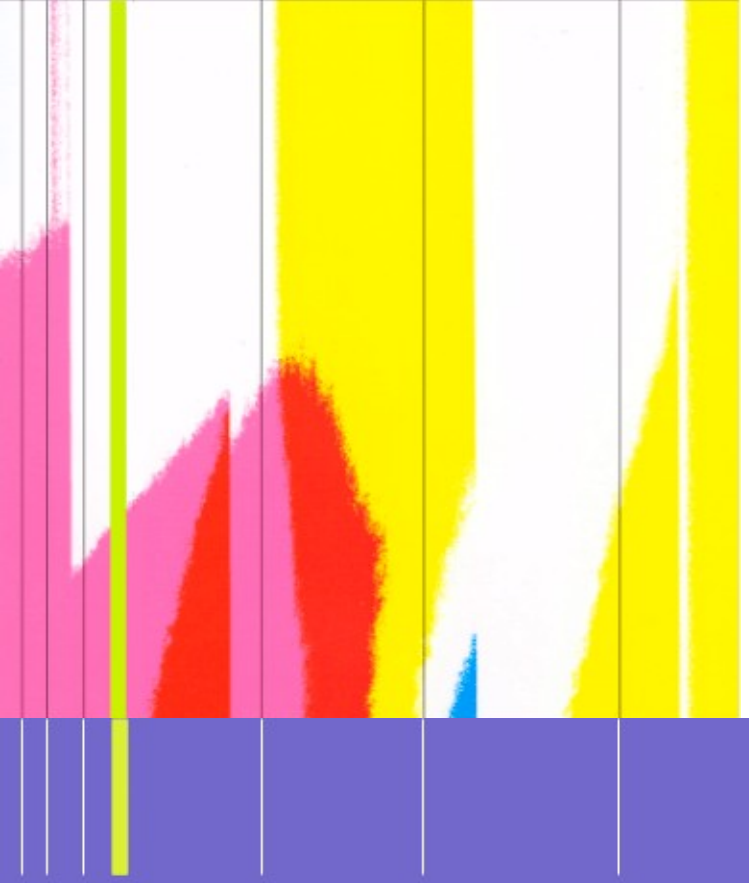


OSCAR KernelPicker: Handling Clients Kernels

Jean Parpaillon
INRIA-IRISA – PARIS Team
May 14, 2007

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

Contents

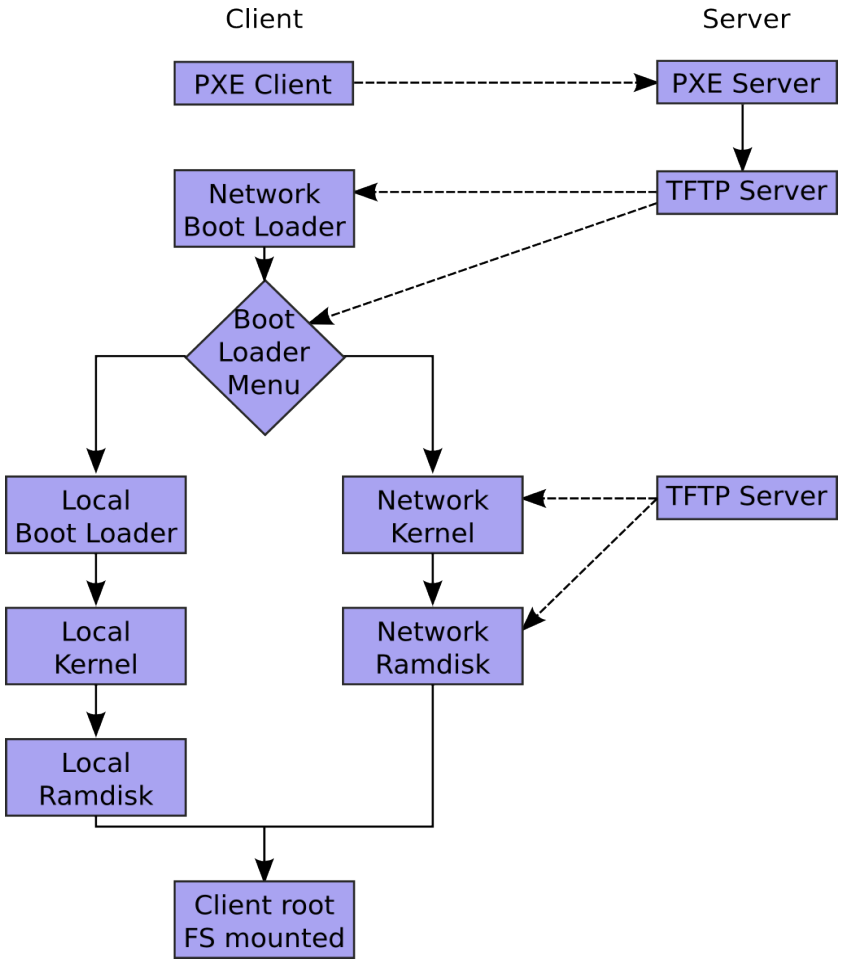
- Introduction
- Boot Management Requirements
- Special Cases
- Existing Solutions
- New KernelPicker Design
- Current Status
- Conclusion

