Damascus University
Faculty of Information Technology Engineering
Department of Computer Systems and Networks

جامعة دمشق
كلية الهندسة المعلوماتية
قسم النظم والشبكات الحاسوبيّة

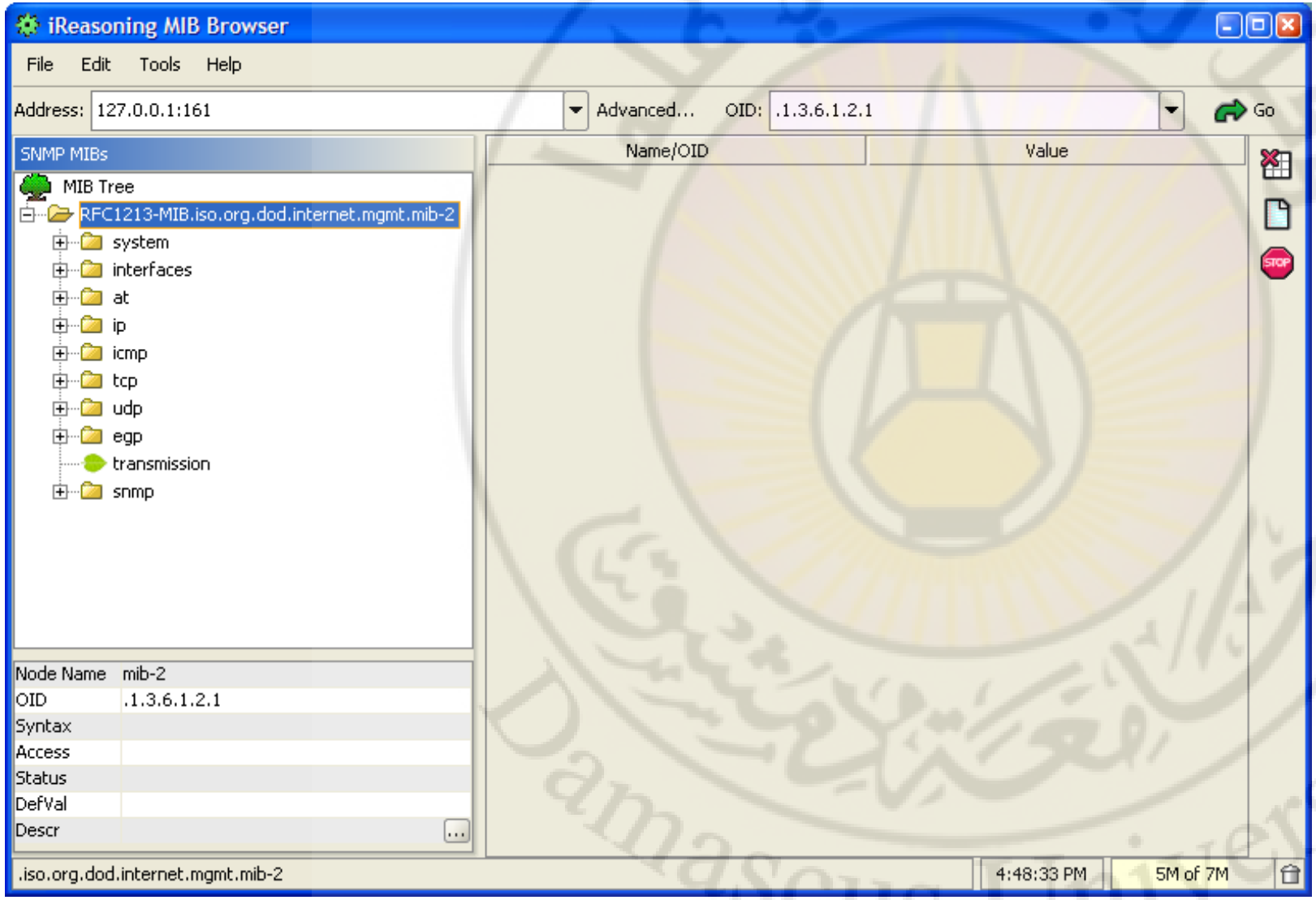
عملي إدارة الشبكات
Network Management

iREASONING NETWORKS

Appendix #2: SNMP Tools - iReasoning's MIB Browser

اعداد: م. غاندي هسام

SNMP – MIB Browser (1)



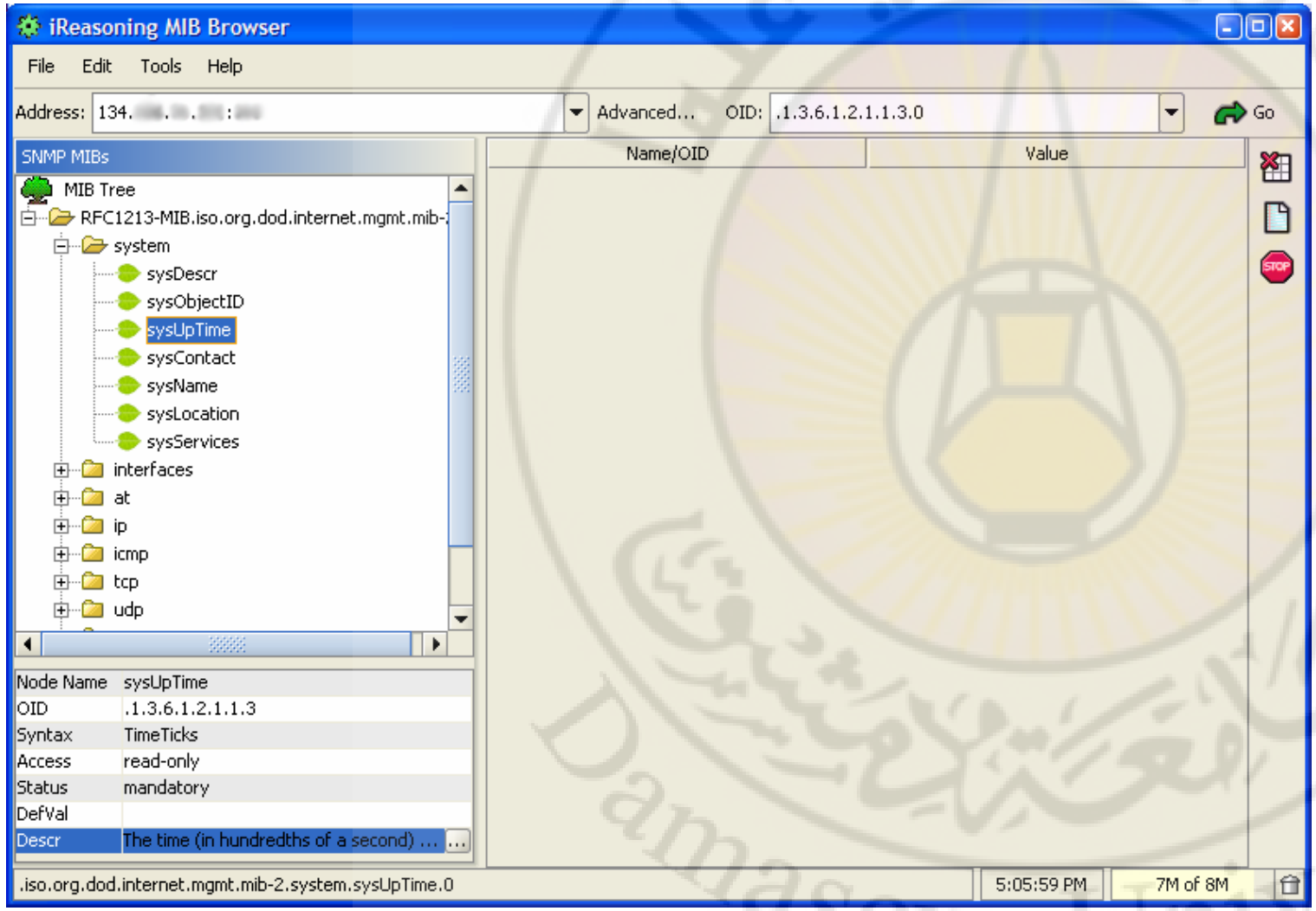
Breakdown…

- LHS is the SNMP MIB structure.

- Lower LHS has details of MIB structure.

- RHS will present MIB values.

SNMP – MIB Browser (3)



Navigation…

- MIB Tree

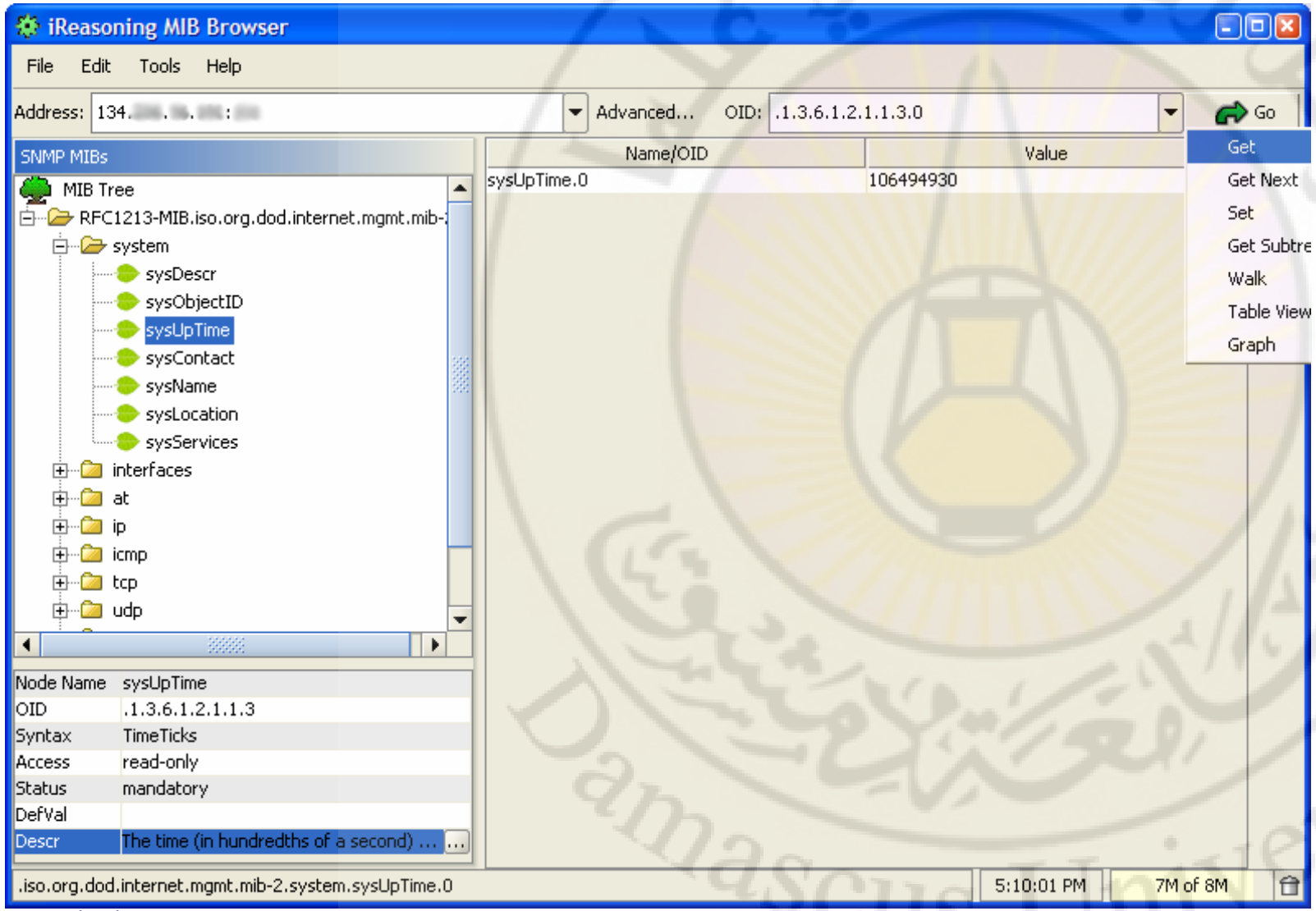
→System

→ sysUpTime

-Notice Lower LHS

- Notice OID

SNMP – MIB Browser (4)



