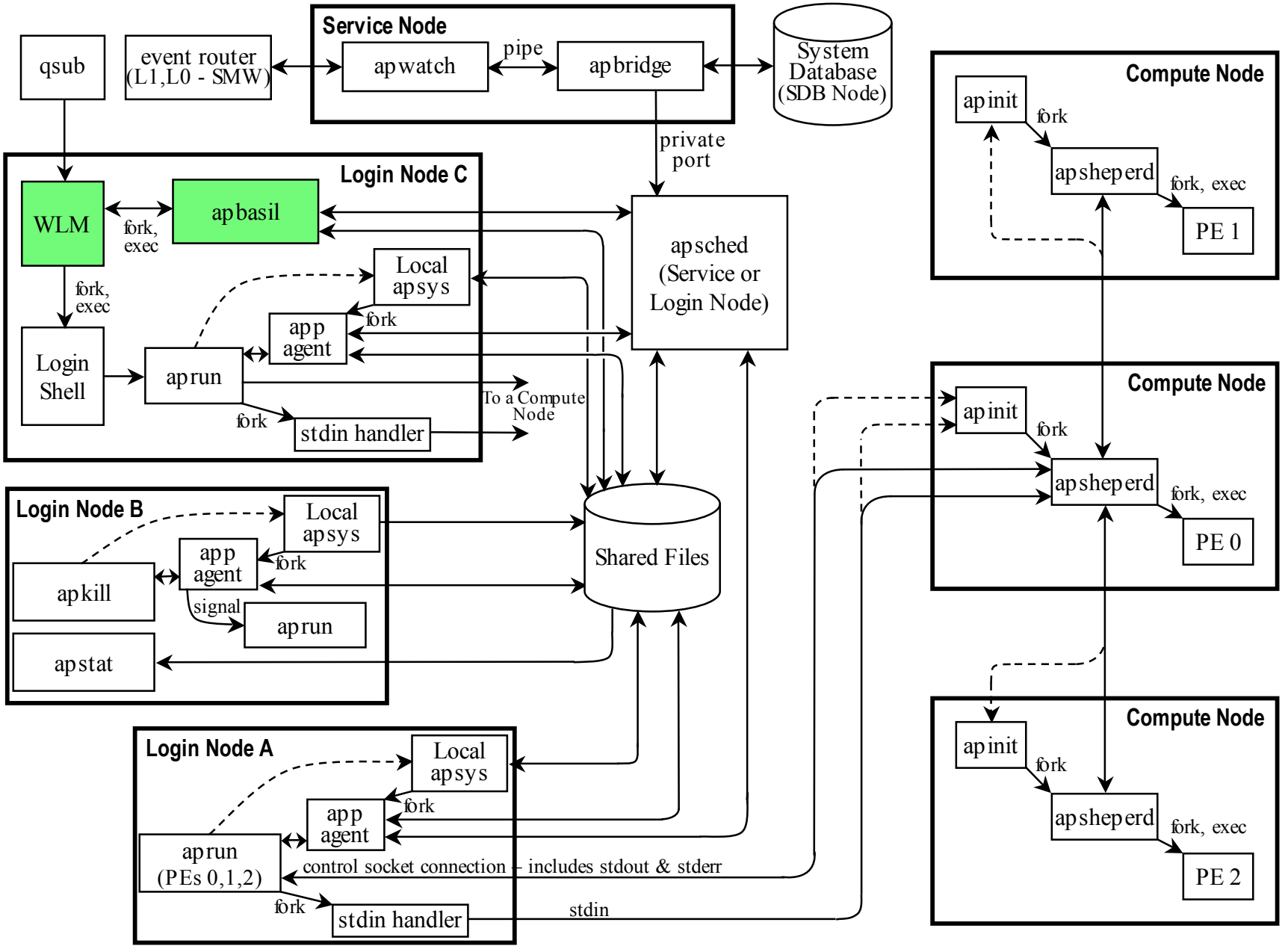


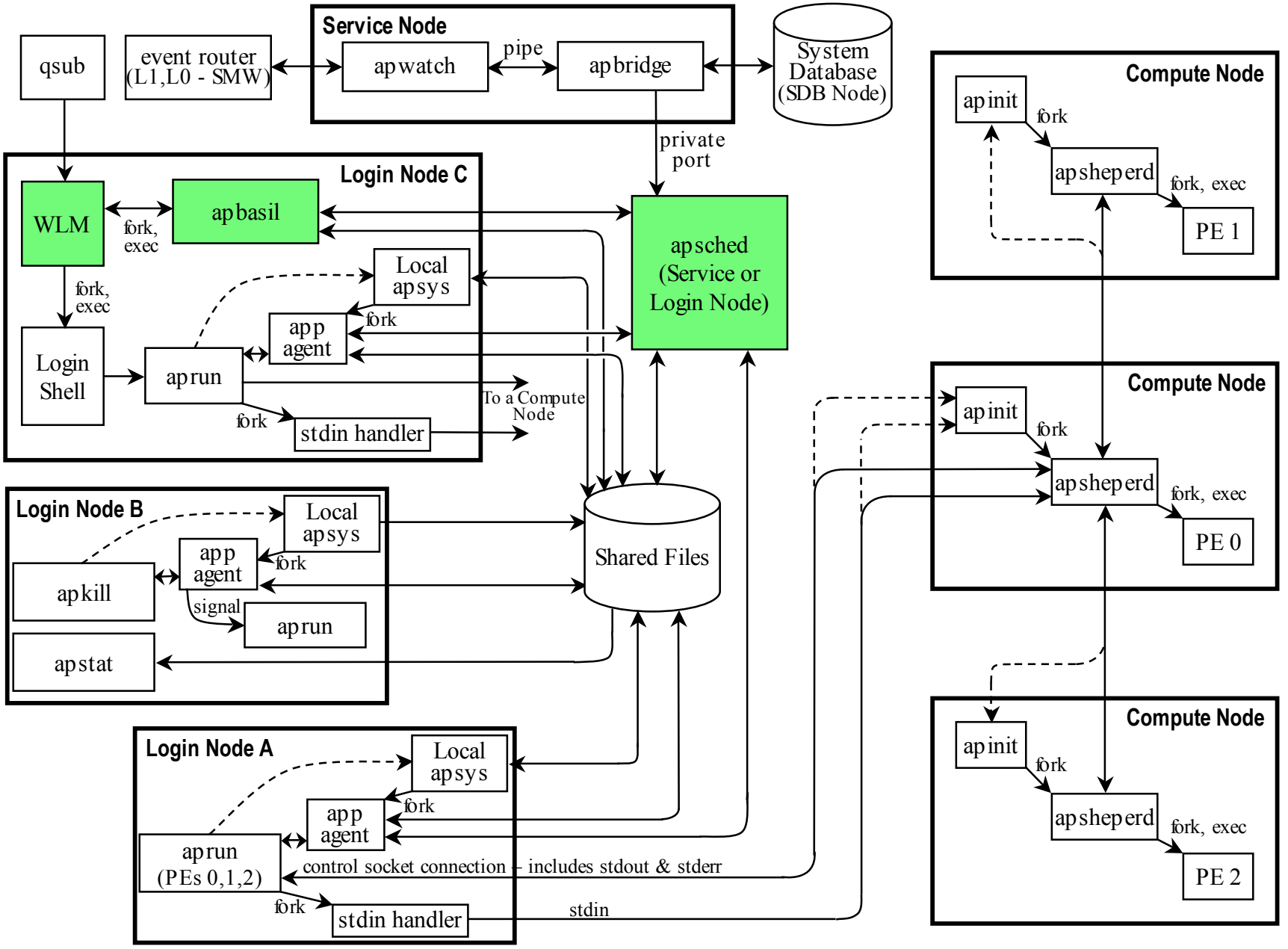


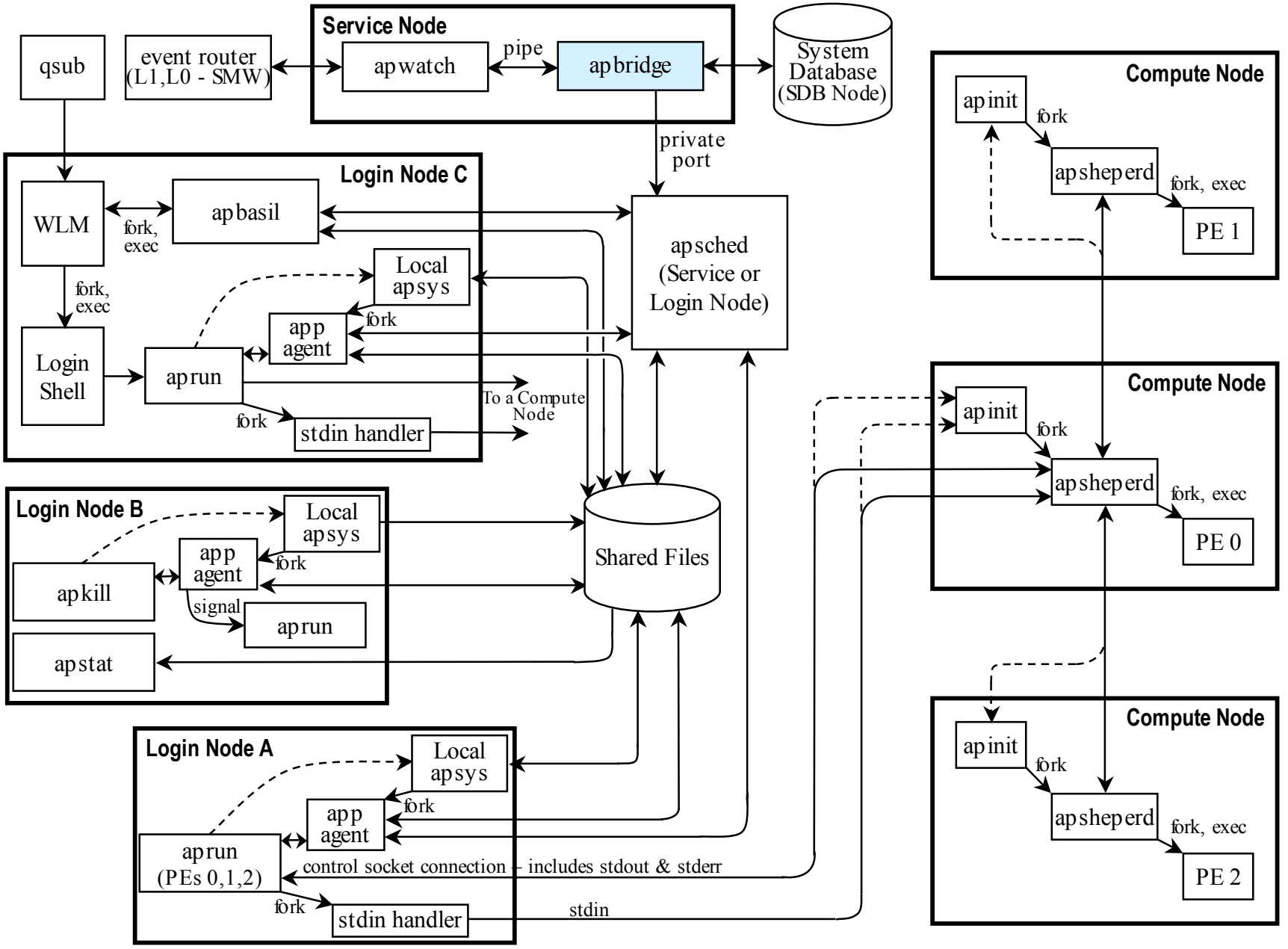


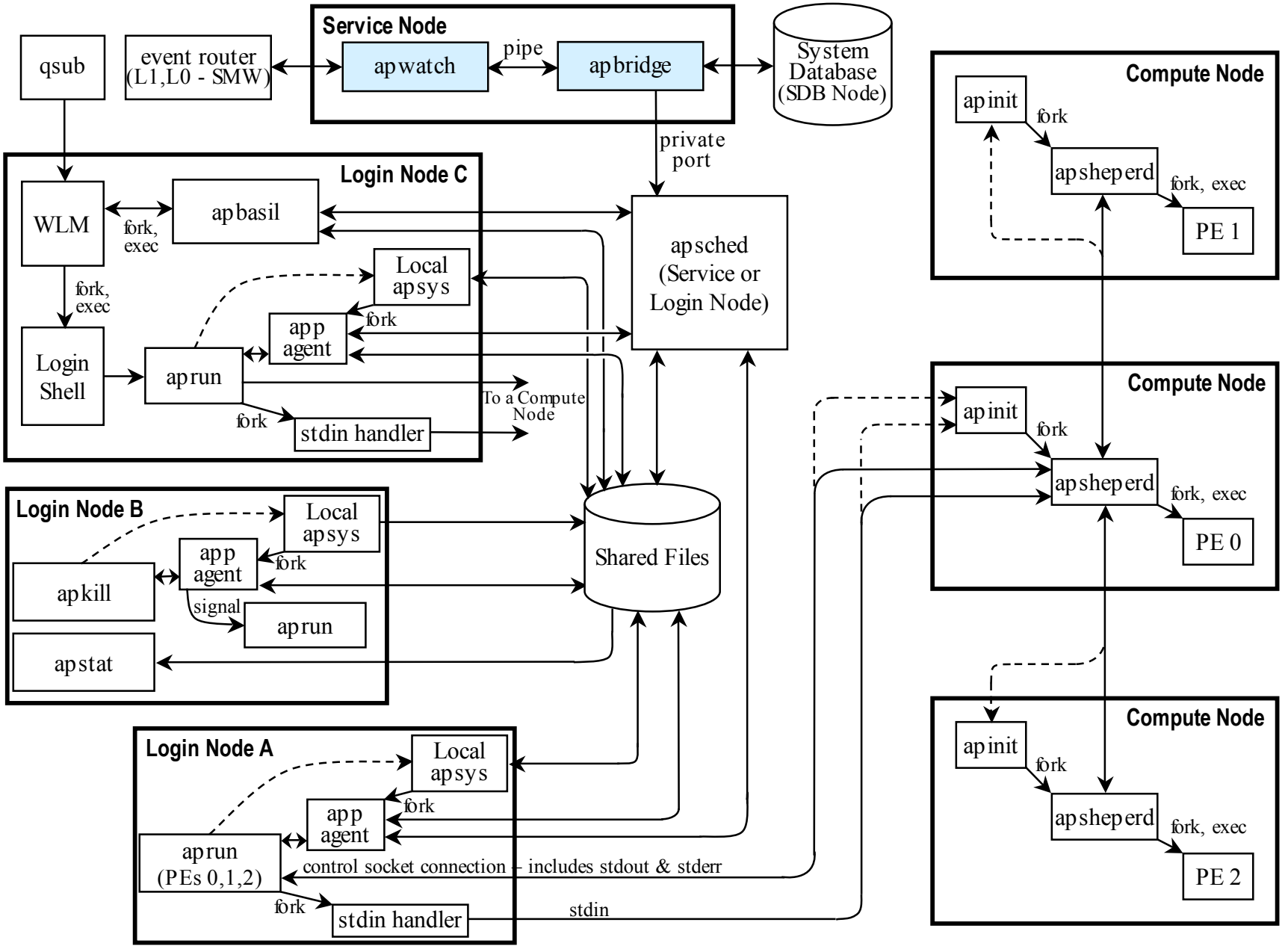

Service Node
qsub
event router
(L1,L0 - SMW)
apwatch
pipe
apbridge
System
Database
(SDB Node)
private
port
Compute Node
apinit
fork
apsheperd