Introduction

- OSCAR
  - deploys standard distribution on client nodes
  - adds cluster specific tools
  - provides cluster wide configuration
- Clients boot process
  - is critical (can make the nodes unavailable)
  - configuration need expertise
  - setup through many different tools

- Need for a single « smart » clients boot manager

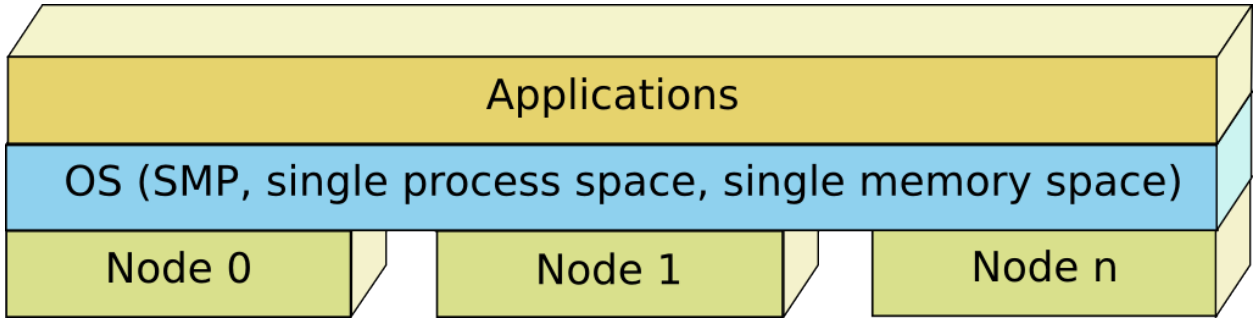
Boot Management Requirements



- Boot methods
  - Network: keep control on nodes
  - Local: nodes independents from server
  - Mix of local and network boot
- Kernel features
  - architecture (x86_64, ia64, etc.)
  - hardware support
  - SSI, virtualization
- Kernel deployment
  - from packages
  - « home made »

