EE981 Network Switching & Routing

Kashif Sharif

NAT: Network Address Translation

- Limited IP addresses
- Usage of private IP address
  - Benefits of reusability
  - Challenge of global reachability

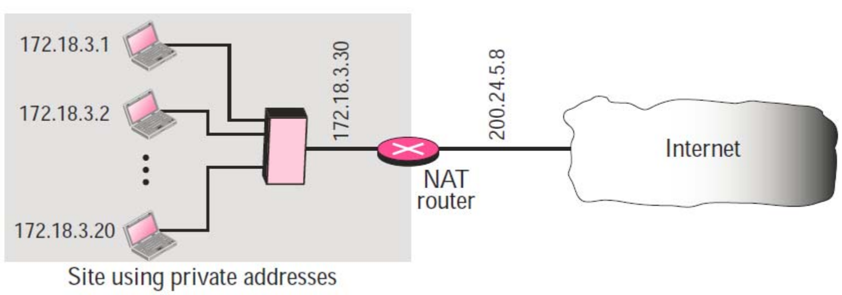
2

NAT: Network Address Translation

- Enables use of private IP addresses inside the network
- Enables communication outside network with a set of (at least one) Internet addresses

3

NAT: Network Address Translation



4

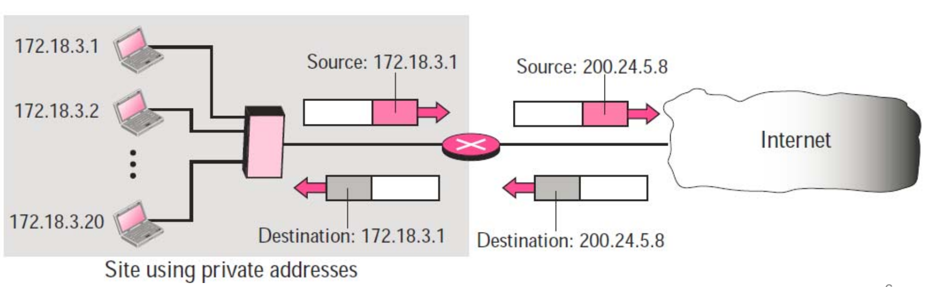
NAT Benefits

- Local network uses just one IP address as far as outside world is concerned:
- Range of addresses not needed from ISP: just one IP address for all devices
- Can change addresses of devices in local network without notifying outside world
- Can change ISP without changing addresses of devices in local network
- Devices inside local net not explicitly addressable, visible by outside world (a security plus)

5

Address Translation

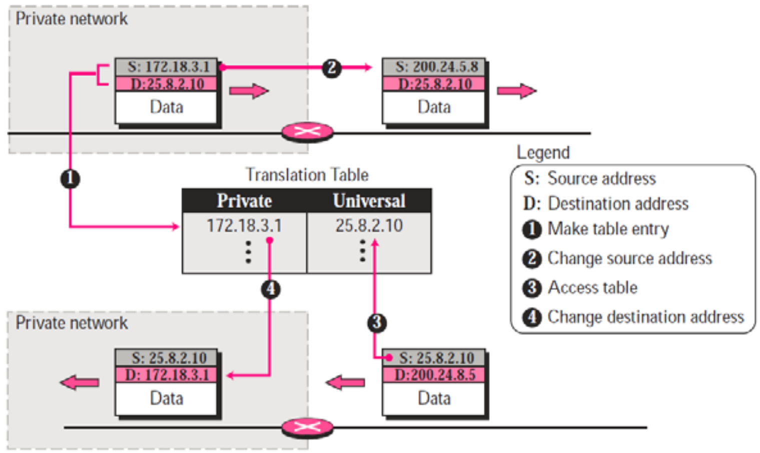
- NAT works at edge router
- Replaces the source address in IP header with Internet address for outgoing packets
- Incoming packets get the destination address changed appropriately



6

Translation Table

The mapping of private source and Internet destination addresses has to be maintained



7

Limitations of Previous Two-Column Table

- Communication must be initiated by private address host
- Communication with same destination by multiple private hosts

8

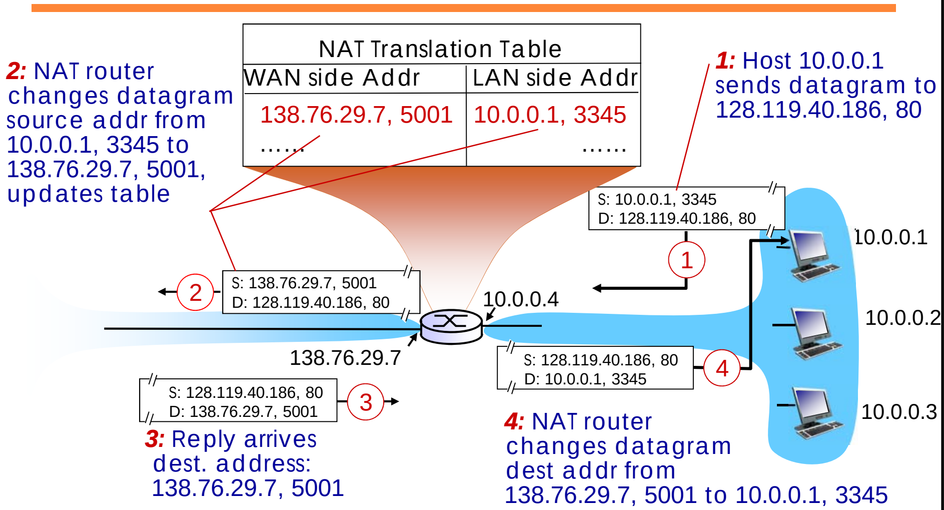
Practical Solution

Keep port information in the NAT Table

Private Address	*Private Port*	*External Address*	*External Port*	*Transport Protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
...	...	...	...	...

9

NAT Working



10

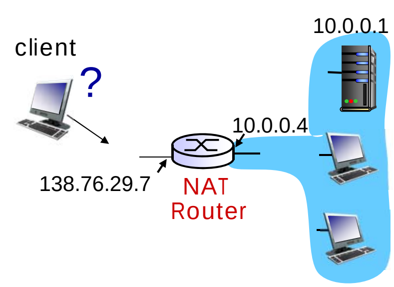
NAT Considerations

- 16-bit port-number field
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - Routers should only process up to layer 3
  - Violates end-to-end argument
    - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - Address shortage should instead be solved by IPv6

11

NAT Traversal Problem

- Client wants to connect to server with address 10.0.0.1
  - Server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - Only one externally visible NATed address: 138.76.29.7



12

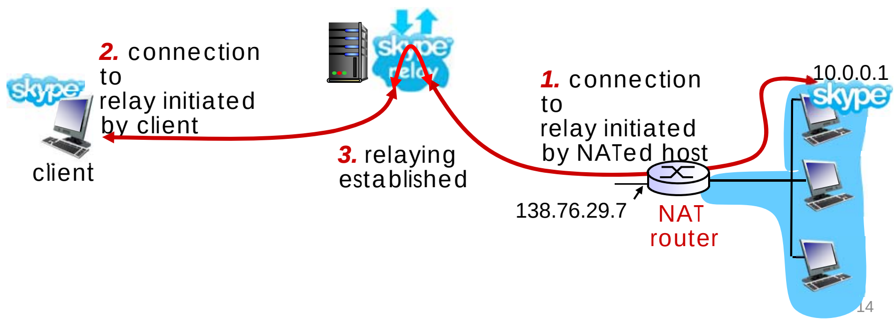
NAT Traversal Problem

- *Solution1:* Statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000
- *Solution 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATed host to:
  - Learn public IP address (138.76.29.7)
  - Add/remove port mappings (with lease times)
    - i.e., automate static NAT port map configuration

13

NAT Traversal Problem

- *Solution 3:* relaying (used in Skype)
  - NATed client establishes connection to relay
  - External client connects to relay
  - Relay bridges packets between to connections



14

Address Resolution Protocol

ARP

Address Mapping

- Hosts & Routers are identified by logical addresses
  - IP, 32 bit
- At physical level every device is identified by a physical identifier
  - Ethernet: 48bit MAC

Address Mapping

Delivery of a packet to a host or router requires two levels of addressing: Logical & Physical