fork, exec
PE 1
Login Node C
WLM
fork,
exec
apbasil
Local
apsys
fork,
exec
app
agent
fork
Login
Shell
aprun
To a Compute
Node
fork
stdin handler
apsched
(Service or
Login Node)
Compute Node
apinit
fork
apsheperd
fork, exec
PE 0
Login Node B
Local
apsys
app
agent
fork
apkill
signal
aprun
apstat
Shared Files
Compute Node
apinit
fork
apsheperd
fork, exec
PE 2
Login Node A
Local
apsys
app
agent
fork
aprun
(PEs 0,1,2)
control socket connection – includes stdout & stderr
fork
stdin handler
stdin

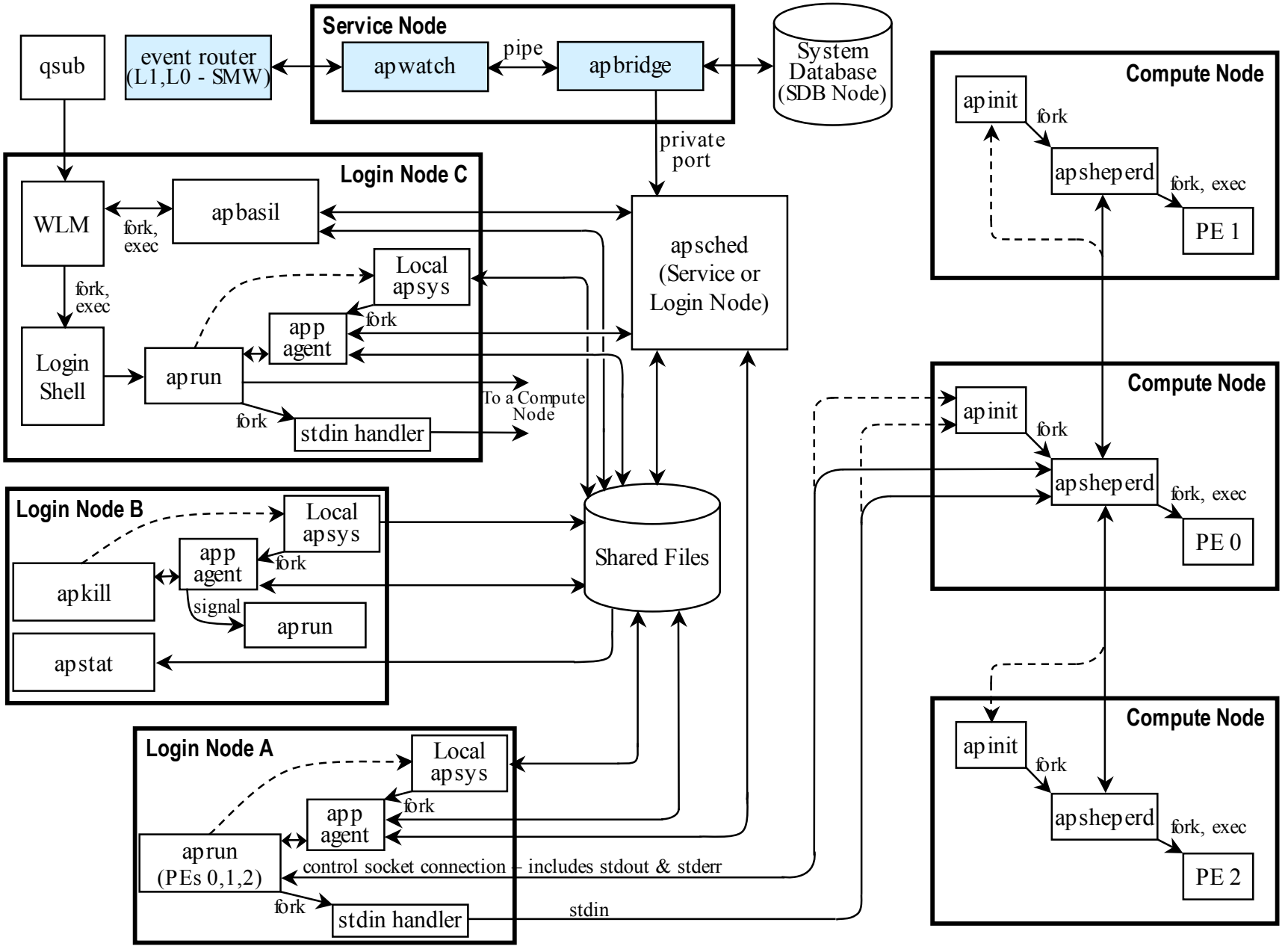


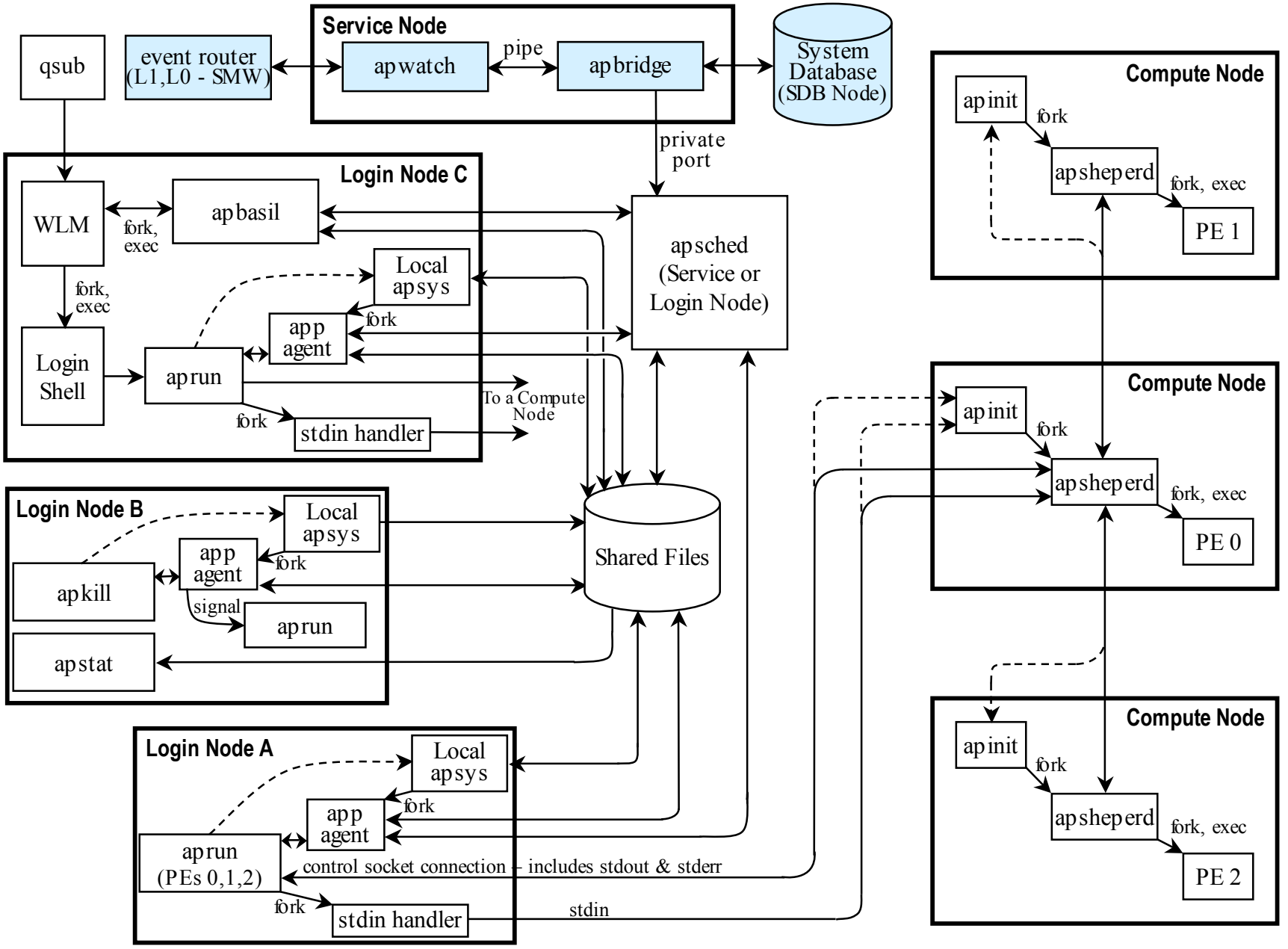


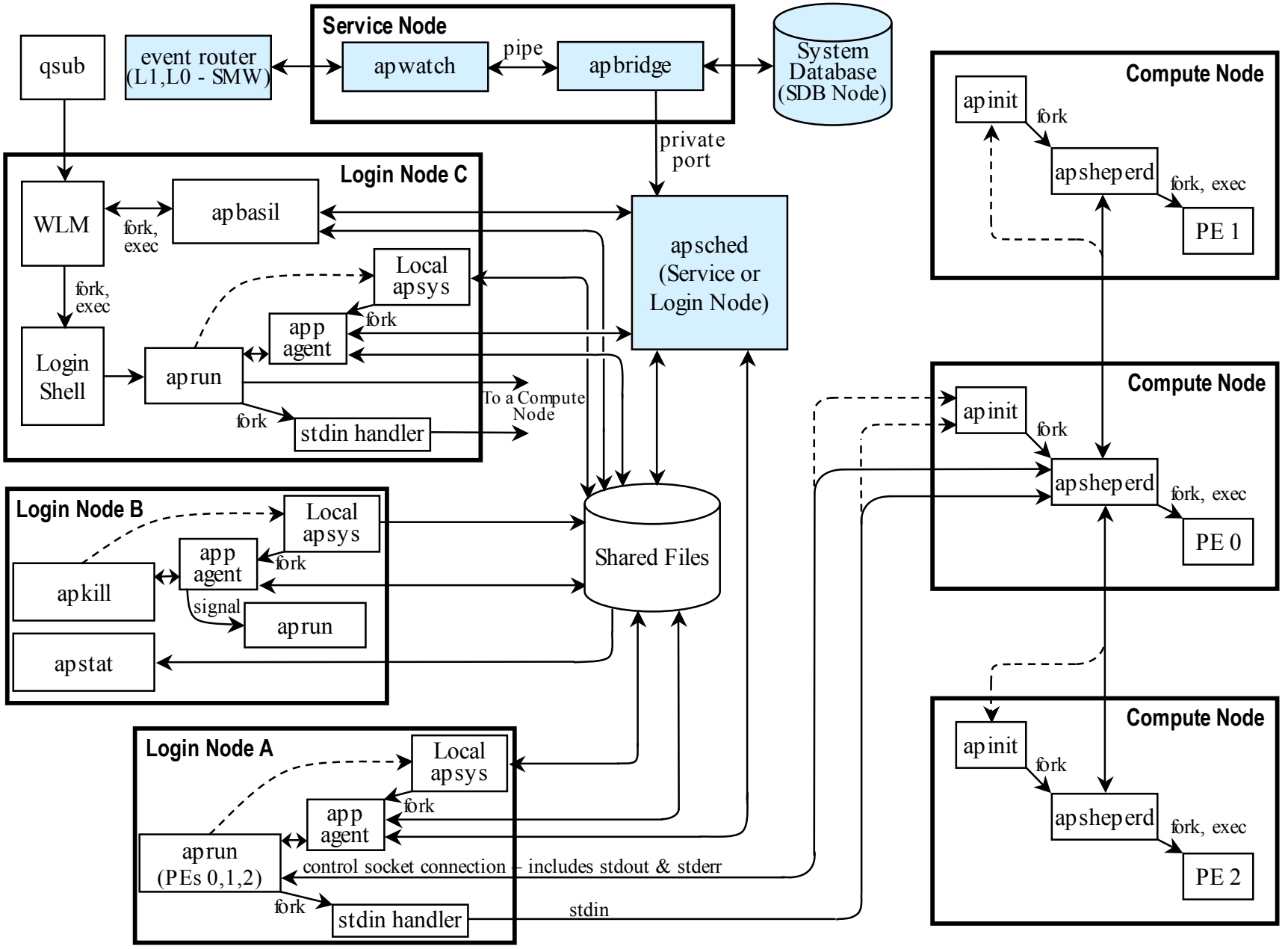


