# Distributed Computing with SAGE

Yi Qiang

yi@yiqiang.net

## Distributed Computation

- **What is distributed computing?**
  - Distributed computing is decentralized and parallel
  - Computers speak to each other over a network, now days known as the “Internet”.
  - Similar to clustering, but much cheaper and infinitely times more scalable.
  - Heterogeneous
    - We don't care what kind of hardware you have
    - We don't care what OS you run
    - We don't care about your geographical location
    - Only two requirements:
      - SAGE
      - Internet connectivity
  - Lots of **IDLE** computer time we could utilize to solve interesting math problems!

## Examples of Distributed Computing

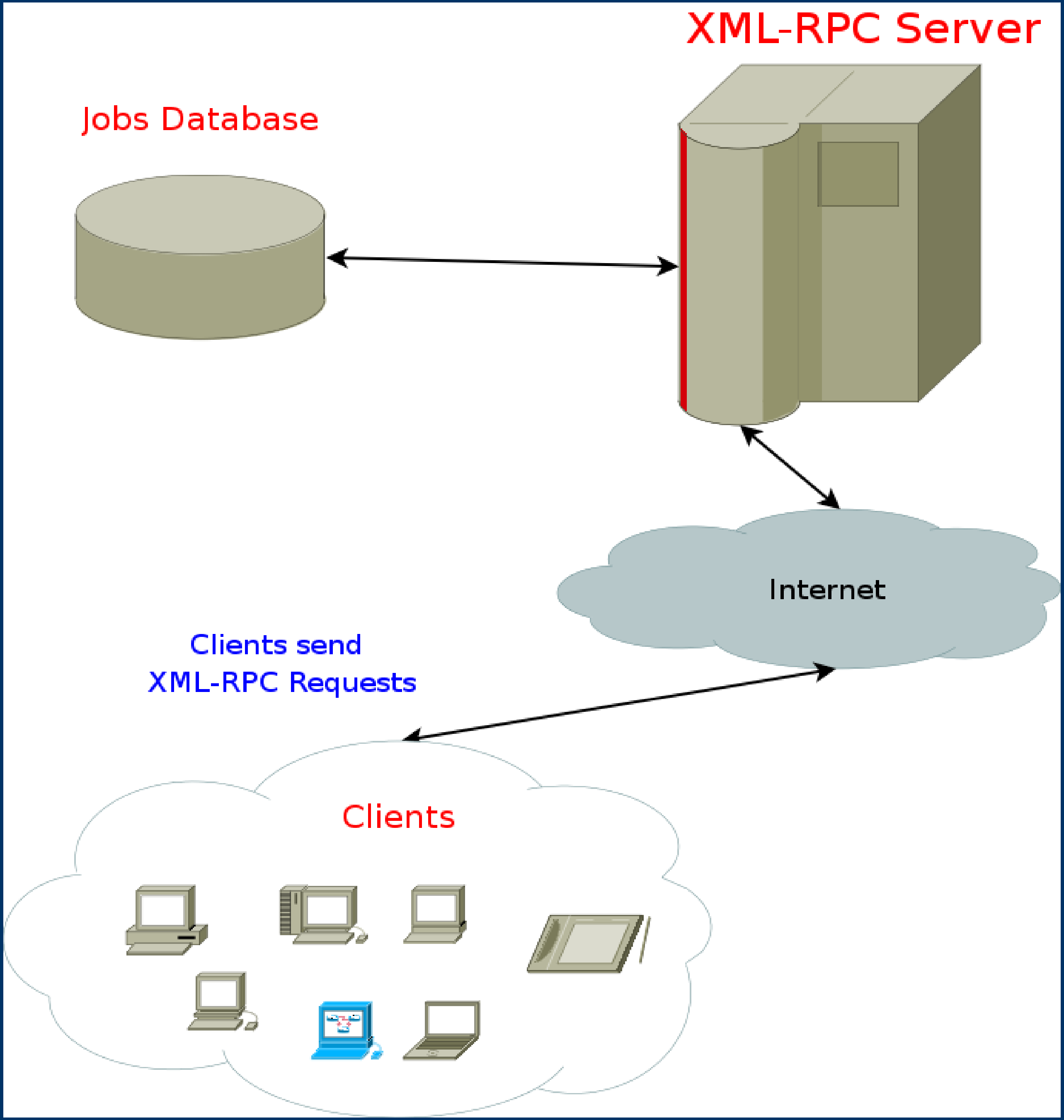
- distributed.net
  - Attemps to break various encryption standards (Finished RC5-64 in 2002, working on RC5-72)
- SETI@Home
  - Searches for signs of extra-terrestrial intelligence
- Folding@Home
  - Tries to understand why proteins misfold
- Many more... for a good list, check out: http://en.wikipedia.org/wiki/List_of_distributed_computing_projects
- Oh...I almost forgot to mention GIMPS, the Great Internet Mersenne Prime Search

## And then there is GIMPS

- Great Internet Mersenne Prime Search
- On December 15, 2005, Dr. Curtis Cooper and Dr. Steven Boone, professors at Central Missouri State University, discovered the 43rd Mersenne Prime, 2^30,402,457-1

## How to integrate distributed computing with SAGE

- Client-Server Model
- Using robust and yet simple python frameworks
- We need XML-RPC, BSDDB and a thread safe Queue.  Those are almost enough to accomplish this!
- What python offers with it's standard library:
  - SimpleXMLRPCServer
  - bsddb
  - thread safe Queue


XML-RPC Server
Jobs Database
Internet
Clients send XML-RPC Requests
Clients

## Just like 1-2-3

- How easy is it to create an XML-RPC server in python? Simple as 1-2-3!

```
(1) import SimpleXMLRPCServer
(2) server = SimpleXMLRPCServer.SimpleXMLRPCServer(('localhost', 8000))
(3) server.serve_forever()
```

## What problems are good for distributed computing?

- Easily parallelized
  - Tasks can be split into smaller chunks
  - Tasks do not depend on other tasks
- Clients do not need to communicate with one another (this is not P2P)
- Tractable verification of answers
  - Faulty hardware/software will cause incorrect answers to be produced!

## *TODO:*

- Discover more problems that we can solve using distributed computing
  - Cremona's Database
- Test scalability of Python's SimpleXMLRPCServer
  - Possible alternatives include Medusa and Twisted, both are supposed to scale much better
- Generate fancy statistics
- Easy job submission by scientists, web interface
- Protection against hackers/crackers/cheaters
- Google Summer of Code.