Both should be map-able to each other

Static Mapping – Dynamic Mapping

17

Address Mapping

Static Mapping

- Create a table on EACH machine to store the logical and associated physical addresses
- Problems
  - Change in NIC, resulting change in mapping
  - Protocol dependent physical addresses change often
  - Computer moved from one network to another will result in logical address change

18

Address Mapping

Dynamic Mapping

- Every time a logical address needs mapping, a protocol determines the physical address

- Address Resolution Protocol
- Reverse Address Resolution Protocol

19

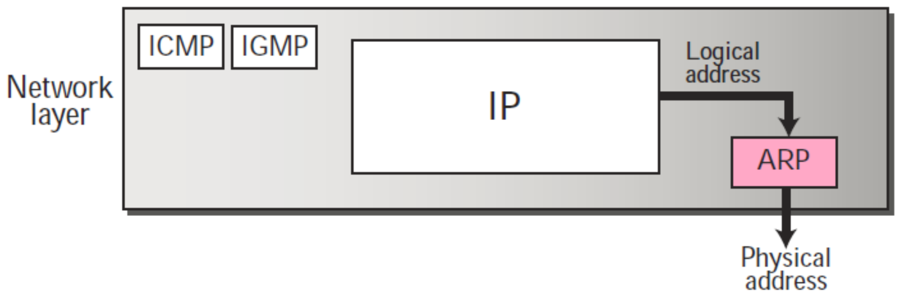
Address Resolution Protocol (ARP)

- Logical Addresses are known before communication
  - DNS protocol, other mechanism

- ARP maintains tables that keep fresh mapping of IP & MAC

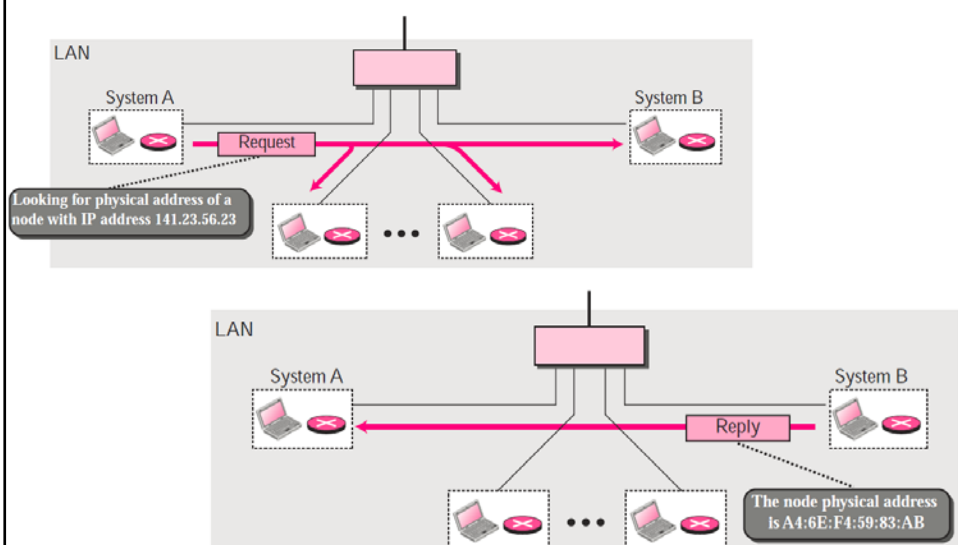
20

Network Layer & ARP



21

ARP Operation



ARP Operation

ARP Request is a Broadcast in the subnet

ARP Response is a Unicast

23

ARP Packet Format

Hardware Type		Protocol Type
Hardware length	Protocol length	Operation Request 1, Reply 2
Sender hardware address (For example, 6 bytes for Ethernet)		
Sender protocol address (For example, 4 bytes for IP)		
Target hardware address (For example, 6 bytes for Ethernet) (It is not filled in a request)		
Target protocol address (For example, 4 bytes for IP)		

24

ARP Packet Format

- Hardware Type
  - 16-bit field defining network type
  - E.g. 1 for Ethernet
- Protocol Type
  - 16-bit hex representation for higher level protocol
  - E.g. 0800 for IPv4
- Hardware Length
  - 8-bit field for defining length of hardware address
  - E.g. 6 for Ethernet Address

