Gestion des données dans les grilles, les clusters et le cloud
– LIPN, AOC Team –

Christophe Cérin[1]

[1]Université de Paris XIII, LIPN, CNRS UMR 7030, France

Marseille Workshop on Scientific Data Preservation

Table of contents

Facts and Objectives

1. We come from the High Performance Computing (HPC) field;

Facts and Objectives

1. We come from the High Performance Computing (HPC) field;
2. Introduce success stories in data movement / storage;

Facts and Objectives

1. We come from the High Performance Computing (HPC) field;
2. Introduce success stories in data movement / storage;
3. Demystify some jargon and clarify some useful notions;

Facts and Objectives

1. We come from the High Performance Computing (HPC) field;
2. Introduce success stories in data movement / storage;
3. Demystify some jargon and clarify some useful notions;
4. *All the communities share the need to put data close to the computation nodes at some time!*

Applications running on what?

- Applications with dependencies are for HPC computers;

Applications running on what?

- Applications with dependencies are for HPC computers;
- Applications with few dependencies (BOT: Bag of Tasks) are for grids;

Applications running on what?

- Applications with dependencies are for HPC computers;
- Applications with few dependencies (BOT: Bag of Tasks) are for grids;
- Applications with no dependencies are for Clouds.

Clusters

Definition

1. Hierarchical (multi-core/node/chassis/...)
2. Homogeneous (almost) for the processor type;
3. Interconnected by High Performance networking technologies (Quadrics, Myrinet, Infiniband, IBM RoCE RDMA);
4. Applications: developed with MPI (Message Passing Interface) or OpenMP or hybrid;
5. HPC: prediction climate, physics particle...

The leader (June 2013)

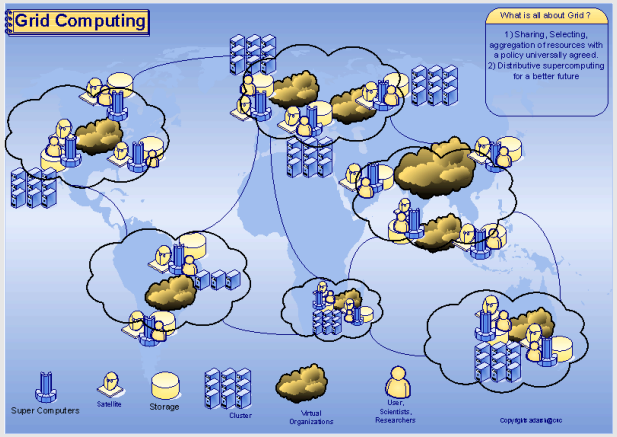
Performance of Tianhe

- Petaflops: 10^{15} operation / second;
- 24MW (with cooling);
- 3.2M cores

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National University of Defense Technology China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3120000	33862.7	54902.4	17808
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560640	17590.0	27112.5	8209
3	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1572864	17173.2	20132.7	7890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705024	10510.0	11280.4	12660

Grid Architectures

Grid in a nutshell



Grid = cluster of clusters

- Grids tend to be more loosely coupled, heterogeneous, and geographically dispersed compared to Clusters;
- Grids are often constructed with general-purpose grid middleware software libraries (to federate clusters resources)
- Internet (or dedicated network) ;
- Many projects are no more open.

Globus

Globus toolkit

Key points

- The Globus Toolkit: open source toolkit for building computing grids developed and provided by the Globus Alliance.
- The Globus Toolkit is an implementation of the following standards: Open Grid Services Architecture (OGSA), Open Grid Services Infrastructure (OGSI); Web Services Resource Framework (WSRF); Job Submission Description Language (JSDL); Distributed Resource Management Application API (DRMAA); WS-Management; WS-BaseNotification; SOAP; WSDL; Grid Security Infrastructure (GSI)...

Globus

Globus toolkit

Key points

- The Globus Toolkit has implementations of the OGF-defined protocols to provide:
  1. Resource management: Grid Resource Allocation & Management Protocol (GRAM)
  2. Information Services: Monitoring and Discovery Service (MDS)
  3. Security Services: Grid Security Infrastructure (GSI)
  4. *Data Movement and Management: Global Access to Secondary Storage (GASS) and GridFTP*

GridFTP (http://www.mcs.anl.gov/~kettimut/)

Definition

- GridFTP is an extension of the standard File Transfer Protocol (FTP) for high-speed, reliable, and secure data transfer;
- The protocol was defined within the GridFTP working group of the Open Grid Forum
- GridFTP also addresses the problem of incompatibility between storage and access systems;

Key points

- Grid Security Infrastructure provides authentication and encryption to file transfers, with user specified levels of confidentiality and data integrity;
- GridFTP achieves much greater use of bandwidth by allowing multiple simultaneous TCP streams.
- GridFTP provides a fault tolerant implem. of FTP. Transfers can also be automatically restarted if a problem occurs.

Globus Online

Researcher

- Globus Online enables you to move, sync, and share your data using just a web browser.
- Online helps you easily move data between your laptop, lab server, campus computing cluster, and supercomputing facility quickly, securely, and reliably.
- Keep your files in sync across multiple systems with just a few clicks.

Key points

- You're a good candidate for Globus Online if:
  1. You run computing resources for users who need to move and share big data
  2. You operate scientific instruments that generate high data volumes
  3. You manage a facility or service for users to analyze large data sets

Trends before the end of the 1990s

- PCs become good enough for scientific computing;

David Anderson notices that in 2012, the landscape "includes over 1 billion privately owned PCs and 100 million GPUs capable of general-purpose computing. These have a total computing capability of roughly 100 ExaFLOPS, and on the order of 10 Exabytes of free disk space, accessible via 1 Petabit/second of network bandwidth."

Trends before the end of the 1990s

- PCs become good enough for scientific computing;
- PCs become cheap;

David Anderson notices that in 2012, the landscape "includes over 1 billion privately owned PCs and 100 million GPUs capable of general-purpose computing. These have a total computing capability of roughly 100 ExaFLOPS, and on the order of 10 Exabytes of free disk space, accessible via 1 Petabit/second of network bandwidth."

Trends before the end of the 1990s

- PCs become good enough for scientific computing;
- PCs become cheap;
- PCs become available at Best-By (supermarket);

David Anderson notices that in 2012, the landscape "includes over 1 billion privately owned PCs and 100 million GPUs capable of general-purpose computing. These have a total computing capability of roughly 100 ExaFLOPS, and on the order of 10 Exabytes of free disk space, accessible via 1 Petabit/second of network bandwidth."

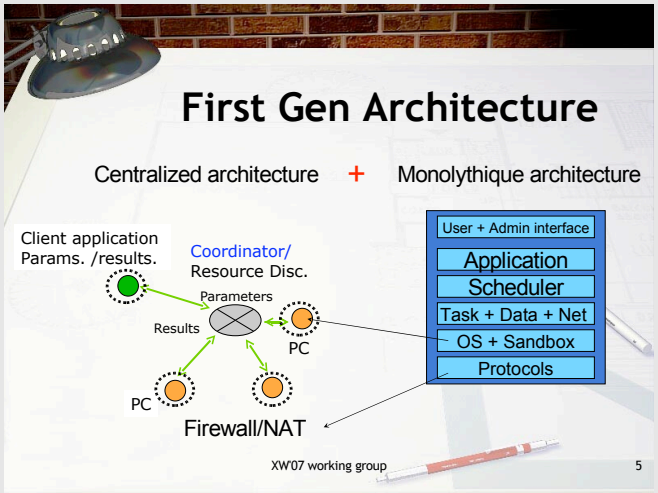
Trends before the end of the 1990s

- PCs become good enough for scientific computing;
- PCs become cheap;
- PCs become available at Best-By (supermarket);
- PCs become connected through Internet

David Anderson notices that in 2012, the landscape "includes over 1 billion privately owned PCs and 100 million GPUs capable of general-purpose computing. These have a total computing capability of roughly 100 ExaFLOPS, and on the order of 10 Exabytes of free disk space, accessible via 1 Petabit/second of network bandwidth."

Desktop Grid Architectures

Desktop Grid

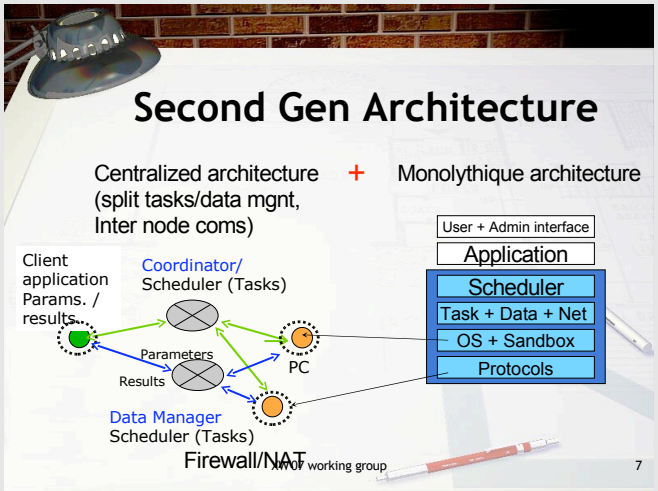


Key Points

- Federation of thousand of nodes;
- Internet as the communication layer: no trust!
- Volatility; local IP; Firewall

Desktop Grid Architectures

Desktop Grid



Future Generation (in 2006)

- Distributed Architecture
- Architecture with modularity: every component is "configurable": scheduler, storage, transport protocol
- Direct communications between peers;
- Security;
- Applications coming from any sciences (e-Science applications)

Desktop Grid and crowdsourcing

Crowdsourcing definition

1. Crowdsourcing = an online, distributed problem-solving and production model that leverages the collective intelligence of online communities to serve specific organizational goals.
2. Where is the locus of control? (in the community? in the organization?)
3. Where is the mutual benefit?

Key points

- With Desktop Grids (2012), the locus of control is in the organization ;
- What is the benefit for participating in DG projects? (BOINC: The credit system is designed to avoid cheating by validating results before granting credit on projects. This ensures users are returning accurate results for both scientific and statistical reasons.)

Main objectives of BonjourGrid

- Count on existing distributed tools for services discovering (publish/subscribe paradigm);

Main objectives of BonjourGrid

- Count on existing distributed tools for services discovering (publish/subscribe paradigm);
- Design and implement a platform able to manage multiple instances of DG middleware;

Main objectives of BonjourGrid

- Count on existing distributed tools for services discovering (publish/subscribe paradigm);
- Design and implement a platform able to manage multiple instances of DG middleware;
- Reduce as much as possible the use of any central element;

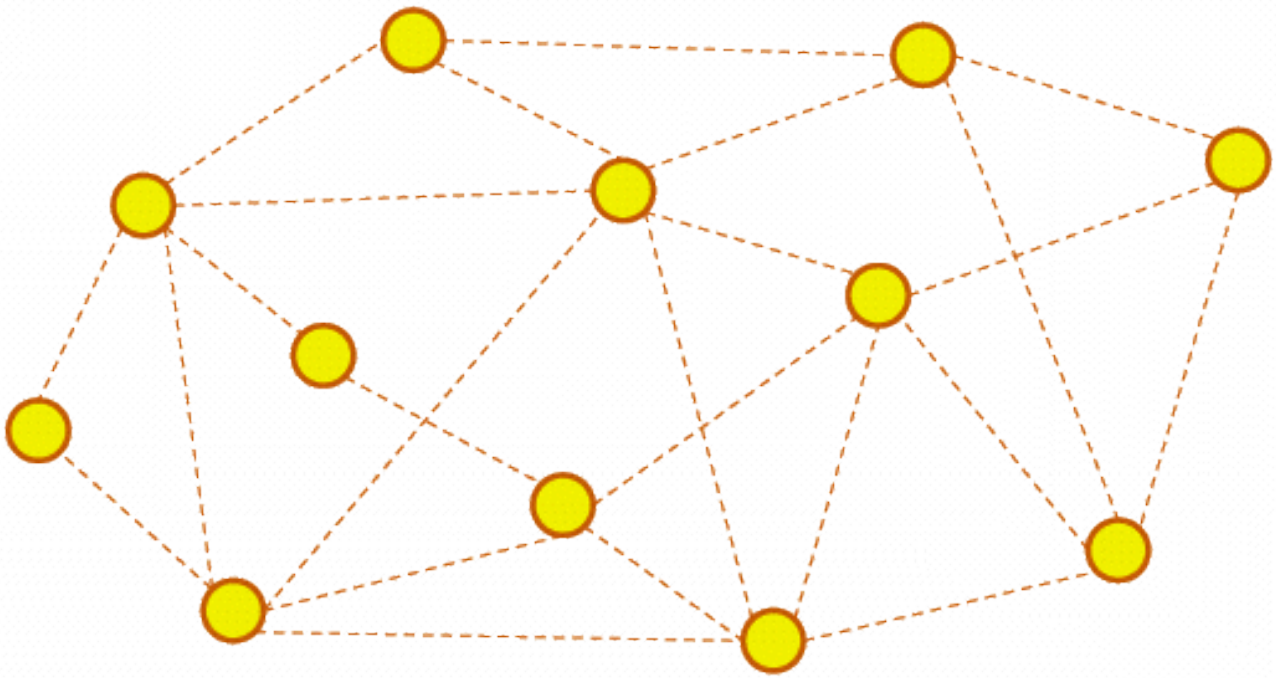
Main objectives of BonjourGrid

- Count on existing distributed tools for services discovering (publish/subscribe paradigm);
- Design and implement a platform able to manage multiple instances of DG middleware;
- Reduce as much as possible the use of any central element;
- Create a coordinator, on the fly, without any system administrator intervention; From a vision with a single coordinator towards a vision with multiple coordinators.

