**Computer Networking**

*(Learning from Udemy’s Mike Meyer’s COMPTIA Network (N10-007))*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Network Models** | | | | | | | | | |
| - Explain devices, applications, protocols and services at their appropriate OS1 layers  - Models are used to represent how networks function  - There are two very popular network models  1. OSI seven-layer model  2. TCP/IP model  - Each layer  - Has a distinct function  - Has a name and a number | | | | | | | | | |
| **OSI model** | | | | **TCP/IP model** | | | | | |
| - Specifies 7 distinct functions that a network must do  - Older, more detailed | | | | - Looks at applications as applications  - E.g. email, FTP, telnet  - More modern, not as detailed but not as complicated | | | | | |
| Layers | | | |  | | | | | |
| 7 - Application | - The APIs etc built into an application that makes the application “network aware”  - E.g. MS Words knows how to search and show what files to open | | | 4 - Application | | | | - Everything to do with application works in this layer | |
| 6 - Presentation | - Converts data into a format that your applications can read | | |
| 5 - Session | - Actual connection between two systems  - Defines what’s taking place in terms of how that connectivity really works  - E.g. TCP connection between a web server and a web client, sending an email, sharing a folder | | | 3 - Transport | | | | - Assembly, disassembly  - What it takes to connect to other system to make sure data gets there  - E.g. TCP, UDP | |
| 4 - Transport | - Assembly, disassembly area for data as it gets broken down into individual packets  - Makes sure that the packets gets into the other system in good order | | |
| 3 - Network | - Has to do with logical addresses  - E.g. IP addresses, routers | | | 2 - Internet | | | | - IP addresses, routers | |
| 2 - Data Link | - Anything that works with a MAC address works at the data link layer  - E.g. Network cards, switches | | | 1 - Network Interface/Link | | | | - Hardware  - Covers all the physical cabling, MAC addresses, network cards  - Exceptions: routers | |
| 1 - Physical | - What can of cables to use and stuff like that | | |
| Flow  (from Layer 1 up) | | **Layer 1/1 (OSI/TCP/IP)**  1. Network layer POV: network card waits for data  2. Network card receives an ethernet frame  3. Checks incoming MAC address and verify that it’s for it (the correct system)  **Layer 2/1**  4. Frame check sequence, check the entire Ethernet frame to make sure its in good shape  5. Strip off MAC address and store it elsewhere in memory, left with an IP packet  6. Network card passes IP packet (frame + source,dest IP address) to next layer  **Layer 3/2**  7. Network layer POV: Look at IP addresses, make sure that its for it  8. Discard its own IP address but keeps IP address of where information came from  9. Depending on what type of packet, what’s left can be e.g. TCP segment (designed and ready for computer, just need to get it to the right applications and in the right format)  10. Passes it to the next higher layer  **Layer 4/3**  11. Transport layer POV: act as the assembler/disassembler  12 (1). If its big-sized data *going out,* disassemble  - Break it into bite-sized chunks  - