

Agenda

- **Last time:**
 - Brief CORBA overview
 - General pros/cons of RPC/RMI
 - Security intro (chapt 7)
 - Assignment #1 Due, Assignment #2 Out (due Thur Mar 1 – 2 weeks)
- **This time**
 - "A Note on Distributed Computing"
 - Security (chapt 7)
- **Next time (Tues Feb 27)**
 - Security (chapt 7)
- **Note: No class next Thursday Feb 22 (Marty at Conference)**

CS451: Distributed Systems (Spring 2007)

Before we start: Assignment #2

- **Note: sample from japan is good, but I don't think you should use this pattern of "embedding" a rregistry in the server – use my sample code instead (on class Web)**
- **To use my sample code**
 - Command prompt (with path set): rregistry 6666 (NOTE: must be in same directory as sample code)
 - Command prompt (with path set):
 - javac MyRemote.java MyRemotImpl.java
 - rmic MyRemotImpl
 - java MyRemotImpl
 - Command prompt (with path set): rregistry 6666
 - javac MyLocal.java MyRemote.java
 - java MyLocal

CS451: Distributed Systems (Spring 2007)

Before we start: Assignment #2

- **Sample java socket code is at <http://java.sun.com/docs/books/tutorial/networking/socket/clientServer.html>**
 - Look at the "knockknockserver" class
- **Cygwin not necessary for code dev for Assignment #2**
- **Visual Studio not necessary**
- **My files**
 - IMServer.java (397 lines)
 - IMServerInterface.java (15 lines)
 - OutwardFacing.java (176 lines)
 - Registered.java (31 lines)
 - testRegistered.java (37 lines)

CS451: Distributed Systems (Spring 2007)

Before we start: Assignment #2

- **My dev env (note: only one machine used)**
 - Jcreator (editing and compiling – configure → options → JDK Profiles → new)
 - Command prompt (with path set): rregistry 6666
 - (NOTE: must be in same directory as server code)
 - Command prompt (with path set):
 - javac IMServer.java IMServerInterface.java
 - java IMServer IMServer1 6666
 - Command prompt (with path set):
 - java IMServer IMServer2 6666
 - Command prompt (with path set):
 - javac OutwardFacing
 - java OutwardFacing args
 - **Cygwin: run your existing client**

CS451: Distributed Systems (Spring 2007)

Before we start: Assignment #2

- **Persistence:**
 - File system
 - DB
- **String []splits = str.split(" ");**
- **The "Vector" class might be useful**
- **Use "rebind", not "bind" (for the java Naming class)**
- **A reasonable design:**
 - Use the file system for persistence
 - **Both partners: [design protocols and IMServerInterface.java](#)**
 - One partner: design and implement OutwardFacing.java
 - One partner: design and implement IMServer.java
- **Get "register" working first**

CS451: Distributed Systems (Spring 2007)

Before we start: Assignment #2

- **By Friday 5pm, your team is required (10% of your grade) to send me email:**
 - Text of email: description of your algorithms for dealing with one server's failure
 - What to do when one server comes back on-line (after failure), and/or
 - How will you ensure that the two servers are synchronized
 - Attachment: IMServerInterface.java
- **Note: this information must "make sense", but you are not bound by this (i.e., you can change it later if you want to)**

CS451: Distributed Systems (Spring 2007)

Before we start: submission of PA #2

•Partner 1 must :

1. Hand in a paper copy of his/her answers to Part 1 (the book questions). This paper copy must clearly state the name of Partner 1, and additionally state "Part 2 done with *Partner 2 name*".
2. Email an electronic copy of his/her answers to humphrey@cs.virginia.edu with the subject line "CS451 Assignment 2 Part 1".

•Partner 2 must :

1. Hand in a paper copy of his/her answers to Part 1 (the book questions). This paper copy must clearly state the name of Partner 2, and additionally state "Part 2 done with *Partner 1 name*".
2. Email an electronic copy of his/her answers to humphrey@cs.virginia.edu with the subject line "CS451 Assignment 2 Part 1".
3. Hand in a paper copy of the team's Part 2. This paper copy must clearly state the names of both partners.
4. Email an electronic copy of the team's Part 2 to humphrey@cs.virginia.edu with the subject line "CS451 Assignment 2 Part 2 (*Partner 1 name, Partner 2 name*)".

CS451: Distributed Systems (Spring 2007)

"A Note on Distributed Computing"

•Jim Waldo: Sun engineer, instrumental in ORB design (while@HP), designed Jini @ Sun

•Java RMI: Feb 1997, this paper: Nov 1994

•"much of the current work in distributed, o-o systems is based on the assumption that objects form a single ontological class."