25

ARP Packet Format

- Protocol Length
  - 8-bit field for defining length for logical address
  - E.g. 4 for IPv4
- Operation
  - 16-bit field defining the type of packet
  - ARP Request → 1
  - ARP Reply → 2

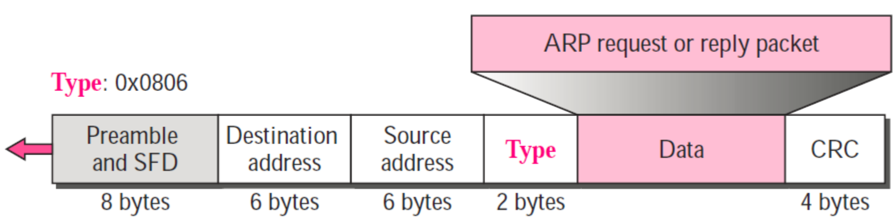
26

ARP Packet Format

- Sender Hardware Address
  - Variable length field, populated by sender with its MAC address
- Sender Protocol Address
  - Variable length field, populated by sender with its IP address
- Target Hardware Address
  - Variable length, containing all 0's from sender side.
- Target Protocol Address
  - Variable field, containing logical address of target

27

ARP Encapsulation



28

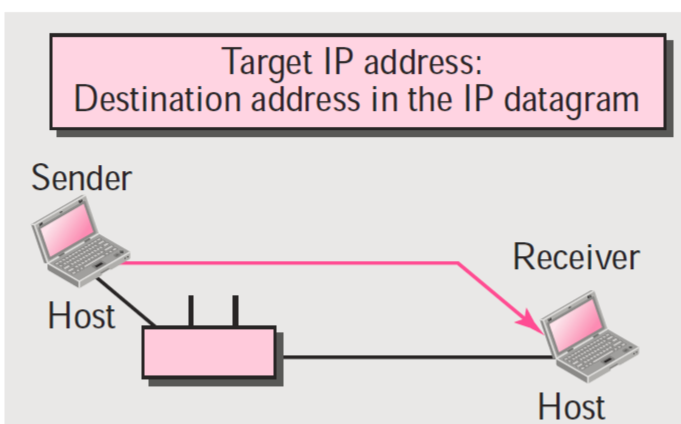
ARP Operation

- Sender knows IP address of destination
- IP asks ARP to determine MAC
  - ARP builds request appropriately
- ARP Req is passed to data link layer
  - Datalink layer broadcasts the frame in subnet
- Every machine process ARP request and drops except the target
- Target machine replies (unicast) with ARP Reply message
- Sender receives reply, and process IP packets accordingly

29

Usage of ARP

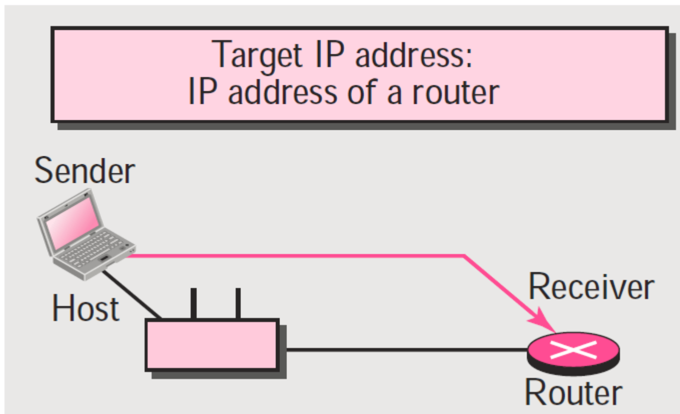
Case 1: A host has a packet to send to a host on the same network.



30

Usage of ARP

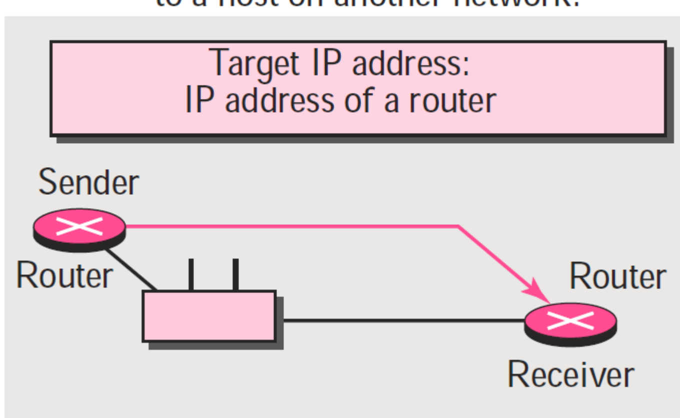
Case 2: A host has a packet to send to a host on another network.