SNMP PDU's…

(1) Get

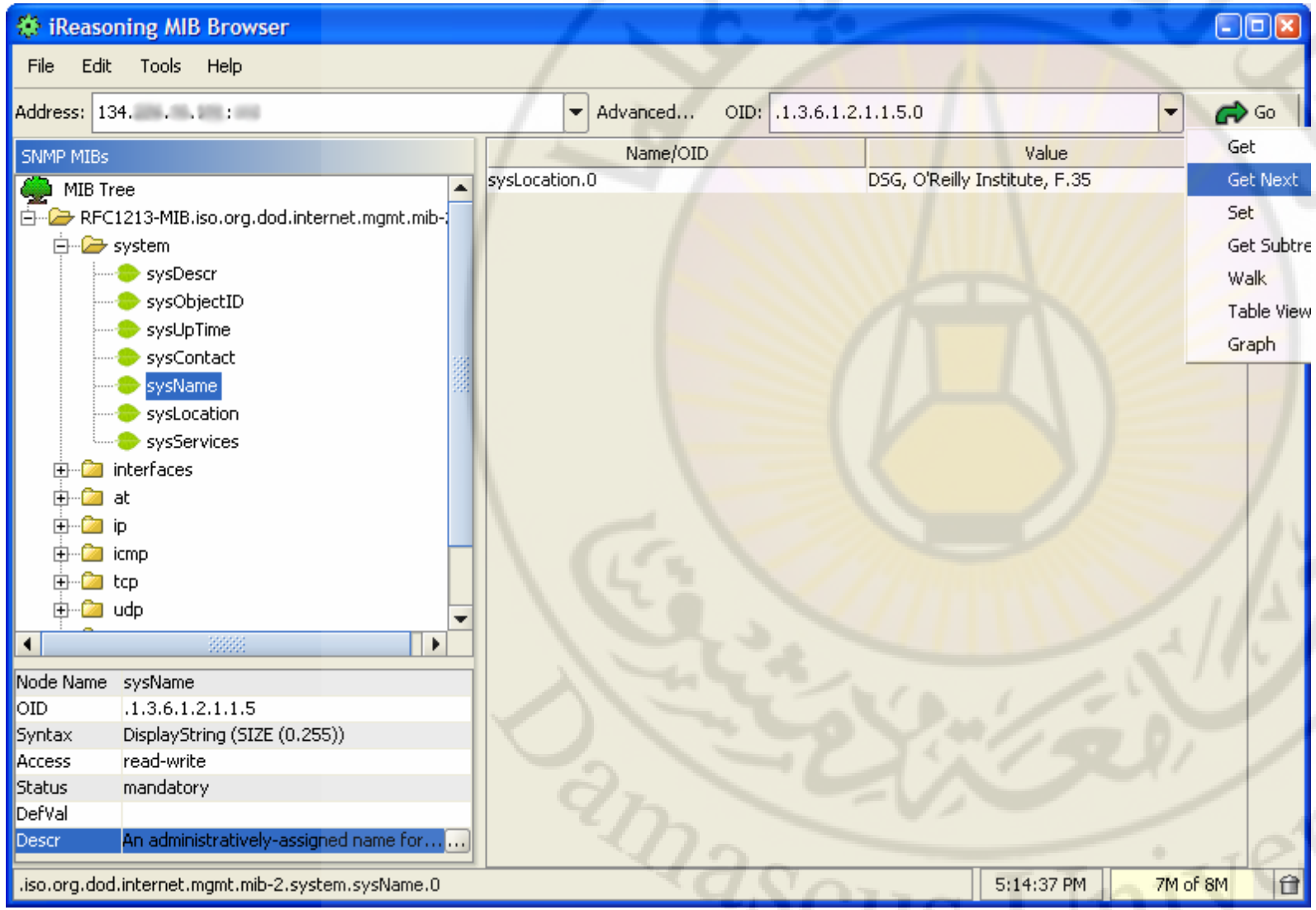
- Select 'Go'

→'Get'

- RHS has values.

- OID – Value

SNMP – MIB Browser (5)



SNMP PDU's…

(2) GetNext

-Selected OID is:

.1.3.6.1.2.1.1.5

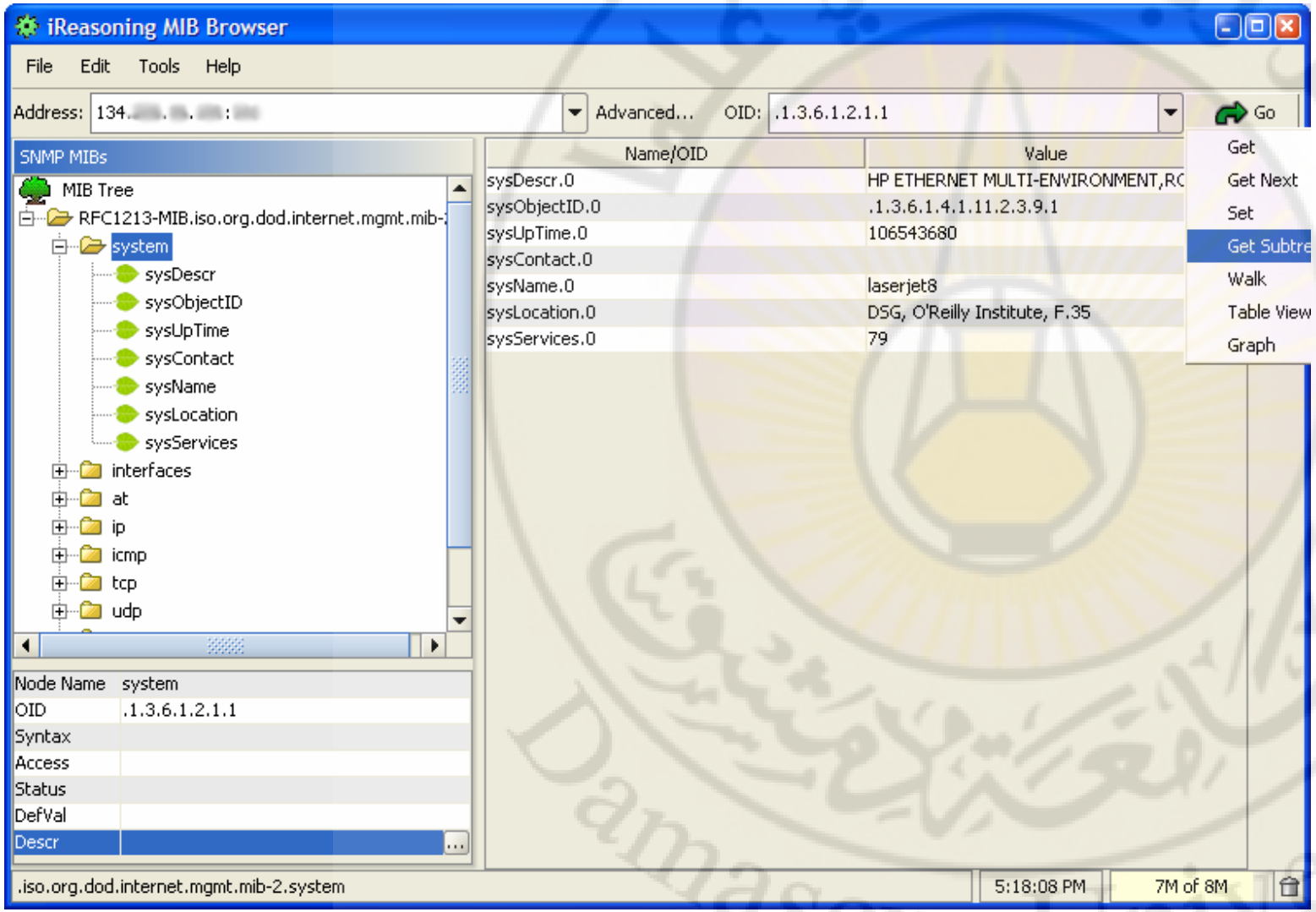
-Returned value:

(.1.3.6.1.2.1.1.6)

or

"DSG, O'Reilly Institute, F.35"

SNMP – MIB Browser (6)



SNMP…

(3) Get SubTree

-Position of MIB:

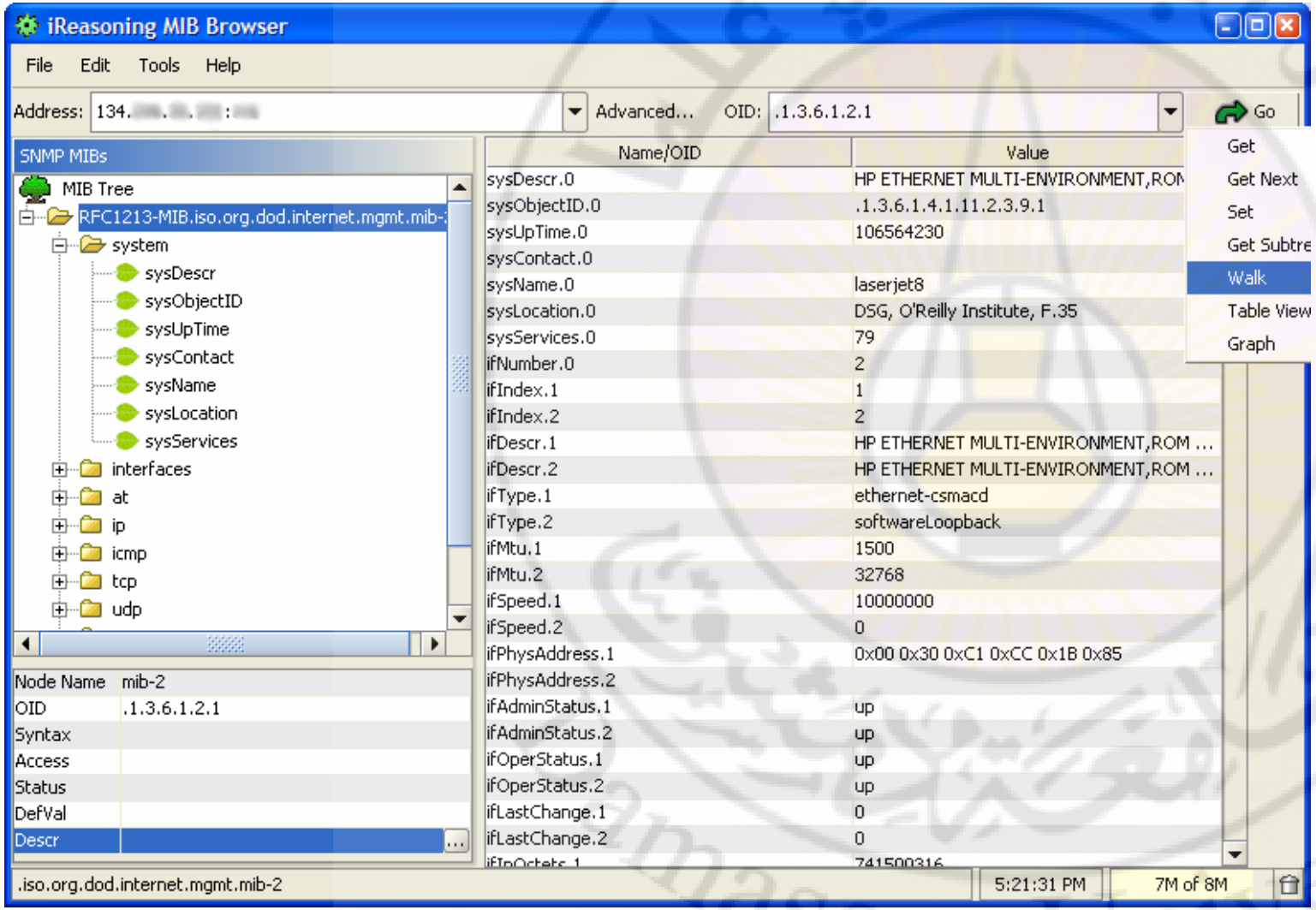
.1.3.6.1.2.1.1

(a.k.a. system)

-RHS values:

Returns all values below system.

SNMP – MIB Browser (7)



SNMP…

(4) Walk

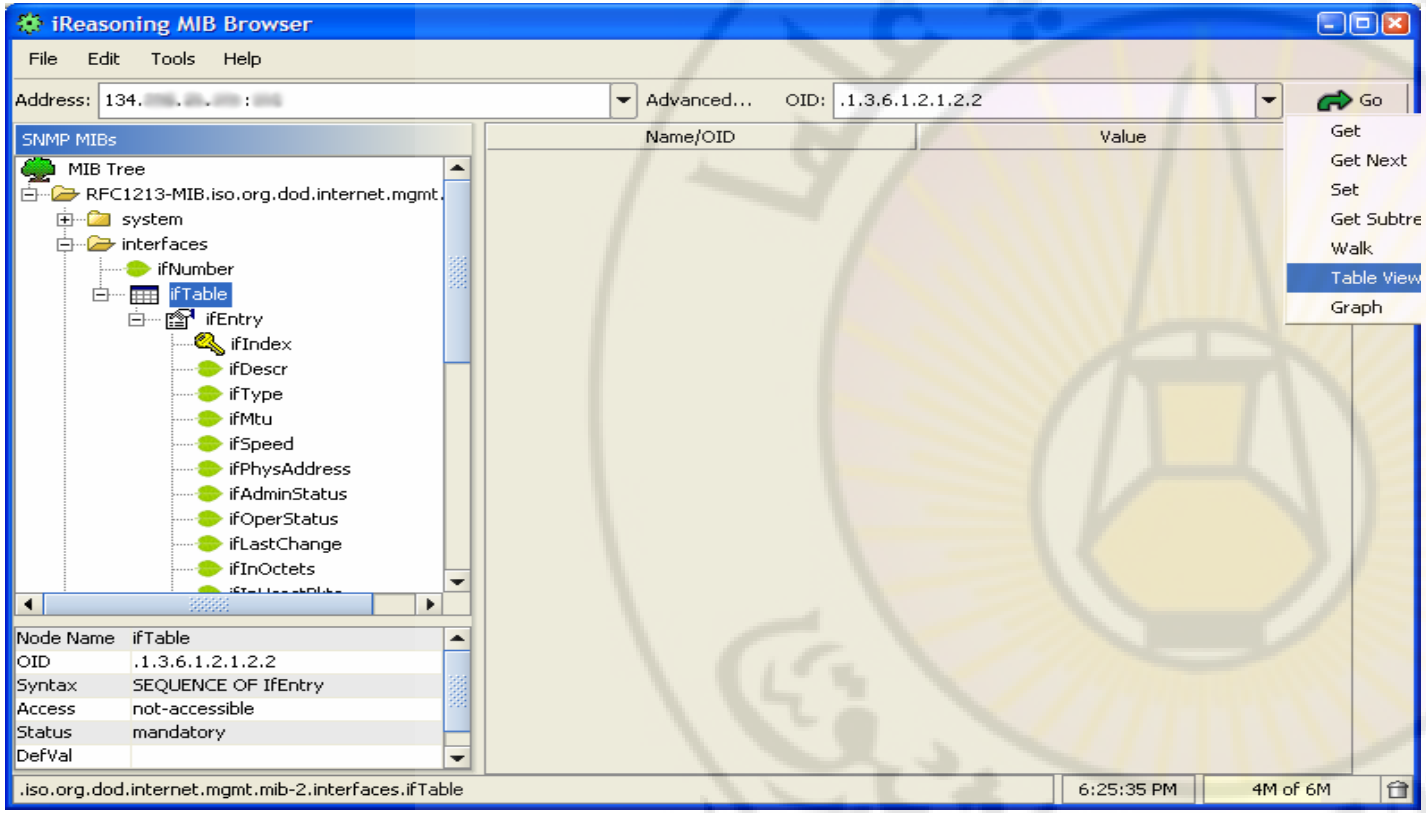
-MIB Location:

