X36DSV – 2. seminar

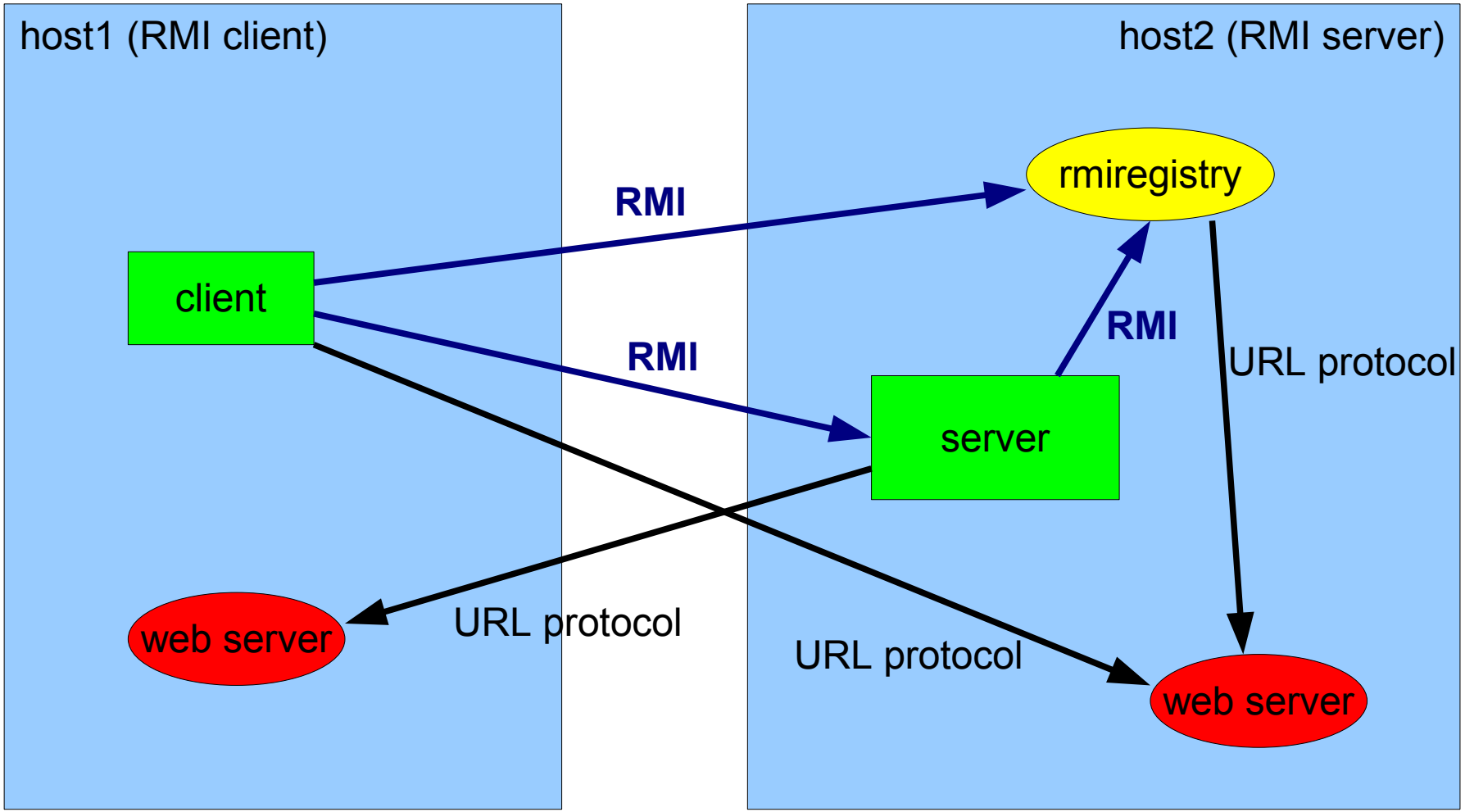
RMI

Remote Method Invocation

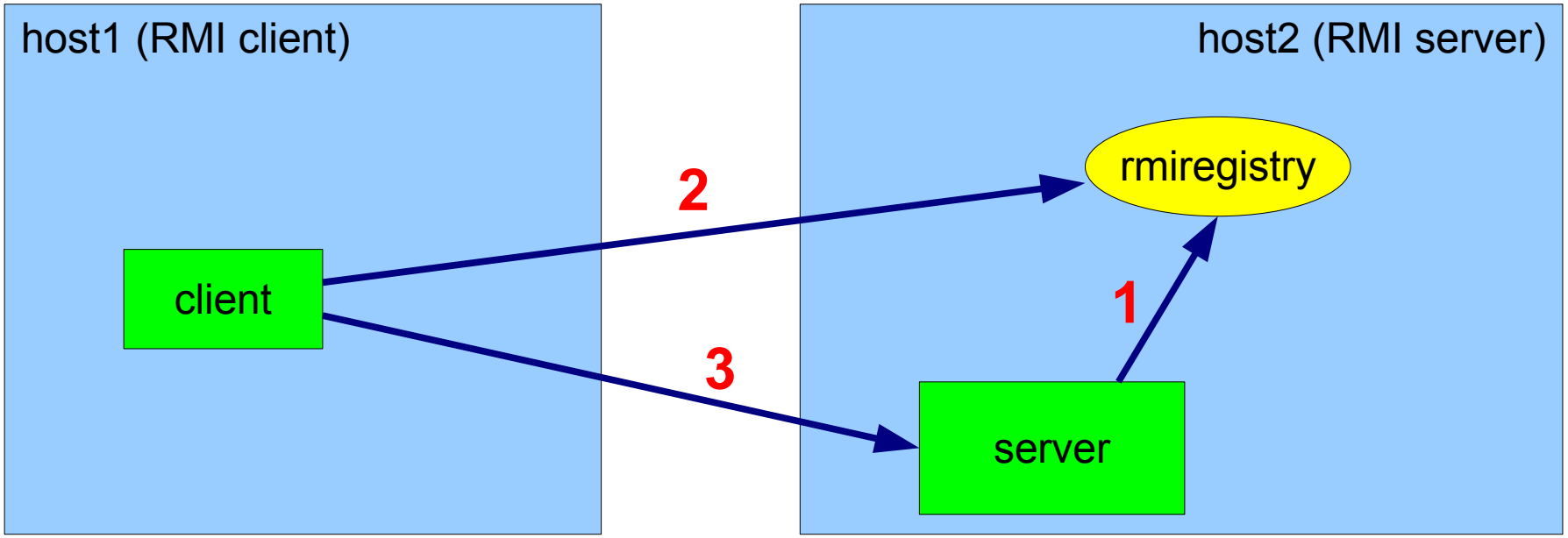
RMI – what it is?