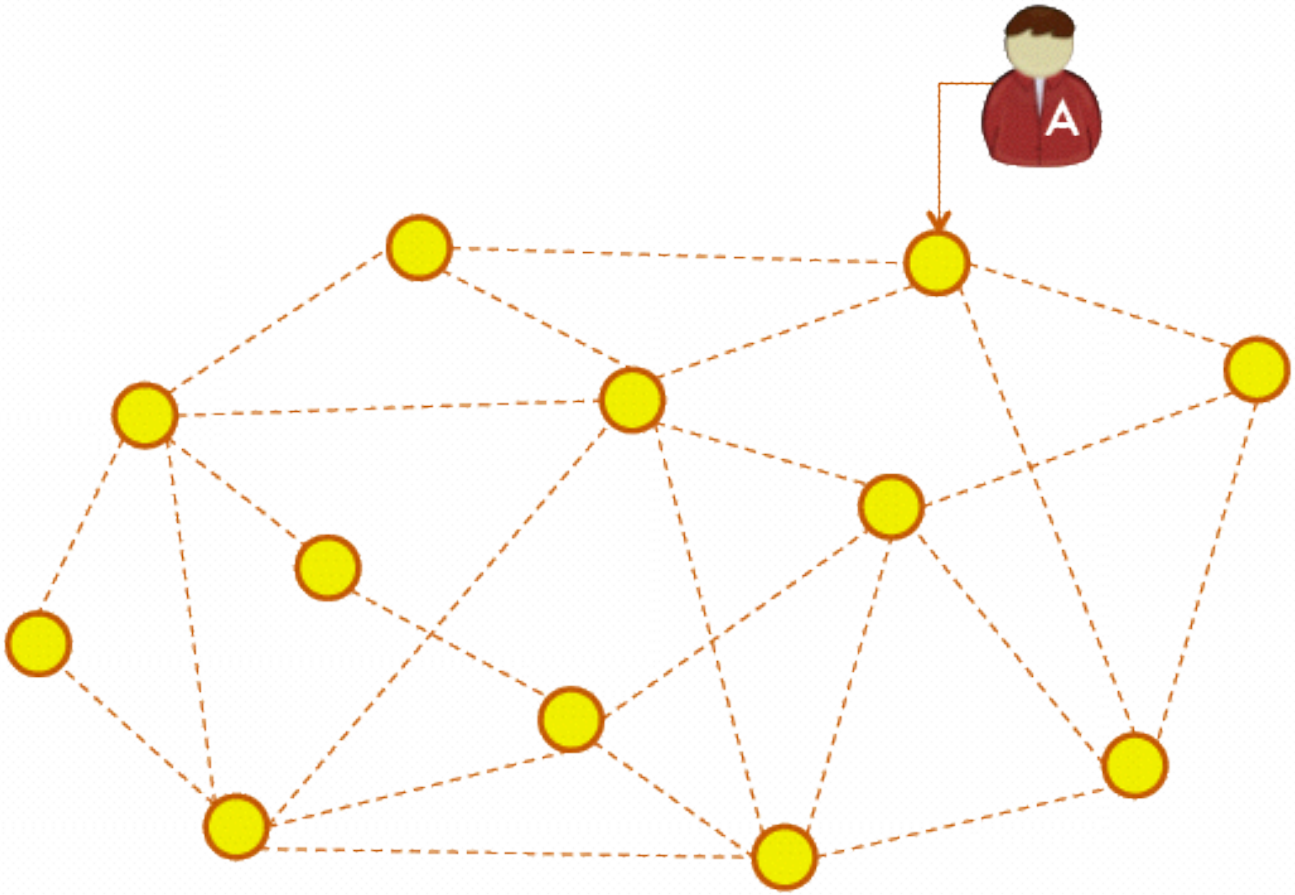
Main objectives of BonjourGrid

- Count on existing distributed tools for services discovering (publish/subscribe paradigm);
- Design and implement a platform able to manage multiple instances of DG middleware;
- Reduce as much as possible the use of any central element;
- Create a coordinator, on the fly, without any system administrator intervention; From a vision with a single coordinator towards a vision with multiple coordinators.
- Each coordinator searches, in a concurrent way, participants (idle machines)