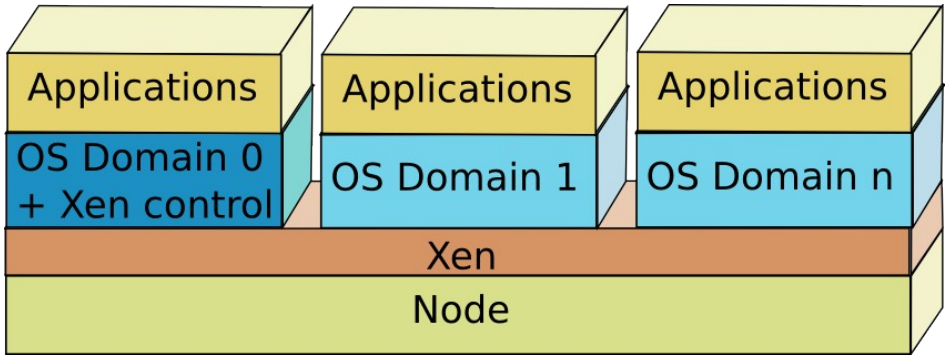
Special Cases - SSI-OSCAR

- Offers the user a unique view of resources
- Based on Kerrighed OS (Linux based)
- Requires variable parameters at boot time (`cluster_id, node_id`)
- Development: easy update of kernel file and boot parameters



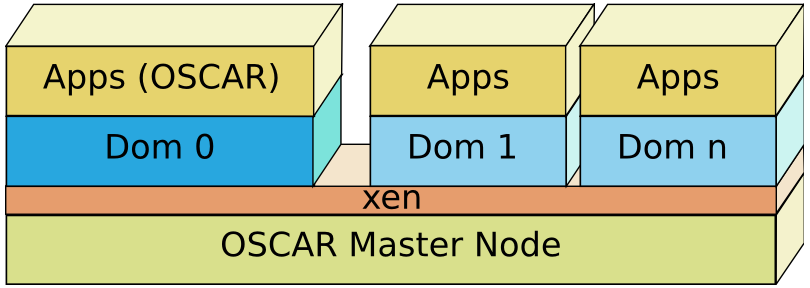
Special Cases – Xen (1/2)

- Paravirtualization solution: host OS + guest OSes
- Booting hypervisor (Xen) + host OS (Linux)
  - at node boot time
  - Xen requires specific parameters
- Booting virtual OSes (Linux)
  - through Xen monitor

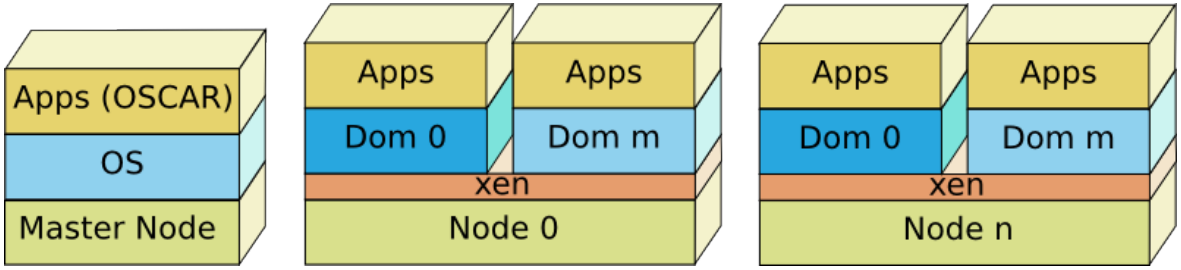


Special Cases – Xen (2/2)

- Xen usages
  - Cluster on master node (testing, development, etc.)



  - Cluster on each cluster nodes (production servers, etc.)



Existing Solutions (1/3)

OSCAR Default

- Master node controls clients boot
- 2 boot entries:
  1) Install mode: network boot with kernel provided by SystemImager
  2) Production mode: local boot with default kernel provided by distribution

UYOK

- SystemImager feature
- Retrieve kernel from running client node
- Production mode use retrieved kernel

Existing Solutions (2/3)

SystemConfigurator

- Setup local boot loader
- Build init RAMdisk

SystemImager Command Line

- Setup network boot

First KernelPicker

- Setup a kernel on an image from files (kernel, modules)
- Can setup install and production modes
- Basic configuration GUI
- Based on SystemImager and SystemConfigurator