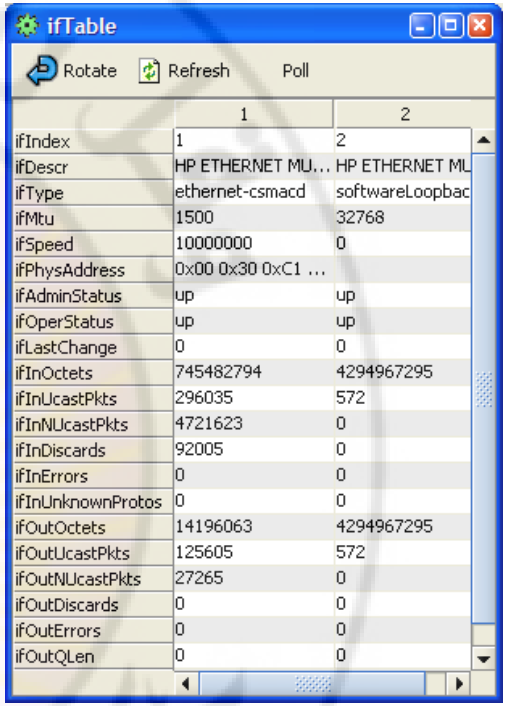
.1.3.6.1.2.1

(a.k.a. mib-2)

- Returns *ALL* values under mib-2

SNMP – MIB Browser (8)





Tables…

- MIB Location:

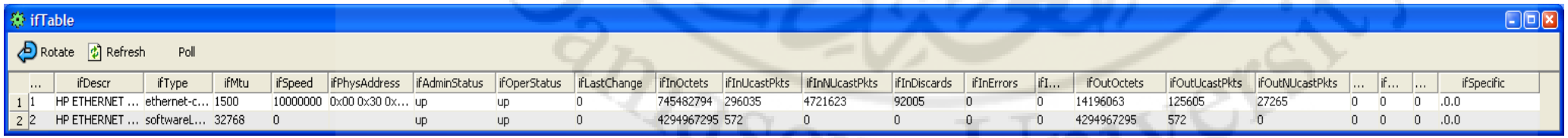
.1.3.6.1.2.1.2.2

(or interfaces)

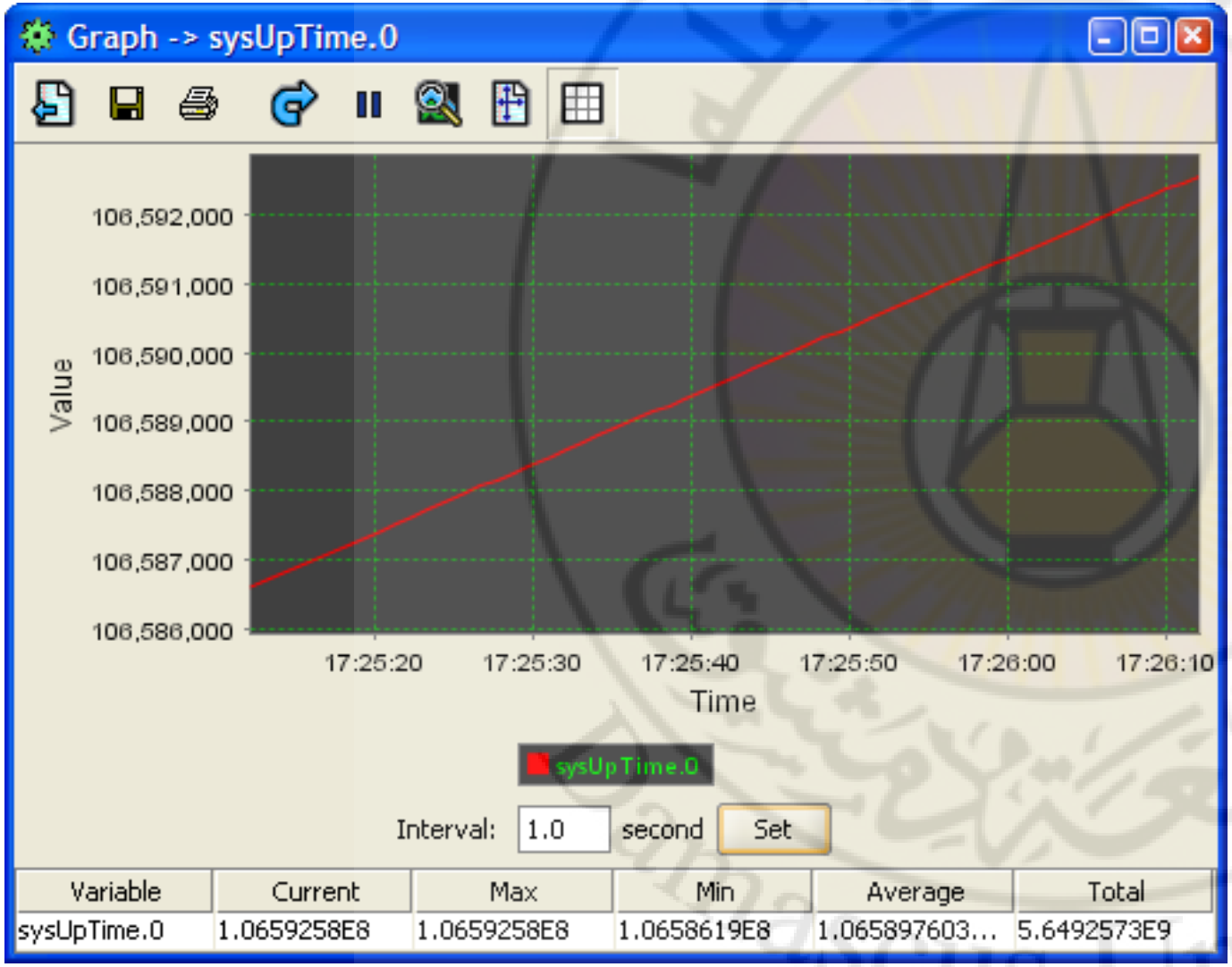
- Select ifTable,

→ Go, then Table View.

- Refresh/Poll



SNMP – MIB Browser (9)



SNMP…

- Graph

- Select a value from the RHS, say sysUpTime

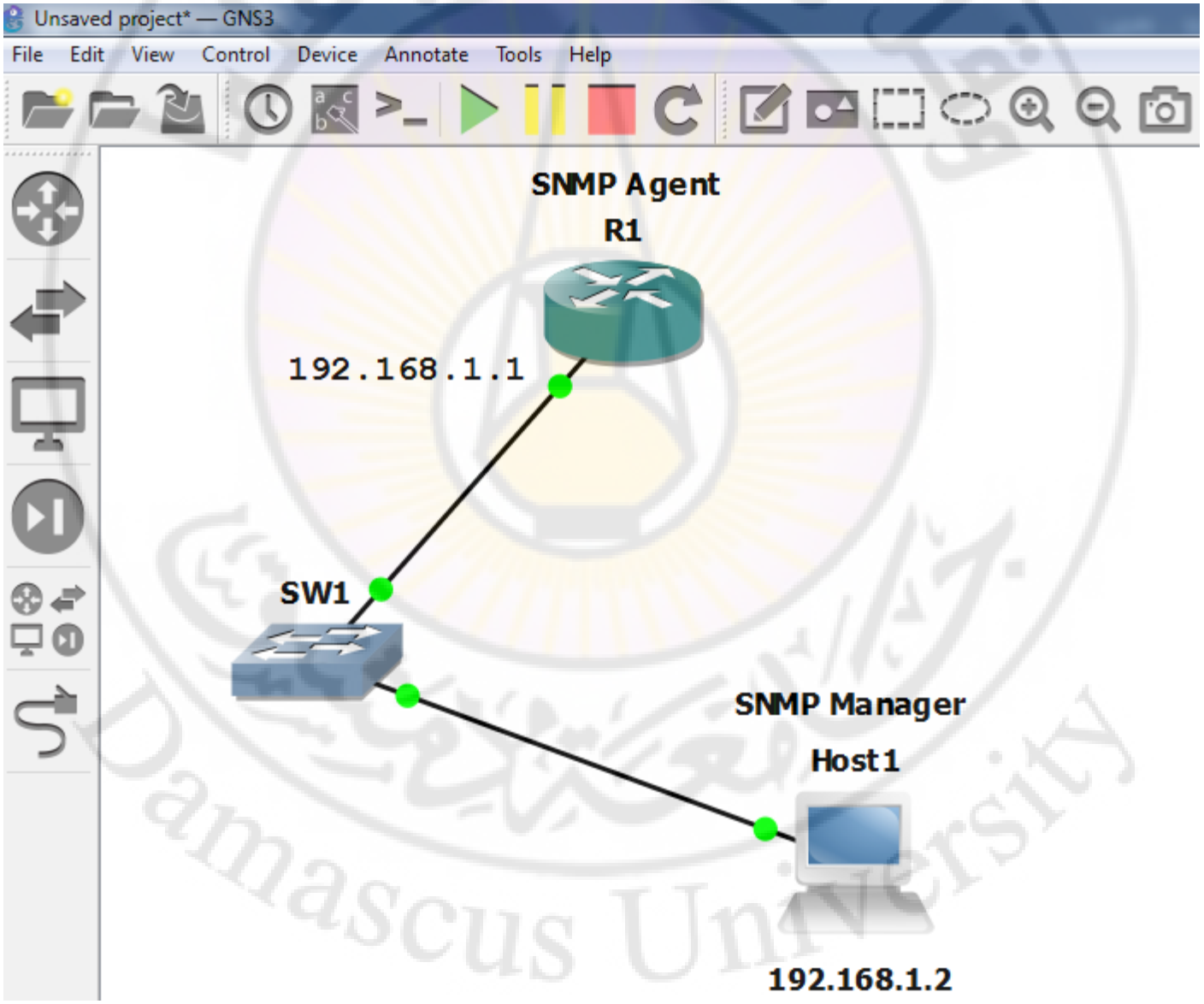
- Highlight and select ‘Go’, then ‘Graph’.

- Interval = 1s → set.