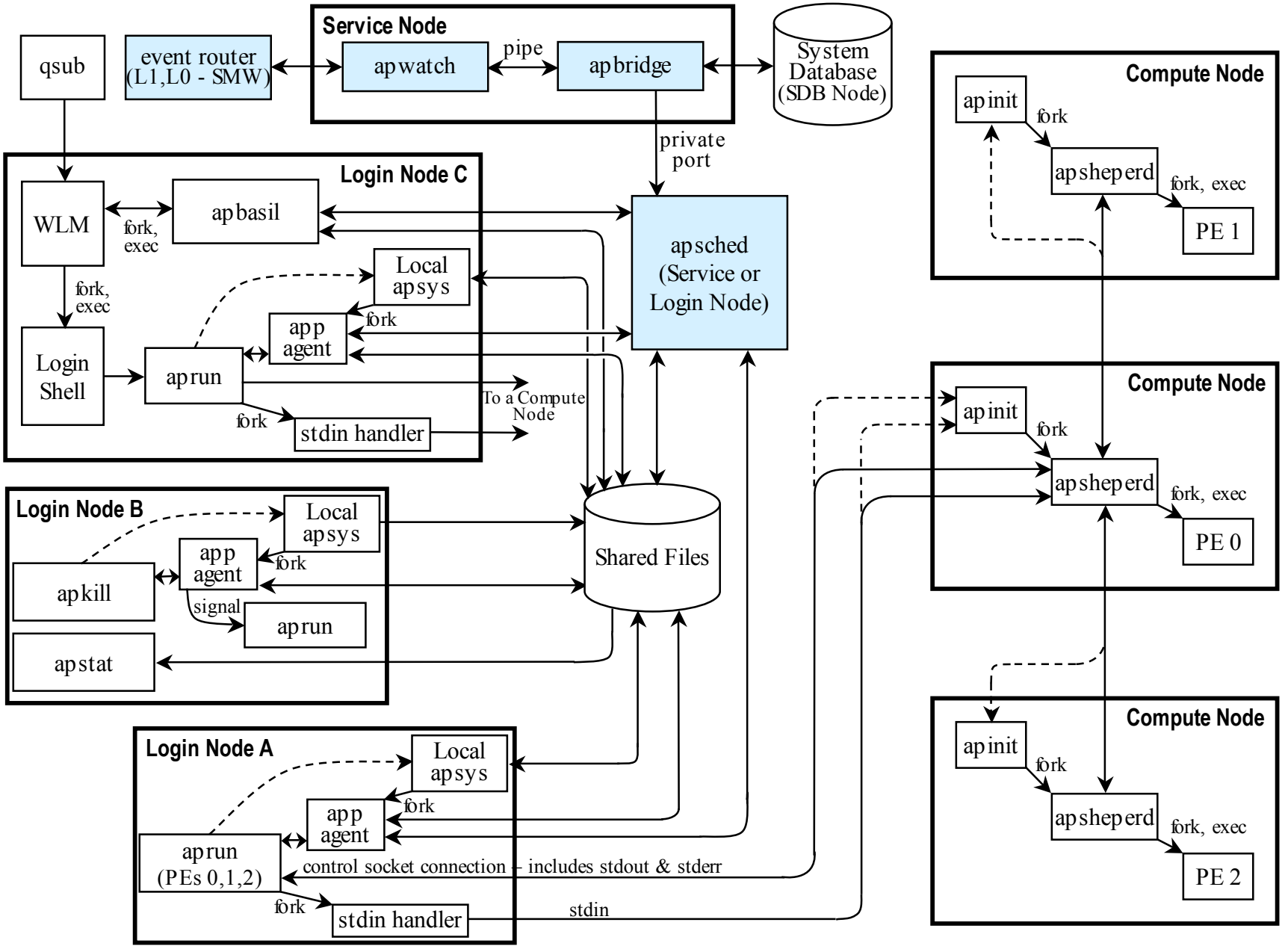

Service Node
qsub
event router
(L1,L0 - SMW)
apwatch
pipe
apbridge
System
Database
(SDB Node)
private
port
Compute Node
apinit
fork
apsheperd
fork, exec
PE 1
Login Node C
WLM
fork,
exec
apbasil
apsched
(Service or
Login Node)
fork,
exec
Local
apsys
fork
app
agent
Login
Shell