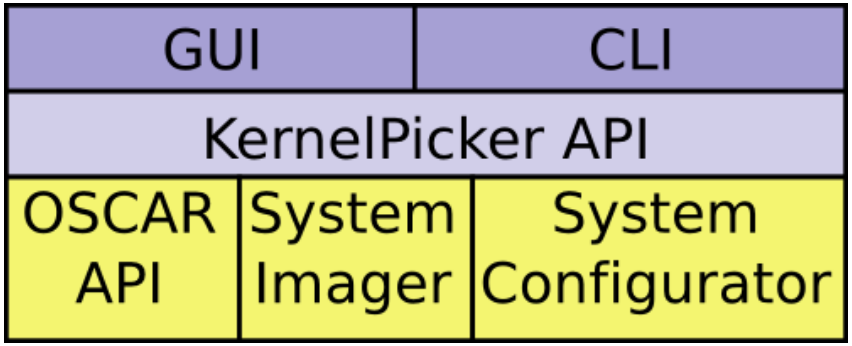
Existing Solutions (3/3)

	Home-made kernel	Kernel package	SSI Config	Xen Config	Boot parameters edition	UI
OSCAR Default	No	Yes	No	No	No	No
UYOK	Yes	No	No	No	No	CLI/GUI
SystemConfigurator	No	No	Yes	No	Yes	No
SystemImager	No	No	No	No	No	CLI/GUI
First KernelPicker	Yes	No	No	No	No	CLI/GUI

- Heterogeneous tools
- No Xen support
- No global UI

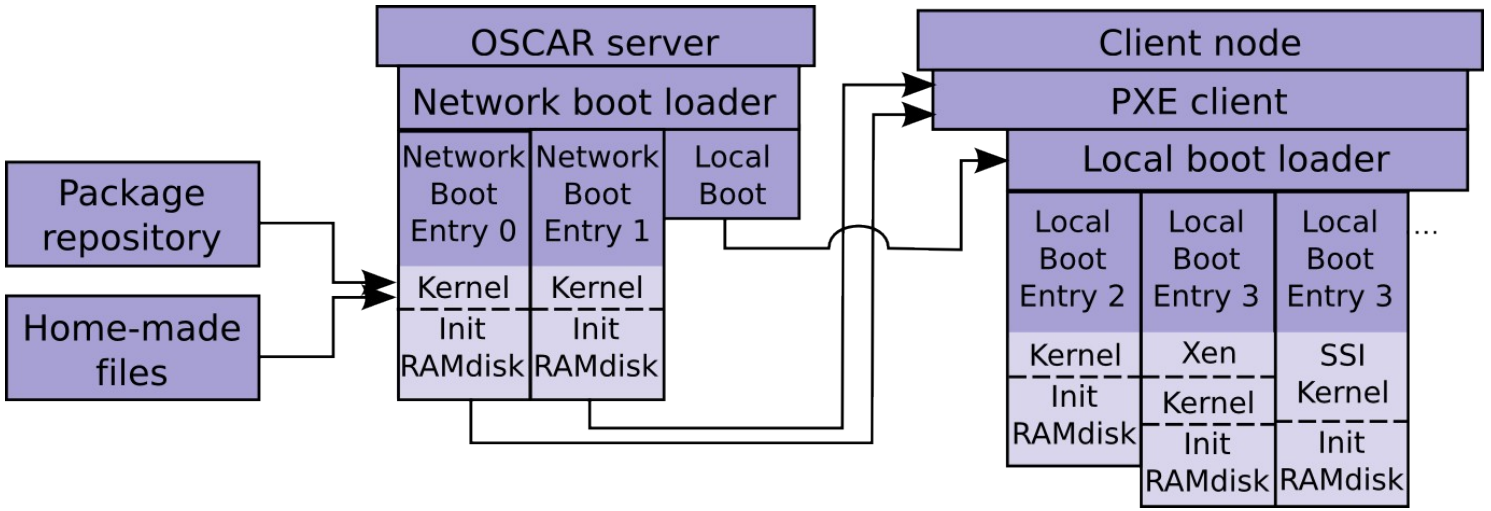
New KernelPicker Design - Architecture

- A unique tool to manage clients boot process
- Based on existing tools
- 3 layers:
  - Backends: existing tools and APIs
  - API: object oriented
  - UI: GUI and CLI (integrated into OSCAR ones)