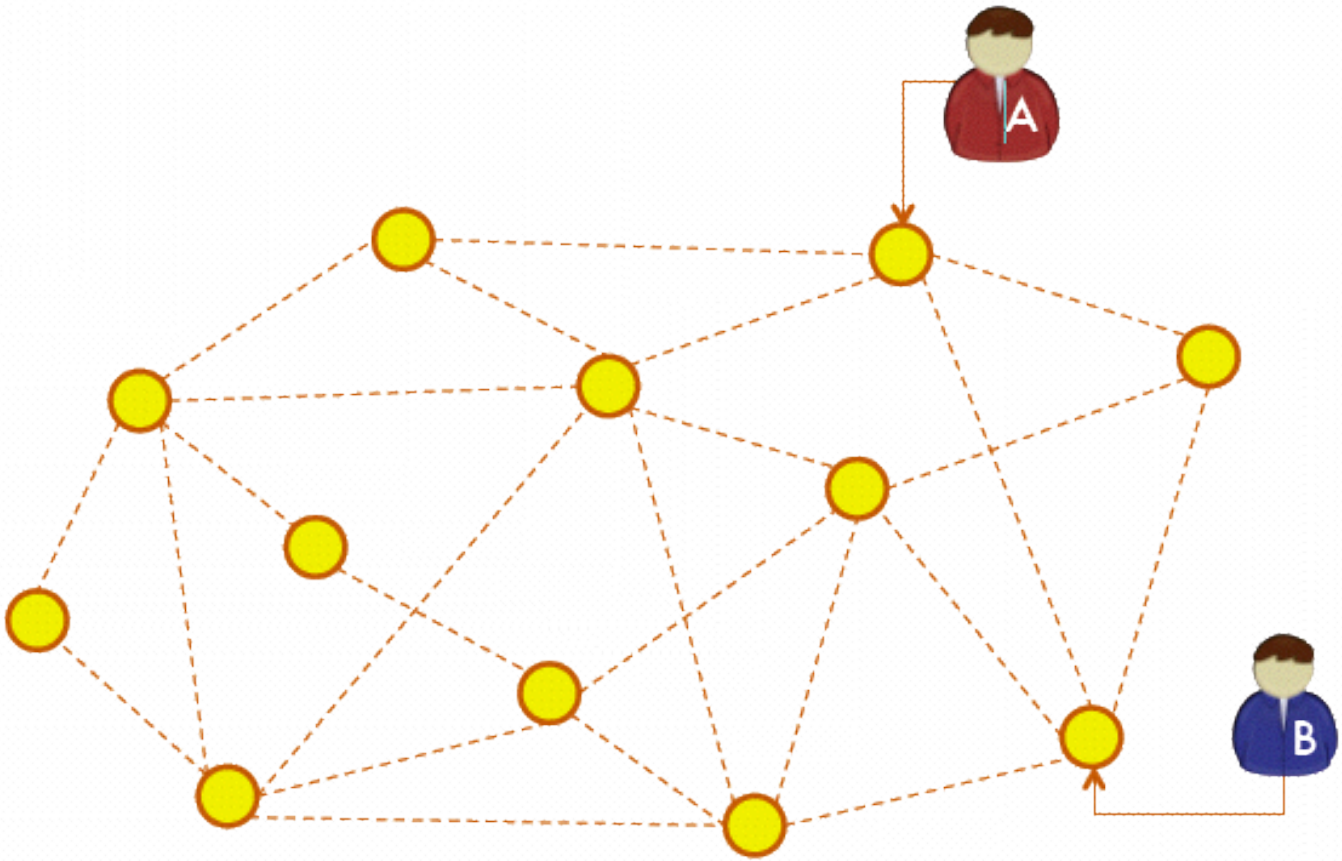
How BonjourGrid works



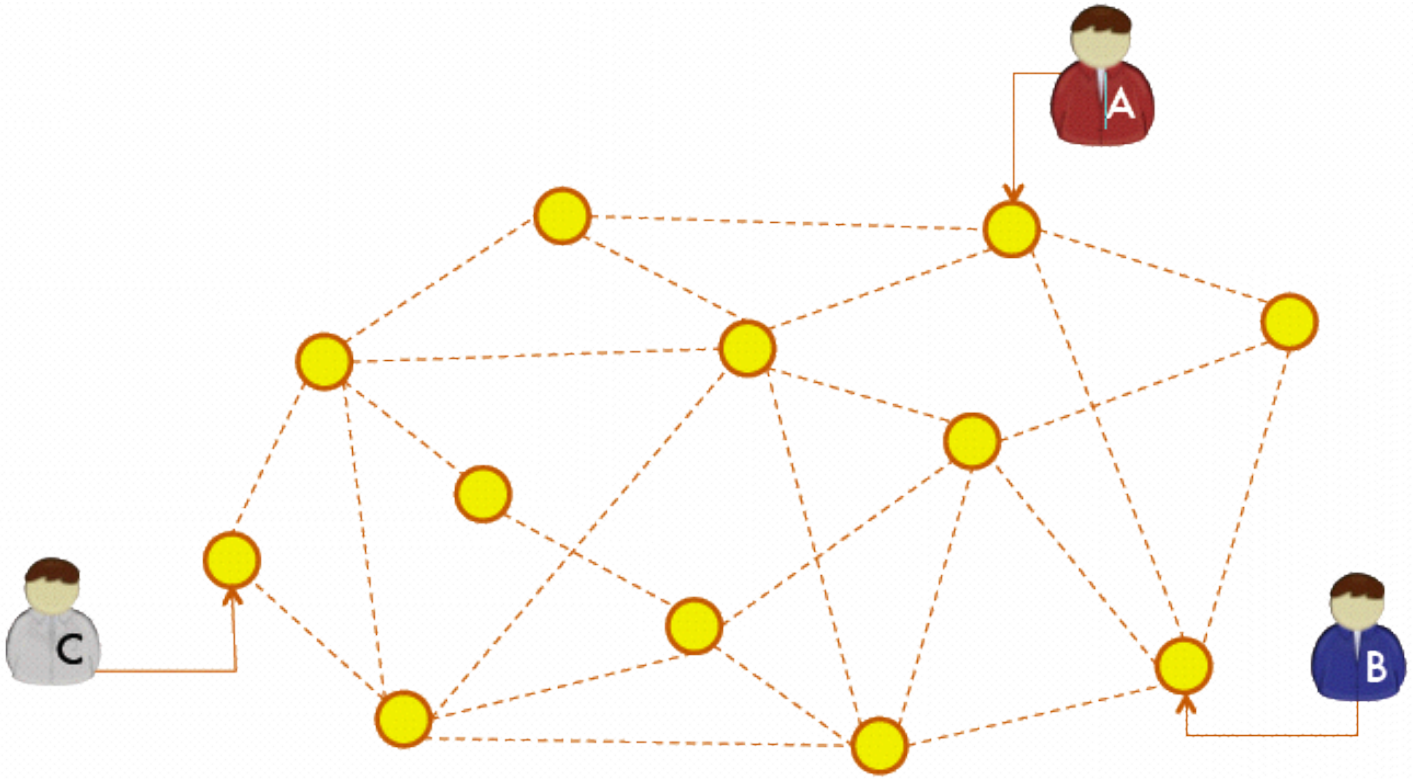
How BonjourGrid works



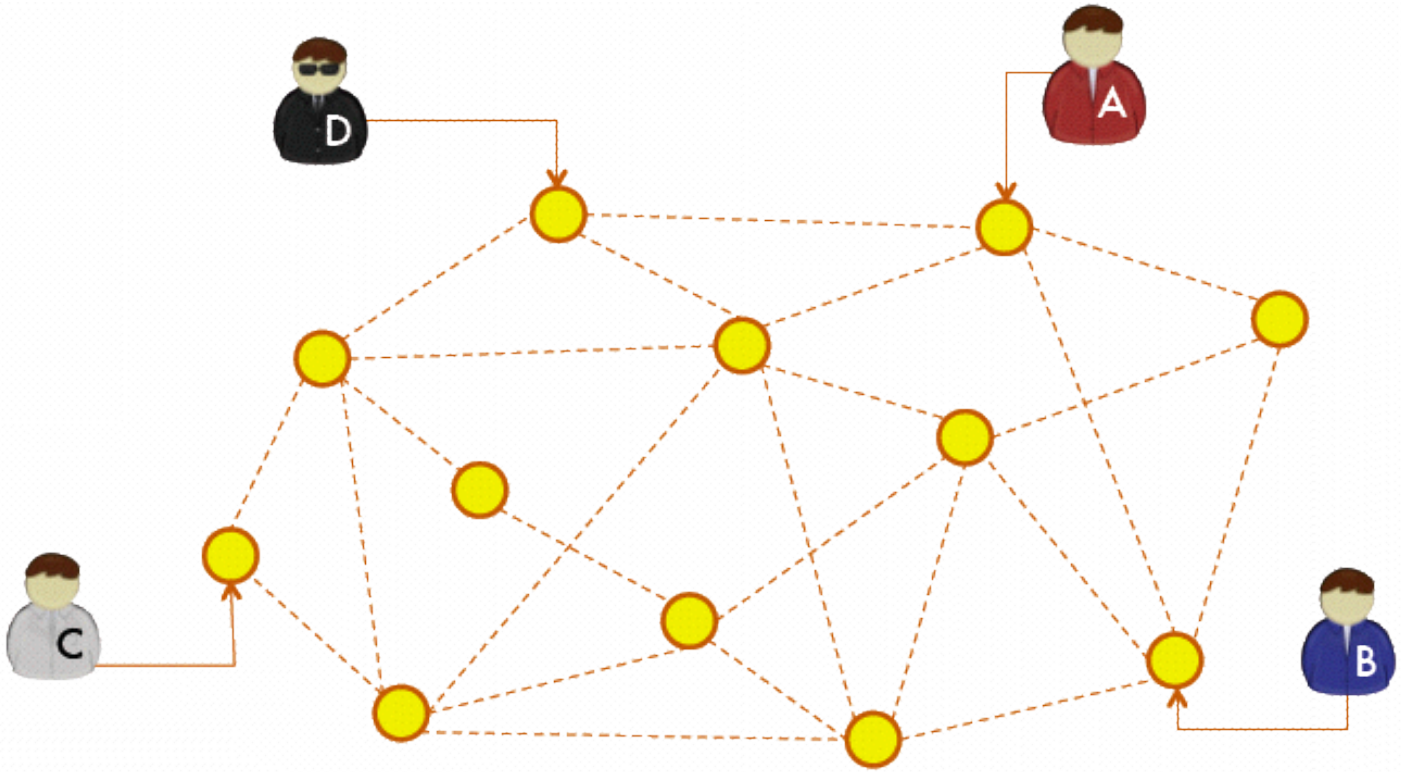
How BonjourGrid works



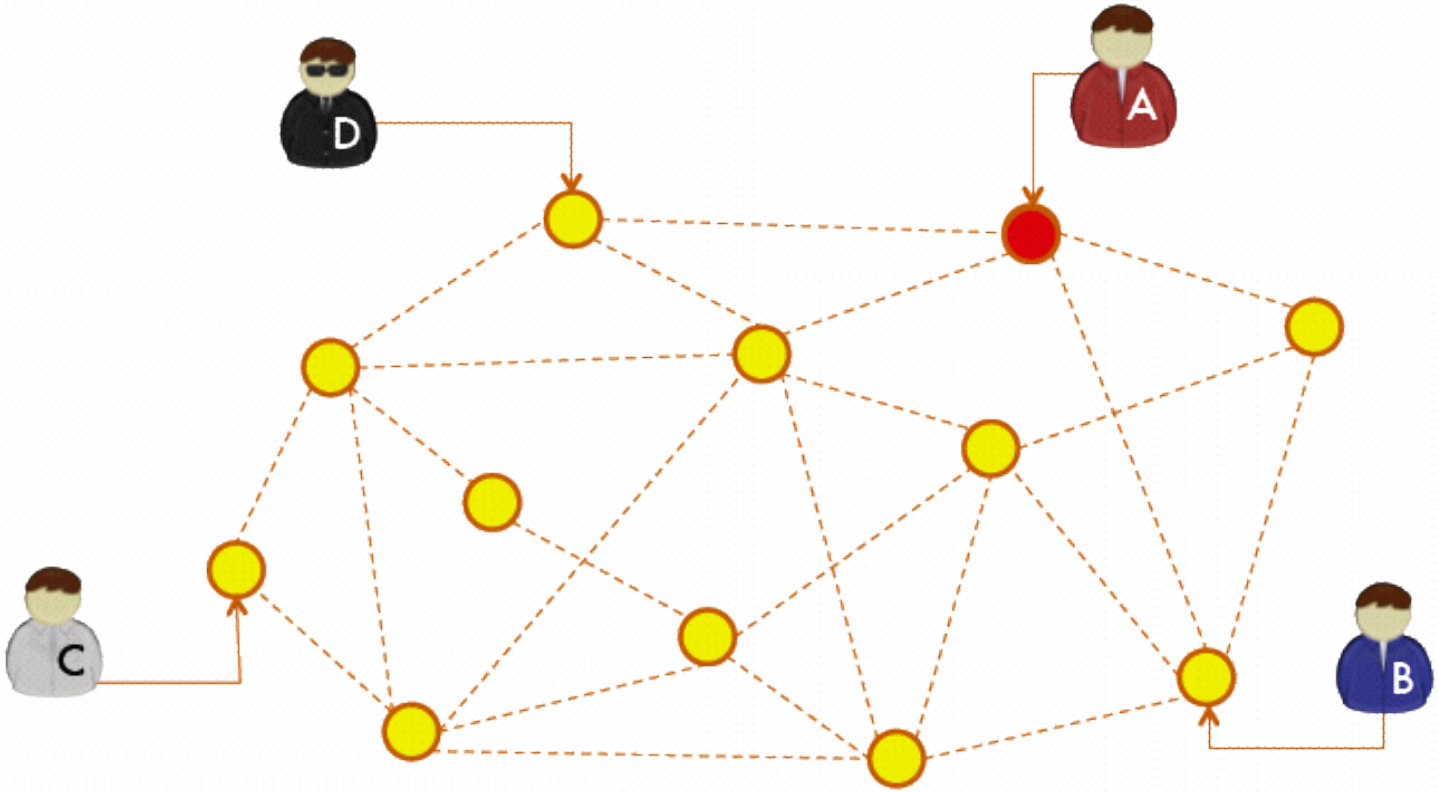
How BonjourGrid works



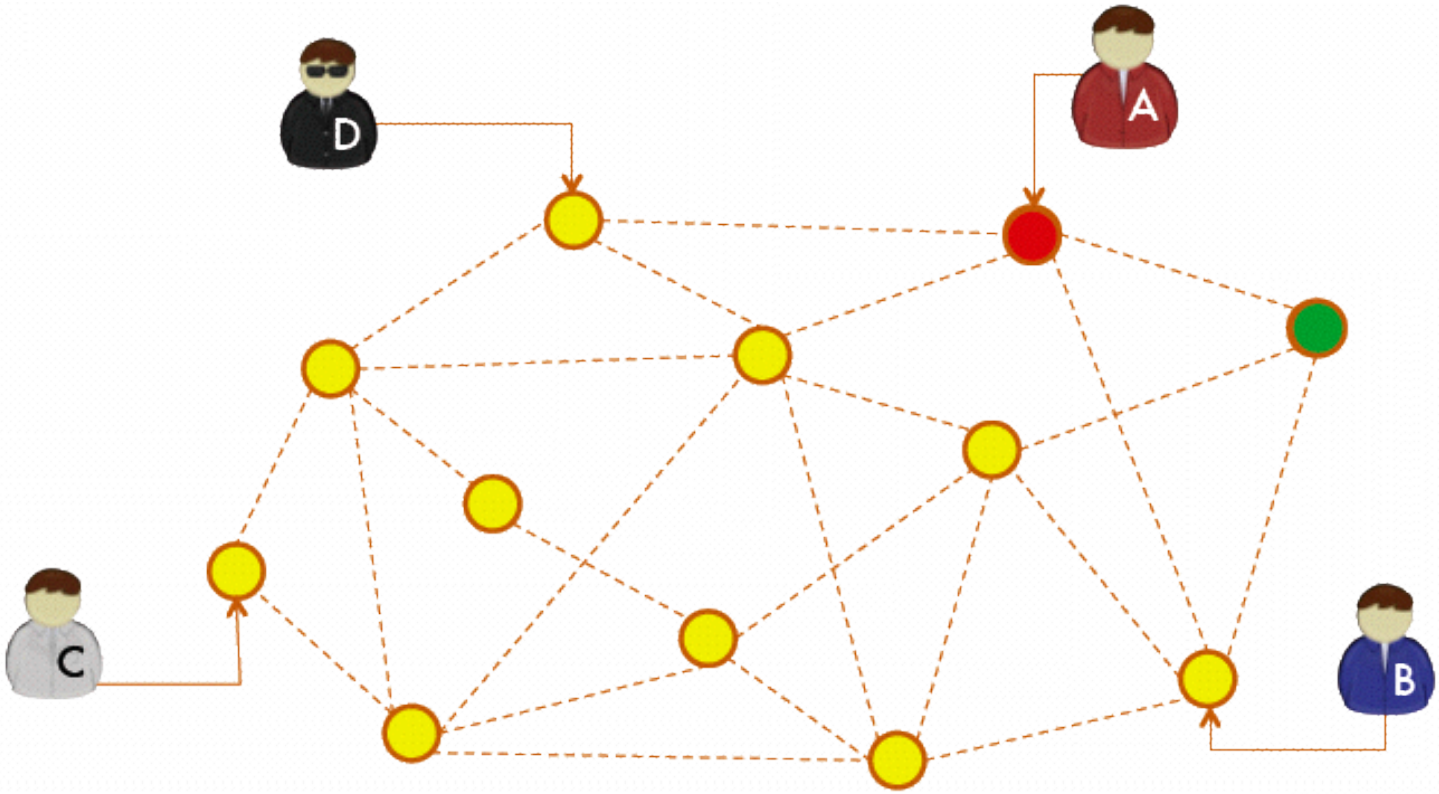
How BonjourGrid works



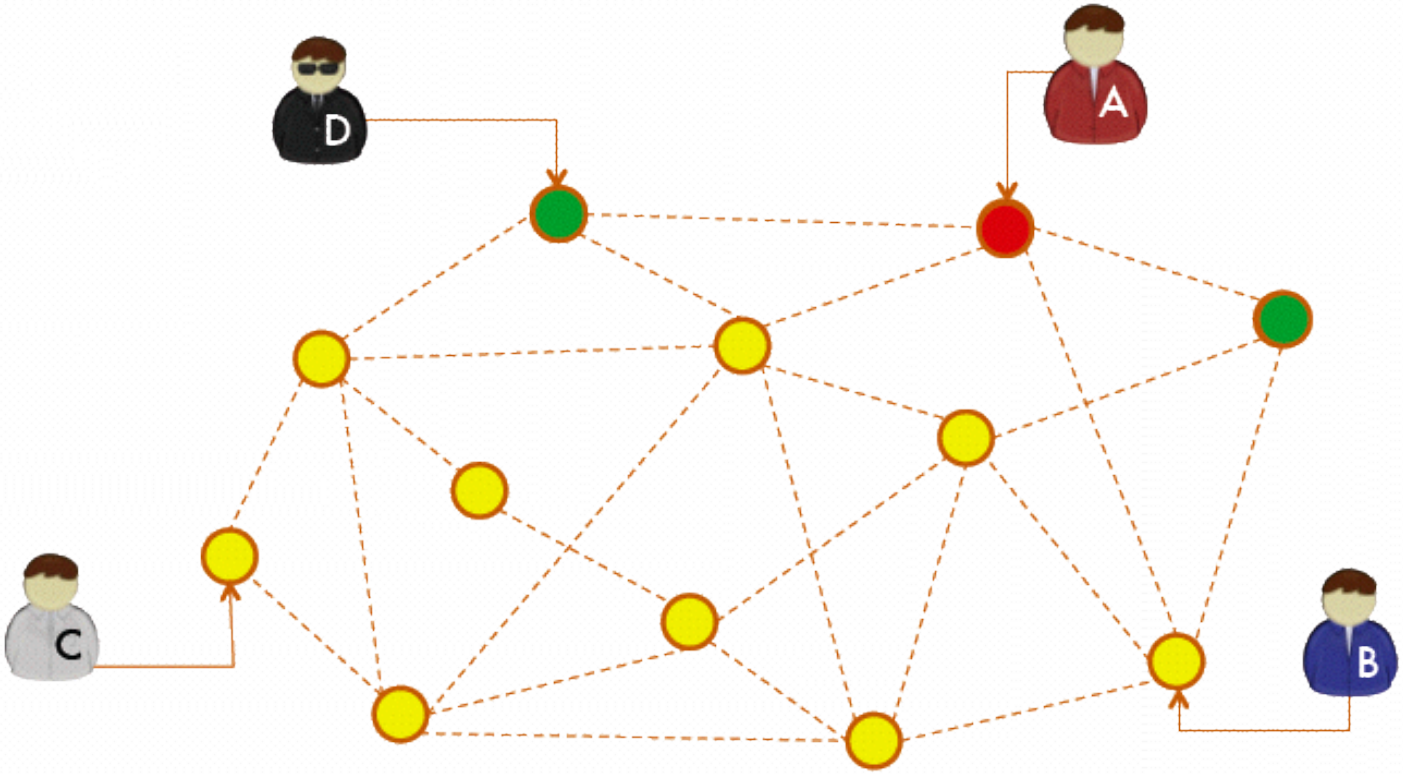
How BonjourGrid works



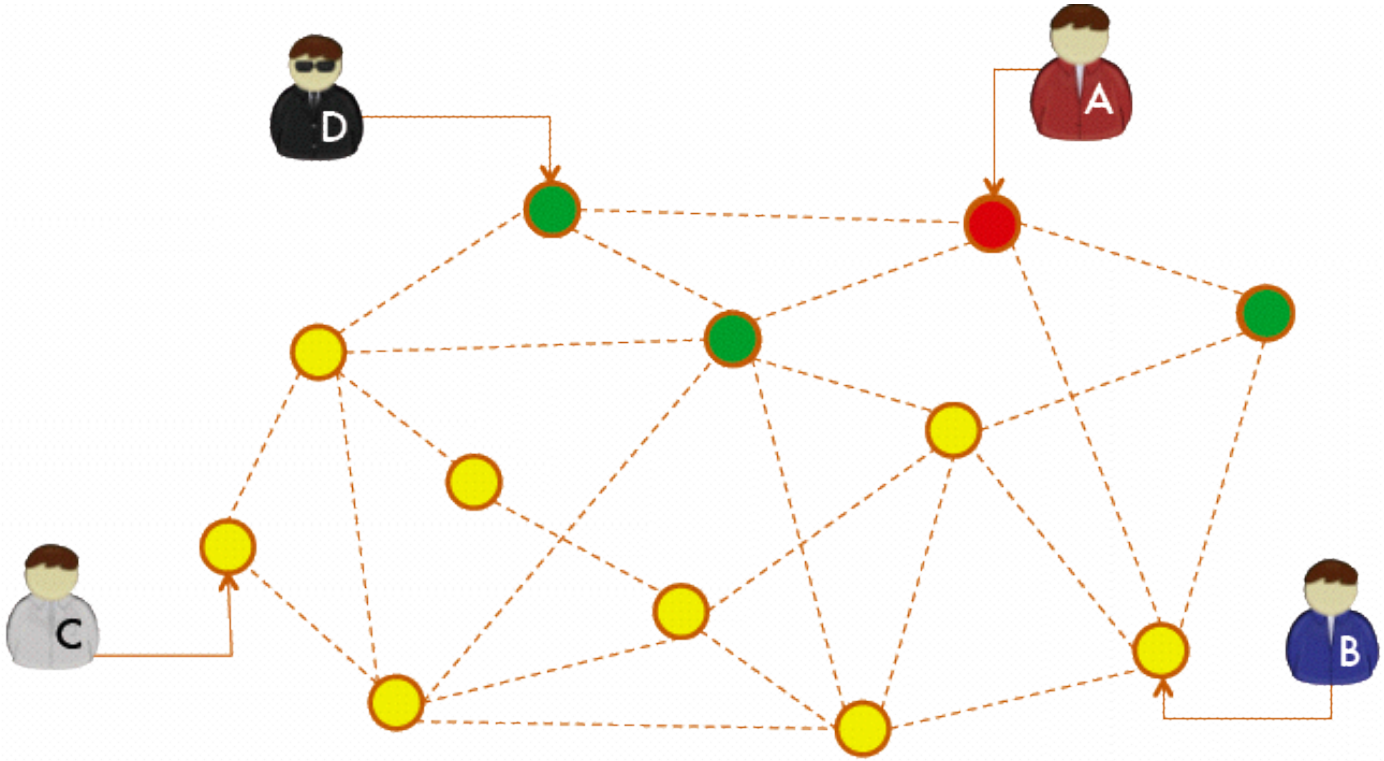
How BonjourGrid works



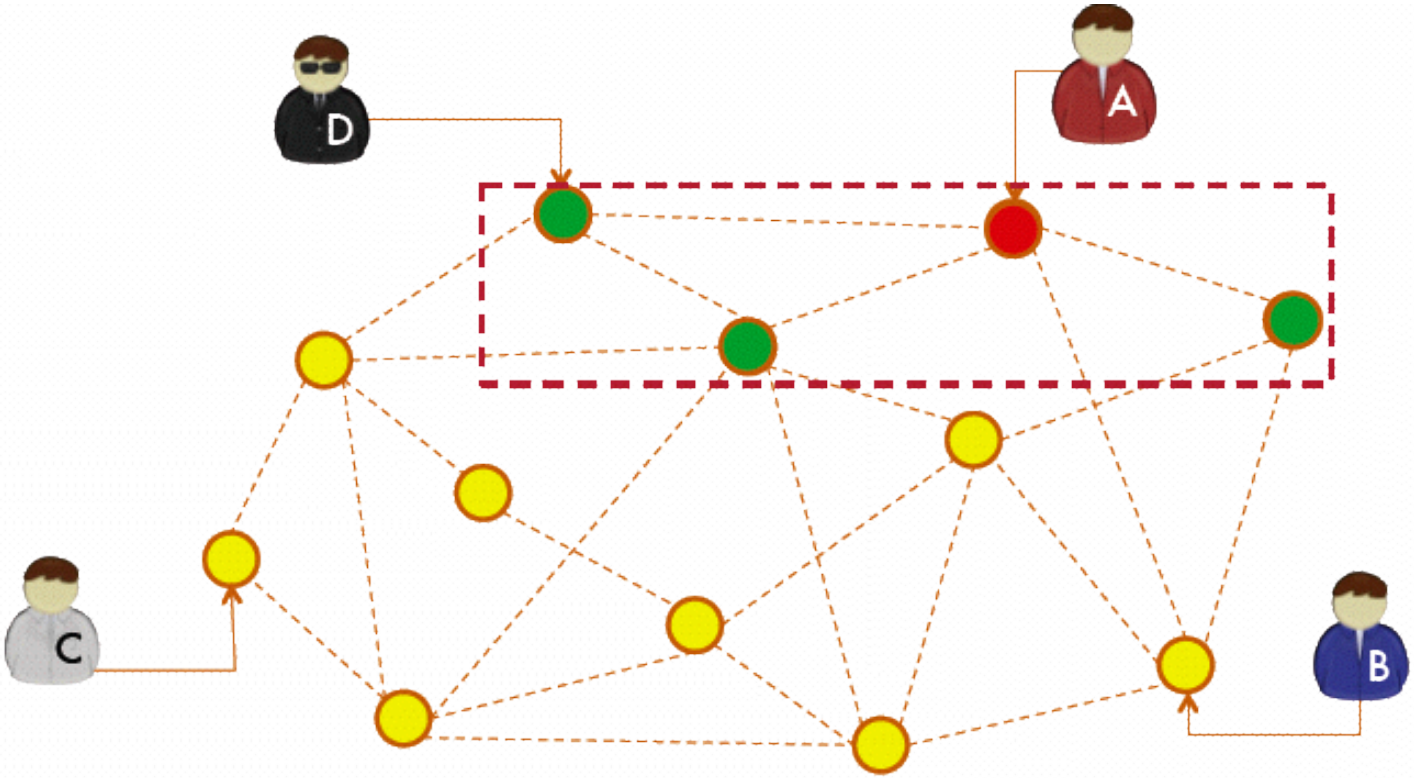
How BonjourGrid works



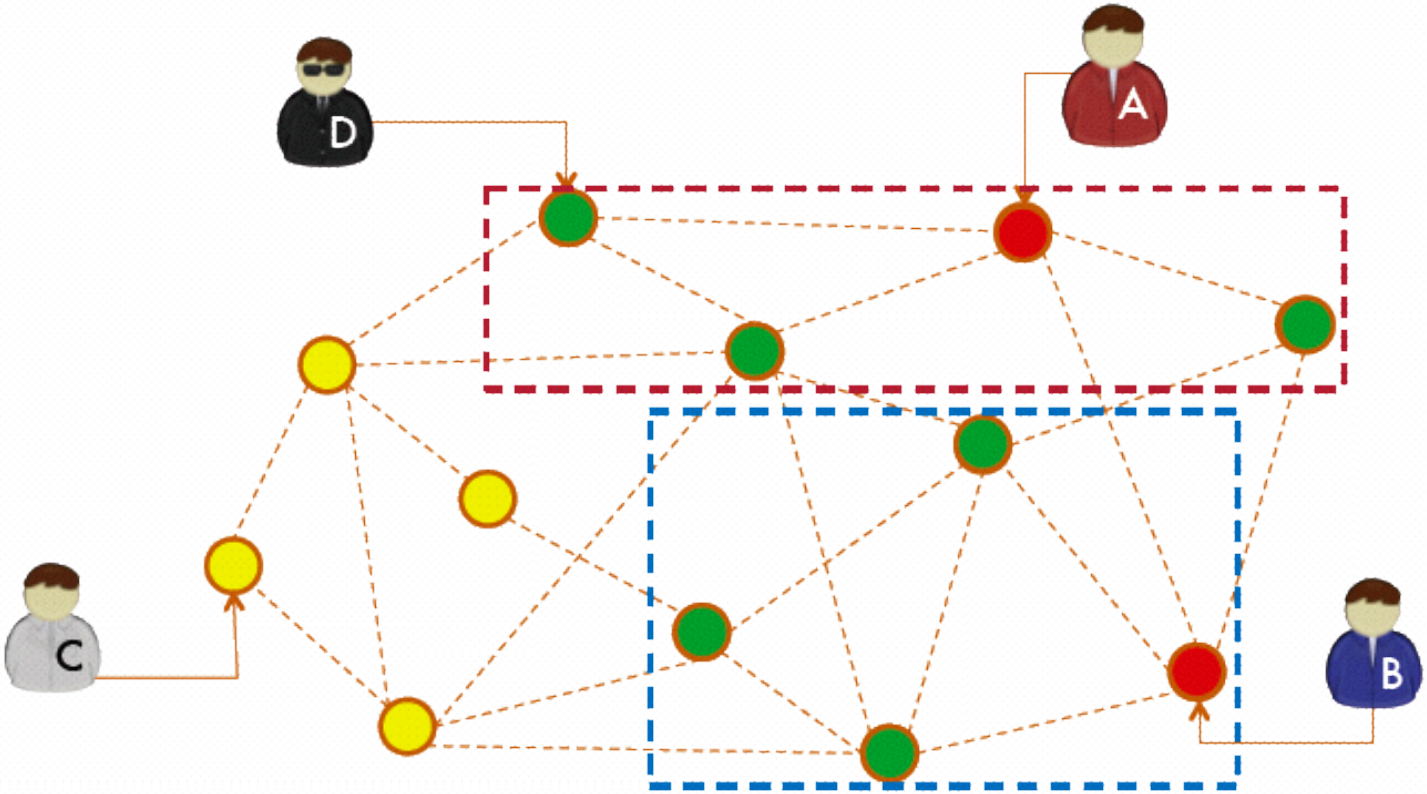
How BonjourGrid works



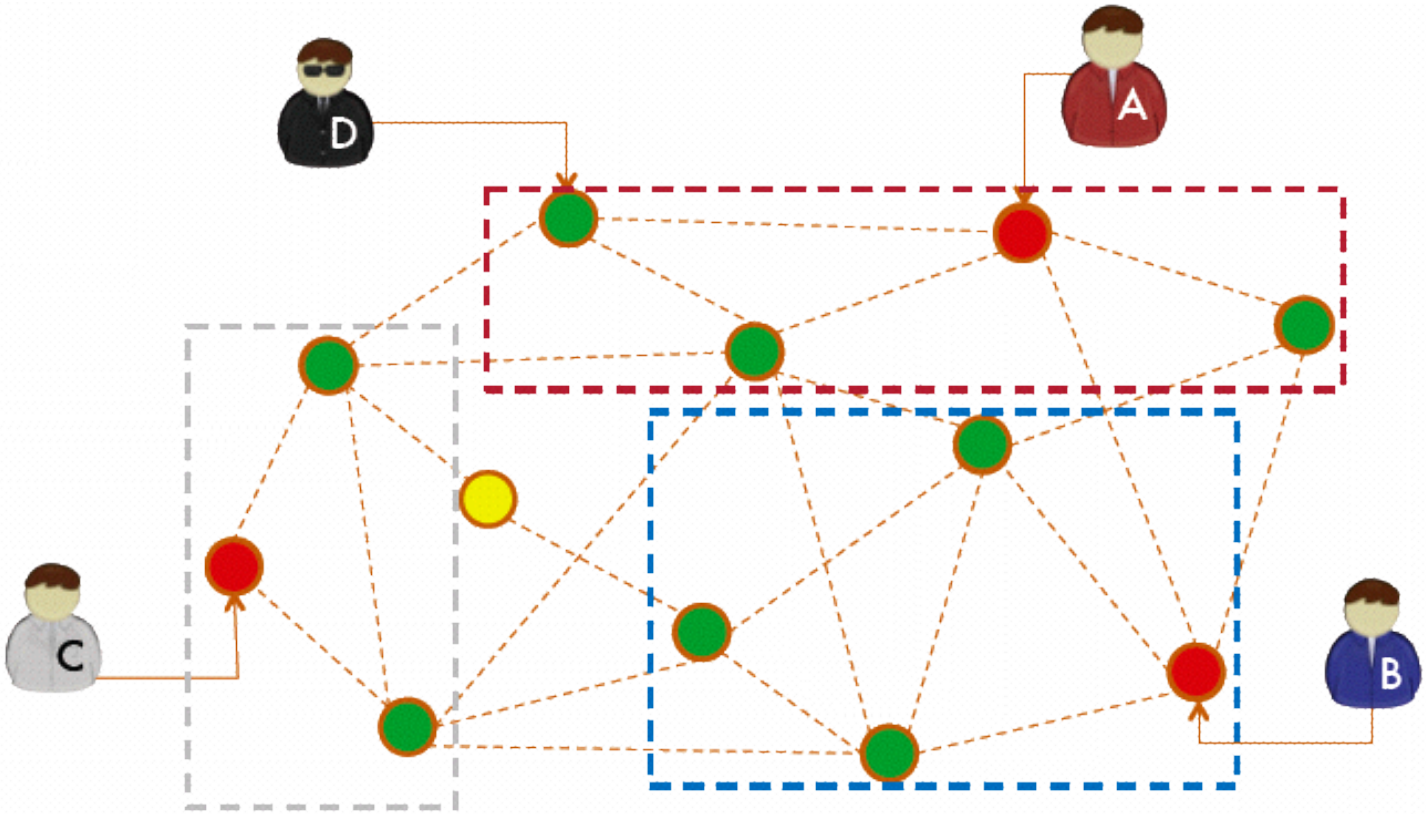
How BonjourGrid works



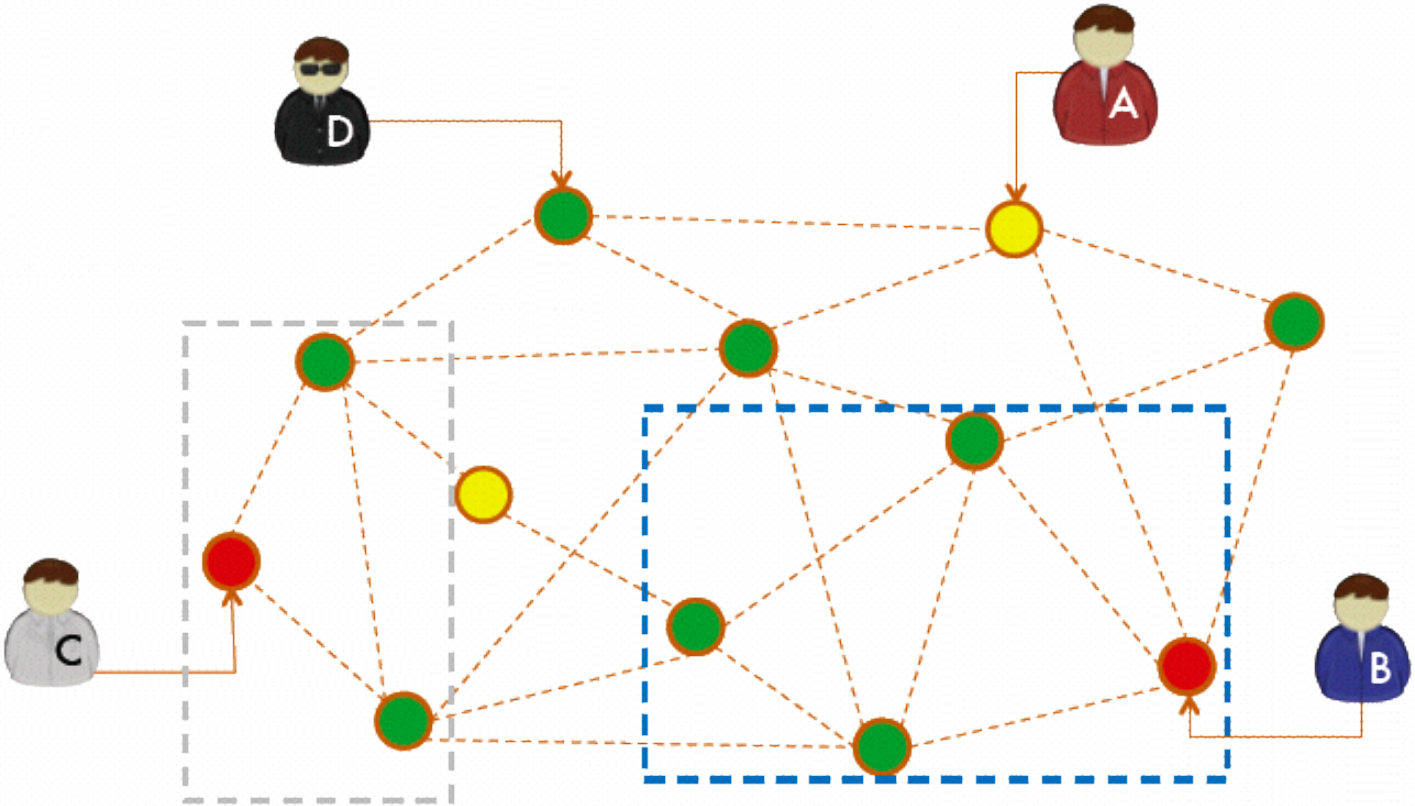
How BonjourGrid works



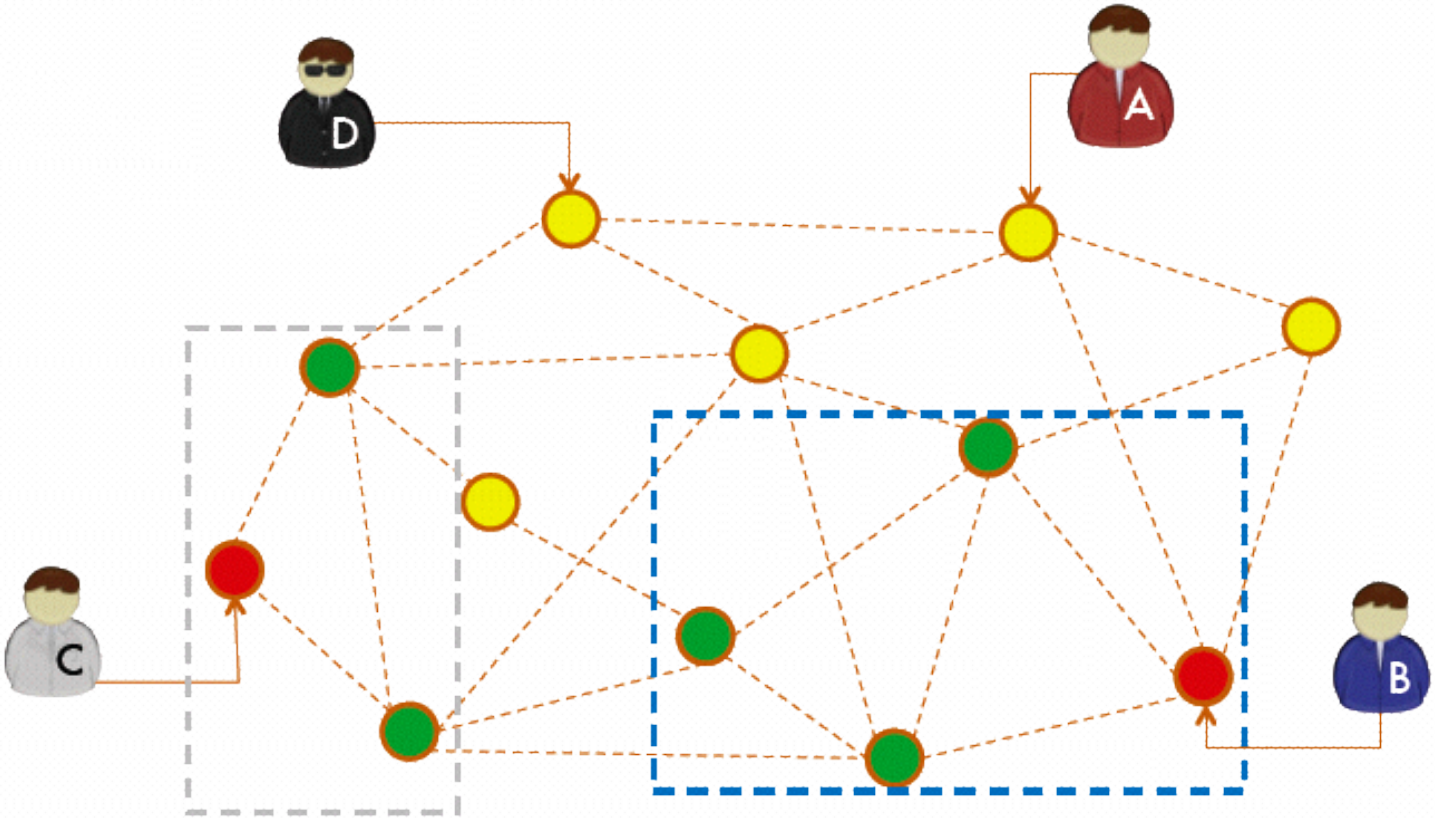
How BonjourGrid works



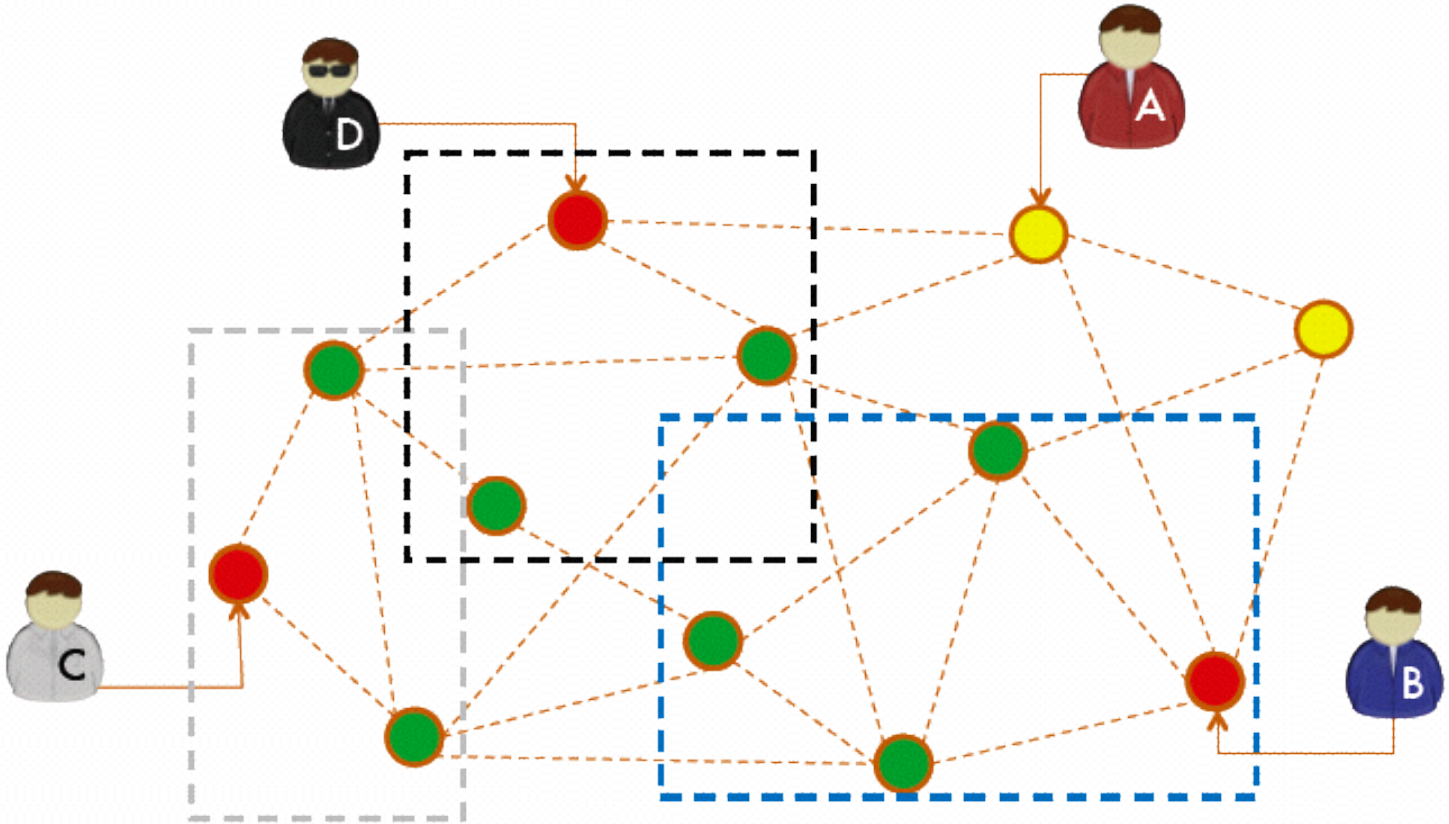
How BonjourGrid works