End of Appendix #2

Using GNS3 and A simple applicating with SNMP4J in JaVA
Appendix #3

- Topology



By Eng.Ghandy Hessam

Class: snmp_get

```
import java.io.IOException;
import org.snmp4j.*;
import org.snmp4j.event.ResponseEvent;
import org.snmp4j.mp.SnmpConstants;
import org.snmp4j.smi.*;
import org.snmp4j.transport.DefaultUdpTransportMapping;

/**
 *
 * @author Ghandy Hessam
 */
public class snmp_get {

    public static void main(String[] args) throws IOException {
        // TODO code application logic here
        OID oid = new OID("1.3.6.1.2.1.1.5.0"); // sysName
        PDU pdu = new PDU();
        pdu.add(new VariableBinding(oid));
        pdu.setType(PDU.GET);
        CommunityTarget target = new CommunityTarget();
        target.setCommunity(new OctetString("public"));
       // Address targetaddress = ;
        target.setAddress(new UdpAddress("192.168.1.1/161"));
        target.setVersion(SnmpConstants.version1);
        Snmp snmp = new Snmp(new DefaultUdpTransportMapping());
        snmp.listen();

        ResponseEvent response = snmp.send(pdu, target);
        if (response.getResponse() == null) {
            // request timed out
            System.out.println("Faield!");
        } else {
            //System.out.println("Received response from: " +
response.getPeerAddress());

            // dump response PDU
            System.out.println(response.getResponse().get(0).toString());
        }


    }
}
```

Class: snmp_set

```
import java.io.IOException;
import org.snmp4j.*;
import org.snmp4j.TransportMapping;
import org.snmp4j.event.ResponseEvent;
import org.snmp4j.event.ResponseListener;
import org.snmp4j.mp.SnmpConstants;
import org.snmp4j.smi.Address;
import org.snmp4j.smi.GenericAddress;
import org.snmp4j.smi.Integer32;
import org.snmp4j.smi.OID;
import org.snmp4j.smi.OctetString;
import org.snmp4j.smi.VariableBinding;
import org.snmp4j.transport.DefaultUdpTransportMapping;
/**
 *
 * @author Ghandy Hessam
 */
public class snmp_set {

    public static void main(String[] args) throws IOException {
        String strAddress = "192.168.1.1/161";
        Address targetAddress = GenericAddress.parse(strAddress);

        try {
            TransportMapping transport = new DefaultUdpTransportMapping();
            Snmp snmp = new Snmp(transport);
            transport.listen();
            CommunityTarget target = new CommunityTarget();
            target.setCommunity(new OctetString("public"));
            target.setAddress(targetAddress);
            target.setRetries(2);
            target.setTimeout(5000);
            target.setVersion(SnmpConstants.version1);
            PDU pdu = new PDU();
            pdu.add(new VariableBinding(new OID("1.3.6.1.2.1.1.5.0"), new
OctetString("RRR1")));
            pdu.setType(PDU.SET);
            ResponseListener listener = new ResponseListener() {

                public void onResponse(ResponseEvent event) {
                    ((Snmp) event.getSource()).cancel(event.getRequest(),
this);
                    System.out.println("Set Status is: " +
event.getResponse().getErrorStatusText());
                }
```

```
            };



            snmp.send(pdu, target, null, listener);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

End of Appendix #3

Damascus University
Faculty of Information Technology Engineering
Department of Computer Systems and Networks

جامعة دمشق
كلية الهندسة المعلوماتية
قسم النظم والشبكات الحاسوبيّة

عملي إدارة الشبكات
Network Management

Session 3: How to use the library snmp4j in JAVA

اعداد: م. غاندي هسام

Package org.snmp4j Description

- Provides classes and interfaces for creating, sending, and receiving SNMP v1/v2c/v3 messages.
- A SNMP message is composed of its message header and its PDU payload.
- Contains three main groups of classes and interfaces:
  - Classes for SNMP message and target creation
  - Classes for SNMP message sending (command generation)
  - Classes for SNMP message dispatching (command responding)

SNMP Messages and Targets

- To exchange a SNMP message with a remote system, that system has to be:
  - Identified
  - Retransmission
  - timeout policy information
- A remote system is specified with SNMP4J by creating a target instance appropriate for the SNMP protocol to be used.
  - For SNMPv1 and SNMPv2c the *community target* has to be used.
  - For SNMPv3 the *User target* has to be used instead.

- A SNMP message consists of:
  - message's payload
  - the SNMP Protocol Data Unit (PDU)
  - message header.
- In SNMP4J the message header information is represented by Target and the PDU is represented by one of the following classes:
  - PDUv1 (snmp v1)
  - PDU (snmp v2)
  - ScopedPDU (snmp v3)

To send a SNMP message with SNMP4J, a PDU instance and a Target instance have to be created.

PDU Examples

❑ SNMPv1/v2c GET PDU:

```
OID oid = new OID("1.3.6.1.2.1.1.1"); // sysDescr
PDU pdu = new PDU();
pdu.add(new VariableBinding(oid));
pdu.setType(PDU.GET);
```

❑ SNMPv3 GETBULK PDU:

```
ScopedPDU pdu = new ScopedPDU();
pdu.add(new VariableBinding(new OID("1.3.6.1.2.1.2.1"))); // ifNumber
pdu.add(new VariableBinding(new OID("1.3.6.1.2.1.2.2.1.10"))); // ifInOctets
pdu.add(new VariableBinding(new OID("1.3.6.1.2.1.2.2.1.16"))); // ifOutOctets
pdu.setType(PDU.GETBULK);
```

❖ Note:
ifNumber: The number of network interfaces
ifInOctets and ifOutOctets: counters.

Target Examples

❑ Community Target:

```
CommunityTarget target = new CommunityTarget();
target.setCommunity(new OctetString("public"));
target.setAddress(targetAddress);
target.setVersion(SnmpConstants.version1);
```

❑ User Target:

```
UserTarget target = new UserTarget();
target.setAddress(targetAddress);
target.setRetries(1);
// set timeout to 500 milliseconds -> 2*500ms = 1s total timeout
target.setTimeout(500);
target.setVersion(SnmpConstants.version3);
target.setSecurityLevel(SecurityLevel.AUTH_PRIV);
target.setSecurityName(new OctetString("MD5DES"));
```

Sending SNMP messages

- By using the *Snmp* class.
- To setup a *Snmp* instance it is sufficient to call its constructor with a *TransportMapping* instance.
- A SNMP4J *Snmp* instance supports SNMP v1, v2c, and v3 by default.
- SNMP messages can be sent synchronously (blocking) and asynchronously (non-blocking).

Example for Sending a Synchronous Message

```
Snmp snmp = new Snmp(new DefaultUdpTransportMapping());
snmp.listen();
...
ResponseEvent response = snmp.send(requestPDU, target);
if (response.getResponse() == null) {
   // request timed out
   ...
}
else {
   System.out.println("Received response from: "+ response.getPeerAddress());
   // dump response PDU
   System.out.println(response.getResponse().toString());
}
```

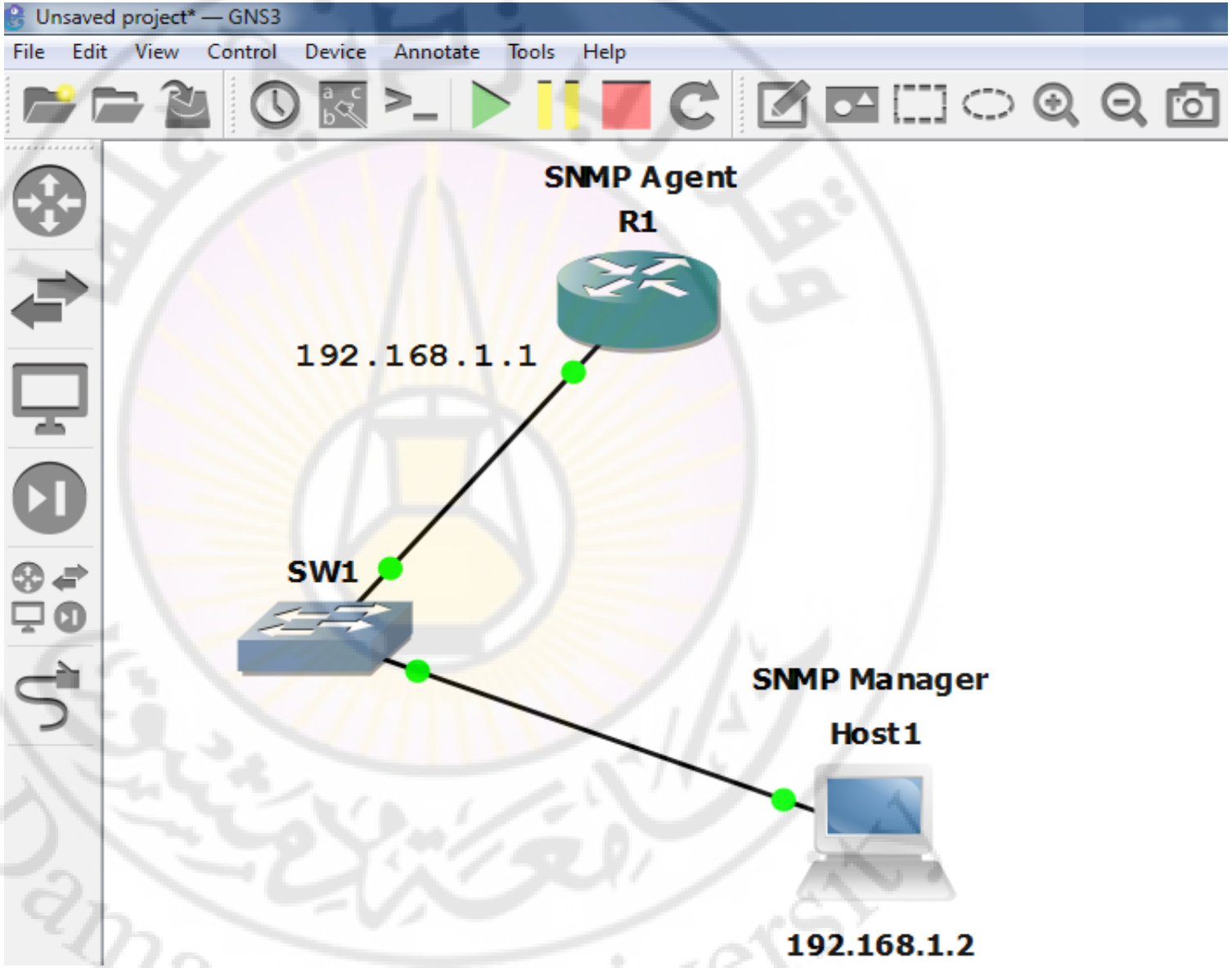
Example for Sending an Asynchronous Message

