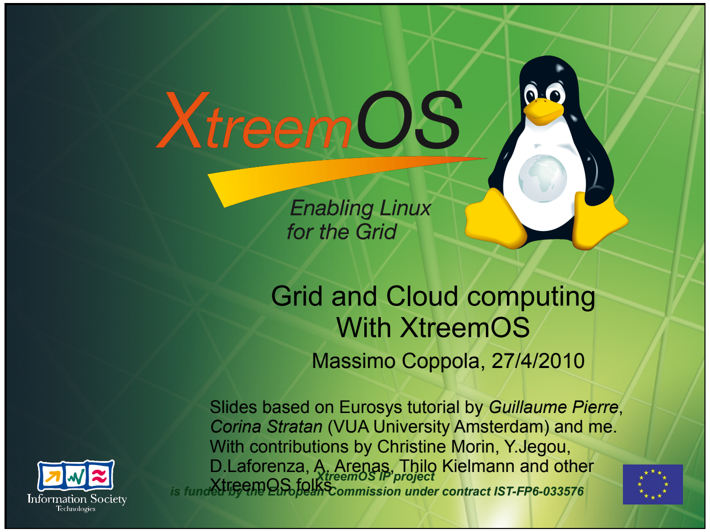


XtreemOS

Enabling Linux for the Grid

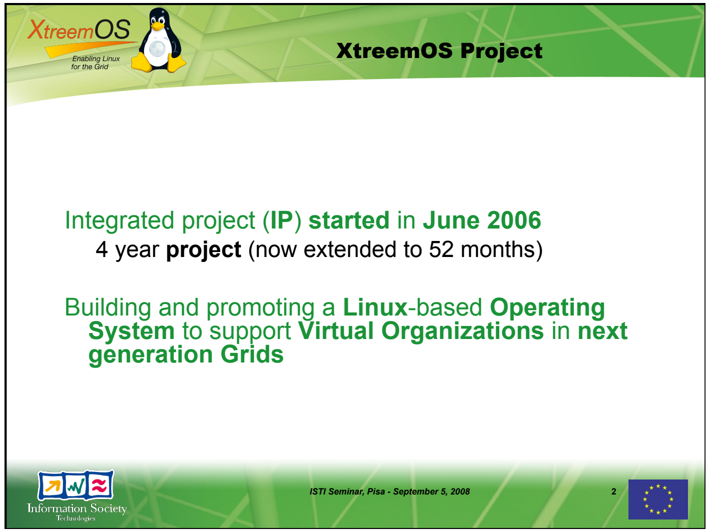
Grid and Cloud computing With XtreemOS

Massimo Coppola, 27/4/2010

Slides based on Eurosys tutorial by *Guillaume Pierre*, *Corina Stratan* (VUA University Amsterdam) and me. With contributions by Christine Morin, Y.Jegou, D.Laforenza, A. Arenas, Thilo Kielmann and other XtreemOS folks

XtreemOS IP project is funded by the European Commission under contract IST-FP6-033576

Information Society Technologies



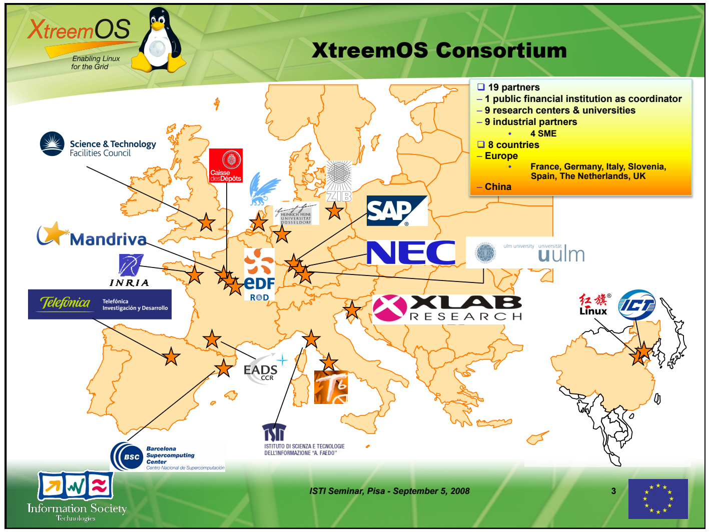
XtreemOS Enabling Linux for the Grid

XtreemOS Project

Integrated project (**IP**) **started** in **June 2006**
4 year **project** (now extended to 52 months)

Building and promoting a **Linux**-based **Operating System** to support **Virtual Organizations** in **next generation Grids**

Information Society Technologies



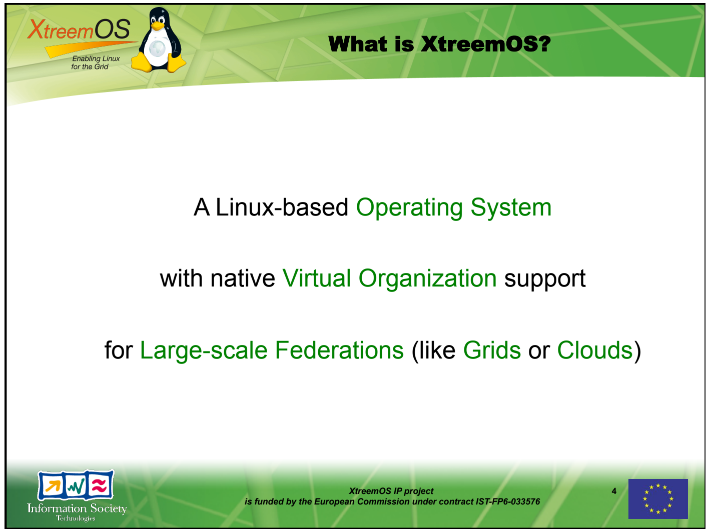
XtreemOS Enabling Linux for the Grid

XtreemOS Consortium

Science & Technology Facilities Council
Caisse des Dépôts
Mandriva
INRIA
Telefónica Investigación y Desarrollo
EDF R&D
SAP
NEC
ulm university universität uulm
XLAB RESEARCH
红旗 Linux
ICT
EADS CCR
Barcelona Supercomputing Center Centro Nacional de Supercomputación
ISTI ISTITUTO DI SCIENZA E TECNOLOGIE DELL'INFORMAZIONE "A. FAEDO"

- ❑ 19 partners
  - – 1 public financial institution as coordinator
  - – 9 research centers & universities
  - – 9 industrial partners
    - 4 SME
- ❑ 8 countries
  - – Europe
    - France, Germany, Italy, Slovenia, Spain, The Netherlands, UK
  - – China

Information Society Technologies



XtreemOS Enabling Linux for the Grid

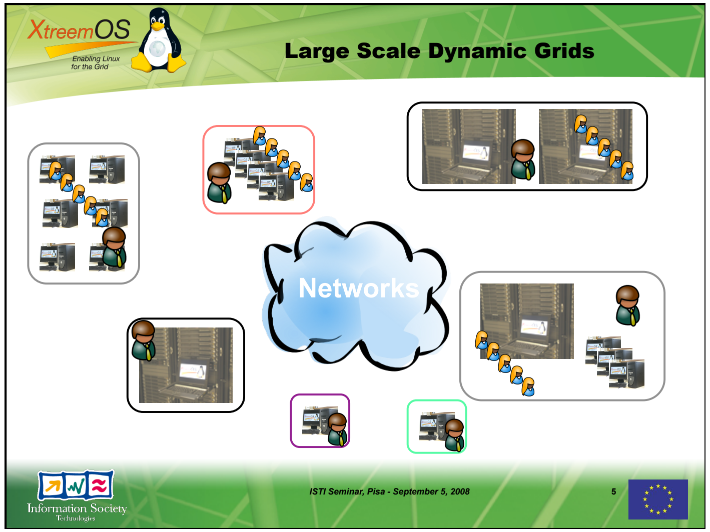
What is XtreemOS?

A Linux-based Operating System

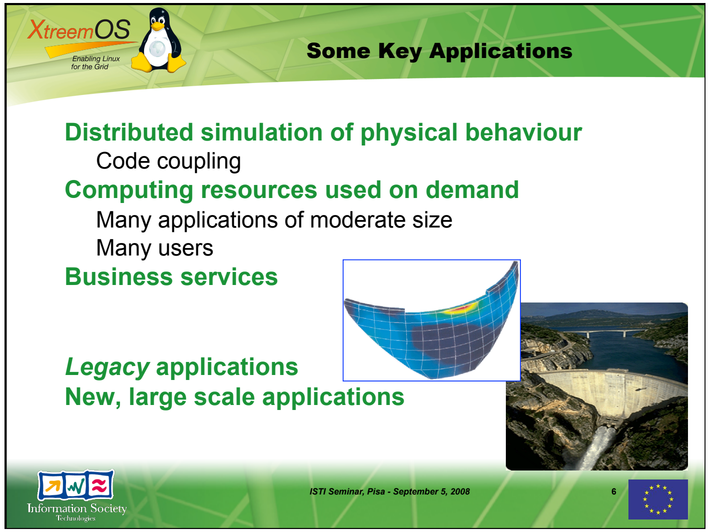
with native Virtual Organization support

for Large-scale Federations (like Grids or Clouds)

Information Society Technologies

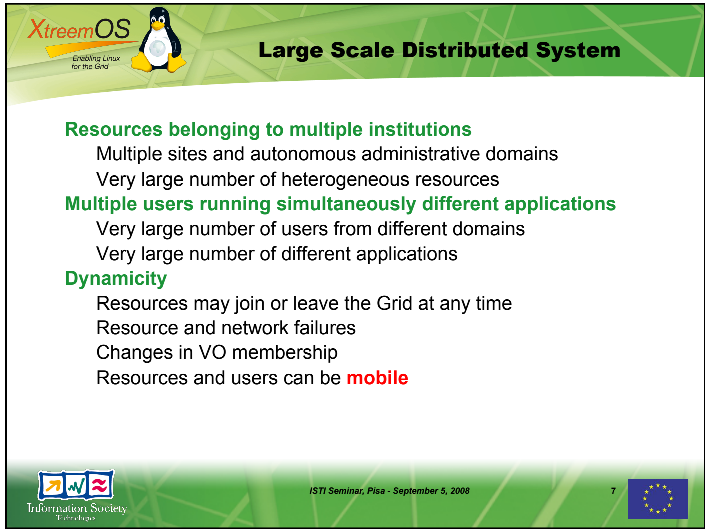


Large Scale Dynamic Grids



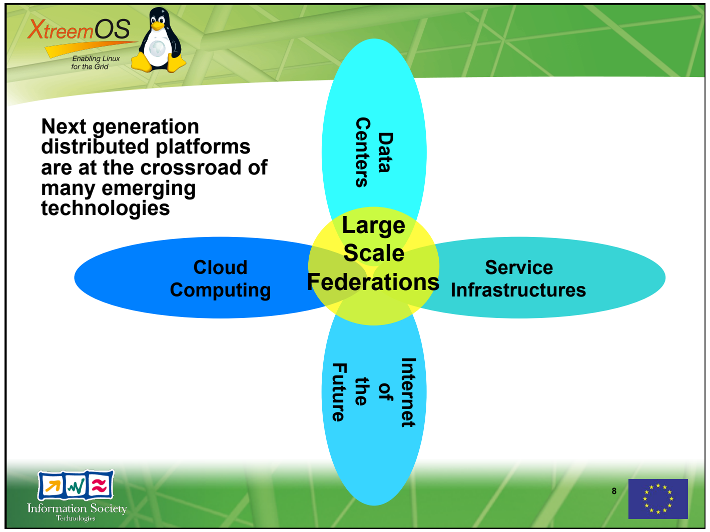
Some Key Applications

- Distributed simulation of physical behaviour
  - Code coupling
- Computing resources used on demand
  - Many applications of moderate size
  - Many users
- Business services
- *Legacy* applications
- New, large scale applications



Large Scale Distributed System

- Resources belonging to multiple institutions
  - Multiple sites and autonomous administrative domains
  - Very large number of heterogeneous resources
- Multiple users running simultaneously different applications
  - Very large number of users from different domains
  - Very large number of different applications
- Dynamicity
  - Resources may join or leave the Grid at any time
  - Resource and network failures
  - Changes in VO membership
  - Resources and users can be **mobile**



Next generation distributed platforms are at the crossroad of many emerging technologies