How BonjourGrid works



How BonjourGrid works



BonjourGrid vision

- The user requests for computation;

BonjourGrid vision

- The user requests for computation;
- The user provides the control flow graph, binaries, input data;

BonjourGrid vision

- The user requests for computation;
- The user provides the control flow graph, binaries, input data;
- The user deploys locally a coordinator and requests for participants; We support XtremWeb, Condor, BOINC.

BonjourGrid vision

- The user requests for computation;
- The user provides the control flow graph, binaries, input data;
- The user deploys locally a coordinator and requests for participants; We support XtremWeb, Condor, BOINC.
- The coordinator selects a set of machines (criteria: RAM, CPU, costs...)

BonjourGrid vision

- The user requests for computation;
- The user provides the control flow graph, binaries, input data;
- The user deploys locally a coordinator and requests for participants; We support XtremWeb, Condor, BOINC.
- The coordinator selects a set of machines (criteria: RAM, CPU, costs...)
- Upon completion, the coordinator returns to the idle state, slaves are freed and the coordination protocol:

BonjourGrid vision

- The user requests for computation;
- The user provides the control flow graph, binaries, input data;
- The user deploys locally a coordinator and requests for participants; We support XtremWeb, Condor, BOINC.
- The coordinator selects a set of machines (criteria: RAM, CPU, costs...)
- Upon completion, the coordinator returns to the idle state, slaves are freed and the coordination protocol:
  - manages and controls resources, services and computing elements;
  - does not depend on any specific machine nor any central element.

