Mingtian SHAO, Kai LU, Wanqing CHI, Ruibo WANG, Yiqin DAI, Wenzhe ZHANG, 2022. TEES: topology-aware execution environment service for fast and agile application deployment in HPC. *Frontiers of Information Technology & Electronic Engineering*, 23(11):1631-1645.
https://doi.org/10.1631/FITEE.2100284

TEES: topology-aware execution environment service for fast and agile application deployment in HPC

Key words: Execution environment; Application deployment; High-performance computing (HPC); Container; Peer-to-peer (P2P); Network topology

Corresponding authors: Kai LU; Wenzhe ZHANG
E-mail: lukainudt@163.com; zhangwenzhe@nudt.edu.cn
ORCID: https://orcid.org/0000-0002-6378-7002;
https://orcid.org/0000-0002-8798-2195

Motivation

1. High-performance computing (HPC) systems have more and more computing resources. Timeliness and convenience are always user concerns.
2. The standard environment cannot support various applications.
3. Deployment overhead and startup latency of the application are not satisfactory, especially when the number of compute nodes is large.

Main idea

1. Based on self-deployed execution environment (SDEE), we design the topology aware execution environment service (TEES) for fast and agile application deployment.

2. TEES combines the advantages of a topology-aware peer-to-peer (P2P) approach with shared storage to greatly reduce the deployment overhead.

3. TEES also reduces the application startup latency through a split-step transport and launch-in advance mechanism.

4. The work has good portability and is suitable for HPC systems.

Method

The framework of TEES

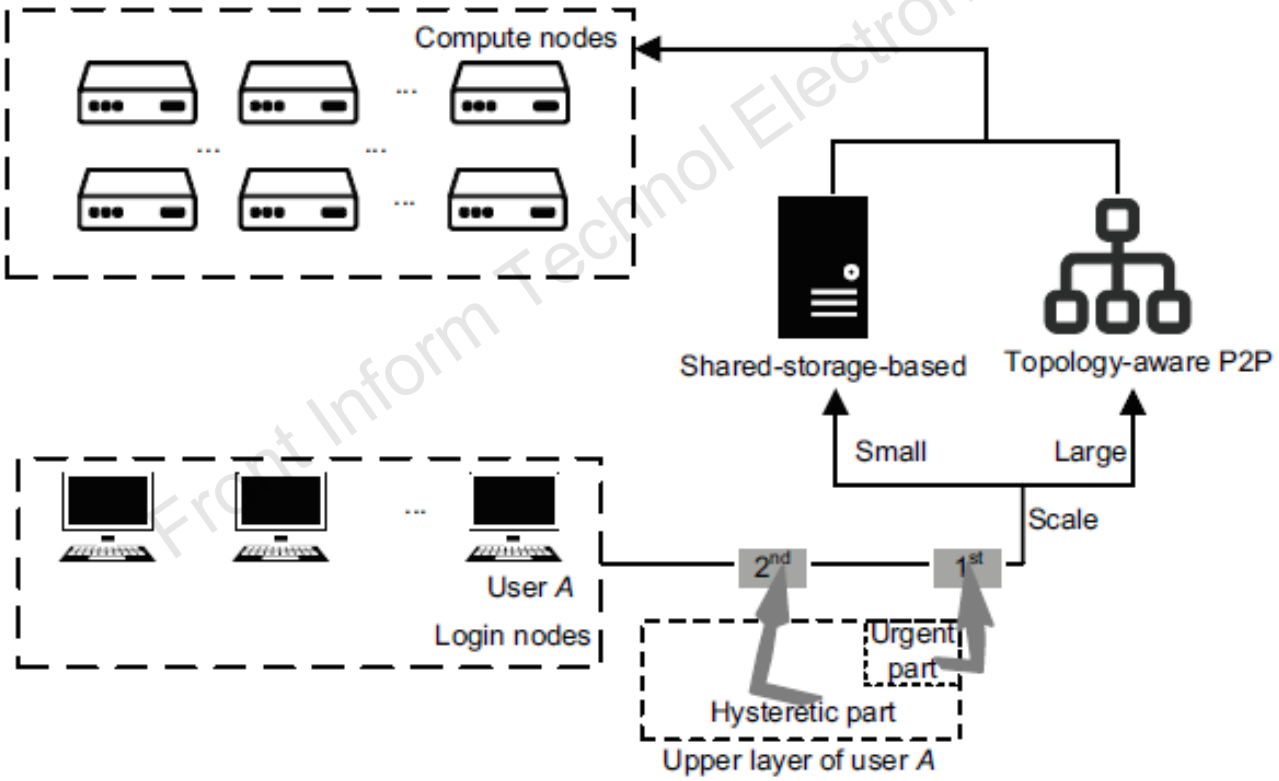

Fig. 2 Framework of the topology-aware execution environment service (TEES)

Method (Cont’d)