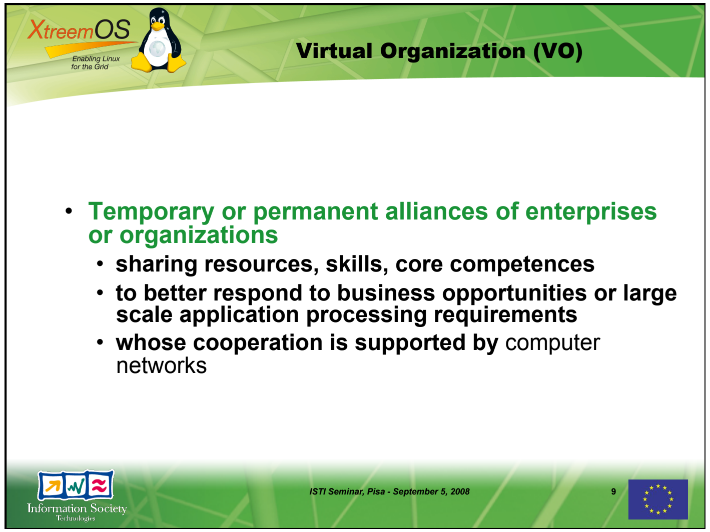
Data Centers

Cloud Computing

Large Scale Federations

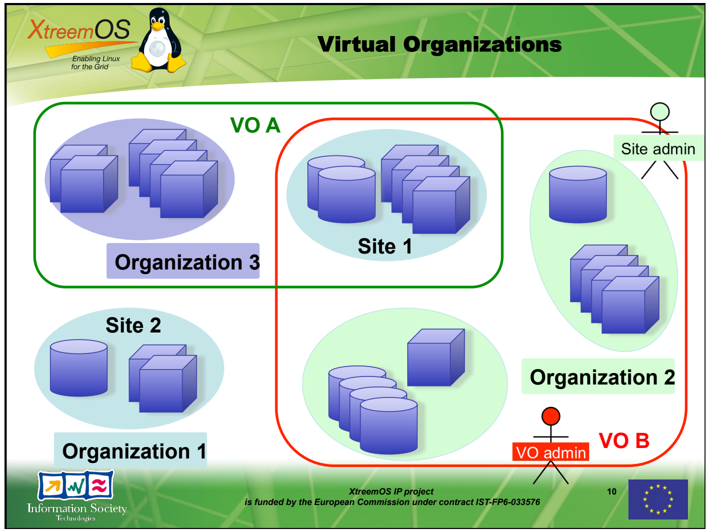
Service Infrastructures

Internet of the Future



Virtual Organization (VO)

- Temporary or permanent alliances of enterprises or organizations
  - sharing resources, skills, core competences
  - to better respond to business opportunities or large scale application processing requirements
  - whose cooperation is supported by computer networks



Virtual Organizations

VO A

Site admin

Organization 3

Site 1

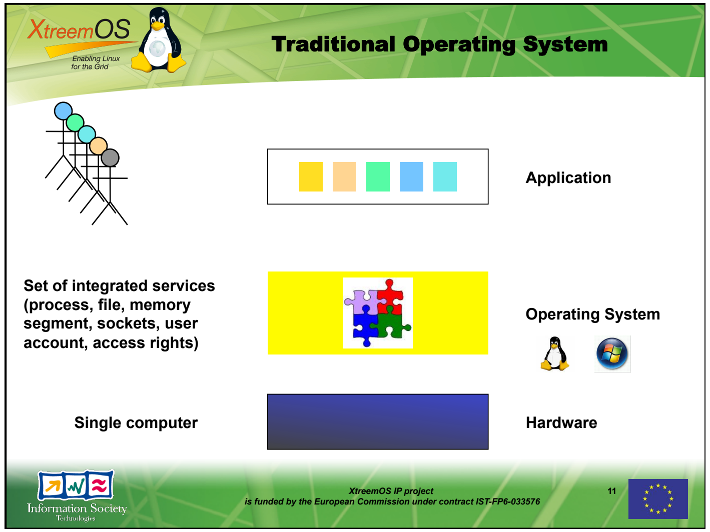
Site 2

Organization 2

Organization 1

VO admin

VO B



Traditional Operating System

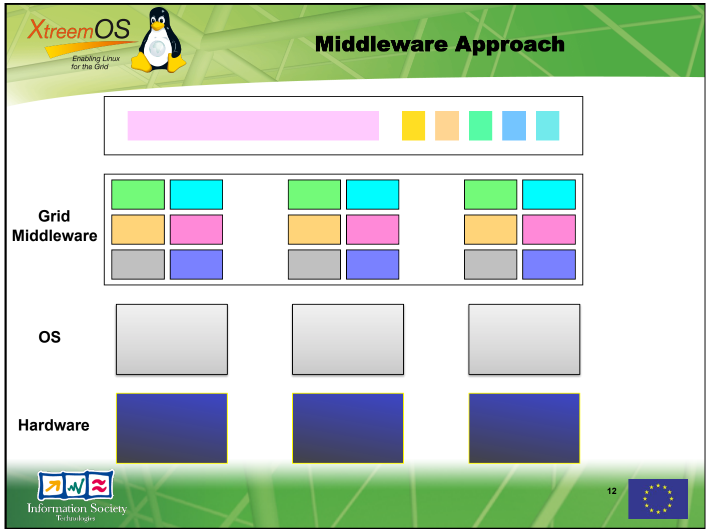
Application

Set of integrated services (process, file, memory segment, sockets, user account, access rights)

Operating System

Single computer

Hardware

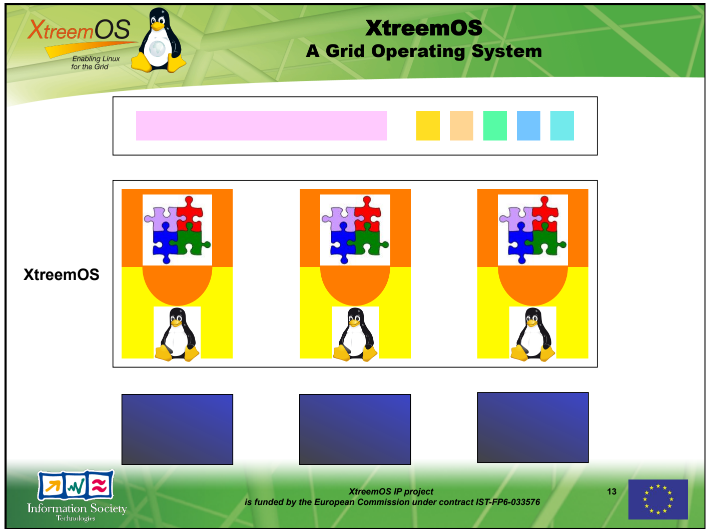


Middleware Approach

Grid Middleware

OS

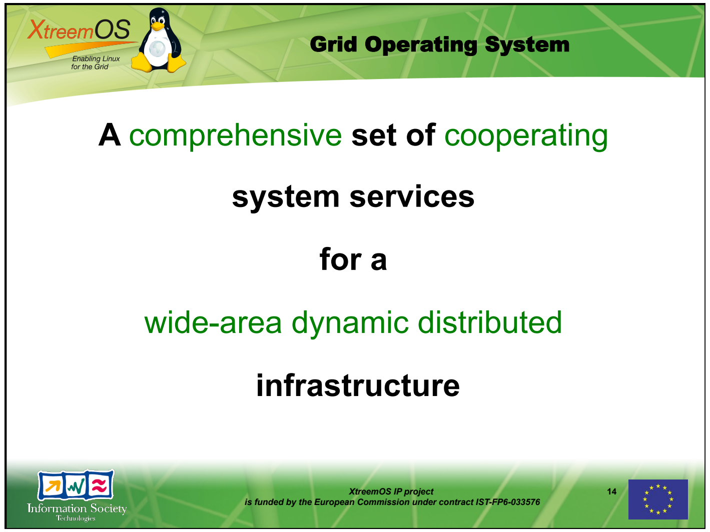
Hardware



XtreemOS
A Grid Operating System

XtreemOS

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576



Grid Operating System

A comprehensive set of cooperating

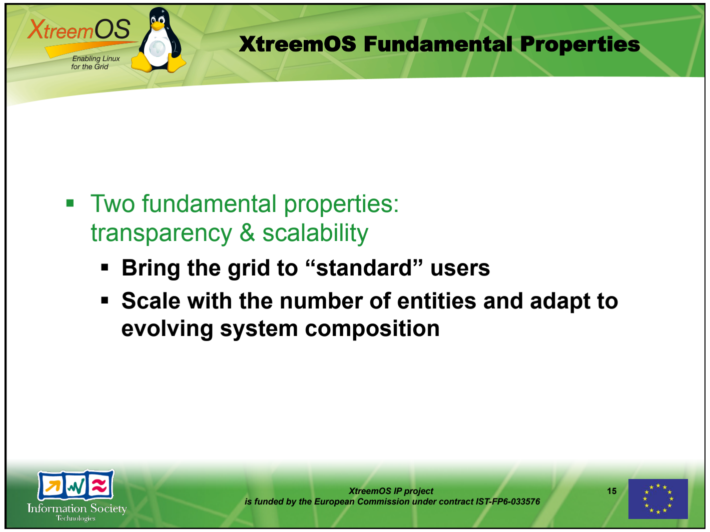
system services

for a

wide-area dynamic distributed

infrastructure

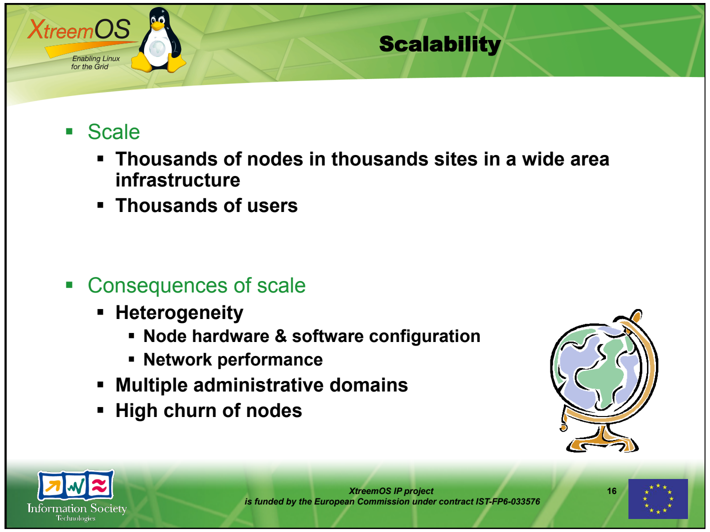
XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576



XtreemOS Fundamental Properties

- Two fundamental properties:
  transparency & scalability
  - Bring the grid to “standard” users
  - Scale with the number of entities and adapt to evolving system composition

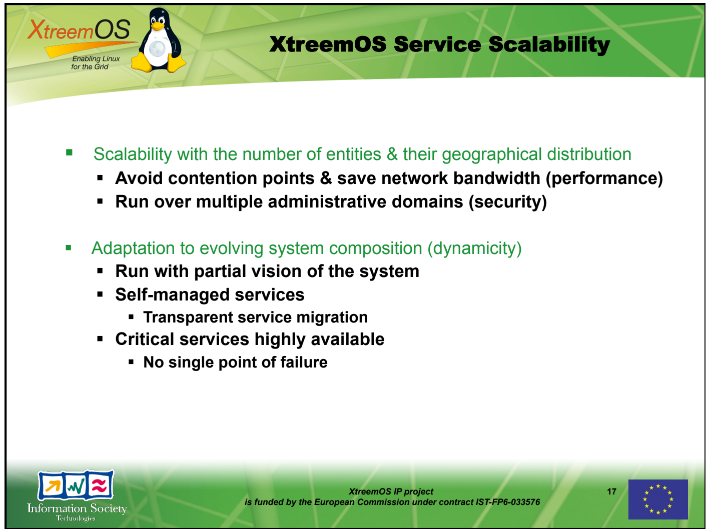
XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576