aprun
To a Compute
Node
fork
stdin handler
Compute Node
apinit
fork
apsheperd
fork, exec
PE 0
Login Node B
Local
apsys
app
agent
fork
apkill
signal
aprun
apstat
Shared Files
Login Node A
Local
apsys
fork
app
agent
aprun
(PEs 0,1,2)
control socket connection – includes stdout & stderr
fork
stdin handler
stdin
Compute Node
apinit
fork
apsheperd
fork, exec
PE 2










qsub
event router (L1,L0 - SMW)
Service Node
apwatch
pipe
apbridge
System Database (SDB Node)
private port
Compute Node
apinit
fork
apsheperd
fork, exec
PE 1
Login Node C
WLM
fork, exec
apbasil
Local apsys
fork
app agent
Login Shell
aprun
fork
stdin handler
To a Compute Node
apsched (Service or Login Node)
Compute Node
apinit
fork
apsheperd
fork, exec
PE 0
Login Node B
Local apsys
app agent
fork
apkill
signal
aprun
apstat
Shared Files
Compute Node
apinit
fork
apsheperd
fork, exec
PE 2
Login Node A
Local apsys
fork
app agent
aprun (PEs 0,1,2)
control socket connection – includes stdout & stderr
fork
stdin handler
stdin

Q & A

- Questions?
- Thanks to the ALPS team:
  - Richard Lagerstrom - development
  - Marlys Kohnke - development
  - Carl Albing – development
  - Bob Gross - testing
  - Jan Gustafson - our current manager
  - Wayne Margotto - our former manager
- Thank You!