1. Execution environment on a node is based on SDEE.
2. Rapid deployment
   - Advantage combination and scheme selection
   - Topology-aware P2P
3. Quick startup
   - Split-step transport and launch-in-advance mechanism
   - File dependencies
4. Portability and scalability
5. Safety and reliability

Method (Cont'd)

Topology-aware P2P

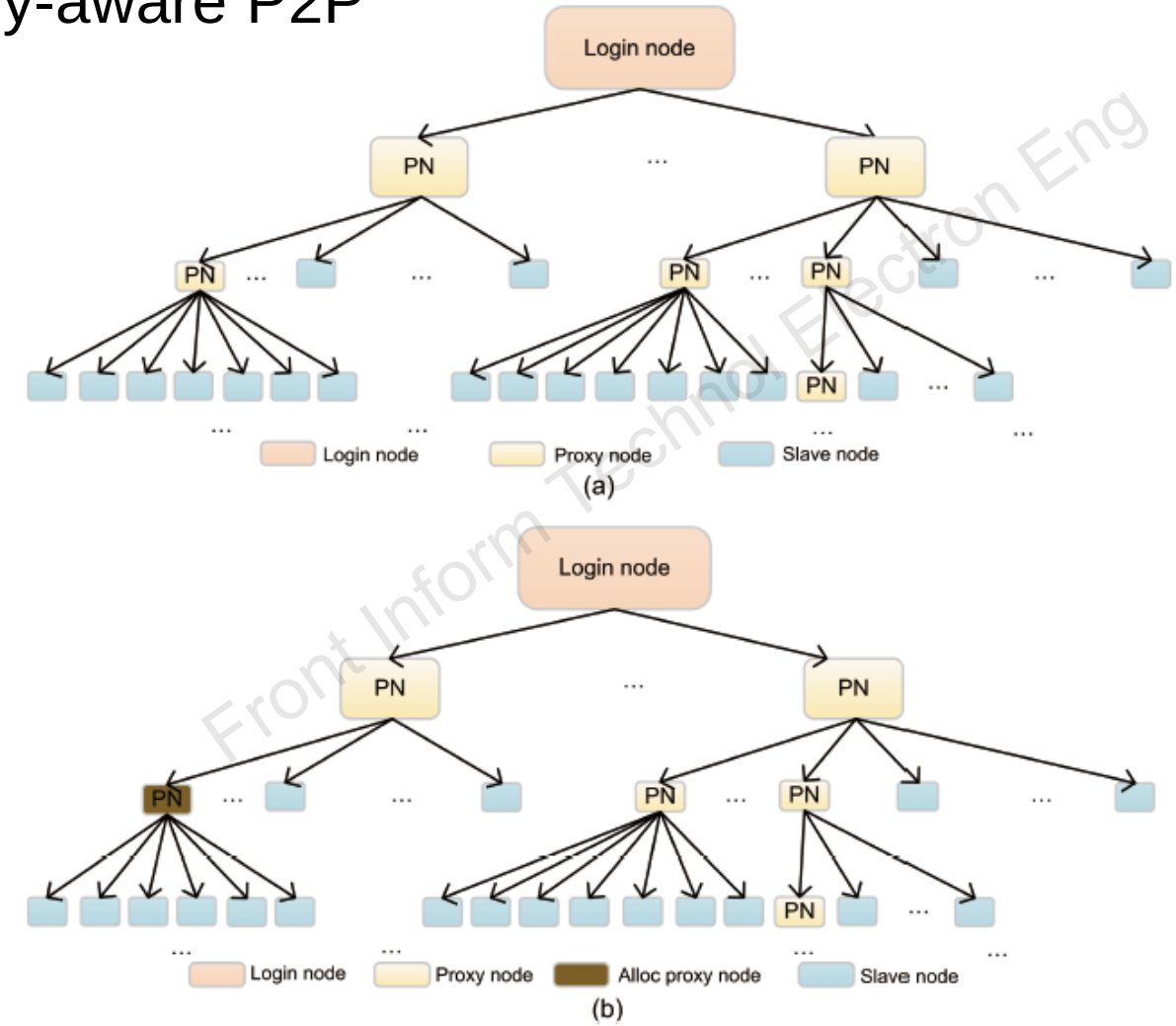

Fig. 3 Structure of the topology-aware peer-to-peer (P2P) tree: (a) structure at the ideal state; (b) structure when P2P tree temporarily occupies the proxy node

Comparison

Table 1 Comparison of TEES with previous works

Performance	TEES	Container-based method (Docker, Singularity, etc.)	Manual work on bare-metal
Main problem addressed	Deployment of application and its execution	Deployment of application and its execution environment	
Size of execution environment	Lightweight upper-layer file system	Cumbersome container image	Small, but scattered
Use mode	Transparent to users	Manually maintain and run the container	Completely manual
Environment configuration permissions	High (configure freely and all these happen in the upper-layer file system)	High (configure freely in the Docker image)	Low (only non-root permission)
Deployment time	Short	Long	Very long
Application startup latency	Low	High	Very high (most of the time is wasted on deployment)
Network load	Low (with topology-aware P2P approach)	High (with typical approach)	High (with typical approach)
SLURM-friendly	Yes	No	No