Scalability

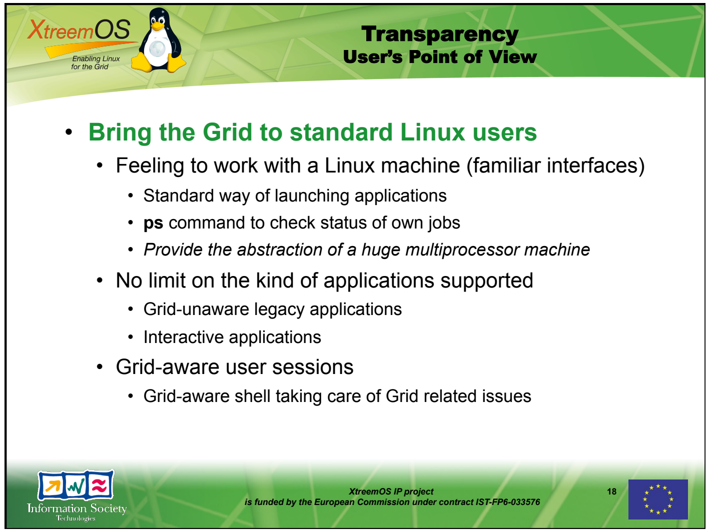
- Scale
  - Thousands of nodes in thousands sites in a wide area infrastructure
  - Thousands of users
- Consequences of scale
  - Heterogeneity
    - Node hardware & software configuration
    - Network performance
  - Multiple administrative domains
  - High churn of nodes

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576



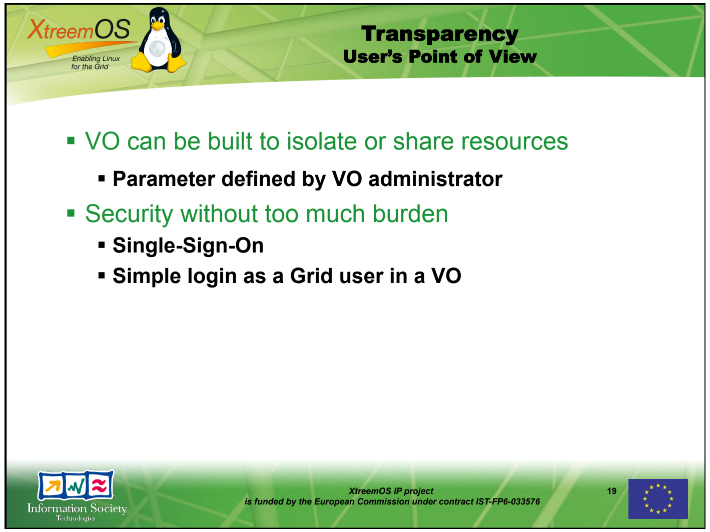
XtreemOS Service Scalability

- Scalability with the number of entities & their geographical distribution
  - **Avoid contention points & save network bandwidth (performance)**
  - **Run over multiple administrative domains (security)**
- Adaptation to evolving system composition (dynamicity)
  - **Run with partial vision of the system**
  - **Self-managed services**
    - **Transparent service migration**
  - **Critical services highly available**
    - **No single point of failure**



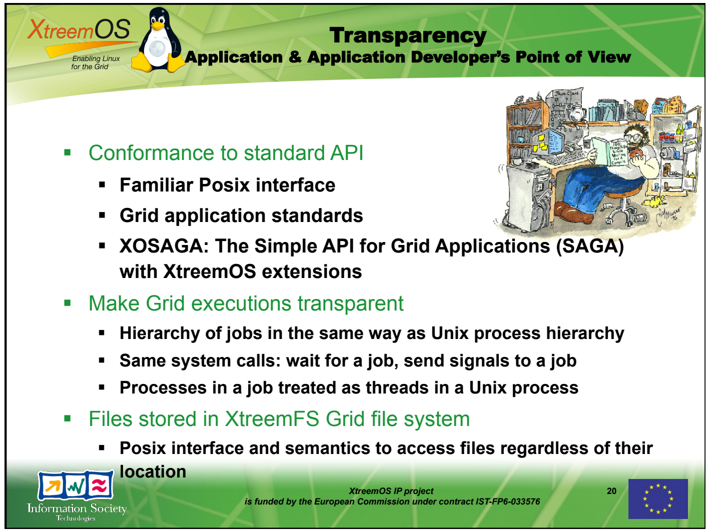
Transparency
User's Point of View

- **Bring the Grid to standard Linux users**
  - Feeling to work with a Linux machine (familiar interfaces)
    - Standard way of launching applications
    - **ps** command to check status of own jobs
    - *Provide the abstraction of a huge multiprocessor machine*
  - No limit on the kind of applications supported
    - Grid-unaware legacy applications
    - Interactive applications
  - Grid-aware user sessions
    - Grid-aware shell taking care of Grid related issues



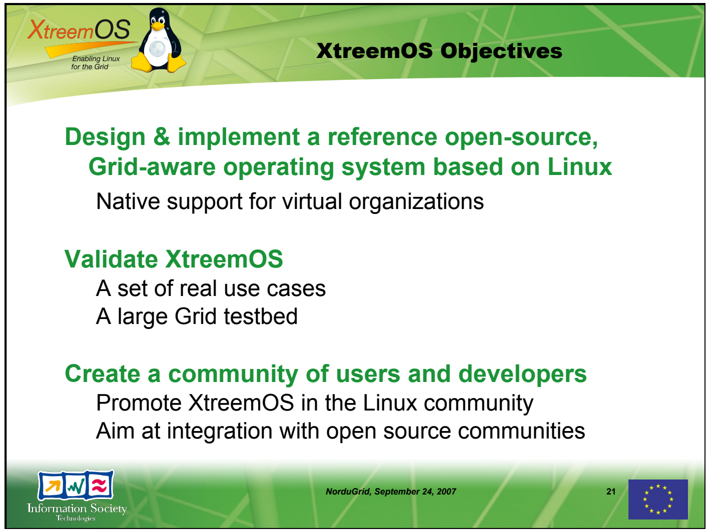
Transparency
User's Point of View

- VO can be built to isolate or share resources
  - **Parameter defined by VO administrator**
- Security without too much burden
  - **Single-Sign-On**
  - **Simple login as a Grid user in a VO**



Transparency
Application & Application Developer's Point of View

- Conformance to standard API
  - **Familiar Posix interface**
  - **Grid application standards**
  - **XOSAGA: The Simple API for Grid Applications (SAGA) with XtreemOS extensions**
- Make Grid executions transparent
  - **Hierarchy of jobs in the same way as Unix process hierarchy**
  - **Same system calls: wait for a job, send signals to a job**
  - **Processes in a job treated as threads in a Unix process**
- Files stored in XtreemFS Grid file system
  - **Posix interface and semantics to access files regardless of their location**



XtreemOS
Enabling Linux for the Grid

XtreemOS Objectives

Design & implement a reference open-source, Grid-aware operating system based on Linux

Native support for virtual organizations

Validate XtreemOS

A set of real use cases

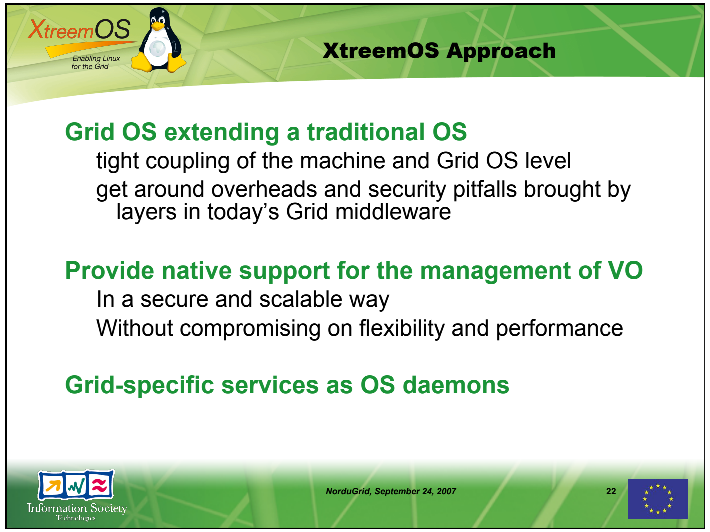
A large Grid testbed

Create a community of users and developers

Promote XtreemOS in the Linux community

Aim at integration with open source communities

Information Society Technologies



XtreemOS
Enabling Linux for the Grid

XtreemOS Approach

Grid OS extending a traditional OS

tight coupling of the machine and Grid OS level

get around overheads and security pitfalls brought by layers in today's Grid middleware

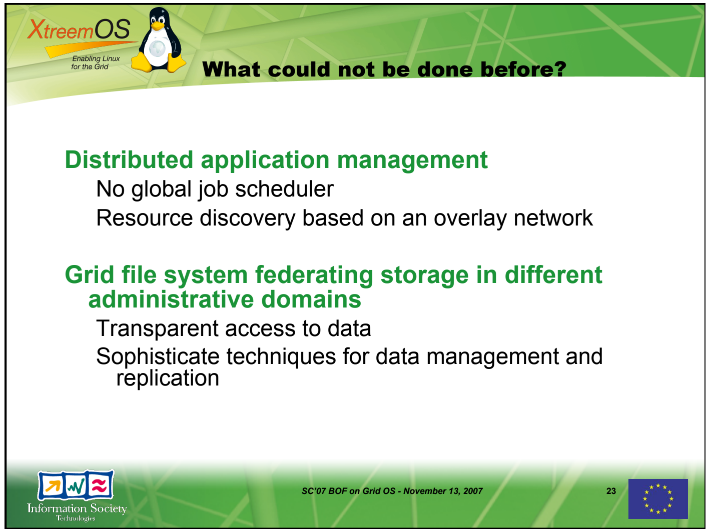
Provide native support for the management of VO

In a secure and scalable way

Without compromising on flexibility and performance

Grid-specific services as OS daemons

Information Society Technologies



XtreemOS
Enabling Linux for the Grid

What could not be done before?

Distributed application management

No global job scheduler

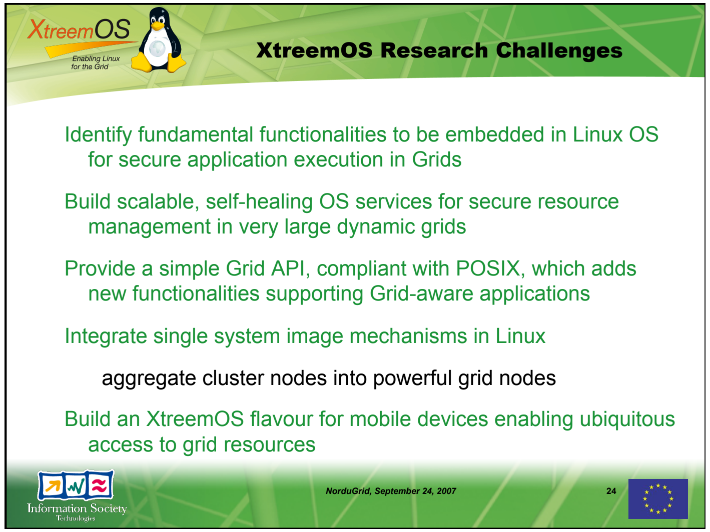
Resource discovery based on an overlay network

Grid file system federating storage in different administrative domains

Transparent access to data

Sophisticate techniques for data management and replication

Information Society Technologies



XtreemOS
Enabling Linux for the Grid

XtreemOS Research Challenges

Identify fundamental functionalities to be embedded in Linux OS for secure application execution in Grids

Build scalable, self-healing OS services for secure resource management in very large dynamic grids

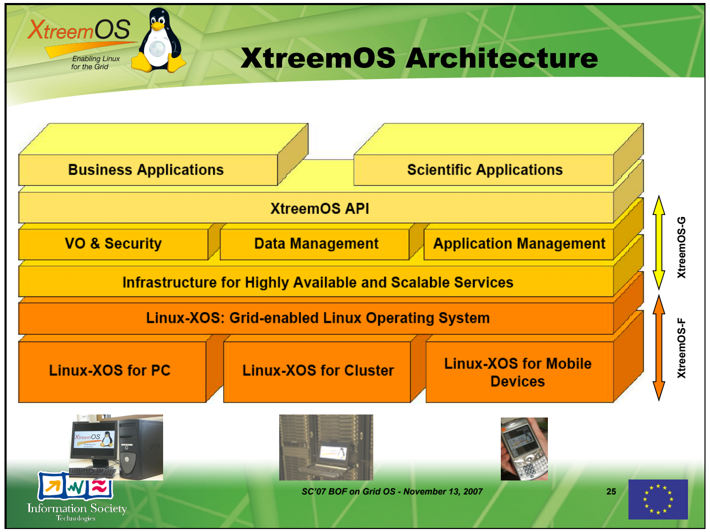
Provide a simple Grid API, compliant with POSIX, which adds new functionalities supporting Grid-aware applications