```
Snmp snmp = new Snmp(new DefaultUdpTransportMapping());
snmp.listen();
...
ResponseListener listener = new ResponseListener() {
   public void onResponse(ResponseEvent event) {
     ((Snmp)event.getSource()).cancel(event.getRequest(), this);
      PDU response = event.getResponse();
      PDU request = event.getRequest();
      if (response == null) {
         System.out.println("Request "+request+" timed out");
      }
      else {
         System.out.println("Received response "+response+" on request "+
                           request);
      }
   }
};
snmp.sendPDU(request, target, null, listener);
```

Using GNS3 and A simple applicating with SNMP4J in JaVA
Lab #2

Topology

- With GNS3 design the following network.



End of Session 3

Damascus University
Faculty of Information Technology Engineering
Department of Computer Systems and Networks

جامعة دمشق
كلية الهندسة المعلوماتية
قسم النظم والشبكات الحاسوبيّة

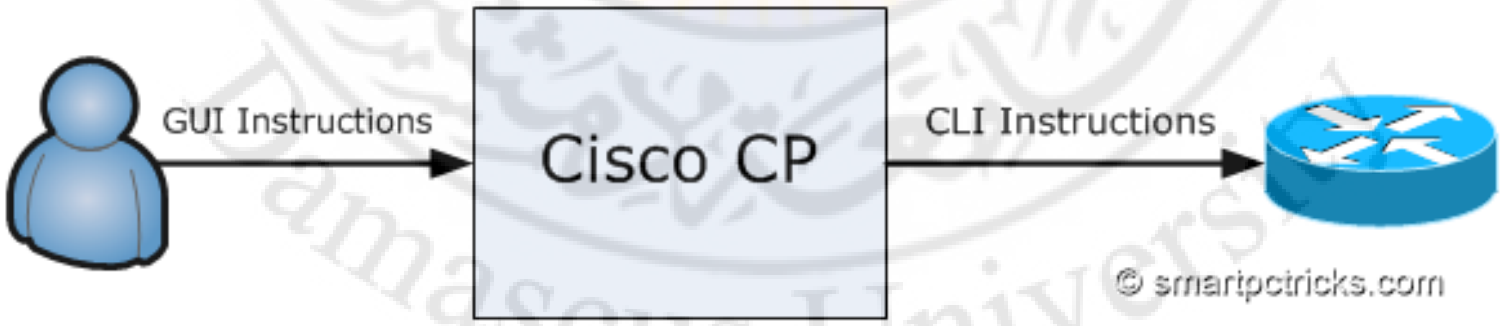
عملي إدارة الشبكات
Network Management

Session 4: Cisco Configuration Professional CCP

اعداد: م. غاندي هسام

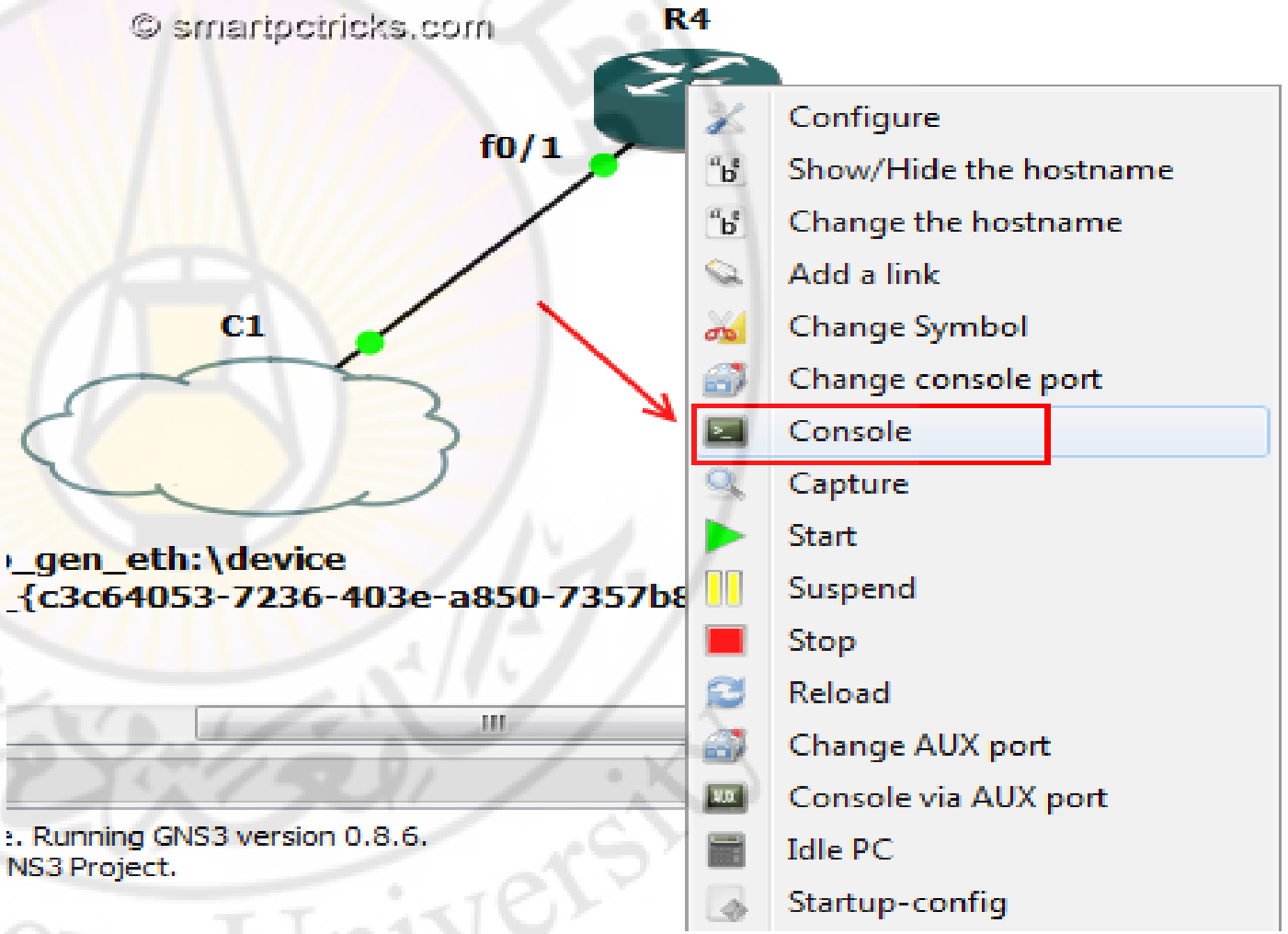
Introduction

- also called **CCP** or **Cisco CP**, it can be downloaded from Cisco website.
- Those who don't like CLI (Command Line Interface) may go for CCP, it's just awesome.
- CCP, the Cisco router configuration tool, delivers GUI interface to the administrator and converts the GUI instructions to configuration commands and updates the router.



Setup CCP for GNS3 to configure Cisco routers Lab#3

- Install **Microsoft LoopBack adapter** in Windows and assign an IP address of **20.0.0.2/8** (you may use your own).
- Build a topology in GNS3, Before getting in to the CCP configuration.

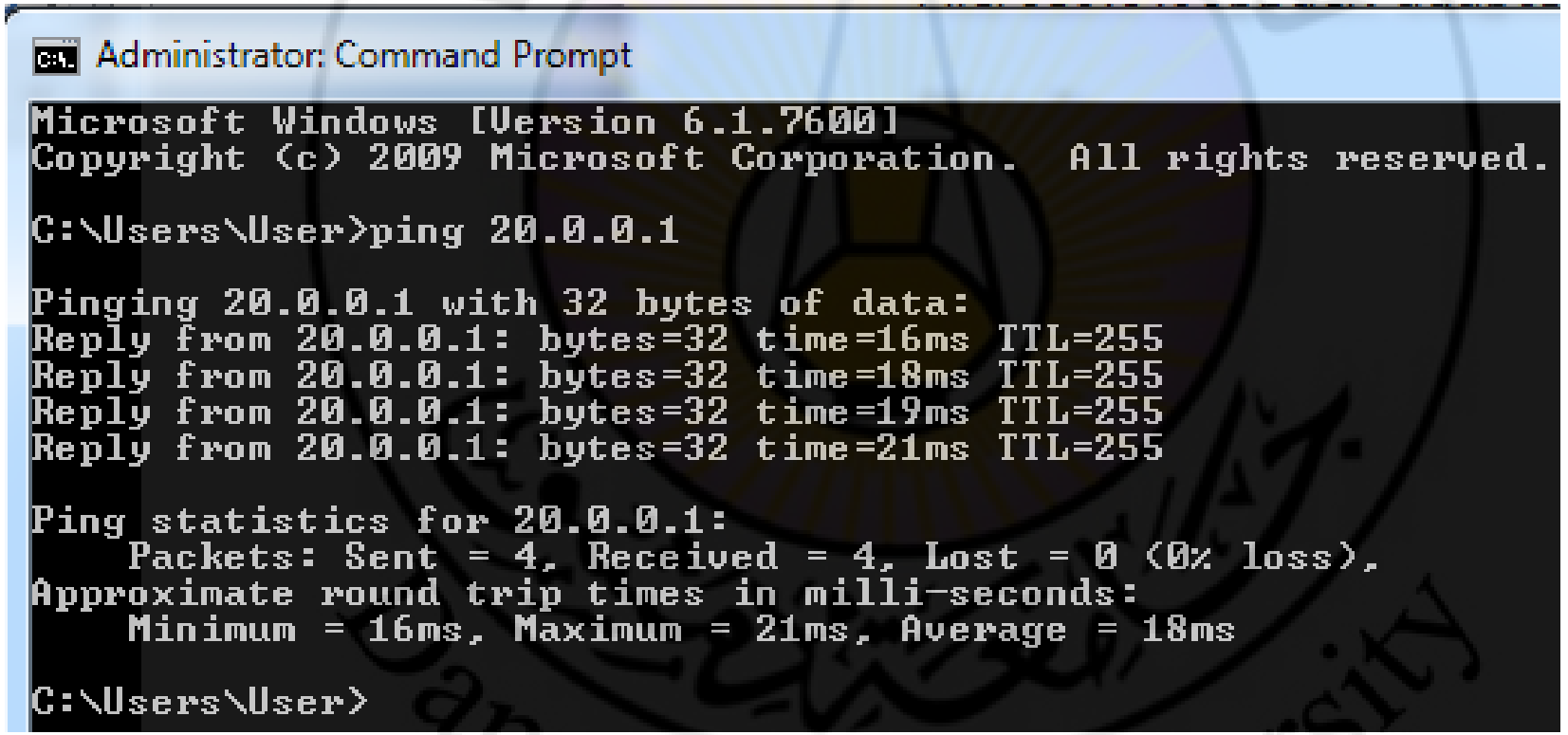


- Enter the following commands in the router via console.
- These commands let Cisco router management software to get access the Router.

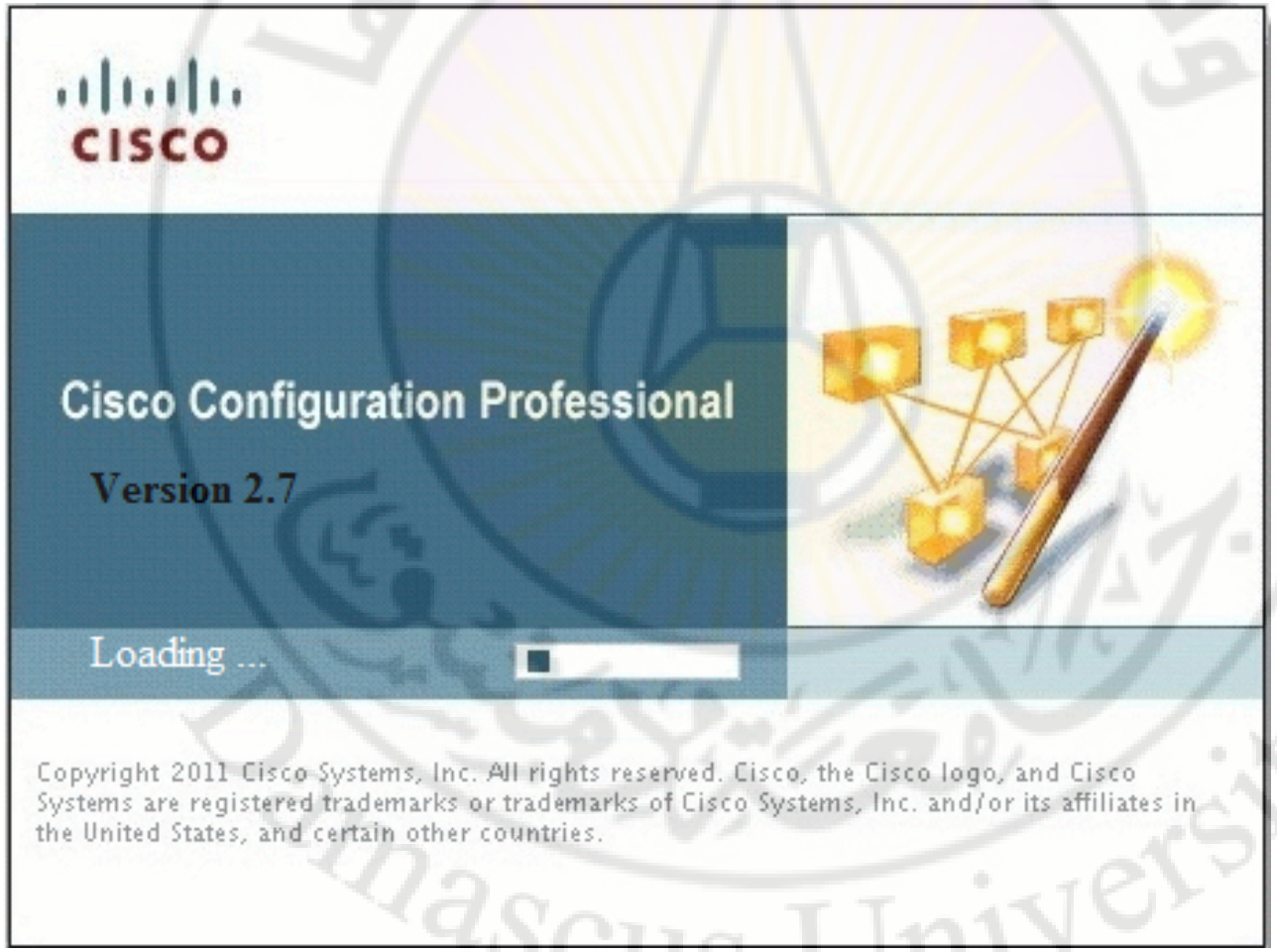
1. #ip http server
2. #ip http secure-server
3. #ip http authentication local
4. #username user privilege 15 password 123
5. Assign an IP to any interface (same range of loopback adapter)

```
R6#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R6(config)#ip http server
R6(config)#ip http secure-server
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

R6(config)#
*Sep 8 16:31:15.207: %SSH-5-ENABLED: SSH 1.99 has been enabled
R6(config)#
*Sep 8 16:31:15.867: %PKI-4-NOAUTOSAVE: Configuration was modified. Issue "write mem

R6(config)#ip http authentication local
R6(config)#username user privilege 15 password 123
R6(config)#interface fastEthernet 0/0
R6(config-if)#ip address 20.0.0.1 255.0.0.0
R6(config-if)#no shutdown
R6(config-if)#
*Sep 8 16:32:47.207: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Sep 8 16:32:48.207: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0
R6(config-if)#
```

- Try to ping from your PC to the router interface 20.0.0.1

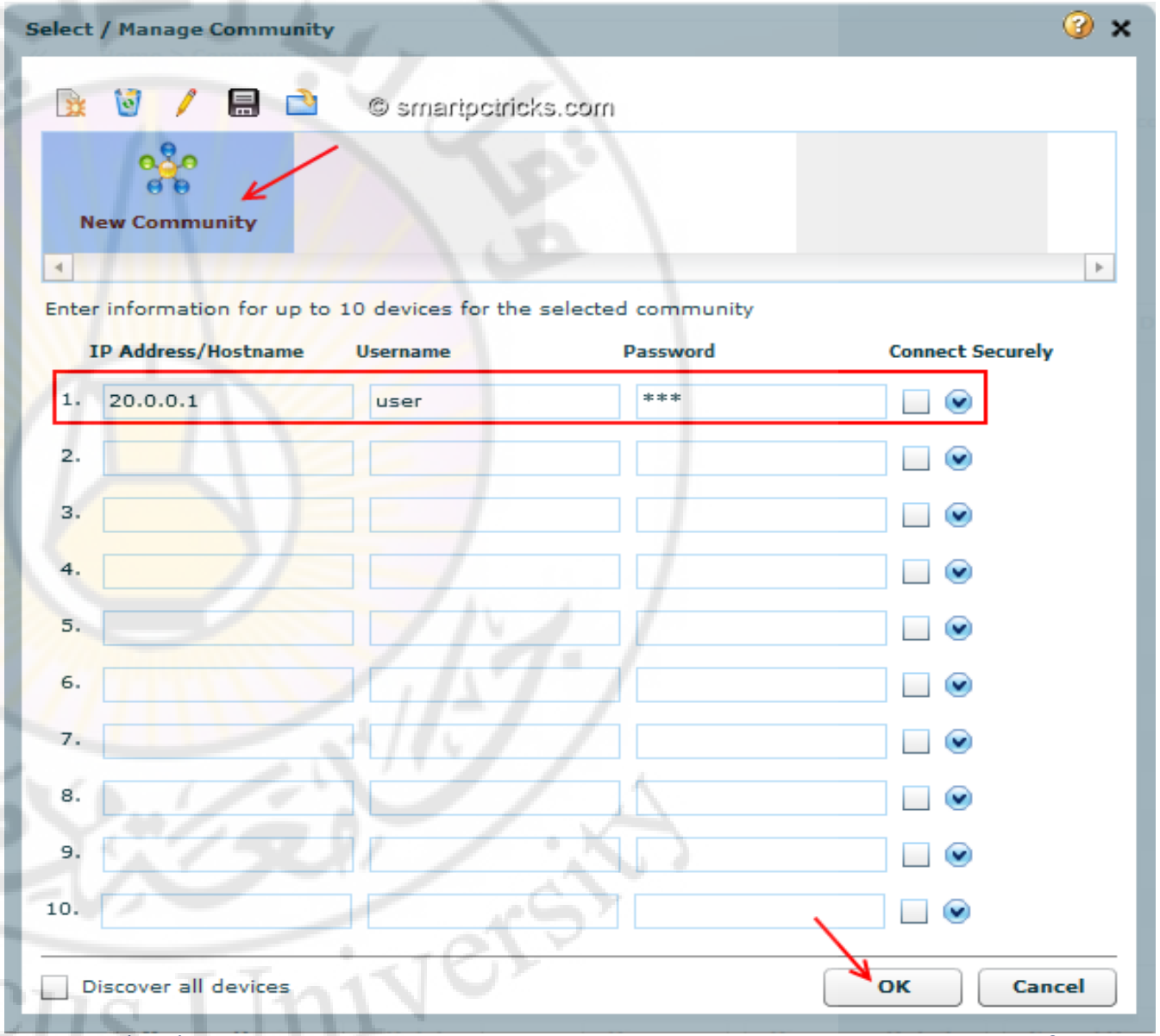


- Launch Cisco Configuration Professional CCP.

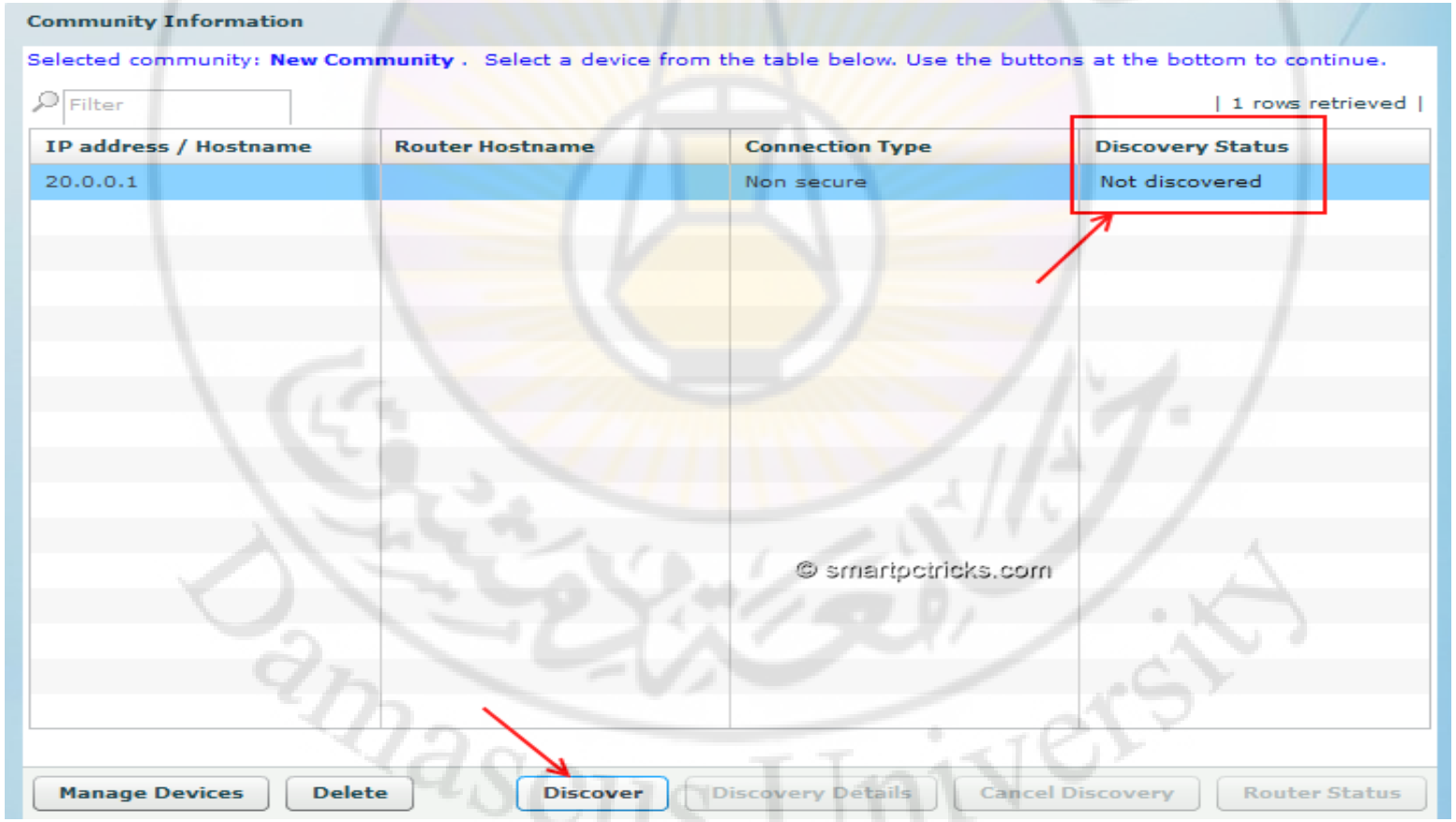


Build community in CCP

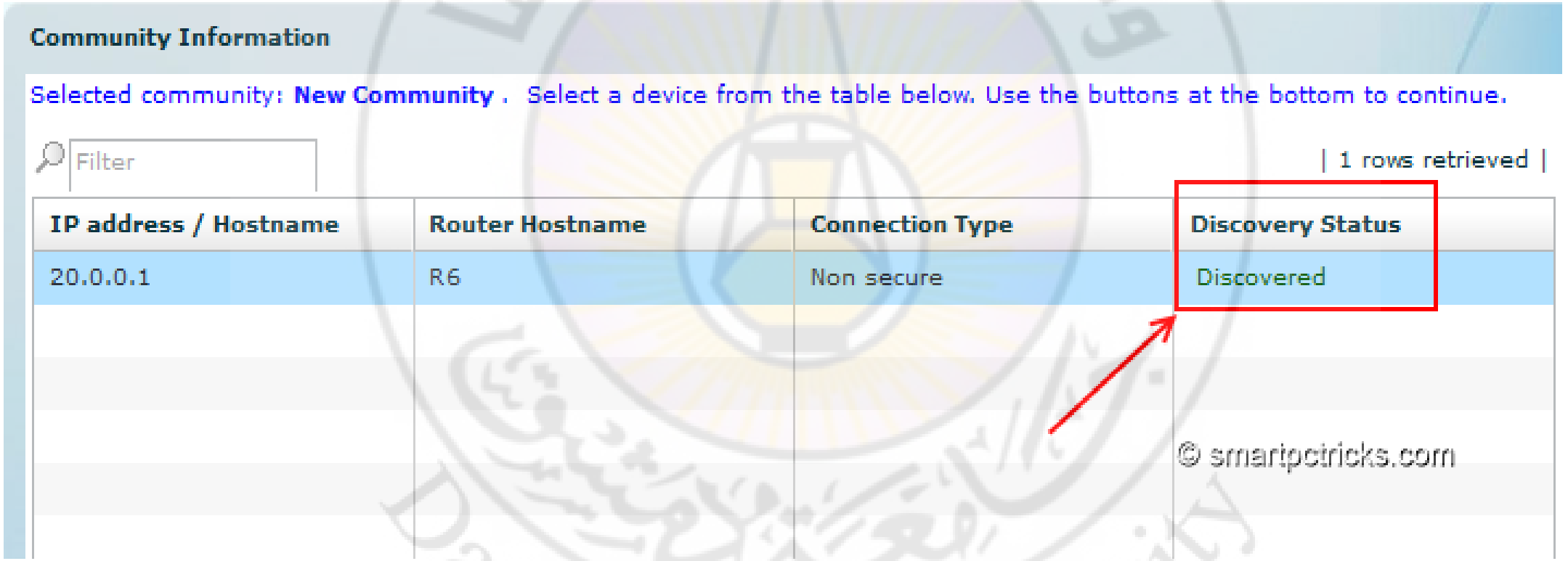
- Community window will open up. Enter the **IP address of the router** interface that you want to configure, with user name and password that you. Click OK.