•"work on distributed O-O systems that ignores or denies these differences is doomed to failure and could easily lead to an industry-wide rejection of the notion of distributed O-O systems"

• Authors specifically identify CORBA

•Phases in "unified object model" development

- Phase 1: Write app without worrying where objects are
- Phase 2: tune performance via "concretizing" object locations
- Phase 3: test with "real bullets"

CS451: Distributed Systems (Spring 2007)

More...

•"seeing location as a part of the implementation of an object and therefore as part of the state that an object hides from the outside world appears to be a natural extension of the o-o paradigm."

•False principles:

- "natural o-o design is independent of the deployment context"
- "Failure and performance are tied to the implementation of an object and should be excluded from the initial design"
- "interface of an object is independent of the context in which the object is used".

•"The desire to merge the programming and computational models of local and remote computing is not new."

CS451: Distributed Systems (Spring 2007)

More...

•"The hard problems in distributed computing are not the problems of how to get things on and off the wire."

•"The hard problems..."

- concern dealing with partial failure and a lack of central resource manager....
- Concern insuring adequate performance and dealing with problems of concurrency...
- have to do with differences in memory access paradigms between local and distributed entities.

CS451: Distributed Systems (Spring 2007)

More...

•Major differences between local and distributed computing:

- Latency: 4-5 orders of magnitude (?) – getting worse?
 - Can't we just rely on faster hw?
 - Tooling can tell us perf issues
- Memory access: the use of pointers
 - Distributed shared memory?
- Partial failure: one component can fail while others continue
 - No global state to allow determination of exactly went wrong!
 - Programs must deal with indeterminacy (at the interface level)
- Concurrency : truly asynchronous operation invocations

•Just part of the QoS?

•So, what about Java RMI? Consistent or inconsistent with this paper?

CS451: Distributed Systems (Spring 2007)

Security – basics

•Security measures must be incorporated into distributed systems.

•Purpose:

- Prevent malicious or mischievous attacks.
- Protect integrity and privacy of information and other resources.

•Means:

- Security policies.
- Security mechanisms.

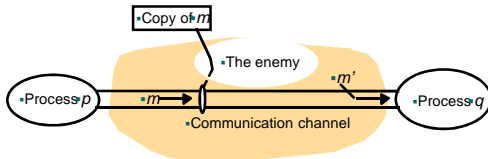
CS451: Distributed Systems (Spring 2007)

Fundamental Models: Security Model

- Chief concern: preventing unauthorized access

- Enemy:

- can send messages to any process
- can read/write messages between processes



CS451: Distributed Systems (Spring 2007)

Fundamental Models: Threats

- Threats to processes

- How can server ensure the identity of the client?
- How can the client know that the server has generated the response?

- Threats to communication channels

- How can the processes ensure that a "man in the middle" cannot understand/replay/delete/alter the messages?

- Denial of service (DOS)

- Approach: crypto (more later in the semester)—doesn't not immediately aid DOS attacks

CS451: Distributed Systems (Spring 2007)

Policy and Mechanism

- Policy

- Determines what should be done
- Determines who should have access to what and when

- Mechanisms

- Implement policies
- Encryption
- Authentication
- Authorisation
- Audit

CS451: Distributed Systems (Spring 2007)

Security - security model

- Processes encapsulate resources and allow clients to access them.

- Principals can be explicitly authorized to operate on resources.

- Resources must be protected against unauthorized access.

- Processes interact through a network that is shared by many users. Enemies can access the network by, e.g.:

- Copying/reading messages.
- Inject arbitrary messages.

CS451: Distributed Systems (Spring 2007)

Designing secure systems – general issues

- Hard to predict all possible attacks and loopholes.

- Balance cost and inconvenience against threats.

- Worst-case assumptions and guidelines:

- Interfaces are exposed.
- Networks are insecure.
- Limit the lifetime and scope of each secret.
- Algorithms and program code are available to attackers.
- Attackers may have access to large resources.
- Minimize the trusted base.

CS451: Distributed Systems (Spring 2007)

Establishing Trust

- "assured reliance on the character, ability, strength, or truth of someone or something"

- Who do you trust in your research life?

- People, software, machines, services

- How do you develop trust in each of these?

- People: personal relationship, "told" to trust them
- Software: Know authors personally, see MD5, sandboxing techniques (Java, .NET),
- Machines: PKI, [Open]SSH
- Services: PKI, [Open]SSH

CS451: Distributed Systems (Spring 2007)

Definition of Computer Security

- **Confidentiality** – assurance that only authorized entities have access to the information
- **Integrity** – assurance that the data has not been altered without proper authorization
- **Availability** – assurance that the information is available when required by authorized users
- **Authentication** – insuring that the identity of an entity is as claimed to be
- **Non-repudiation** – insuring that an entity can not deny that it has taken an action