Integrate single system image mechanisms in Linux

aggregate cluster nodes into powerful grid nodes

Build an XtreemOS flavour for mobile devices enabling ubiquitous access to grid resources

Information Society Technologies



XtreemOS Architecture

Business Applications

Scientific Applications

XtreemOS API

VO & Security

Data Management

Application Management

Infrastructure for Highly Available and Scalable Services

Linux-XOS: Grid-enabled Linux Operating System

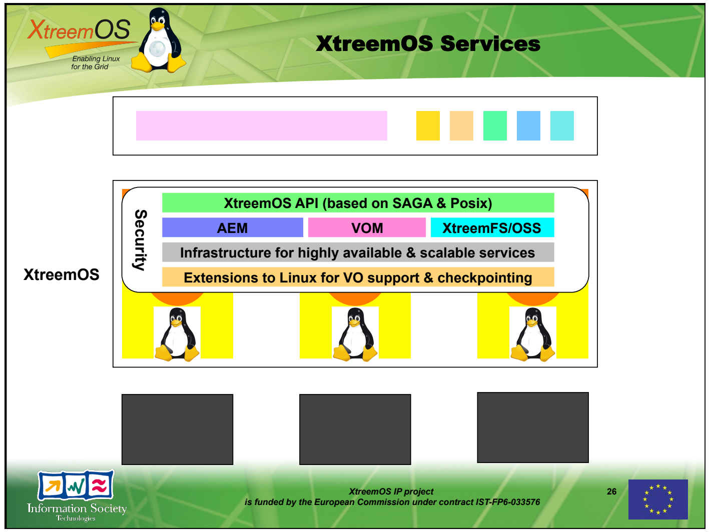
Linux-XOS for PC

Linux-XOS for Cluster

Linux-XOS for Mobile Devices

XtreemOS-G

XtreemOS-F



XtreemOS Services

XtreemOS

Security

XtreemOS API (based on SAGA & Posix)

AEM	VOM	XtreemFS/OSS

Infrastructure for highly available & scalable services

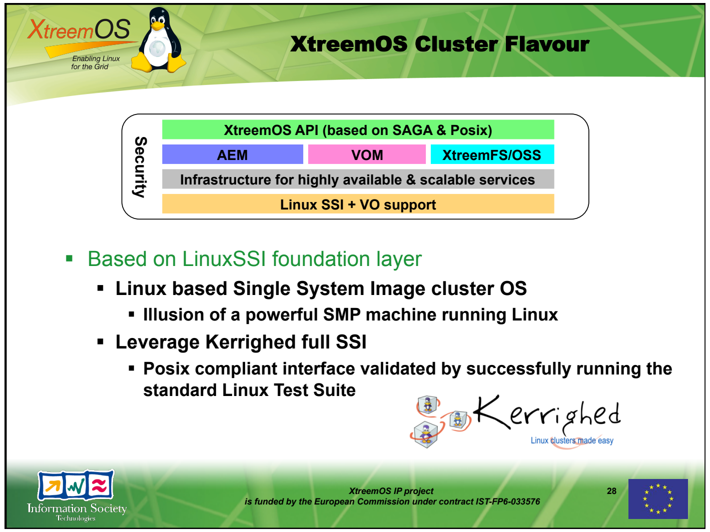
Extensions to Linux for VO support & checkpointing

XtreemOS Flavours

Stand-alone PC

Cluster

Mobile device



XtreemOS Cluster Flavour

Security

XtreemOS API (based on SAGA & Posix)

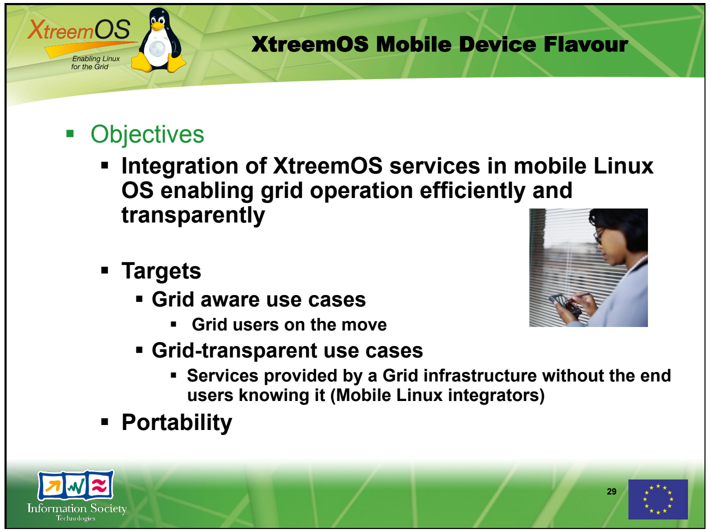
AEM	VOM	XtreemFS/OSS

Infrastructure for highly available & scalable services

Linux SSI + VO support

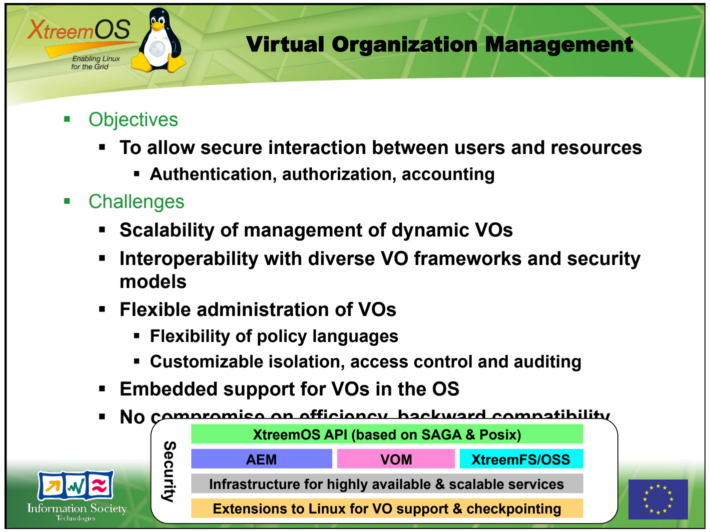
- Based on LinuxSSI foundation layer
  - Linux based Single System Image cluster OS
    - Illusion of a powerful SMP machine running Linux
  - Leverage Kerrighed full SSI
    - Posix compliant interface validated by successfully running the standard Linux Test Suite

Kerrighed
Linux clusters made easy



XtreemOS Mobile Device Flavour

- Objectives
  - Integration of XtreemOS services in mobile Linux OS enabling grid operation efficiently and transparently
  - Targets
    - Grid aware use cases
      - Grid users on the move
    - Grid-transparent use cases
      - Services provided by a Grid infrastructure without the end users knowing it (Mobile Linux integrators)
  - Portability



Virtual Organization Management

- Objectives
  - To allow secure interaction between users and resources
    - Authentication, authorization, accounting
- Challenges
  - Scalability of management of dynamic VOs
  - Interoperability with diverse VO frameworks and security models
  - Flexible administration of VOs
    - Flexibility of policy languages
    - Customizable isolation, access control and auditing
  - Embedded support for VOs in the OS
  - No compromise on efficiency, backward compatibility

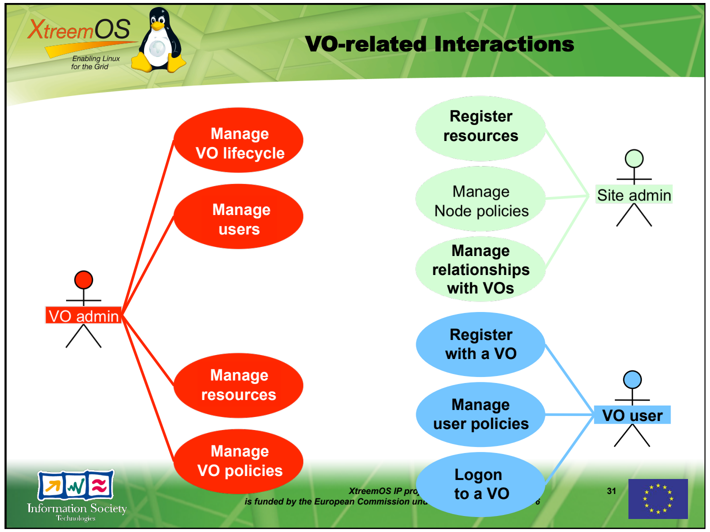
Security

XtreemOS API (based on SAGA & Posix)

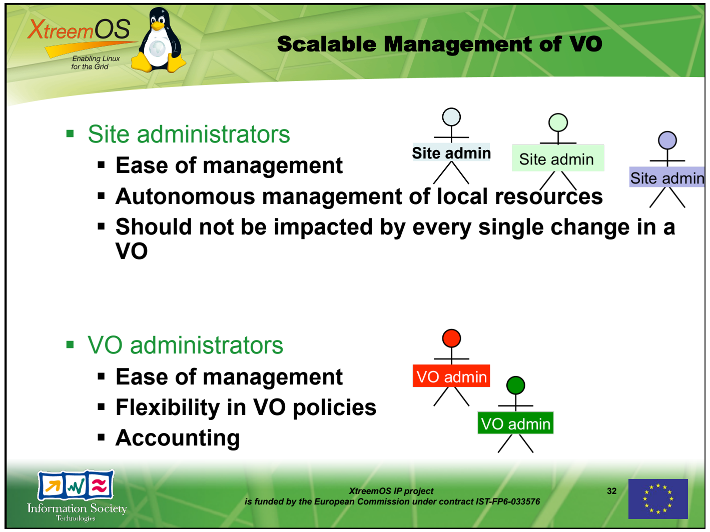
AEM | VOM | XtreemFS/OSS

Infrastructure for highly available & scalable services

Extensions to Linux for VO support & checkpointing



VO-related Interactions



Scalable Management of VO

- Site administrators
  - Ease of management
  - Autonomous management of local resources
  - Should not be impacted by every single change in a VO
- VO administrators
  - Ease of management
  - Flexibility in VO policies
  - Accounting

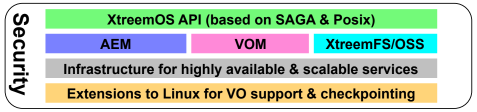
Site admin
Site admin
Site admin
VO admin
VO admin

Summary: XtreemOS VO Management

- Maximum transparency
  - Grid unaware applications & tools can be used without being modified or recompiled
- Integration of Grid level authentication with node level authentication
  - Creation of dynamic on-the-fly mappings for Grid users in a clean & scalable way
  - No centralized Grid wide data base
- Grid user mappings invisible to local users
- VO's are easy to setup and manage
  - No grid map file needed
  - Independent user and resource management
    - User management does not necessitate any resource reconfiguration

Application Execution Management

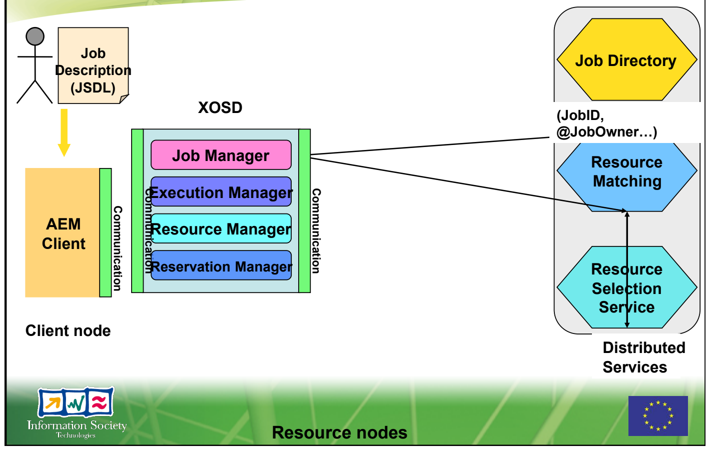
- Objectives
  - Start, monitor, control applications
  - Discover, select, allocate resources to applications