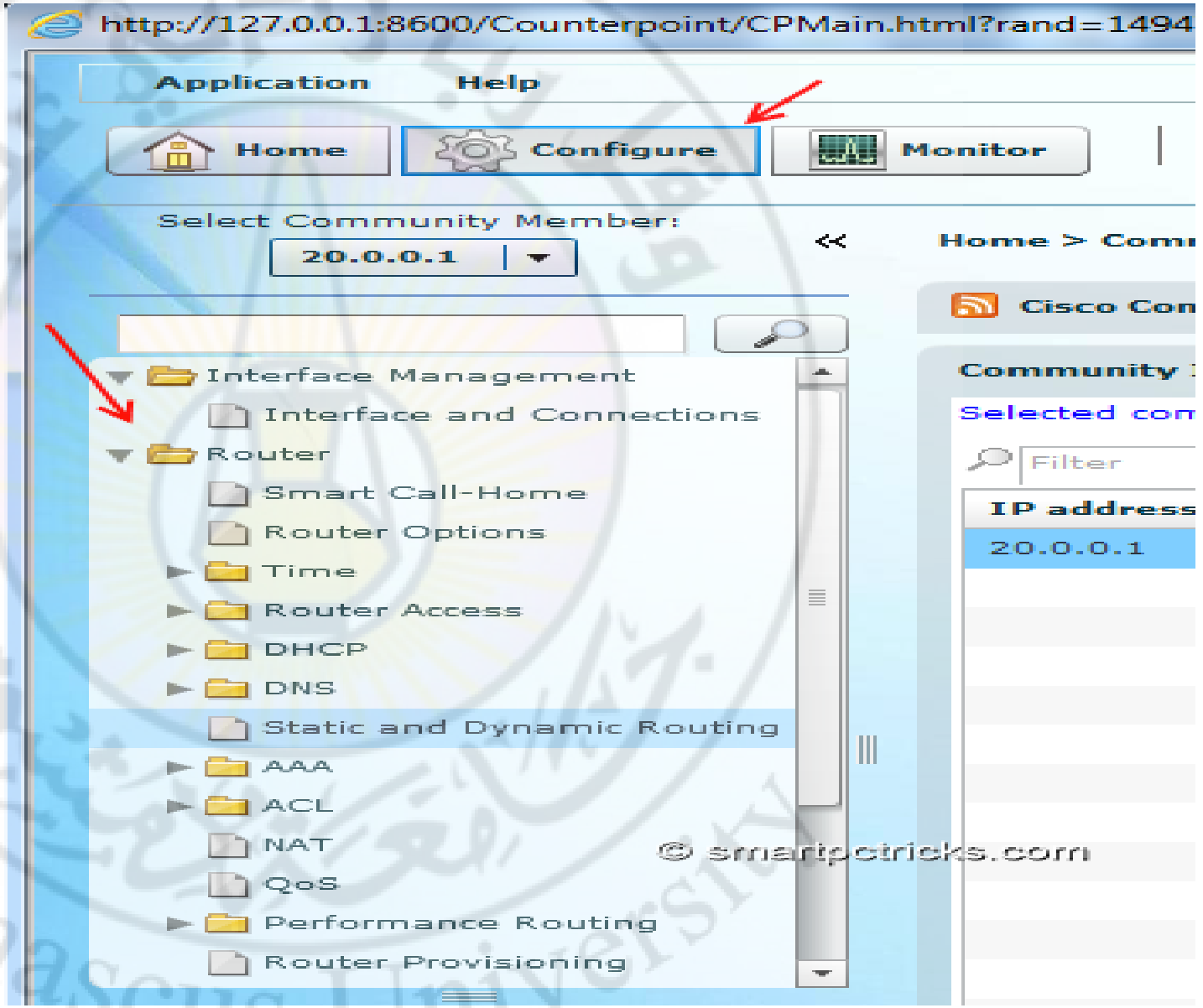
- Now the router will be added to the community with '**Not Discovered**' status. Click '**Discover**' button.



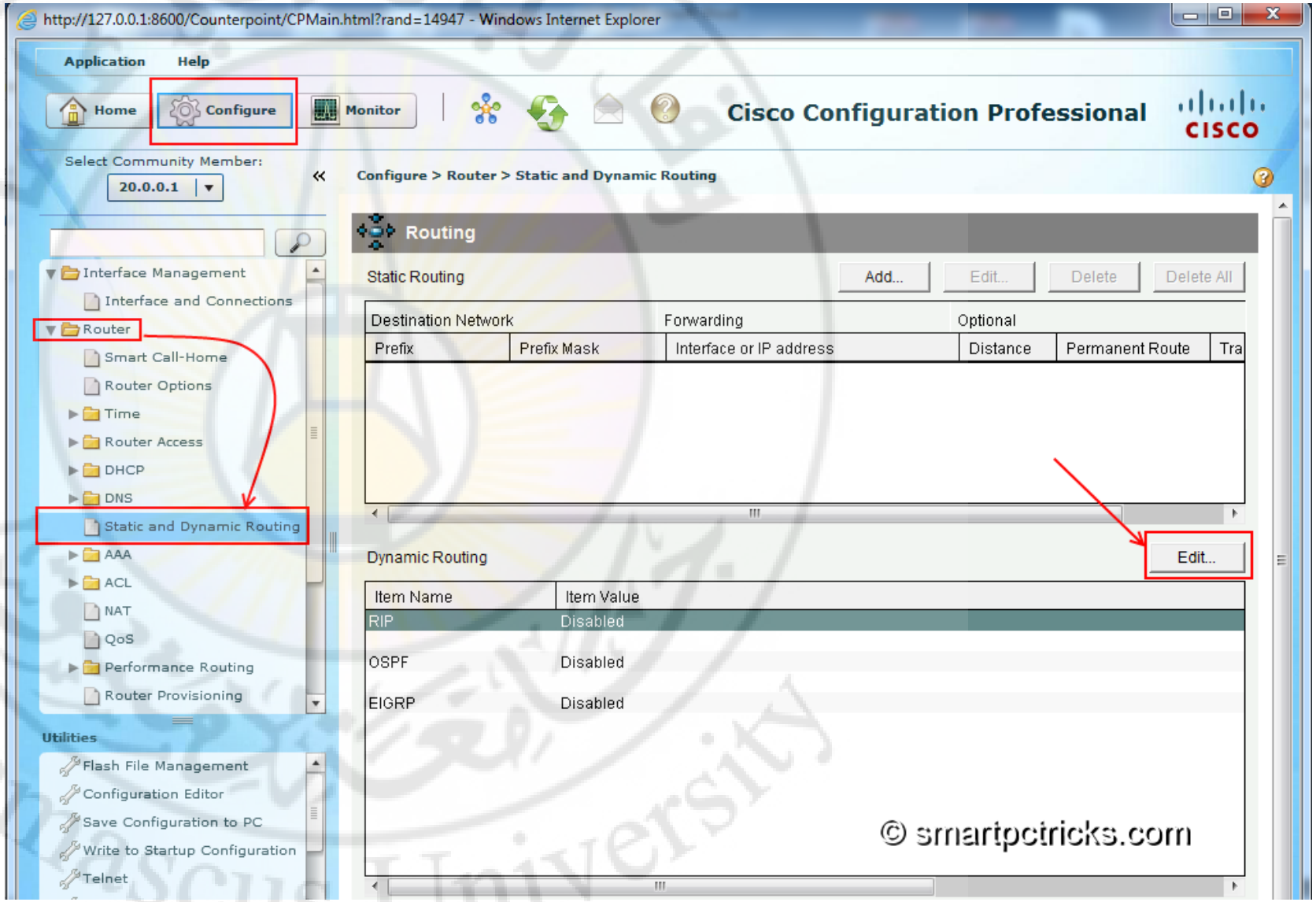
- Once discovered choose **Configure** button in the CCP window.



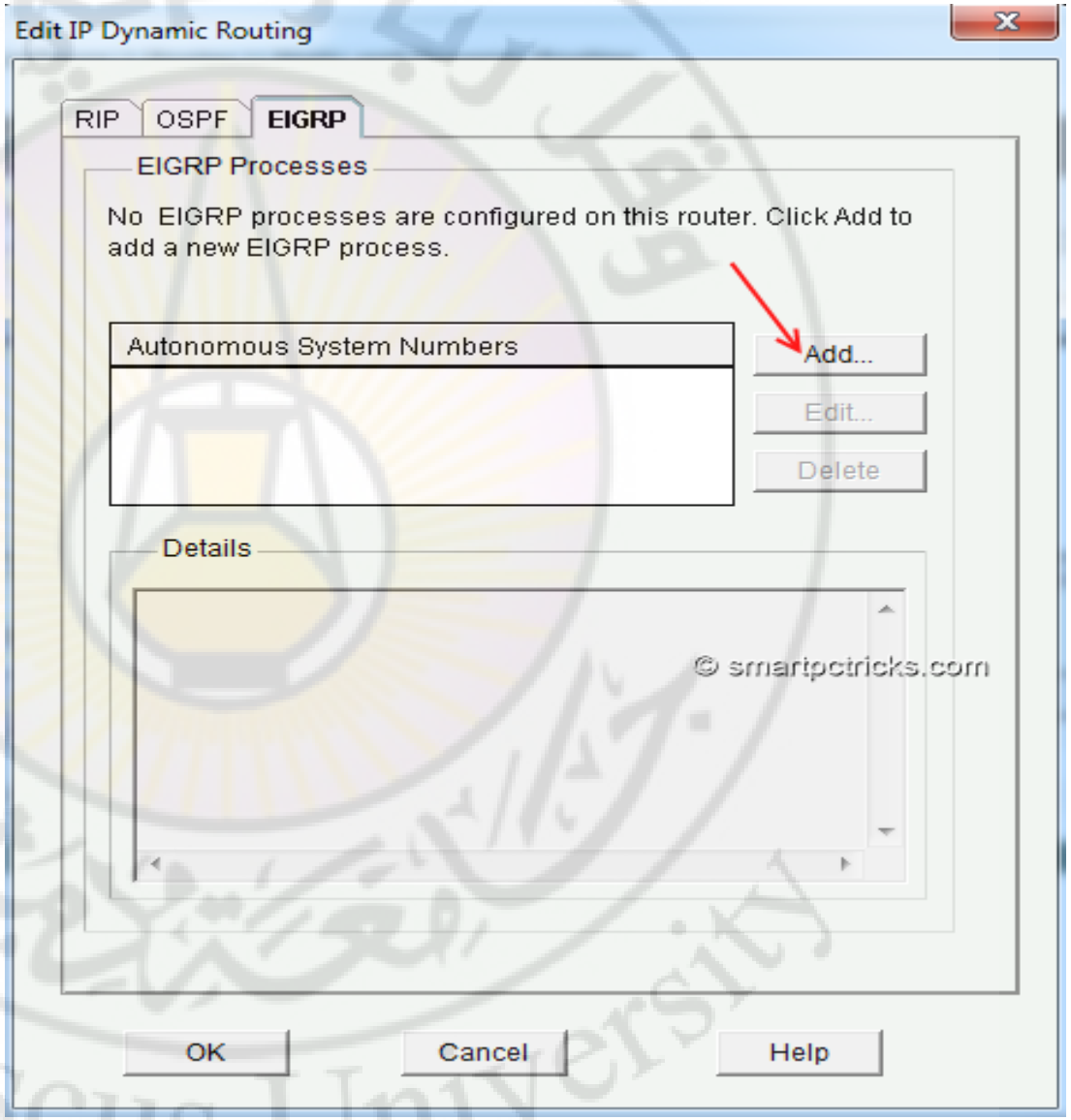
- Here you will see the entire supporting configuration for the specified router like a tree diagram.



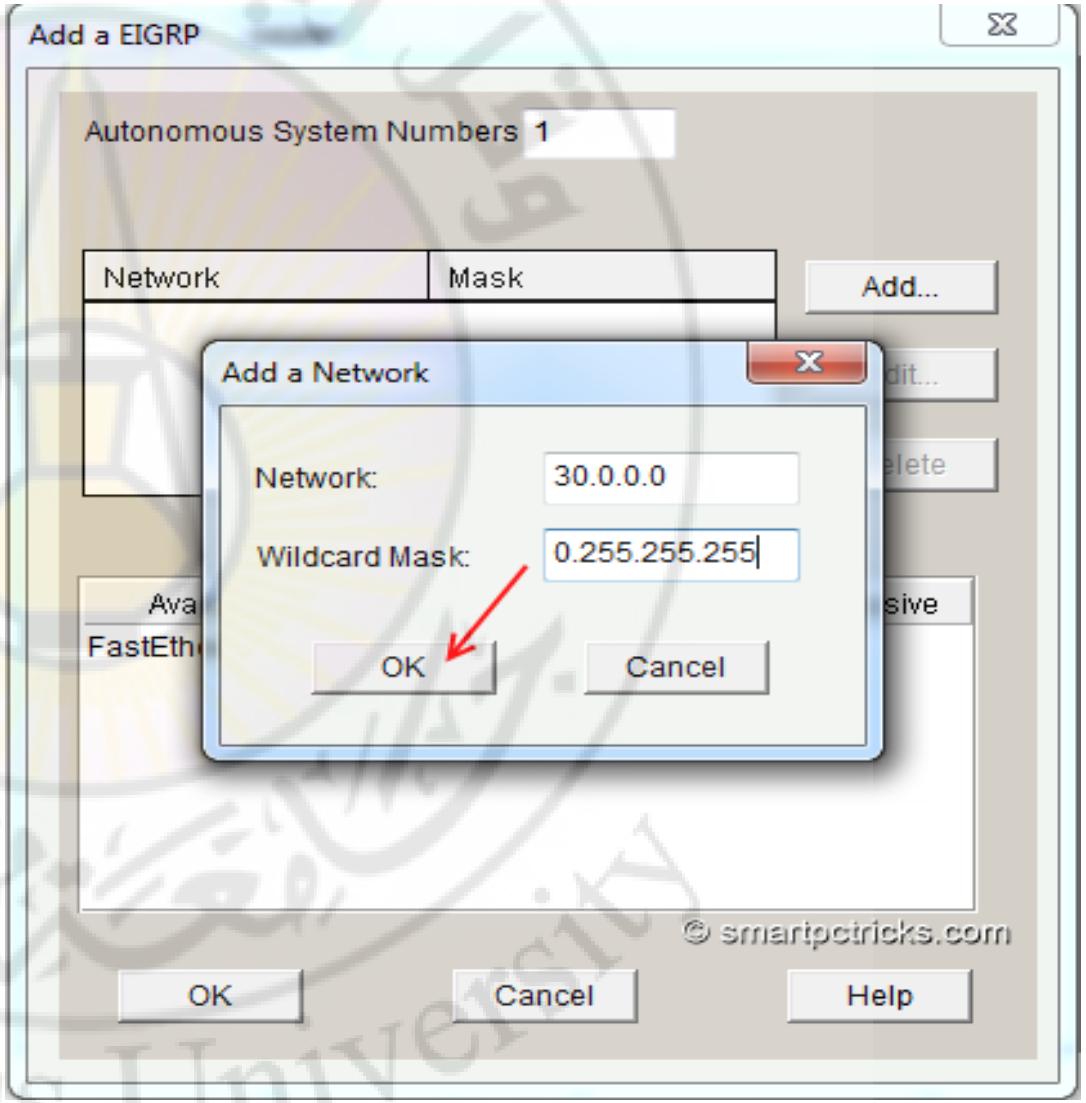
- I just selected '**Static and Dynamic Routing**' under Router tree.
- Our aim is to enable **Dynamic Routing EIGRP** in the router, so click **Edit** button near to the Dynamic Routing section.



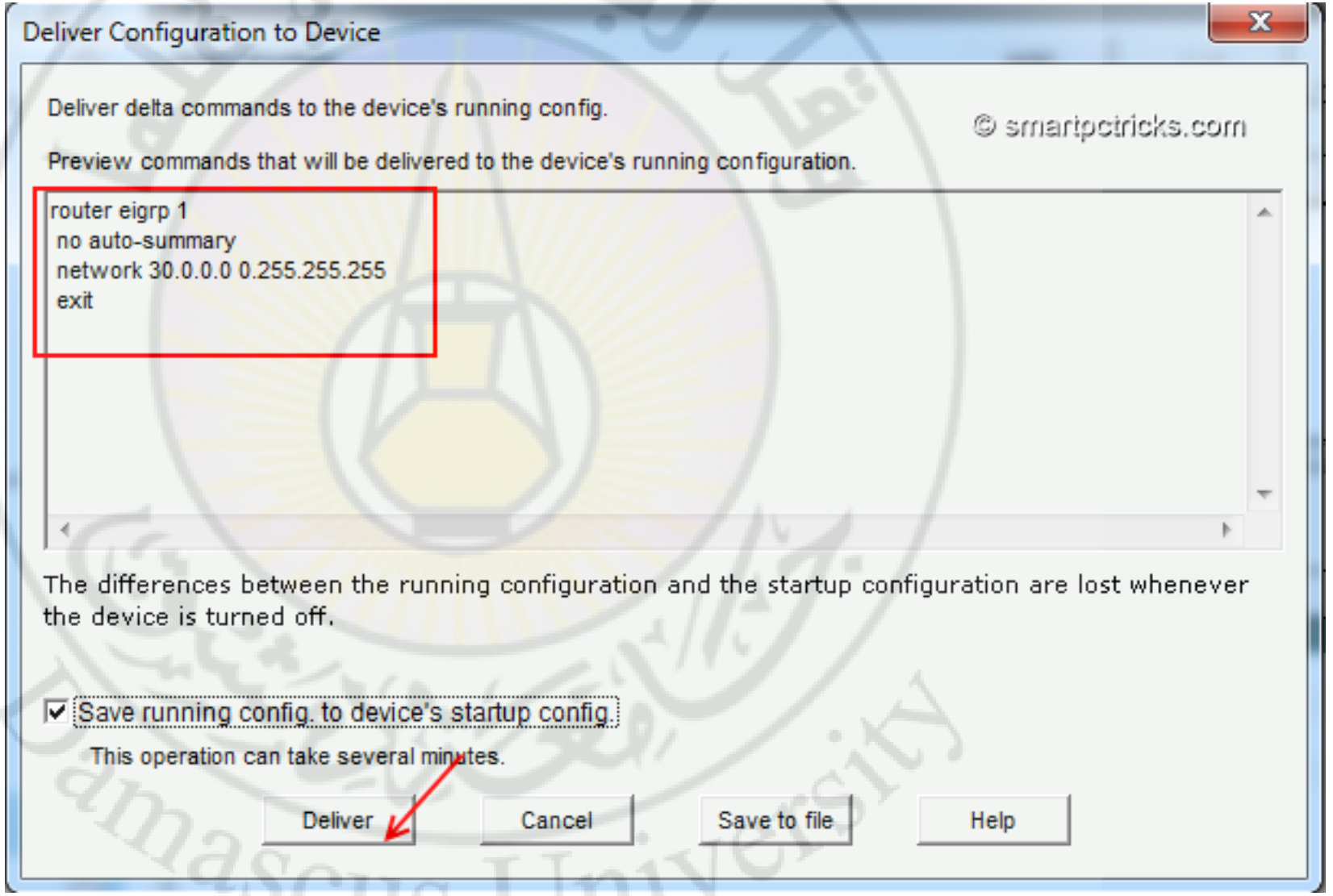