31

Usage of ARP

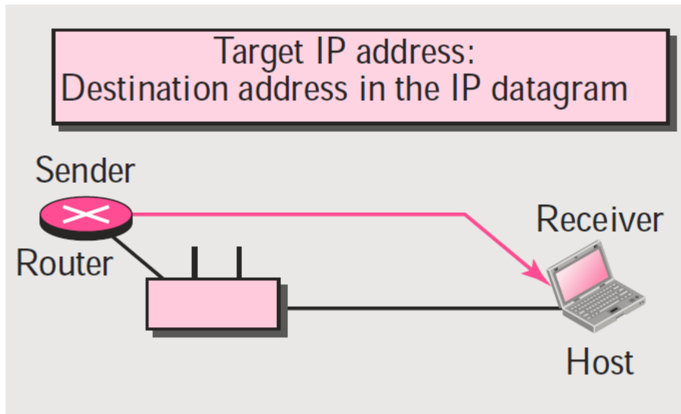
Case 3: A router has a packet to send to a host on another network.



32

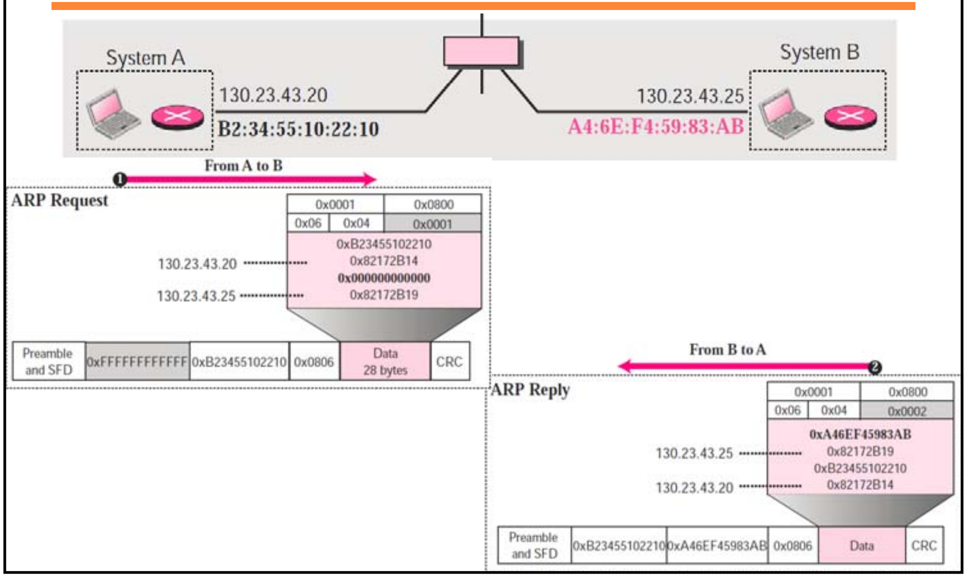
Usage of ARP

Case 4: A router has a packet to send to a host on the same network.