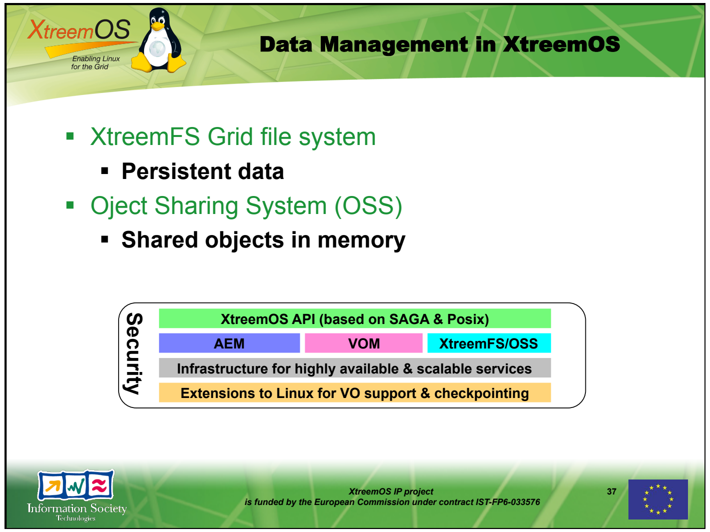

Security
XtreemOS API (based on SAGA & Posix)
AEM
VOM
XtreemFS/OSS
Infrastructure for highly available & scalable services
Extensions to Linux for VO support & checkpointing

AEM

- Features
  - "Self-scheduling" jobs
    - No global job scheduler
  - Resource discovery based on overlay networks
  - Unix-like job control
  - Monitoring & accounting
    - Accurate and flexible monitoring of job execution
  - Resource reservation & co-allocation
  - Interface for workflow engine
  - Checkpointing service for grid jobs

AEM Architecture





Data Management in XtreemOS

- XtreemFS Grid file system
  - Persistent data
- Oject Sharing System (OSS)
  - Shared objects in memory

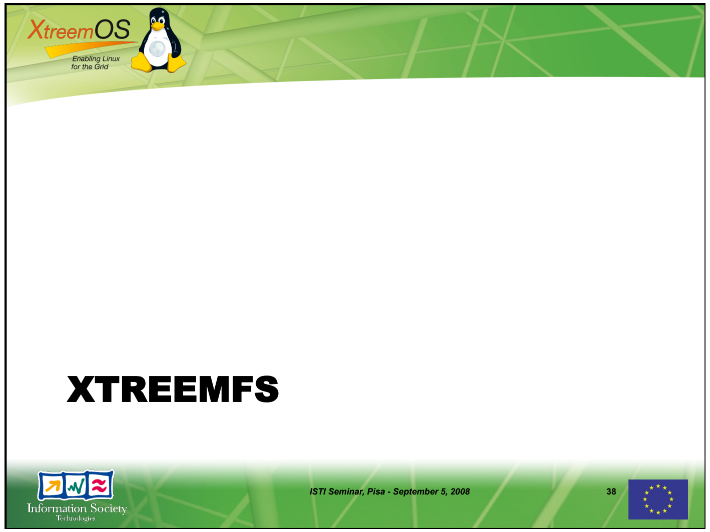
Security

XtreemOS API (based on SAGA & Posix)

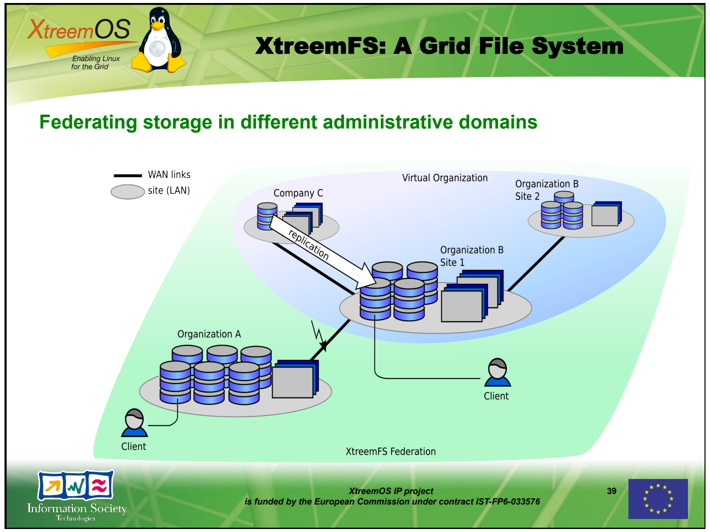
AEM | VOM | XtreemFS/OSS

Infrastructure for highly available & scalable services

Extensions to Linux for VO support & checkpointing

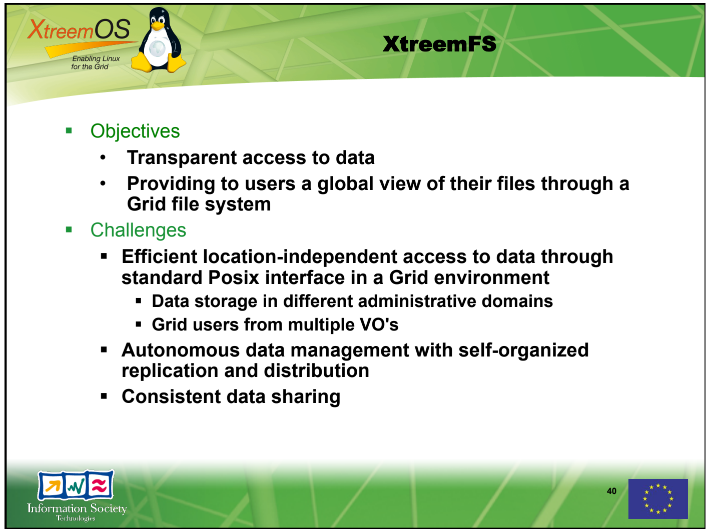


XTREEMFS



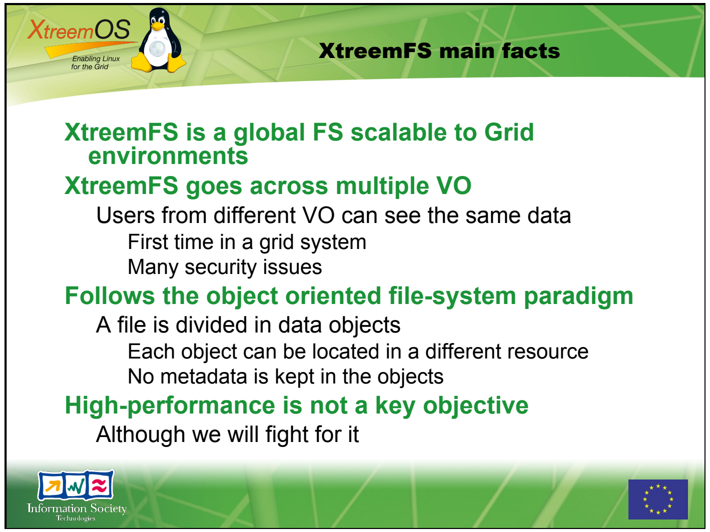
XtreemFS: A Grid File System

Federating storage in different administrative domains



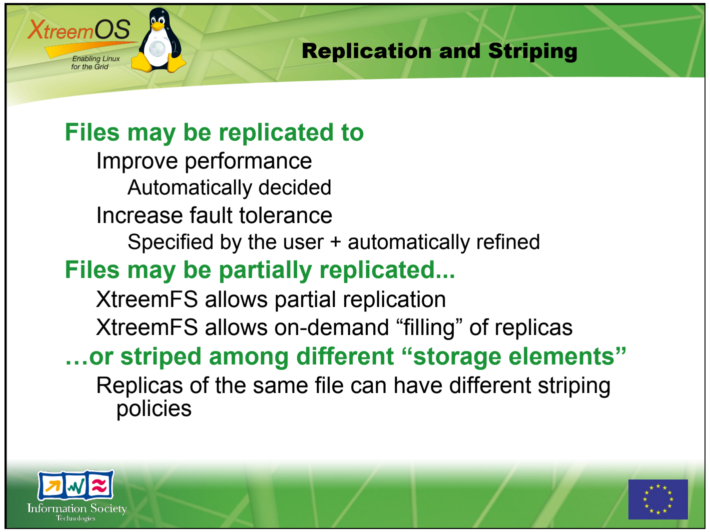
XtreemFS

- Objectives
  - Transparent access to data
  - Providing to users a global view of their files through a Grid file system
- Challenges
  - Efficient location-independent access to data through standard Posix interface in a Grid environment
    - Data storage in different administrative domains
    - Grid users from multiple VO's
  - Autonomous data management with self-organized replication and distribution
  - Consistent data sharing



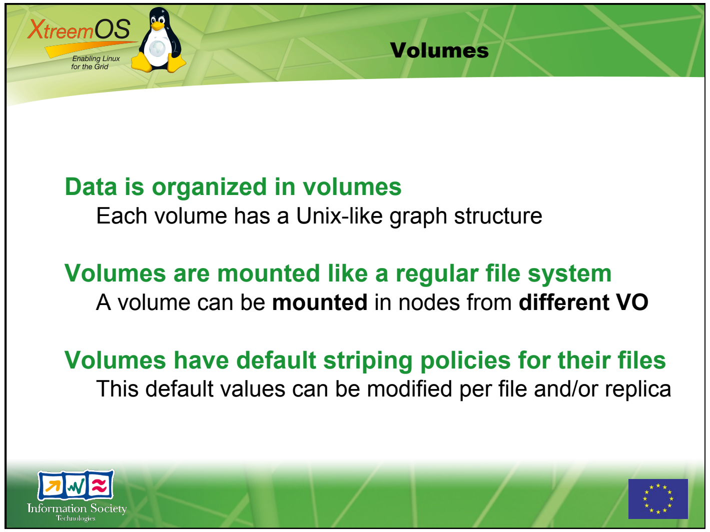
XtreemFS main facts

- **XtreemFS is a global FS scalable to Grid environments**
- **XtreemFS goes across multiple VO**
  - Users from different VO can see the same data
    - First time in a grid system
    - Many security issues
- **Follows the object oriented file-system paradigm**
  - A file is divided in data objects
    - Each object can be located in a different resource
    - No metadata is kept in the objects
- **High-performance is not a key objective**
  - Although we will fight for it



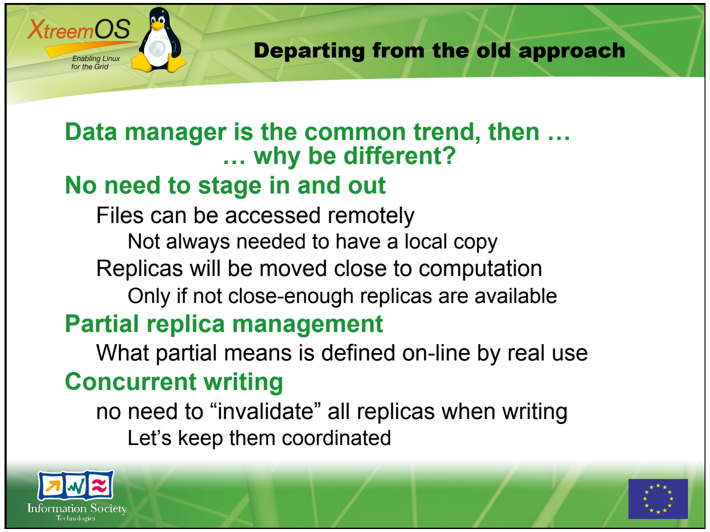
Replication and Striping

- **Files may be replicated to**
  - Improve performance
    - Automatically decided
  - Increase fault tolerance
    - Specified by the user + automatically refined
- **Files may be partially replicated...**
  - XtreemFS allows partial replication
  - XtreemFS allows on-demand “filling” of replicas
- **…or striped among different “storage elements”**
  - Replicas of the same file can have different striping policies