Major results

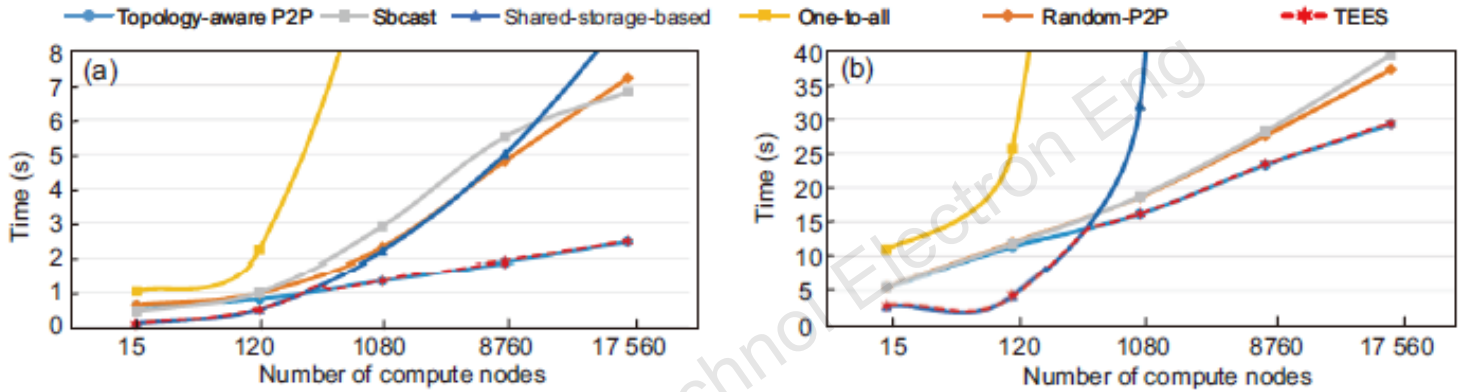

Fig. 4 Deployment time of the six transmission methods when the size of the file deployed is 16 MB (a) and 333 MB (b)

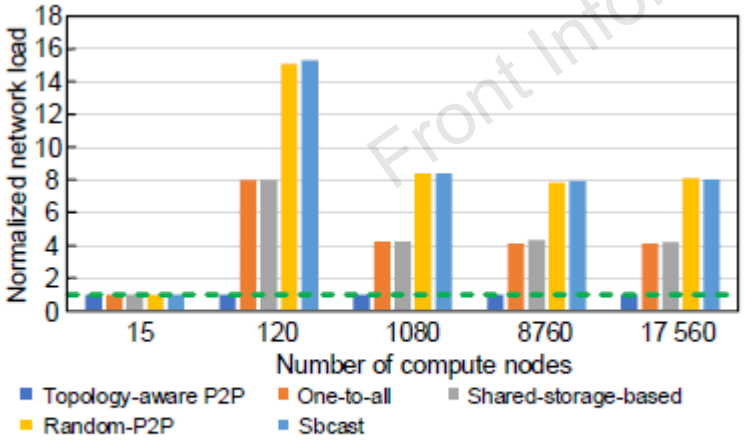

Fig. 5 Network load status of the five transmission methods

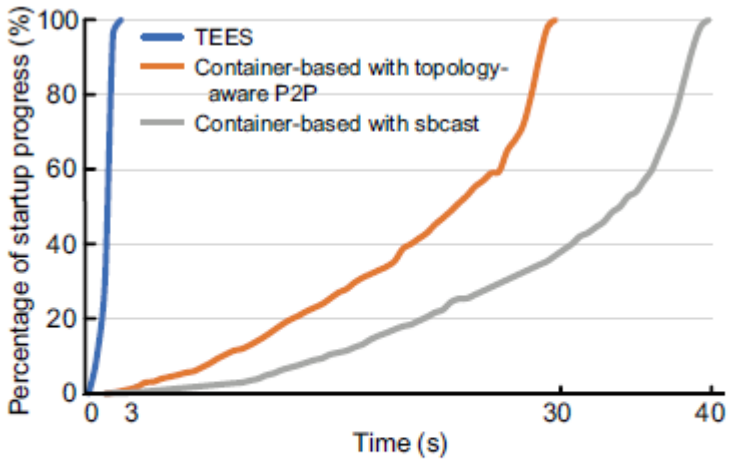

Fig. 6 Application startup latency of the TEES and container-based methods (the number of compute nodes is 17 560)

Conclusions

1. TEES provided a private execution environment for each user in the HPC system, and it realized rapidly automatic deployment of applications and their execution environments.

2. Experimental results showed that TEES is faster and can effectively reduce network load compared to traditional container-based application deployments.