- Popup will give you **Edit IP Dynamic Routing** configur ation window, choose **Add** butt on under EIGRP tab.



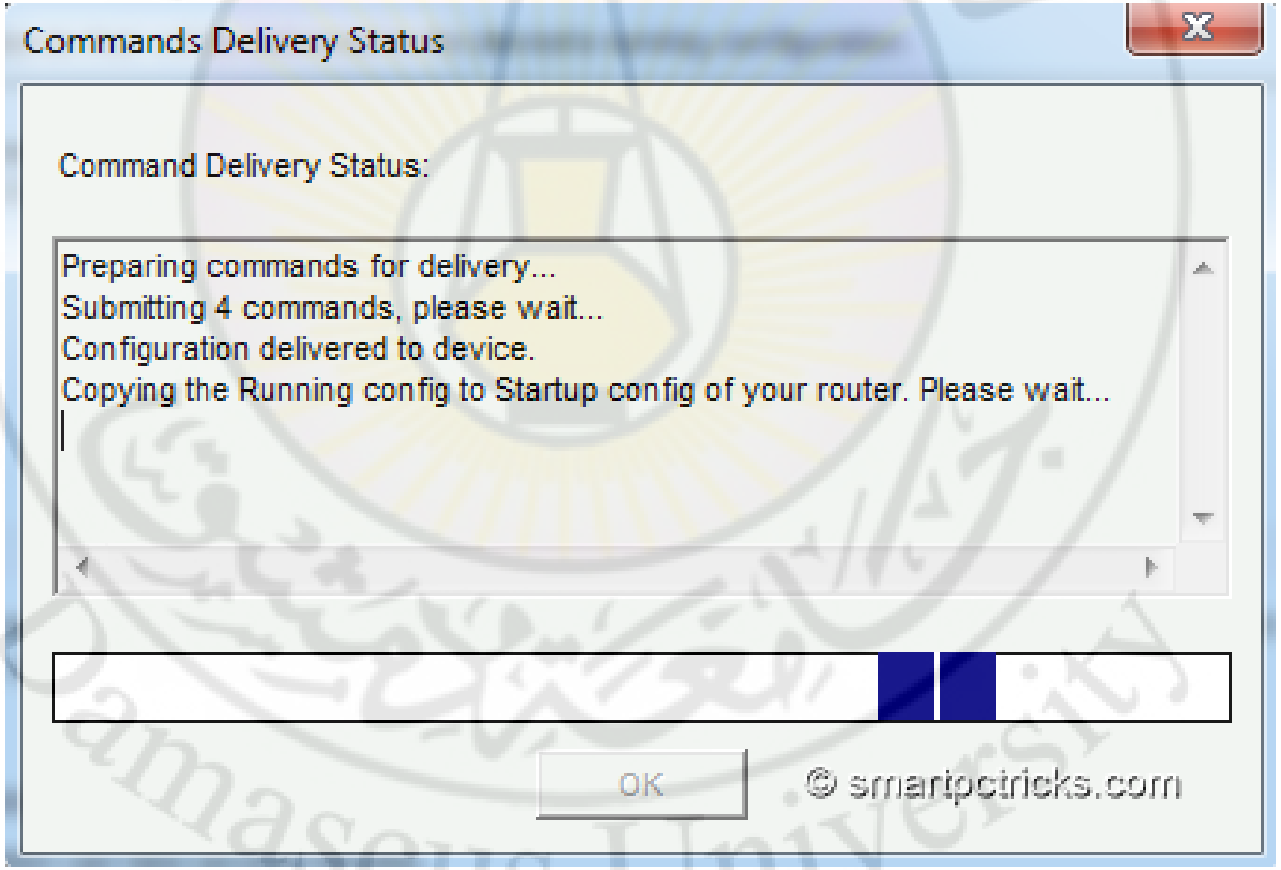
- Now we are at Add a EIGRP window, enter any **Autonomous System Number** (I entered 1), then click **Add Network** button. Enter the network **IP** with wildcard mask that you would like to **advertise via EIGRP**. Click **OK**.



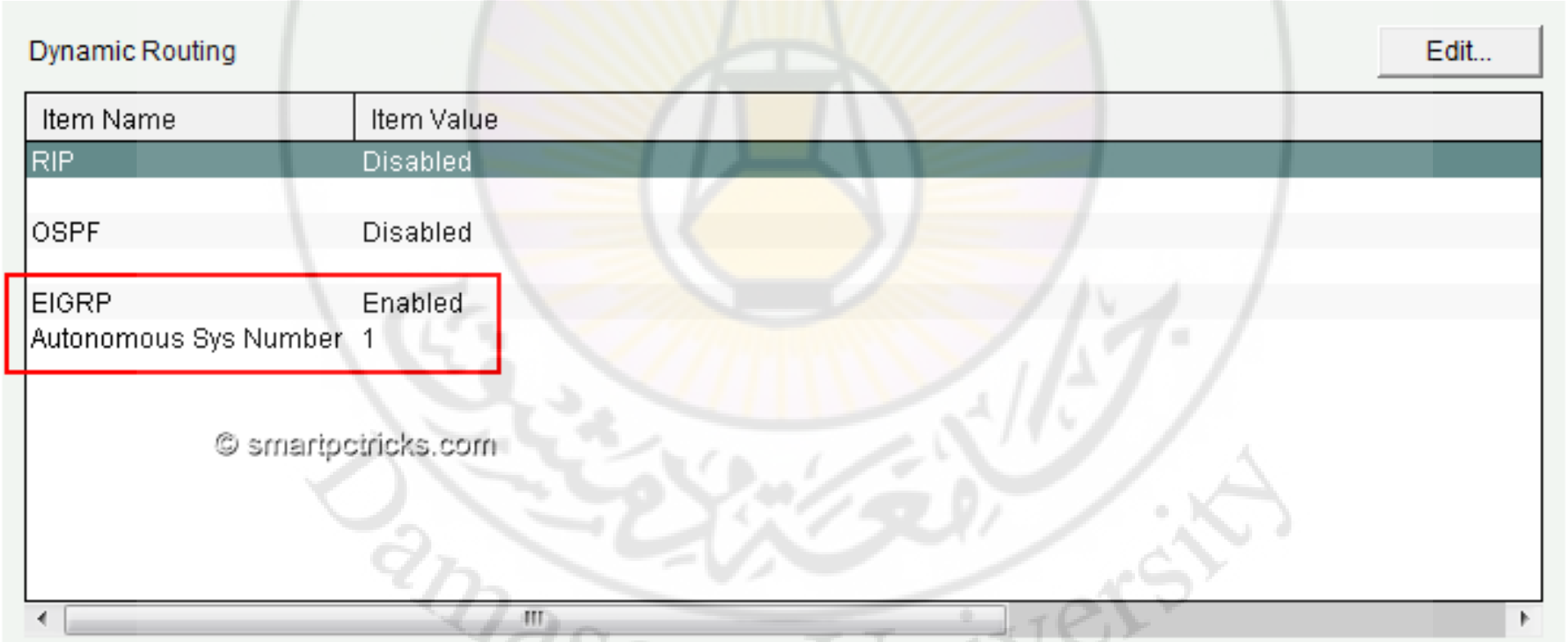
- So next is to deliver the configurati on to our router.



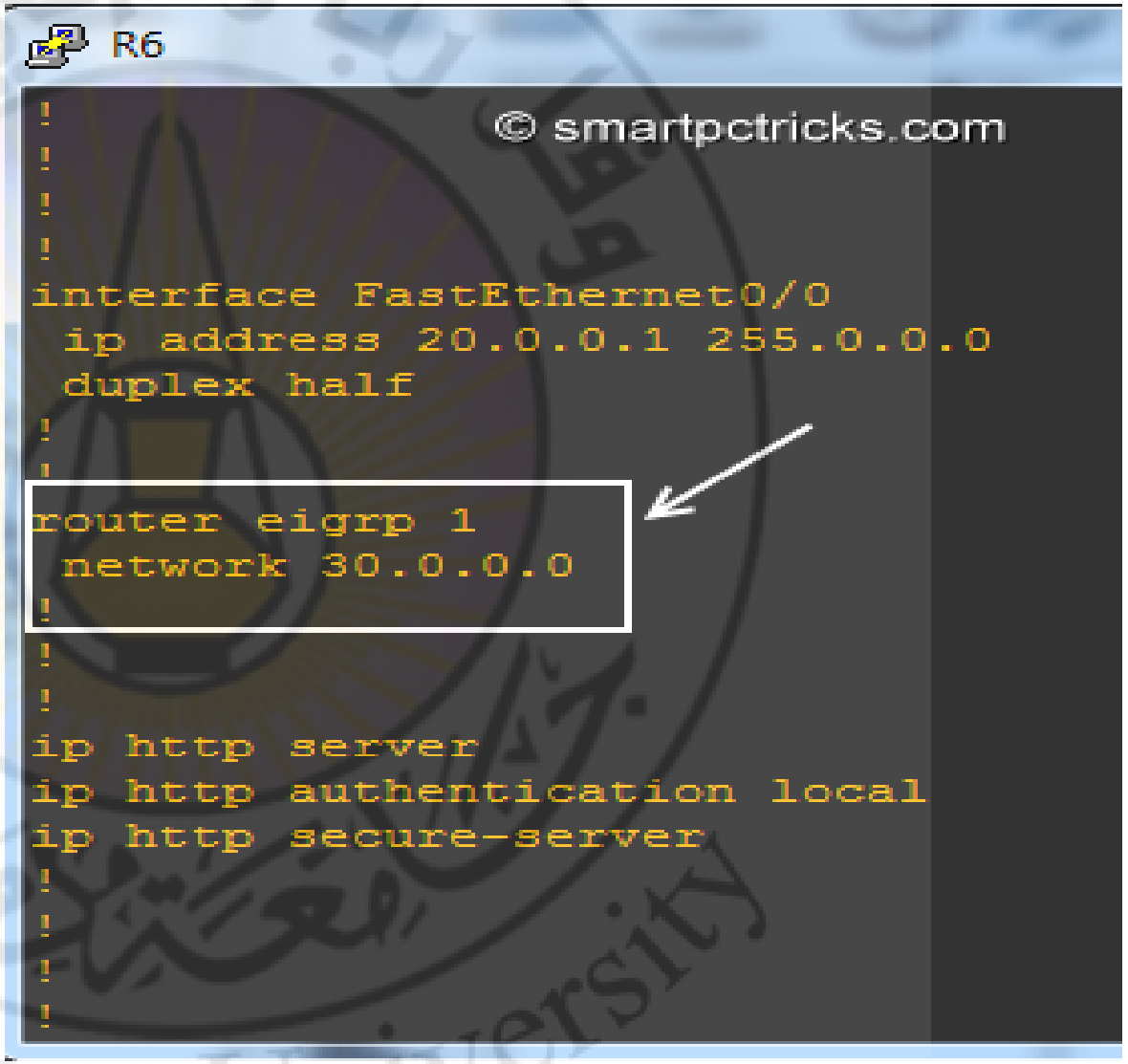
- Setback and relax while CCP deliver the commands to our device.



- After these steps you may see **EIGRP Enabled** in the Dynamic Routing section of CCP.



- Now let’s go to Putty configuration and verify the configuration changes by entering ‘**show run**’ command.



End of Session 4