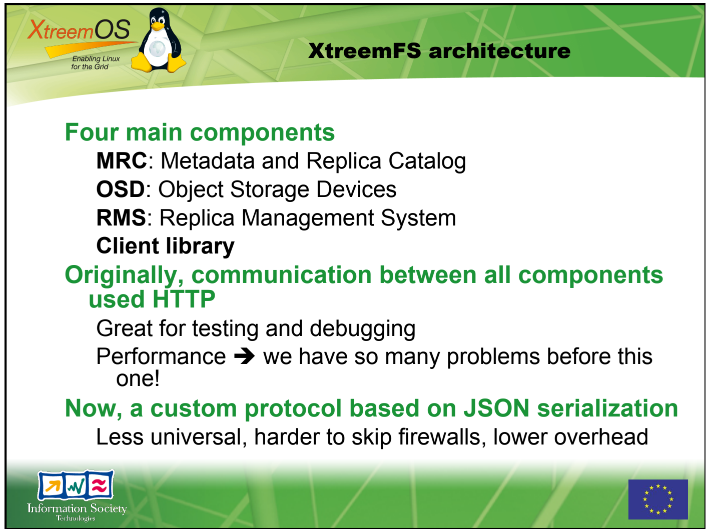
Volumes

- **Data is organized in volumes**
  - Each volume has a Unix-like graph structure
- **Volumes are mounted like a regular file system**
  - A volume can be **mounted** in nodes from **different VO**
- **Volumes have default striping policies for their files**
  - This default values can be modified per file and/or replica



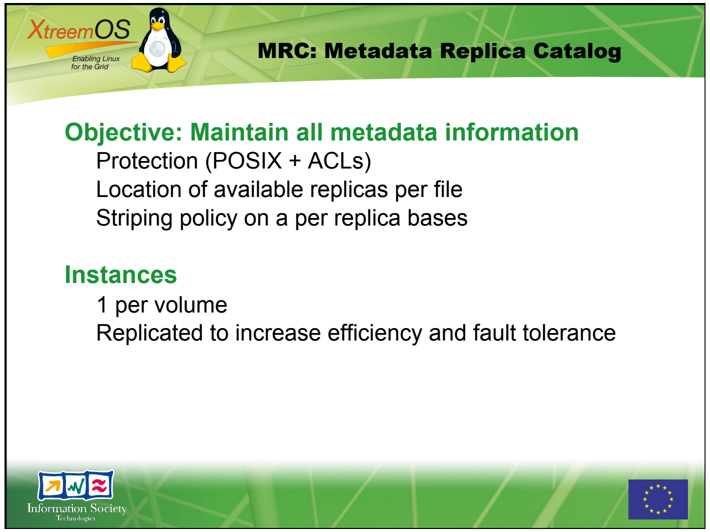
Departing from the old approach

- **Data manager is the common trend, then … … why be different?**
- **No need to stage in and out**
  - Files can be accessed remotely
    - Not always needed to have a local copy
  - Replicas will be moved close to computation
    - Only if not close-enough replicas are available
- **Partial replica management**
  - What partial means is defined on-line by real use
- **Concurrent writing**
  - no need to “invalidate” all replicas when writing
    - Let’s keep them coordinated



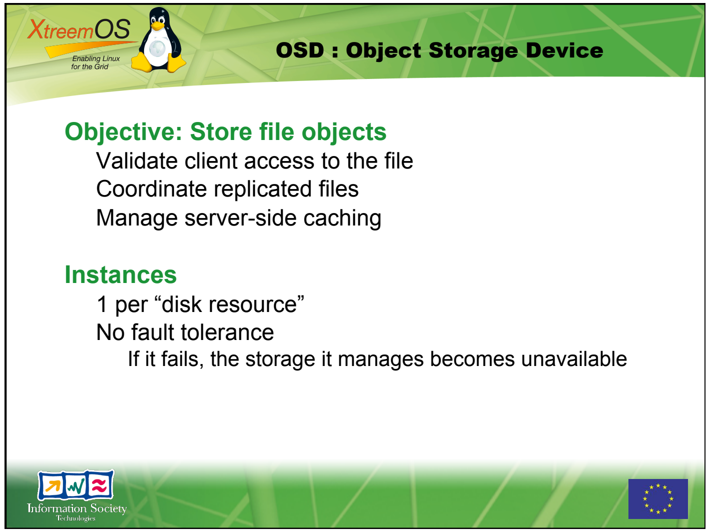
XtreemFS architecture

- **Four main components**
  - **MRC**: Metadata and Replica Catalog
  - **OSD**: Object Storage Devices
  - **RMS**: Replica Management System
  - **Client library**
- **Originally, communication between all components used HTTP**
  - Great for testing and debugging
  - Performance ➔ we have so many problems before this one!
- **Now, a custom protocol based on JSON serialization**
  - Less universal, harder to skip firewalls, lower overhead



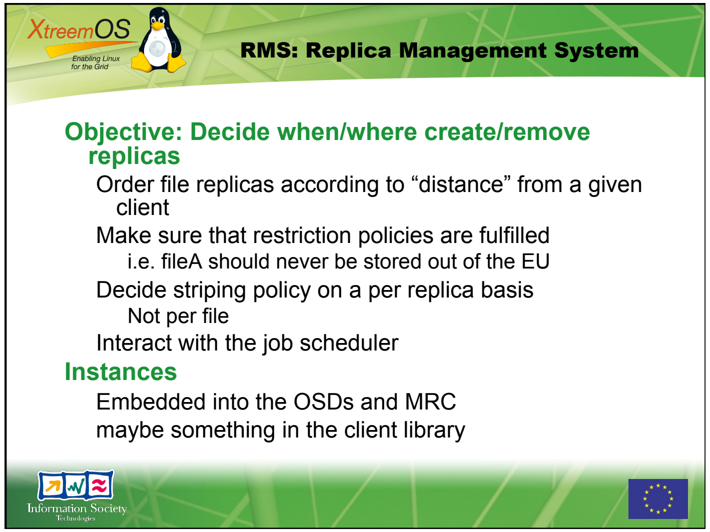
MRC: Metadata Replica Catalog

- **Objective: Maintain all metadata information**
  - Protection (POSIX + ACLs)
  - Location of available replicas per file
  - Striping policy on a per replica bases
- **Instances**
  - 1 per volume
  - Replicated to increase efficiency and fault tolerance



OSD : Object Storage Device

- **Objective: Store file objects**
  - Validate client access to the file
  - Coordinate replicated files
  - Manage server-side caching
- **Instances**
  - 1 per “disk resource”
  - No fault tolerance
    - If it fails, the storage it manages becomes unavailable



RMS: Replica Management System

- **Objective: Decide when/where create/remove replicas**
  - Order file replicas according to “distance” from a given client
  - Make sure that restriction policies are fulfilled
    - i.e. fileA should never be stored out of the EU
  - Decide striping policy on a per replica basis
    - Not per file
  - Interact with the job scheduler
- **Instances**
  - Embedded into the OSDs and MRC
  - maybe something in the client library

Client library

Objective: Translate Linux system calls into messages

Contact the MRC for metadata information

Contact OSDs for real data

Manage striping

And parity (if needed)

Manage client-side caches

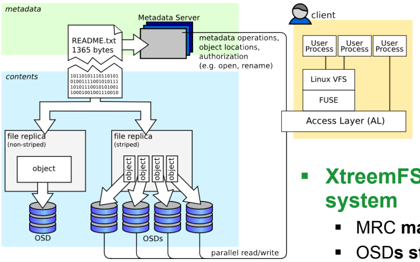
Kernel page cache ➔ a huge problem

Instances

One per machine that mounts an XtreemFS volume (i.e. all)

Implemented as a FUSE module

XtreemFS: Architecture



- **XtreemFS: an object-based file system**
  - MRC **maintains metadata**
  - OSDs **store file content**
  - Client **(Access Layer) provides client access**

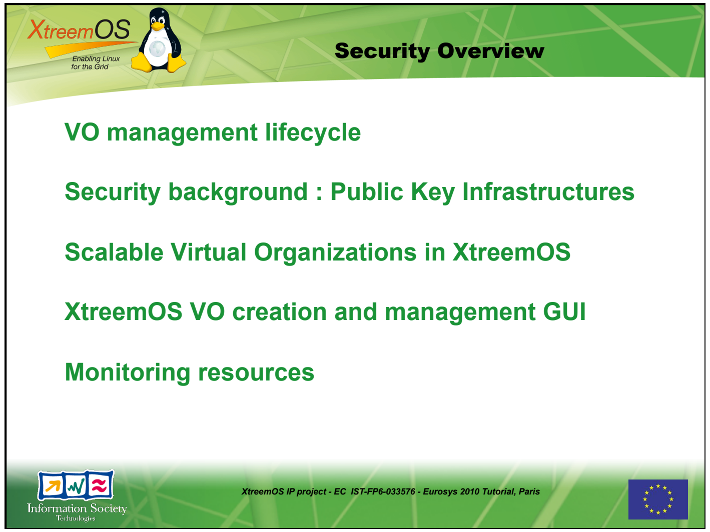
XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemFS Features

- POSIX compatible file system
  - **File system API**
  - **Behaviour as defined by POSIX or local file system**
- Advanced metadata management
  - **Replication**
  - **Partitioning**
  - **Extended attributes and queries**

Summary of XtreemFS Features

- Replication of files
  - **primary/secondary with automatic failover**
  - **fully synchronous to lazy data replication**
  - **POSIX compatible by default**
- Striping (parallel read and write)
- RAID and end-to-end checksums
- Client-side caching and cache consistency
- Access pattern -based replica management (RMS service)



Security Overview

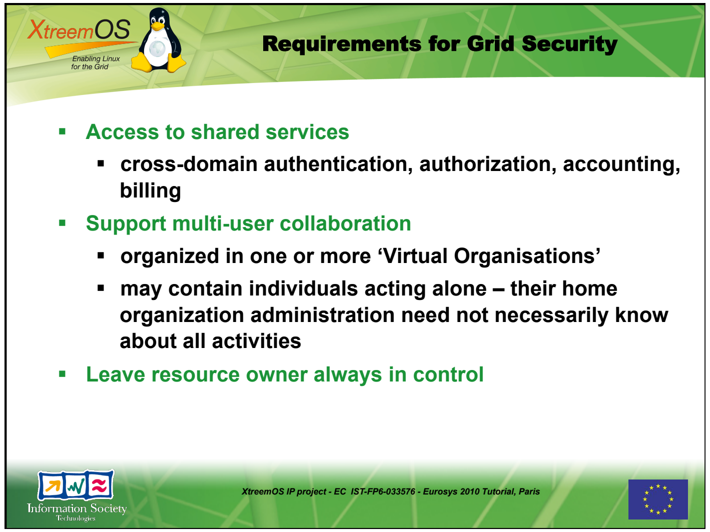
VO management lifecycle

Security background : Public Key Infrastructures

Scalable Virtual Organizations in XtreemOS

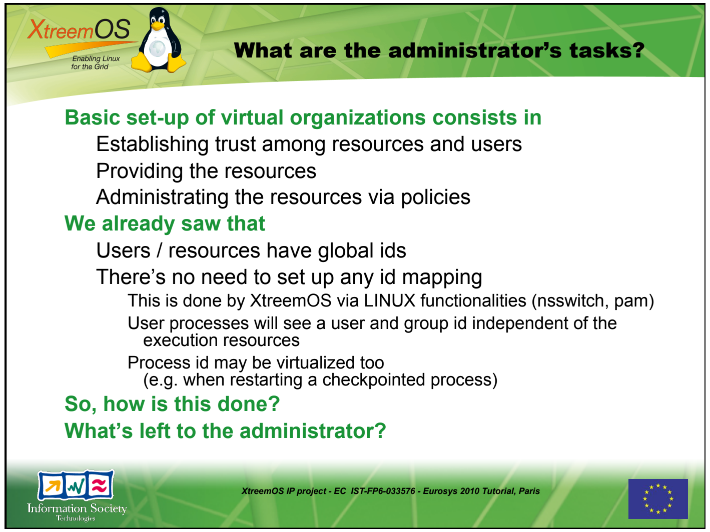
XtreemOS VO creation and management GUI

Monitoring resources



Requirements for Grid Security

- Access to shared services
  - cross-domain authentication, authorization, accounting, billing
- Support multi-user collaboration
  - organized in one or more ‘Virtual Organisations’
  - may contain individuals acting alone – their home organization administration need not necessarily know about all activities
- Leave resource owner always in control



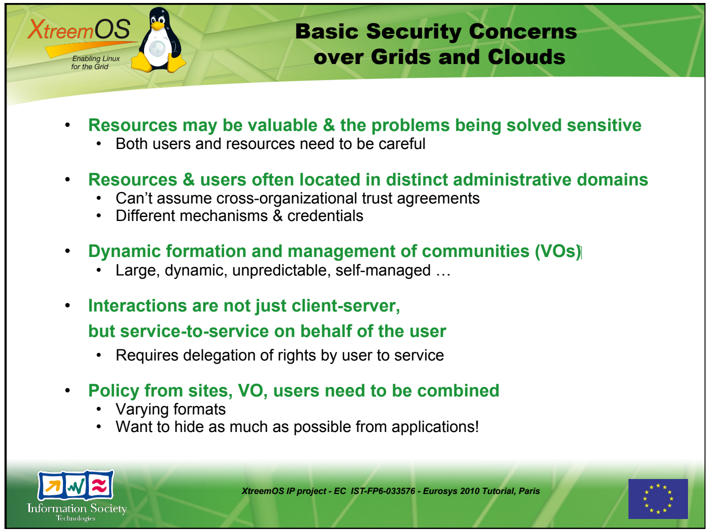
What are the administrator’s tasks?