- method invocation from other JVM
  - localization of remote object
  - communicate with remote objects
  - transfer object between JVM in bytecode
- typically client – server
- http://java.sun.com/docs/books/tutorial/rmi/index.html

RMI application



RMI application



1 – object registration (binding name with object) - `Registry.rebind`
2 – lookup for object with known name - `Registry.lookup`
3 – calling remote object methods - `Remote_object.remote_method`

Parts of Java RMI application

- remote interface
  - declaration of remote methods used by clients
- remote object
  - remote interface implementation
- client
- comunication
  - stub, skeleton (Java 2 don't have skeleton)

Remote

- remote methods
  - methods called between JVM
- remote objects
  - objects with remote methods
- remote interface
  - specifies remote object, extends interface *java.rmi.Remote*
  - every remote method must throw *java.rmi.RemoteException*
- remote stub
  - substitute remote object with local representation (proxy)
  - same methods like remote interface, [un|]marshaling

Application checklist

- remote interface
- implementation of remote objects
- client implementation
- compilation of source code
- stub creation
- distribution of application
- start using application

Implementation

- declaration remote interface
- constructor definition
- remote method implementation
- creation and installation of remote objects
  - main method
  - creation and installation of security manager
  - create instance of remote object
  - object registration in name service
    - rmiregistry, JNDI

RMI - Interface

```
package compute;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface MathServer extends Remote {
    public int secti(int a, int b) throws RemoteException;
}
```

RMI – Remote object

```
package server;

import java.rmi.*;
import compute.*;

public class MathServerImpl implements MathServer {

    // pro potrebu serializace (marshalingu)
    private static final long serialVersionUID = -386L;

    public MathServerImpl() throws RemoteException {
        super();
        }

    public int secti(int a, int b) throws RemoteException {
        int result;
        result=a+b;
        System.out.println(a + " + " + b + " = " + result);
        return result;
    }
}
```

RMI - Server

```
package server;

import java.rmi.*;
import compute.*;

public class Server {
    public static void main(String[] args) {
        if (System.getSecurityManager() == null)
            System.setSecurityManager(new RMISecurityManager());
            String name = "MathD";
        try {
            MathServer msi = new MathServerImpl();
            MathServer stub =
                  (MathServer) UnicastRemoteObject.exportObject(msi);
            Registry registry = LocateRegistry.createRegistry(2010);
            registry.rebind(name, stub);
        }
        catch (Exception e) {
            System.err.println("Data exception: " + e.getMessage());
        }
    }
}
```

RMI - Client

```
package client;

import java.rmi.*;
import compute.*;

public class Client {
    public static void main(String args[]) {
        if (System.getSecurityManager() == null)
            System.setSecurityManager(new RMISecurityManager());
        try {
            MathServer mth;
            String name = "MathD";
            Registry registry = LocateRegistry.getRegistry("localhost",
2010);
            mth = (MathServer) registry.lookup(name);
            int a = Integer.valueOf(args[0]).intValue();
            int b = Integer.valueOf(args[1]).intValue();
            System.out.println(a + " + " + b + " = " + mth.secti(a, b));
        }
        catch (Exception e) {
            System.err.println("Data exception: " + e.getMessage());
        }
    }
}
```

Security

- default methods
- policytool.exe || policytool
- java.security.policy

```
grant {
  permission java.net.SocketPermission "*:1024-65535","connect,accept";
  permission java.io.FilePermission "\data\-","read";
  permission java.io.FilePermission "c:\\home\\ann\\classes\\-","read";
};
```

```
grant {
  permission java.security.AllPermission;
};
```

Compilation and running

```
# source compilation (Win i Lin)
javac compute/MathServer.java
javac server/MathServerImpl.java server/Server.java
javac client/Client.java

# if you don't start rmiregistri within server code then ...
start rmiregistry
rmiregistry &

# server start (in Lin ';' -> ':')
java -cp compute.jar;server.jar
    -Djava.security.policy=java.policy server.Server

# client start (in Lin ';' -> ':')
java -cp compute.jar;client.jar
    -Djava.security.policy=java.policy client.Client 64 46
```