BonjourGrid vision

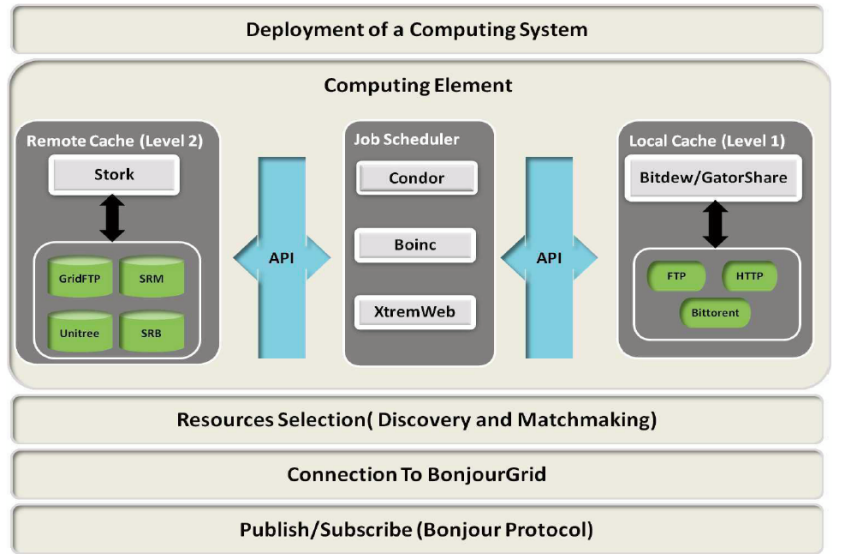
- The user requests for computation;
- The user provides the control flow graph, binaries, input data;
- The user deploys locally a coordinator and requests for participants; We support XtremWeb, Condor, BOINC.
- The coordinator selects a set of machines (criteria: RAM, CPU, costs...)
- Upon completion, the coordinator returns to the idle state, slaves are freed and the coordination protocol:
  - manages and controls resources, services and computing elements;
  - does not depend on any specific machine nor any central element.

A Self-Configurable Desktop Grid System On-demand

Solution

- A Remote Cache for data placement activities, in order to reserve the disk space for application and to transfer the required data from the user site to BonjourGrid.
- Once data are placed on the platform, a second cache called Local Cache is automatically launched to publish and to distribute data over workers.

Architecture



A Data Prefetching Model for Desktop Grids

Issues

- Data scheduling: the user must know how to use FTP, SRM tools and Globus GridFTP;
- Enables high-throughput data management for unmodified data-intensive applications;
- Hiding data management from users and applications;

Solution

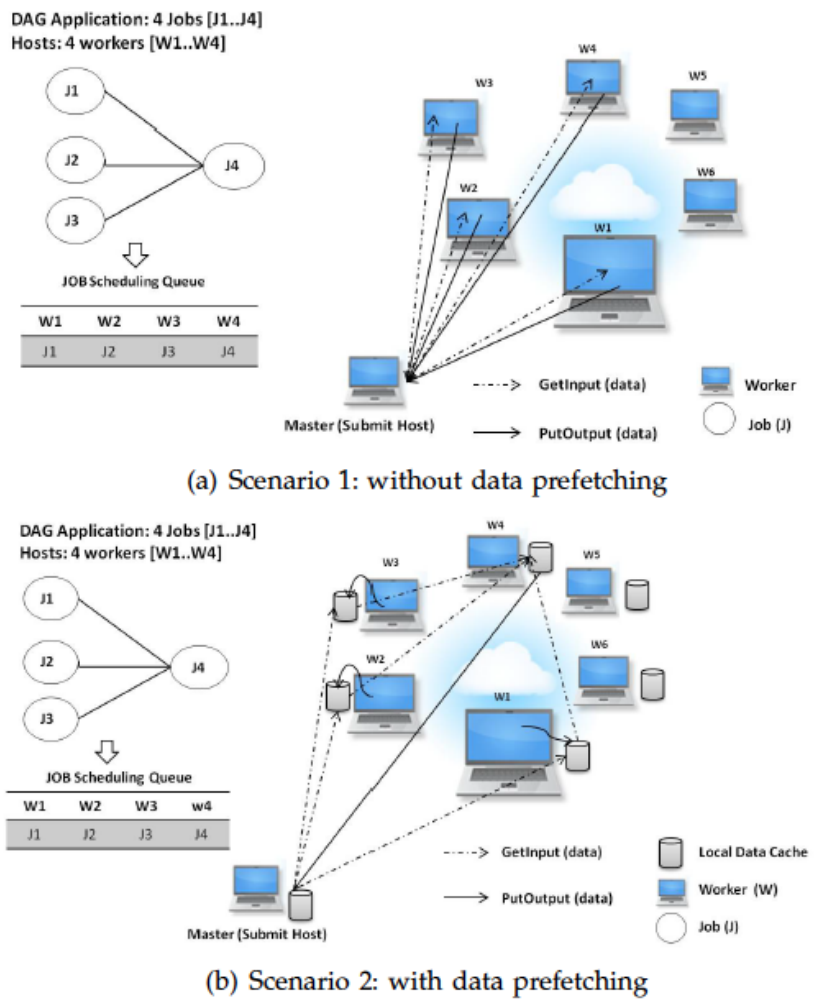
- Data prefetching: The strategies employed to manage and schedule data are as follows: 1) Input files are staged into computation site before jobs execution 2) Output files that are produced by jobs are transferred from a worker node to the storage space of the submitter node.

A Data Prefetching Model for Desktop Grids

Motivations

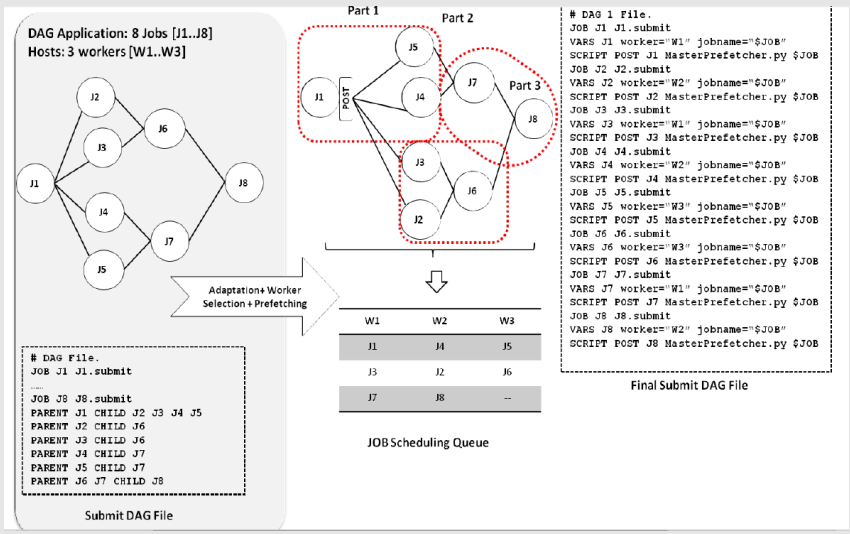
- We assume that prefetching can be used for optimizing various objectives such as:
  1. Mask data access latency between Middleware (the scheduler) and computation site (workers);
  2. Optimize the total execution time and application performance;
  3. Reduce the hierarchical master-worker paradigm impact on performance by avoiding the bottleneck induced by a single master.

Key points

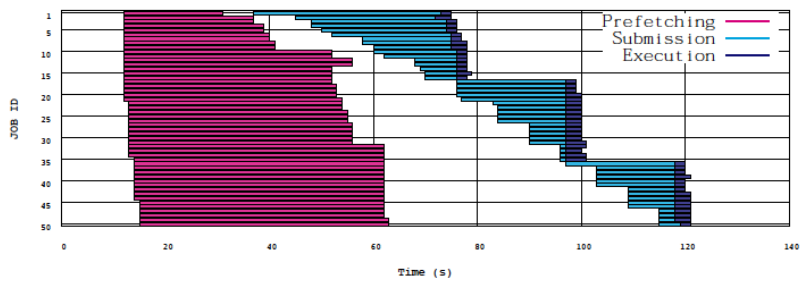


A Data Prefetching Model for Desktop Grids

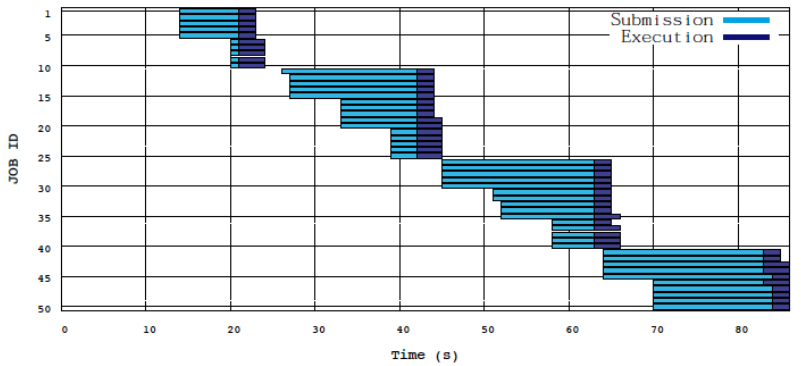
Condor use case



Key points



(a) Condor Diagram With Data Prefetching 1..50 of 300 jobs



(b) Condor Diagram Without Data Prefetching 1..50 of 300 jobs

Towards PaaS and Clouds

The new context: Platform as a Service and Cloud

- Outsourcing of software resources (Google word/spreadsheet online) and hardware resources (Amazon EC2);

Towards PaaS and Clouds

The new context: Platform as a Service and Cloud

- Outsourcing of software resources (Google word/spreadsheet online) and hardware resources (Amazon EC2);
- A variant: Platform as a Service (PaaS) where users also inherit from a complete development infrastructure (based on Javascript for the future?);

Towards PaaS and Clouds

The new context: Platform as a Service and Cloud

- Outsourcing of software resources (Google word/spreadsheet online) and hardware resources (Amazon EC2);
- A variant: Platform as a Service (PaaS) where users also inherit from a complete development infrastructure (based on Javascript for the future?);
  - No hosting problem for the user;

Towards PaaS and Clouds

The new context: Platform as a Service and Cloud

- Outsourcing of software resources (Google word/spreadsheet online) and hardware resources (Amazon EC2);
- A variant: Platform as a Service (PaaS) where users also inherit from a complete development infrastructure (based on Javascript for the future?);
  - No hosting problem for the user;
  - No update problem for the user (he always uses the last release);

Towards PaaS and Clouds

The new context: Platform as a Service and Cloud

- Outsourcing of software resources (Google word/spreadsheet online) and hardware resources (Amazon EC2);
- A variant: Platform as a Service (PaaS) where users also inherit from a complete development infrastructure (based on Javascript for the future?);
  - No hosting problem for the user;
  - No update problem for the user (he always uses the last release);
  - No maintenance, no local storage.

Towards PaaS and Clouds

The new context: Platform as a Service and Cloud

- Outsourcing of software resources (Google word/spreadsheet online) and hardware resources (Amazon EC2);
- A variant: Platform as a Service (PaaS) where users also inherit from a complete development infrastructure (based on Javascript for the future?);
  - No hosting problem for the user;
  - No update problem for the user (he always uses the last release);
  - No maintenance, no local storage.

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

- *Automate (application service provisioning)*;

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

- *Automate (application service provisioning)*;
- Availability of a service → mastering FT → redundancy;

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

- *Automate (application service provisioning)*;
- Availability of a service $\rightarrow$ mastering FT $\rightarrow$ redundancy;
- Data Lock-In: API must not be proprietary but should rely on open standards;

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

- *Automate (application service provisioning)*;
- Availability of a service → mastering FT → redundancy;
- Data Lock-In: API must not be proprietary but should rely on open standards;
- Data Transfer Bottlenecks → use P2P techniques;

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

- *Automate (application service provisioning)*;
- Availability of a service → mastering FT → redundancy;
- Data Lock-In: API must not be proprietary but should rely on open standards;
- Data Transfer Bottlenecks → use P2P techniques;
- Bugs in Large-Scale Distributed Systems → use Formal Methods to specify and to analyse architecture and protocols;