Basic set-up of virtual organizations consists in
- Establishing trust among resources and users
- Providing the resources
- Administrating the resources via policies

We already saw that
- Users / resources have global ids
- There’s no need to set up any id mapping
  - This is done by XtreemOS via LINUX functionalities (nsswitch, pam)
  - User processes will see a user and group id independent of the execution resources
  - Process id may be virtualized too (e.g. when restarting a checkpointed process)

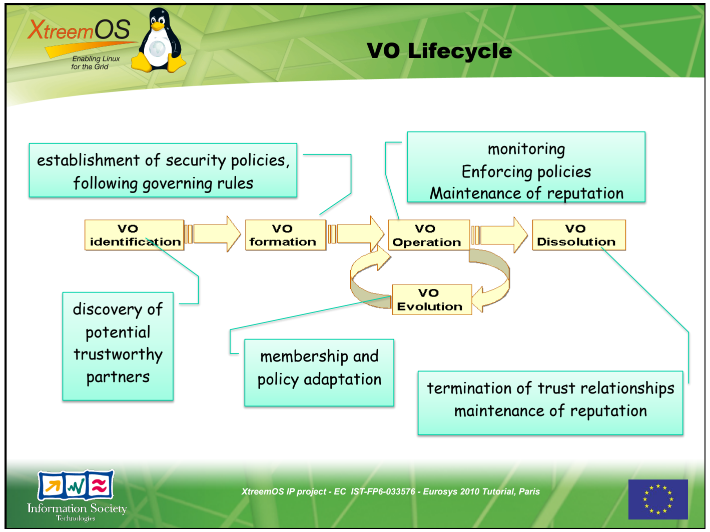
So, how is this done?

What’s left to the administrator?



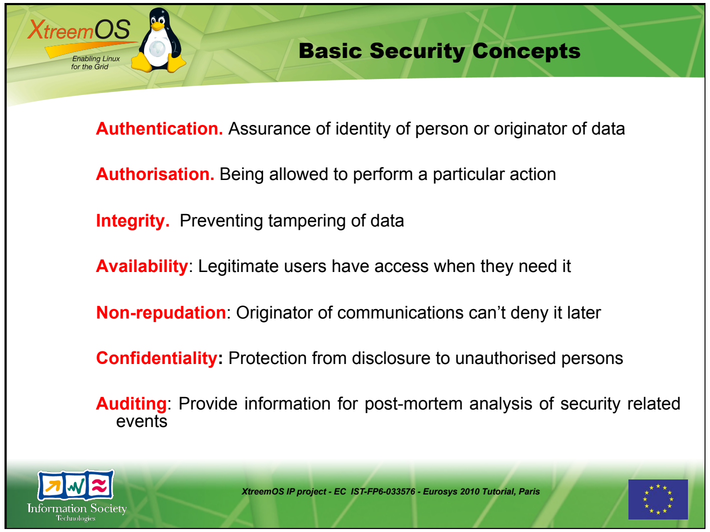
Basic Security Concerns over Grids and Clouds

- Resources may be valuable & the problems being solved sensitive
  - Both users and resources need to be careful
- Resources & users often located in distinct administrative domains
  - Can’t assume cross-organizational trust agreements
  - Different mechanisms & credentials
- Dynamic formation and management of communities (VOs)
  - Large, dynamic, unpredictable, self-managed …
- Interactions are not just client-server, but service-to-service on behalf of the user
  - Requires delegation of rights by user to service
- Policy from sites, VO, users need to be combined
  - Varying formats
  - Want to hide as much as possible from applications!



VO Lifecycle

establishment of security policies, following governing rules
monitoring
Enforcing policies
Maintenance of reputation
VO identification
VO formation
VO Operation
VO Dissolution
VO Evolution
discovery of potential trustworthy partners
membership and policy adaptation
termination of trust relationships
maintenance of reputation



Basic Security Concepts

Authentication. Assurance of identity of person or originator of data

Authorisation. Being allowed to perform a particular action

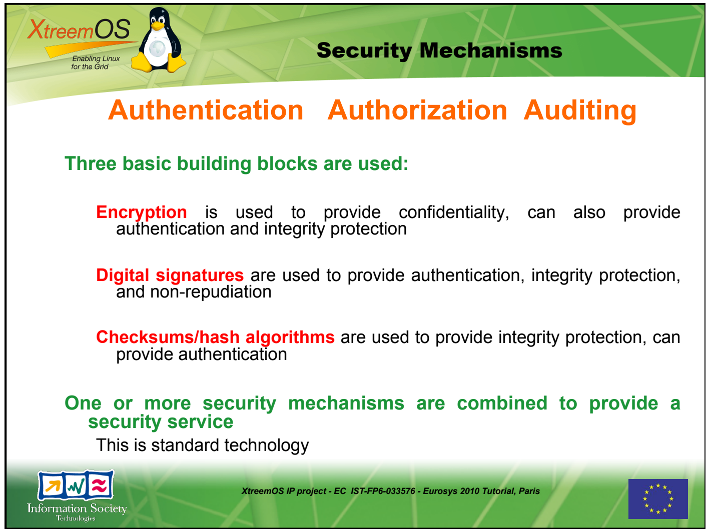
Integrity. Preventing tampering of data

Availability: Legitimate users have access when they need it

Non-repudation: Originator of communications can't deny it later

Confidentiality: Protection from disclosure to unauthorised persons

Auditing: Provide information for post-mortem analysis of security related events



Security Mechanisms

Authentication Authorization Auditing

Three basic building blocks are used:

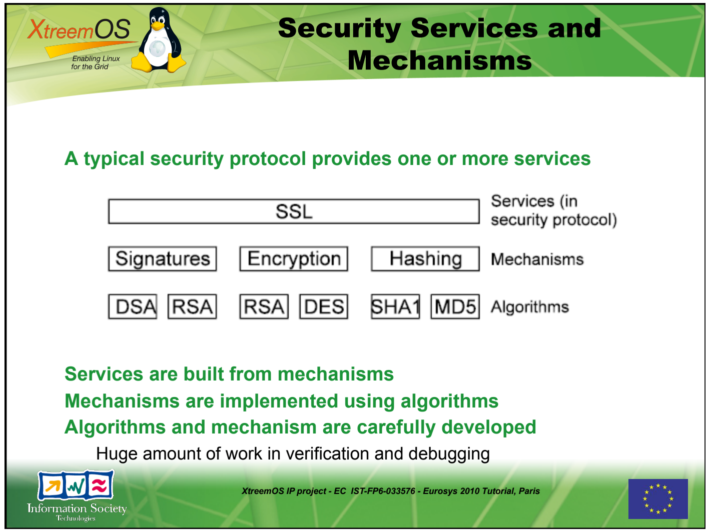
Encryption is used to provide confidentiality, can also provide authentication and integrity protection

Digital signatures are used to provide authentication, integrity protection, and non-repudiation

Checksums/hash algorithms are used to provide integrity protection, can provide authentication

One or more security mechanisms are combined to provide a security service

This is standard technology



Security Services and Mechanisms

A typical security protocol provides one or more services

SSL						Services (in security protocol)
Signatures		Encryption		Hashing		Mechanisms
DSA	RSA	RSA	DES	SHA1	MD5	Algorithms

Services are built from mechanisms

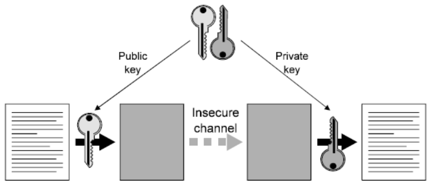
Mechanisms are implemented using algorithms

Algorithms and mechanism are carefully developed

Huge amount of work in verification and debugging

Public-Key Encryption

Users possess public/private key pairs

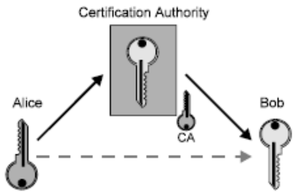


Anyone can encrypt with the public key, only one person can decrypt with the private key

- Communication can be made secure
- The problem is how to authenticate the keys

Certification Authority

A Certification Authority (CA) solves this problem



CA signs Alice's key to guarantee its authenticity to Bob

- Mallet can't substitute his key since the CA won't sign it

Public Key Infrastructure (PKI)

PKI allows one to know that a given key belongs to a given user

- Based on asymmetric encryption

The public key is given to the world encapsulated in a X.509 certificate

Certificates: Similar to passport or driver license

- Identity signed by a trusted party (a CA)

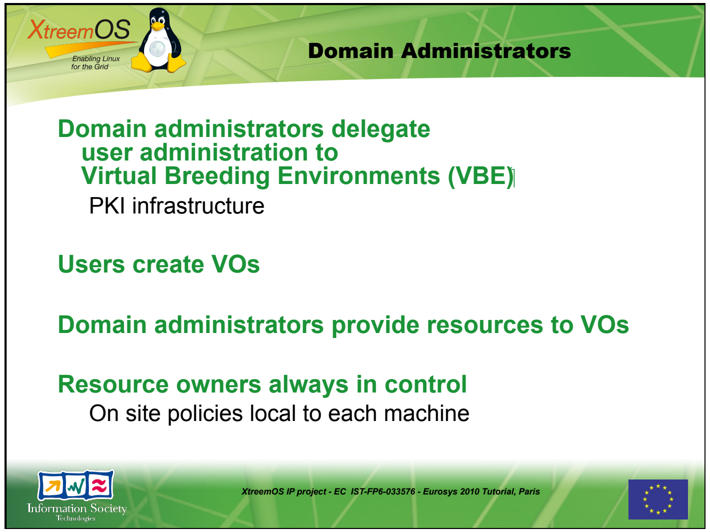
Virtual Breeding Environment and Actors

VO are created in the context of a Virtual Breeding Environment (VBE)

- A Virtual Breeding Environment is composed of users and service providers. It provides user and service provider registration, certificate management, and VO lifecycle management.