New KernelPicker Design - Model

- Kernel origin: packages, “home-made” (files, Linux source tree)
- Kernel package selection from features
- Boot method: network, network + local
- SSI handling
- Xen handling (on client nodes)

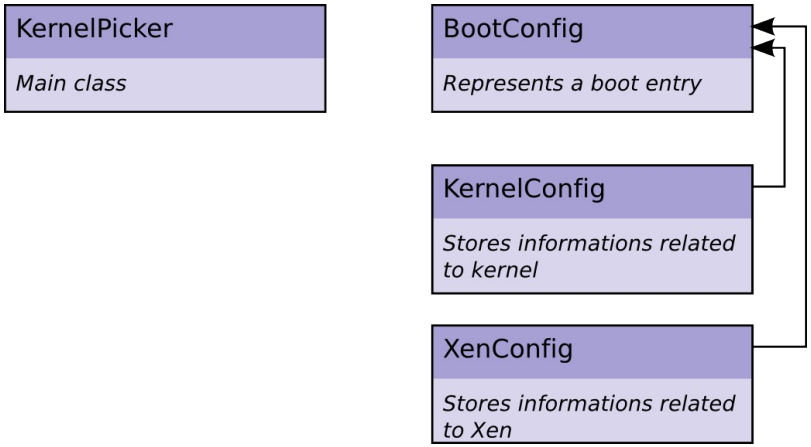


New KernelPicker Design - Backends

- OSCAR API
  - Multi distribution access to packages repositories
- SystemImager
  - Access to images and clients informations
  - PXE setup
- SystemConfigurator
  - Local boot loader setup
    - SSI boot parameters handling
    - Xen boot parameters handling
  - Init RAMdisk setup

New KernelPicker Design - API

- *KernelPicker:* main class
  - Get informations about kernel related packages
  - Setup boot configurations
  - Get informations about boot configurations
- *BootConfig*
- *KernelConfig*
- *XenConfig*
  - *Store informations about boot configurations*