33

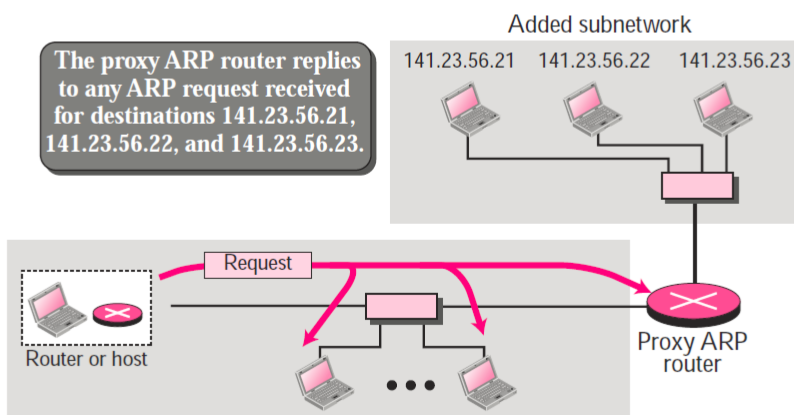
ARP Example



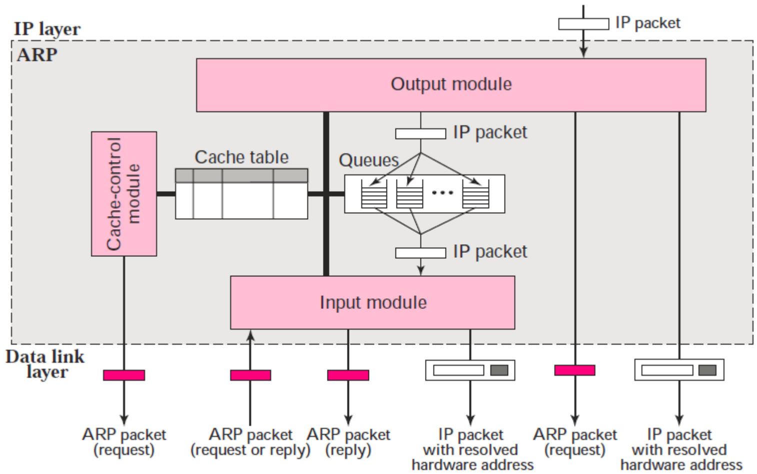
Proxy ARP

- Hides hosts & creates a subnet effect
- Router runs *Proxy ARP*
- Replies with own MAC on behalf of hosts

Proxy ARP



ARP Module



37

ARP Table

State	*Queue*	*Attempt*	*Time-Out*	*Protocol Addr.*	*Hardware Addr.*
R	5		900	180.3.6.1	ACAE32457342
P	2	2		129.34.4.8	
P	14	5		201.11.56.7	
R	8		450	114.5.7.89	457342ACAE32
P	12	1		220.55.5.7	
F					
R	9		60	19.1.7.82	4573E3242ACA
P	18	3		188.11.8.71	

- Typical maximum attempts: 8
- Cisco Timeout: 4 hours

38

ARP: Output Module

```
ARP_Output_Module ( )
{
  Sleep until an IP packet is received from IP software.
  Check cache table for an entry corresponding to the
       destination of IP packet.
  If (entry is found)
  {
      If (the state is RESOLVED)
      {
        Extract the value of the hardware address from the entry.
        Send the packet and the hardware address to data
             link layer.
        Return
      } // end if
      If (the state is PENDING)
      {
        Enqueue the packet to the corresponding queue.
        Return
      }//end if
  }//end if
  If (entry is not found)
  {
      Create a cache entry with state set to PENDING and
           ATTEMPTS set to 1.
      Create a queue.
      Enqueue the packet.
      Send an ARP request.
      Return
  }//end if
} //end module
```

39

ARP: Input Module

```
ARP_Input_Module ( )
{
    Sleep until an ARP packet (request or reply) arrives.
    Check the cache table to find the corresponding entry.
    If (found)
    {
        Update the entry.
        If (the state is PENDING)
        {
            While (the queue is not empty)
            {
                Dequeue one packet.
                Send the packet and the hardware address.
            }//end if
        }//end if
    }//end if
    If (not found)
    {
        Create an entry.
        Add the entry to the table.
    }//end if
    If (the packet is a request)
    {
        Send an ARP reply.
    }//end if
    Return
}//end module
```

40

ARP: Cache-Control Module

```
ARP_Cache_Control_Module ( )
{
    Sleep until the periodic timer matures.
    Repeat for every entry in the cache table
    {
      If (the state is FREE)
      {
         Continue.
      }//end if
       If (the state is PENDING)
       {
          Increment the value of attempts by 1.
          If (attempts greater than maximum)
          {
              Change the state to FREE.
              Destroy the corresponding queue.
          }// end if
          else
          {
            Send an ARP request.
          }//end else
          continue.
       }//end if
       If (the state is RESOLVED)
       {
          Decrement the value of time-out.
          If (time-out less than or equal 0)
          {
             Change the state to FREE.
             Destroy the corresponding queue.
          }//end if
       }//end if
    }//end repeat
    Return.
}//end module
```