Actors

- VBE administrator
- VO administrator
- Domain/site administrators
- End-users – VO members



Domain Administrators

Domain administrators delegate user administration to Virtual Breeding Environments (VBE)

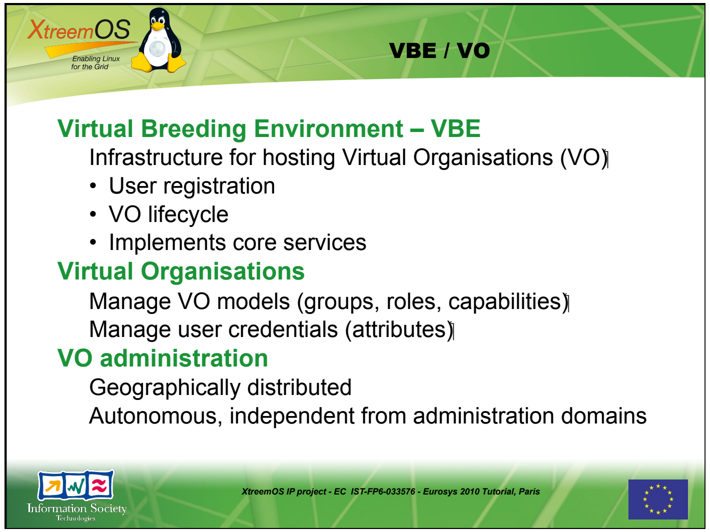
PKI infrastructure

Users create VOs

Domain administrators provide resources to VOs

Resource owners always in control

On site policies local to each machine



VBE / VO

Virtual Breeding Environment – VBE

Infrastructure for hosting Virtual Organisations (VO)

- User registration
- VO lifecycle
- Implements core services

Virtual Organisations

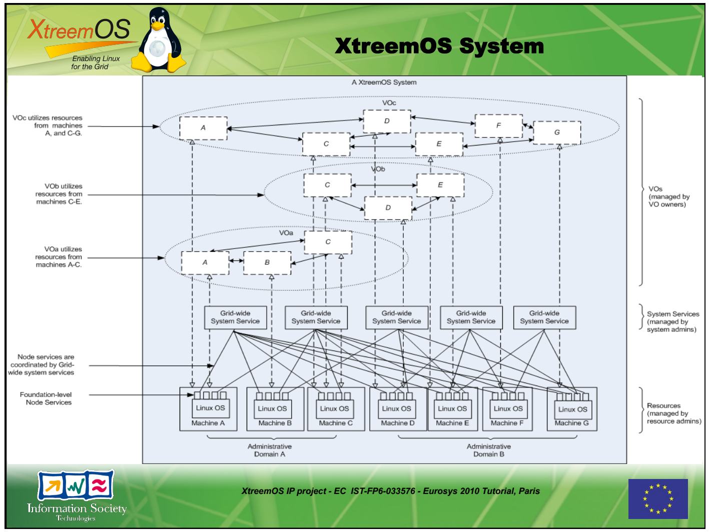
Manage VO models (groups, roles, capabilities)

Manage user credentials (attributes)

VO administration

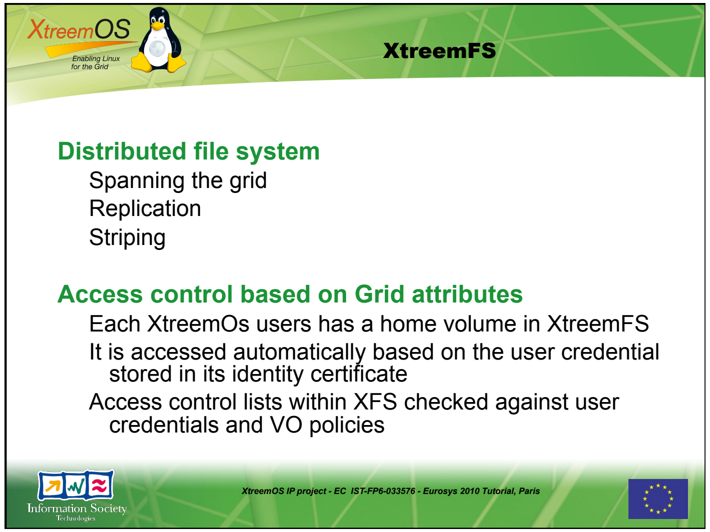
Geographically distributed

Autonomous, independent from administration domains



XtreemOS System

A XtreemOS System
VOc
A
D
F
G
C
E
VOb
C
E
D
VOa
C
A
B
VOc utilizes resources from machines A, and C-G.
VOb utilizes resources from machines C-E.
VOa utilizes resources from machines A-C.
Node services are coordinated by Grid-wide system services
Foundation-level Node Services
Grid-wide System Service
Grid-wide System Service
Grid-wide System Service
Grid-wide System Service
Grid-wide System Service
Linux OS
Machine A
Linux OS
Machine B
Linux OS
Machine C
Linux OS
Machine D
Linux OS
Machine E
Linux OS
Machine F
Linux OS
Machine G
Administrative Domain A
Administrative Domain B
VOs (managed by VO owners)
System Services (managed by system admins)
Resources (managed by resource admins)



XtreemFS

Distributed file system

Spanning the grid

Replication

Striping

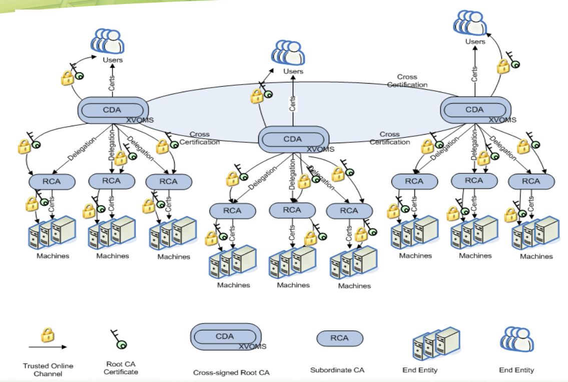
Access control based on Grid attributes

Each XtreemOs users has a home volume in XtreemFS

It is accessed automatically based on the user credential stored in its identity certificate

Access control lists within XFS checked against user credentials and VO policies

Trust Model



XtreemOS VO management

XtreemOS Security Architecture Components and VO management

At least one node (a core node) will host a CDA

XVOMS: the database holding all information about active VOs within an XtreemOS platform

Controls the other key services providing security and platform management

We will see the web GUI

Same functionalities available via shell commands, thus scriptable

XtreemOS security components

VOPS

Policy management point

Policy decision point

Filters to distribute policy decisions in a scalable way

RCA

Resource registration

Distributes certificates to resources

Attributes define resource capabilities for resource discovery (#cpus, memory, ...)

User Session Service

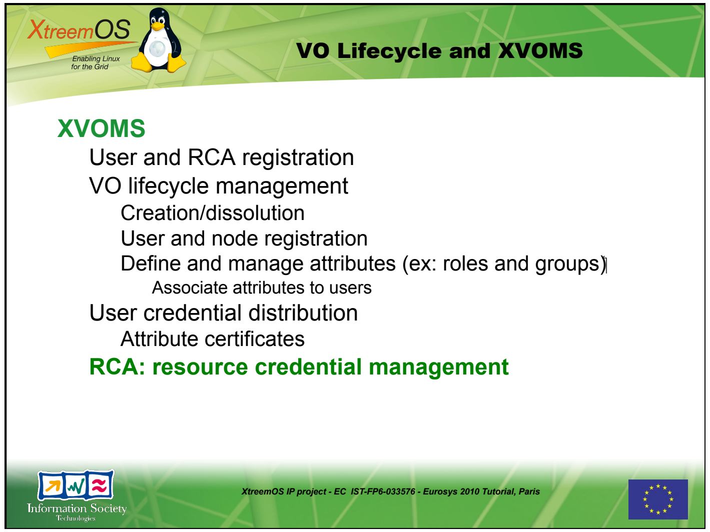
User session services

- Started when the user logs in
  - In charge of validating user credentials
  - Trusted by XtreemOS operating system services
- Bridging the user space with the operating system space
  - All grid requests go through the user session service
- Support untrusted client nodes

Provide Single-Sign-On

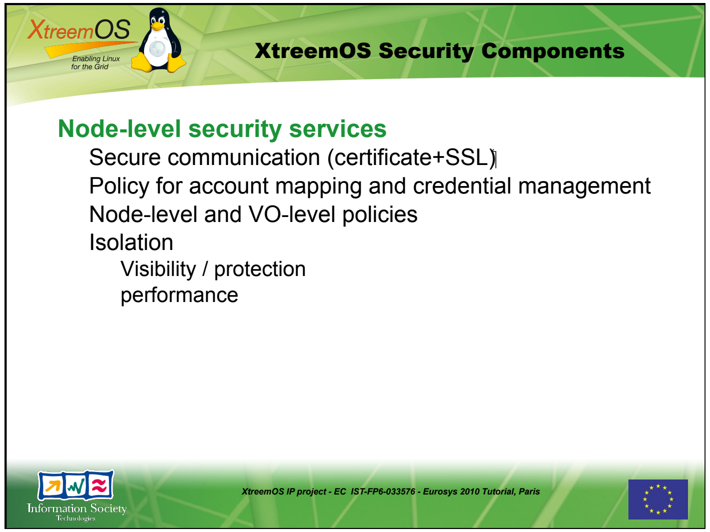
Provide Delegation

Can be replicated on resource nodes



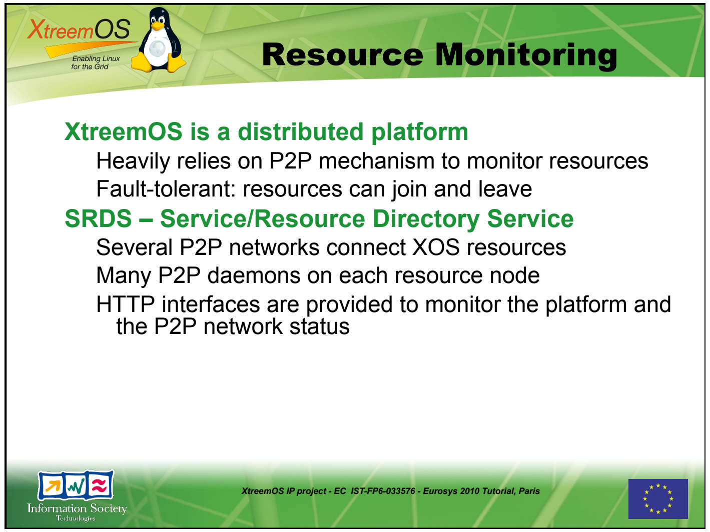
VO Lifecycle and XVOMS

- **XVOMS**
  - User and RCA registration
  - VO lifecycle management
    - Creation/dissolution
    - User and node registration
    - Define and manage attributes (ex: roles and groups)
      - Associate attributes to users
  - User credential distribution
    - Attribute certificates
  - **RCA: resource credential management**



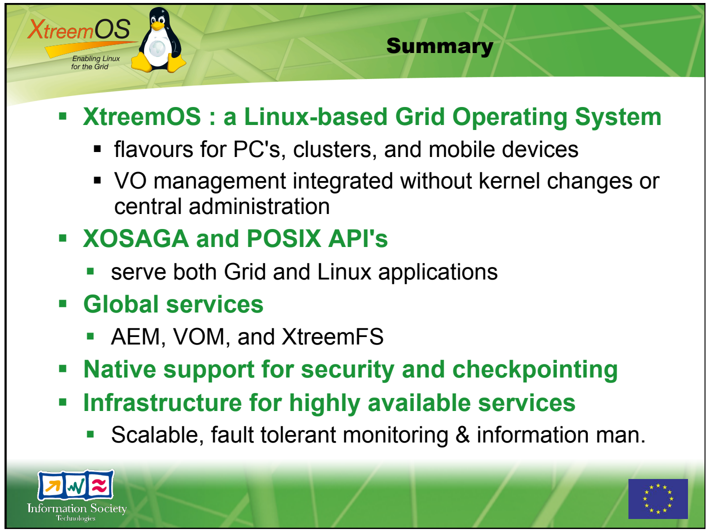
XtreemOS Security Components

- **Node-level security services**
  - Secure communication (certificate+SSL)
  - Policy for account mapping and credential management
  - Node-level and VO-level policies
  - Isolation
    - Visibility / protection
    - performance



Resource Monitoring

- **XtreemOS is a distributed platform**
  - Heavily relies on P2P mechanism to monitor resources
  - Fault-tolerant: resources can join and leave
- **SRDS – Service/Resource Directory Service**
  - Several P2P networks connect XOS resources
  - Many P2P daemons on each resource node
  - HTTP interfaces are provided to monitor the platform and the P2P network status



Summary

- **XtreemOS : a Linux-based Grid Operating System**
  - flavours for PC's, clusters, and mobile devices
  - VO management integrated without kernel changes or central administration
- **XOSAGA and POSIX API's**
  - serve both Grid and Linux applications
- **Global services**
  - AEM, VOM, and XtreemFS
- **Native support for security and checkpointing**
- **Infrastructure for highly available services**
  - Scalable, fault tolerant monitoring & information man.

CONCLUSIONS : What have we seen?

Scalable VO management

- Independent user and resource management
- Interoperability with VO management frameworks and security models
- Customizable isolation, access control and auditing
- Scalable Hierarchical and P2P management of resources

Distributed application management

- No global job scheduler
- Resource discovery based on an overlay network

Grid file system federating storage in different administrative domains

- Transparent access to data

Resources

Information

www.xtreemos.eu

Open source software repository

http://gforge.inria.fr/projects/xtreemos/

Official WWW	http://www.xtreeemos.eu
XtreemOS Blog	https://www.xtreemos.org/blog
IRC channel for user support	irc.freenode.netchannel #xtreemos

XtreemOS 2.1

Mirrors for ISO Downloads and Package Updates

http://www.xtreemos.eu/software/mirror-websites

http://www.xtreemos.eu/software/experimenting-xtreemos-on-virtual-machines