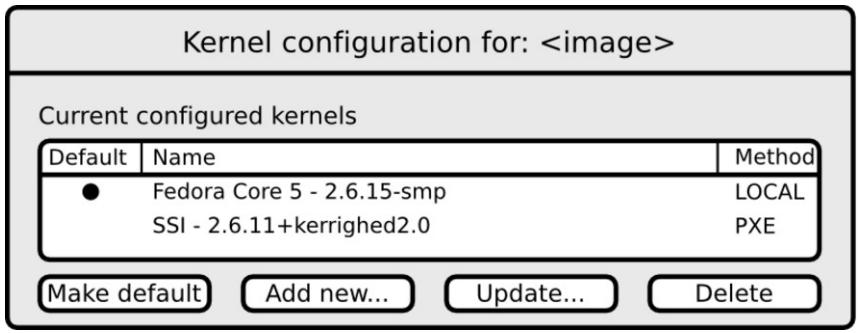


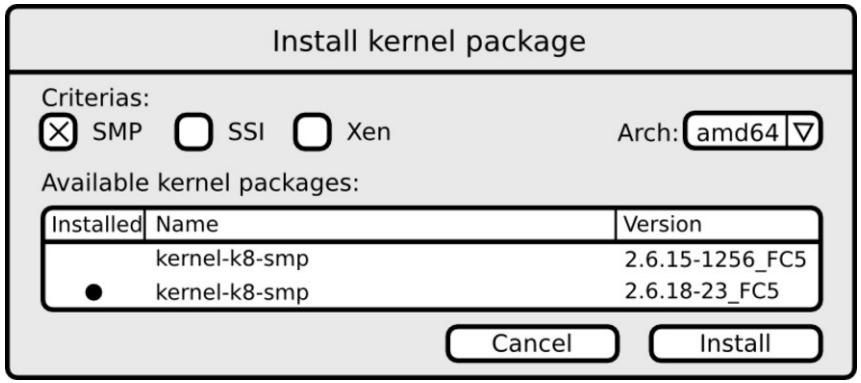
New KernelPicker Design - CLI

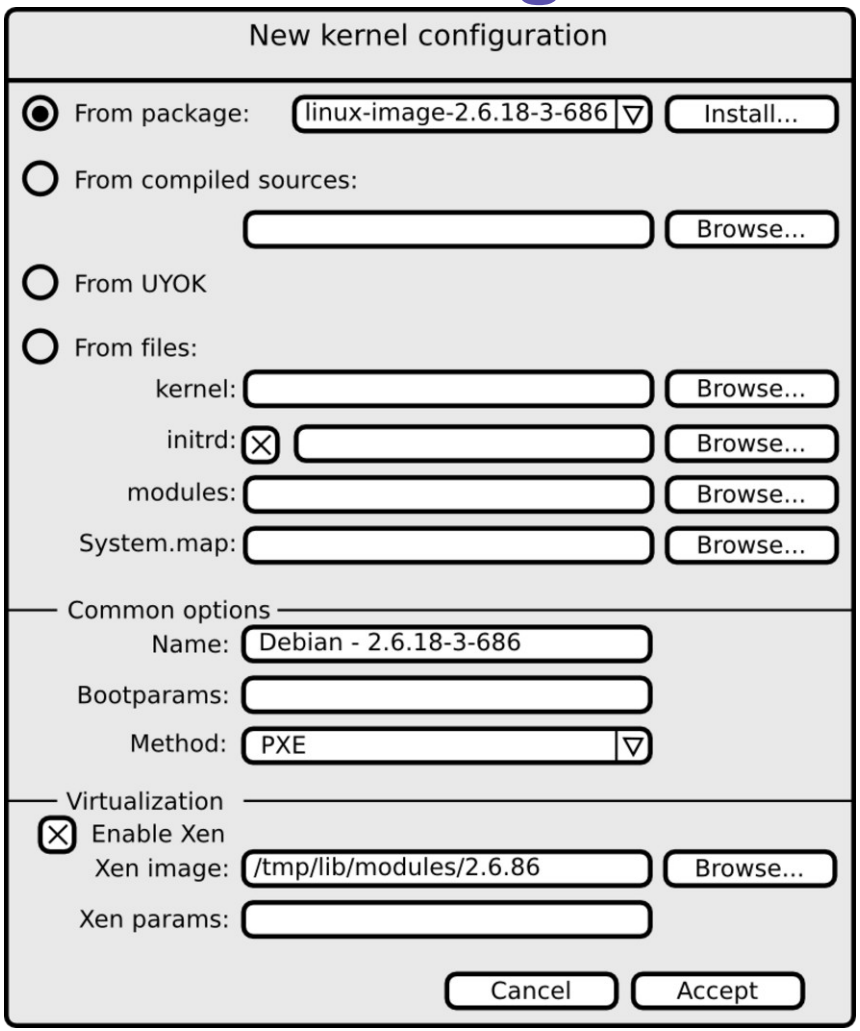
- A single command to
  - list kernel related packages
  - get boot entries related to an image
  - setup a new boot entry for an image
  - update an existing boot entry for an image

New KernelPicker Design - GUI

- Integrated into OSCAR wizard







Current Status

- Not fully implemented
- SystemConfigurator
  - Updated to handle Xen configurations (Geoffroy Vallée, ORNL)
- Network boot entries GUI: *netbootmgr* (Erich Focht, NEC)

Conclusion

- Current client boot process management in OSCAR:
  - Simple, use default values
  - Advanced configuration done through heterogeneous tools
- New KernelPicker
  - Offers a single UI
  - Handles new cases
    - SSI
    - Xen (clusters on client nodes)
- Future
  - Diskless clusters
  - Xen (cluster on master node, etc.)

Questions ?