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

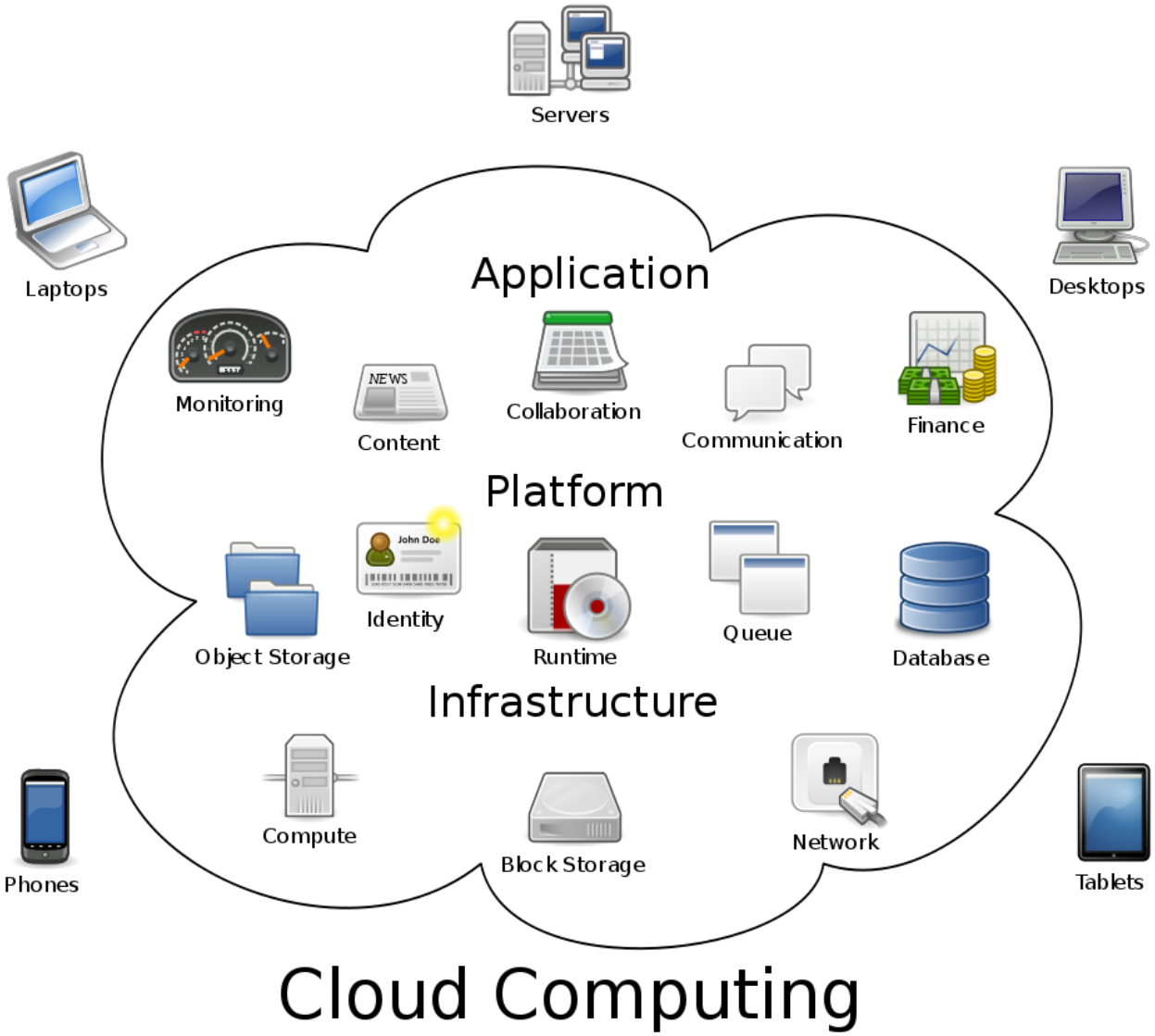
- *Automate (application service provisioning)*;
- Availability of a service → mastering FT → redundancy;
- Data Lock-In: API must not be proprietary but should rely on open standards;
- Data Transfer Bottlenecks → use P2P techniques;
- Bugs in Large-Scale Distributed Systems → use Formal Methods to specify and to analyse architecture and protocols;
- Scaling Quickly → instantiate new PaaS on the fly → define a model for cooperation and interaction;

Research opportunities

Above the Clouds: A Berkeley View of Cloud Computing

- *Automate (application service provisioning)*;
- Availability of a service → mastering FT → redundancy;
- Data Lock-In: API must not be proprietary but should rely on open standards;
- Data Transfer Bottlenecks → use P2P techniques;
- Bugs in Large-Scale Distributed Systems → use Formal Methods to specify and to analyse architecture and protocols;
- Scaling Quickly → instantiate new PaaS on the fly → define a model for cooperation and interaction;

Cloud Computing logical diagram



Some types of public cloud computing

- Infrastructure as a service (IaaS): cloud providers offer computers, as physical or more often as virtual machines, and other resources. The virtual machines are run as guests by a hypervisor, such as Xen or KVM.
- Platform as a service (PaaS): cloud providers deliver a computing platform typically including operating system, programming language execution environment, database, and web server. Application developers can develop and run their software solutions on a cloud platform without the cost and complexity of buying and managing the underlying hardware and software layers.
- Software as a service (SaaS): cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. The cloud users do not manage the cloud infrastructure and platform on which the application is running.

Issues

- Privacy: where are stored the SVG-EDIT drawings? Where is stored my calendar information?
- Compliance: how to exchange data between Google and Amazon?
- Open source: has provided the foundation for many cloud computing implementations, prominent examples being the Hadoop framework and VMware's Cloud Foundry
- Open Standards: Most cloud providers expose APIs that are typically well-documented but also unique to their implementation and thus not interoperable.
- Security: too vast problem!

Apache Libcloud: a unified interface to the cloud

- libcloud is a client library for interacting with many of the popular cloud server providers. It was created to make it easy for developers to build products that work between any of the services that it supports.
- The current version allows users to manage four different cloud resources:
  - Cloud Servers - services such as Amazon EC2 and Rackspace CloudServers (libcloud.compute.*)
  - Cloud Storage - services such as Amazon S3 and Rackspace CloudFiles (libcloud.storage.*)
  - Load Balancers as a Service, LBaaS (libcloud.loadbalancer.*)
  - DNS as a Service, DNSaaS (libcloud.dns.*)
- Compute part of the library is the oldest one and it currently supports more than 26 different providers.

Interoperability & Conclusion

- Libcloud focuses on interoperability for clouds... but are you sure that all systems run the same version of libcloud (some versions may not operate with others)?
- *Openstack (http://www.openstack.org/) is a global collaboration of developers and cloud computing technologists producing the ubiquitous open source cloud computing platform for public and private clouds. The project aims to deliver solutions for all types of clouds by being simple to implement, massively scalable, and feature rich. The technology consists of a series of interrelated projects delivering various components for a cloud infrastructure solution.* Same problem! The technology is *heavy*!
- SlapOS (https://slapos.cloud.univ-paris13.fr) adopts an orthogonal view for interoperability: deploy your own OS, then download the software release for the hardware node... like with Grid5000!

The Holly Trinity

To build a cloud, you need...

1. An ERP (Enterprise Resource Planning): catalog of services ; finance/accounting, manufacturing, sales and service, customer relationship management, etc.;
2. A model for the deployment of applications;
3. Computing nodes / volunteers;

The Holly Trinity

To build a cloud, you need...

1. An ERP (Enterprise Resource Planning): catalog of services ; finance/accounting, manufacturing, sales and service, customer relationship management, etc.;
2. A model for the deployment of applications;
3. Computing nodes / volunteers;

- *FORGET WHAT I HAVE SAID BEFORE!*

The Holly Trinity

To build a cloud, you need...

1. An ERP (Enterprise Resource Planning): catalog of services ; finance/accounting, manufacturing, sales and service, customer relationship management, etc.;
2. A model for the deployment of applications;
3. Computing nodes / volunteers;

- *FORGET WHAT I HAVE SAID BEFORE!*
- Please, count only on the OS for isolation, security, elasticity... Why stacking middleware, virtual machines?
- And how many different abstractions developer really needs? Thread? Process? VM Instance? VM Instance Group? Could this hierarchy be radically simplified with just one abstraction? Could POSIX process abstraction be a candidate here?

Some research question

To build a cloud, you need...

- Traditional virtualization has a major advantage, they are backward compatible on binary level, making it easy to run any existing pre-cloud application on the cloud. But what if backward compatibility is not a requirement? What about new, cloud-native application? Is traditional virtualization still a good platform for them?

Some research question

To build a cloud, you need...

- Traditional virtualization has a major advantage, they are backward compatible on binary level, making it easy to run any existing pre-cloud application on the cloud. But what if backward compatibility is not a requirement? What about new, cloud-native application? Is traditional virtualization still a good platform for them?
- Does user really need to manage that emulated virtual interrupt controller on every VM instance of his compute cloud?

Some research question

To build a cloud, you need...

- Traditional virtualization has a major advantage, they are backward compatible on binary level, making it easy to run any existing pre-cloud application on the cloud. But what if backward compatibility is not a requirement? What about new, cloud-native application? Is traditional virtualization still a good platform for them?
- Does user really need to manage that emulated virtual interrupt controller on every VM instance of his compute cloud?
- VMWare price. Offer the security/isolation/elasticity services without VM;

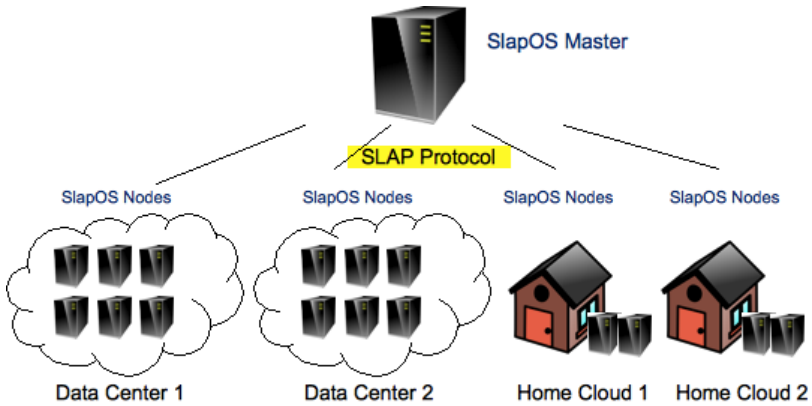
Some research question

To build a cloud, you need...

- Traditional virtualization has a major advantage, they are backward compatible on binary level, making it easy to run any existing pre-cloud application on the cloud. But what if backward compatibility is not a requirement? What about new, cloud-native application? Is traditional virtualization still a good platform for them?
- Does user really need to manage that emulated virtual interrupt controller on every VM instance of his compute cloud?
- VMWare price. Offer the security/isolation/elasticity services without VM;
- Where is and who manage the data?

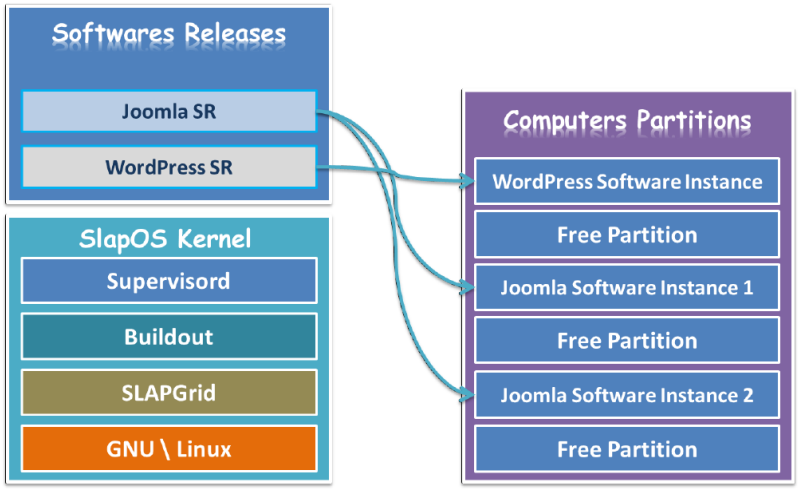
SlapOS Technologies / Architecture

- GNU/Linux Operating system;
- SLAPGrid daemon for installing software and instantiate any number of processes from an installed software;
- Buildout environment for bootstrapping applications;
- Supervisord to control running processes;



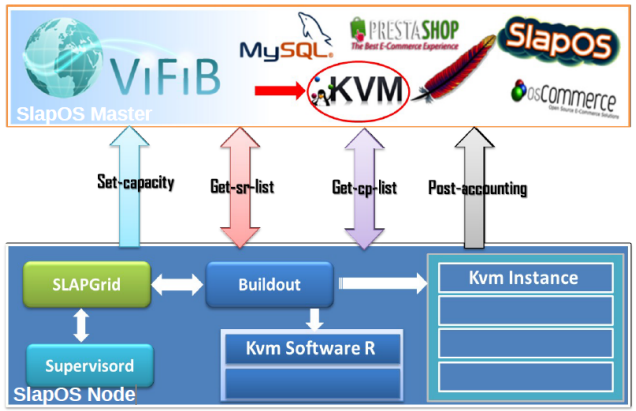
SlapOS Node (Worker/Volunteer) Architecture

- A node: set of predefined partitions (Unix);
- A partition hosts a software release;
- Minimal isolation (LXC);

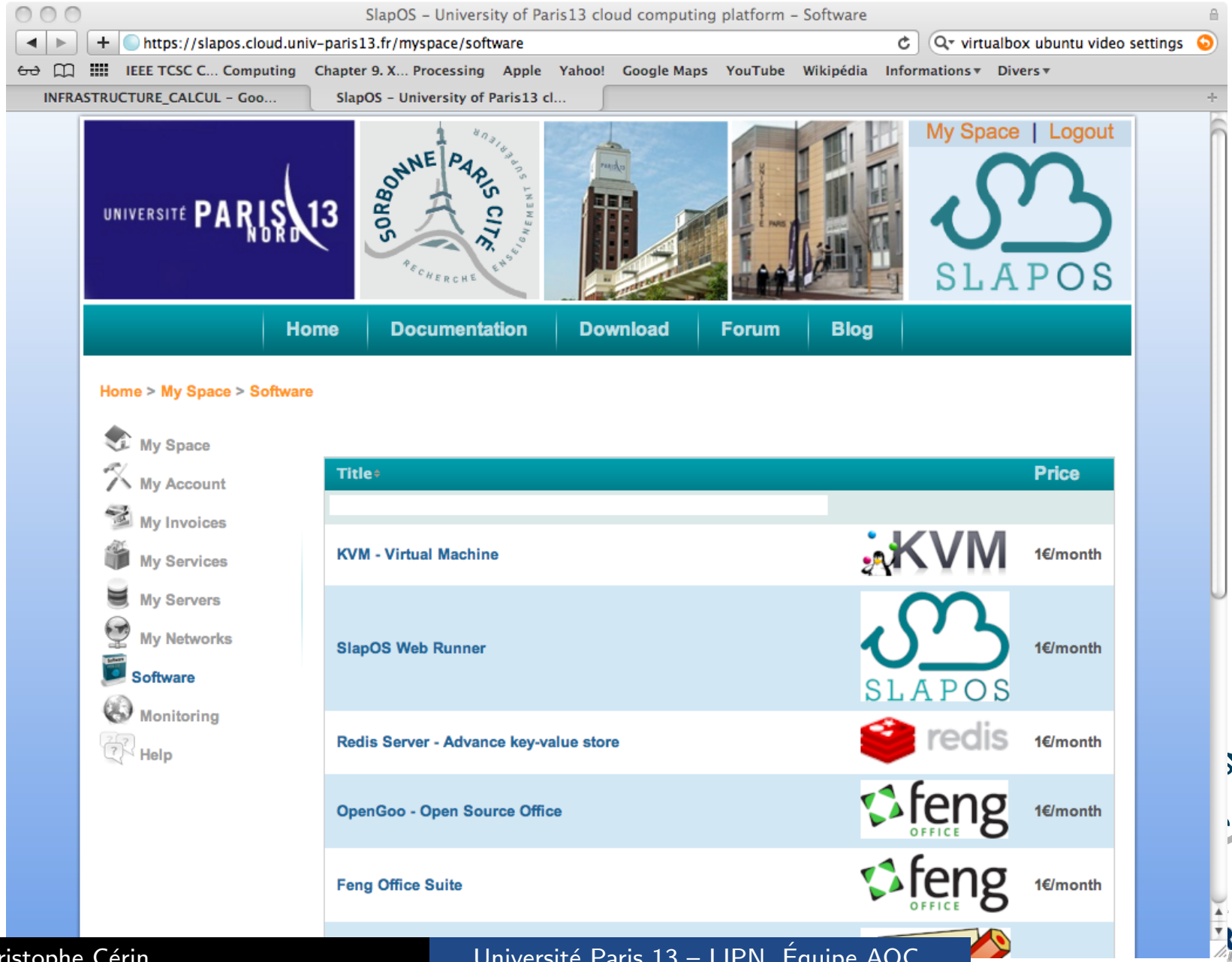


SlapOS Node (Worker/Volunteer) Architecture

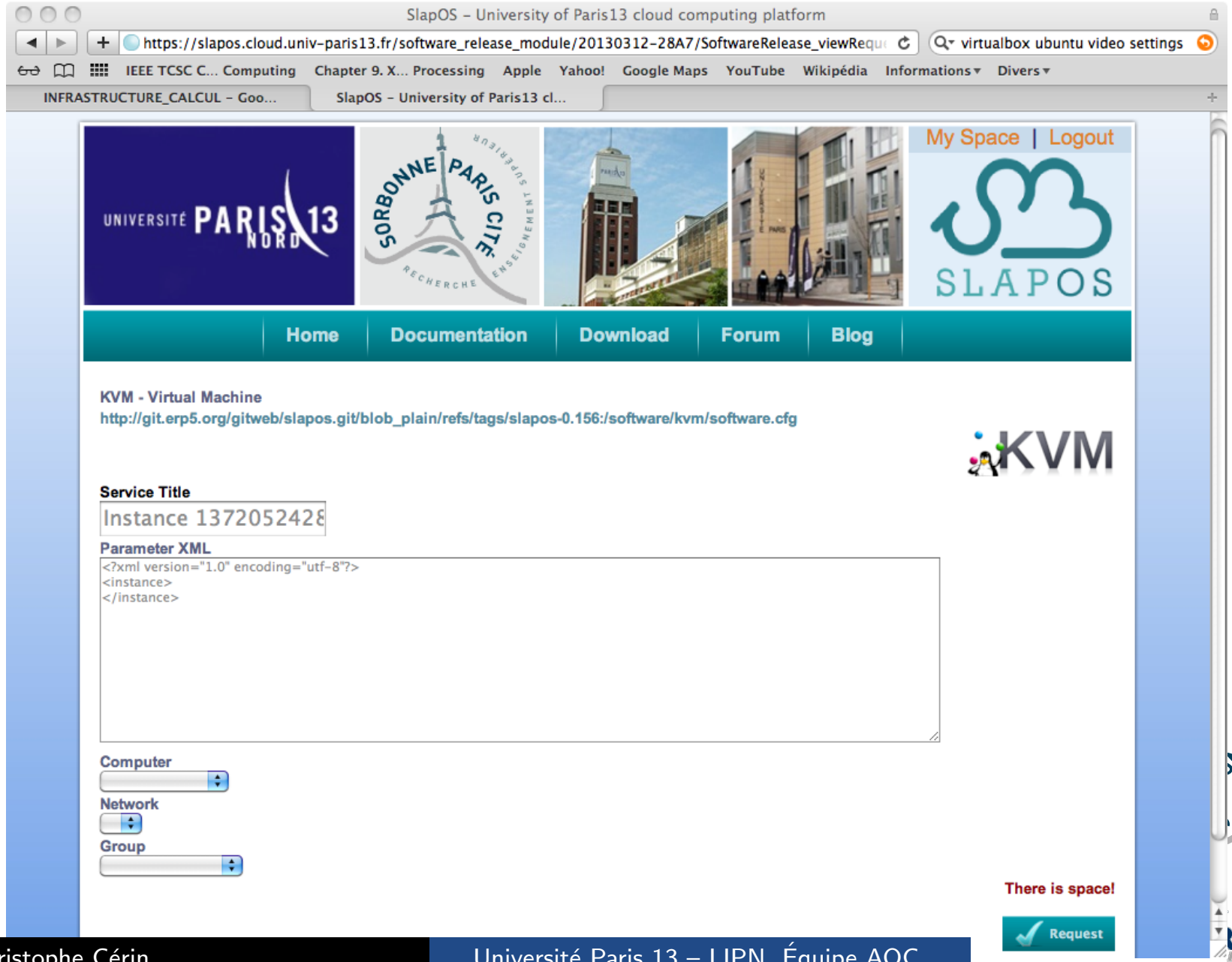
- Software Release: provide all components and programs required to run a specific software;
- Software Instance: instance of a specific software release;
- Buildout profile: Buildout configuration file with rules for install or deployment of software;
- Computer Partition: a directory with a dedicated user, private ipv4, global ipv6, network interfaces ⇒ isolated Runtime environment with ssl certificate.



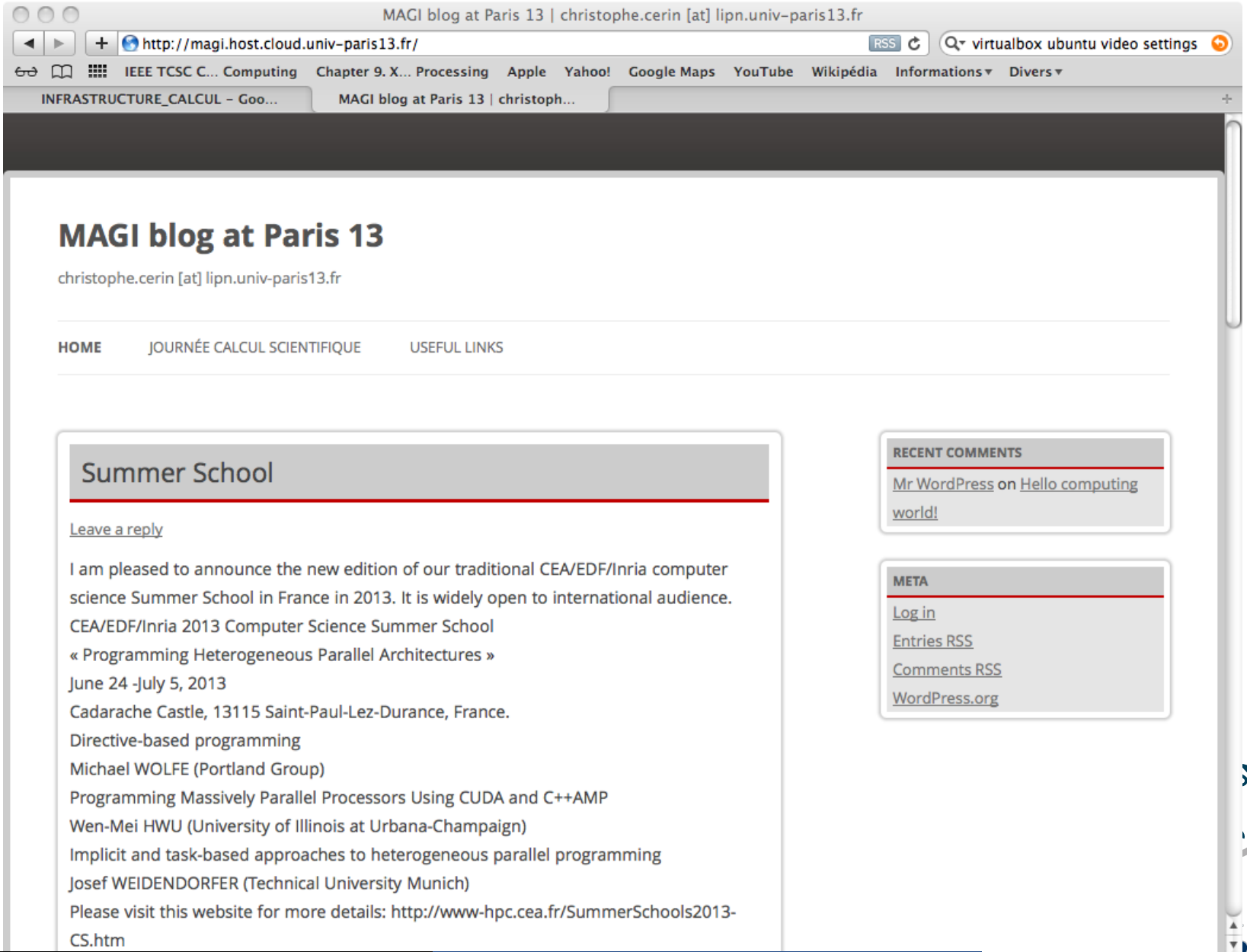
SlapOS is running



SlapOS is running



SlapOS is running



Lessons learned from the use cases

- The lack of IPv6 in companies → external VPN use → latency increase;

Lessons learned from the use cases

- The lack of IPv6 in companies → external VPN use → latency increase;
- IPv6 Compatibility of applications: impossibility of using the IPv6 address for compiling/configuring the BOINC-client app;

Lessons learned from the use cases

- The lack of IPv6 in companies → external VPN use → latency increase;
- IPv6 Compatibility of applications: impossibility of using the IPv6 address for compiling/configuring the BOINC-client app;
- Impossibility to run an application as root ⇒ no problem for BOINC, Condor, BonjourGrid... but problems with Openstack (we use a VM and in the VM we configure the network interfaces);

Lessons learned from the use cases

- The lack of IPv6 in companies → external VPN use → latency increase;
- IPv6 Compatibility of applications: impossibility of using the IPv6 address for compiling/configuring the BOINC-client app;
- Impossibility to run an application as root ⇒ no problem for BOINC, Condor, BonjourGrid... but problems with Openstack (we use a VM and in the VM we configure the network interfaces);
- Impossibility for SlapOS to generate a hostname (no management of DNS). The service is viewed as an IPv6 address, not through a logical name. Problem with the Condor configuration;

Lessons learned from the use cases

- The lack of IPv6 in companies → external VPN use → latency increase;
- IPv6 Compatibility of applications: impossibility of using the IPv6 address for compiling/configuring the BOINC-client app;
- Impossibility to run an application as root ⇒ no problem for BOINC, Condor, BonjourGrid... but problems with Openstack (we use a VM and in the VM we configure the network interfaces);
- Impossibility for SlapOS to generate a hostname (no management of DNS). The service is viewed as an IPv6 address, not through a logical name. Problem with the Condor configuration;

Lessons learned from the use cases

- Configuring the mail: some applications (Condor) send mails! It is still difficult to automate the system for sending messages with SMTP for the SlapOS instances;

Lessons learned from the use cases

- Configuring the mail: some applications (Condor) send mails! It is still difficult to automate the system for sending messages with SMTP for the SlapOS instances;
- Openstack in SlapOS: we need redirection between IPv4 and IPv6 to let SlapOS running with IPv6 and Openstack in IPv4 (currently it is easier to work with IPv4 for Openstack to configure network interfaces. . . not yet automated!)

Conclusion

What we have seen?

- System is a rich field. . . but hidden ;

Conclusion

What we have seen?

- System is a rich field. . . but hidden ;
- Many opportunities to work with people coming from Applications, Mining, Analytics. . .

Conclusion

What we have seen?

- System is a rich field. . . but hidden ;
- Many opportunities to work with people coming from Applications, Mining, Analytics. . .
- Clusters will survive. Grids will survive more or less for special needs (elasticity). Clouds will absorb / fusion with HPC, Business driven applications.

Special Thanks and announcement

Leila, Camille, Alain, Walid, Yanik, Heithem, Nicolas. . . MAGI

Special Thanks and announcement

Leila, Camille, Alain, Walid, Yanik, Heithem, Nicolas. . . MAGI

Seminar (Dec. 2nd, 2013 @ UP7): The big seminar on Systems for intensive computing. Location: Amphiteatre at Condorcet (10 rue Alice Domon et Leonie Duquet, 75013 Paris)
Three speakers: Cécile Cavet (UP7), Yanik Ngoko (UP13), Nadjib Ait Saadi University of Paris-Est Creteil Val de Marne (UPEC).

Gestion des données dans les grilles, les clusters et le cloud – LIPN, AOC Team –

Christophe Cérin[1]

[1]Université de Paris XIII, LIPN, CNRS UMR 7030, France

Marseille Workshop on Scientific Data Preservation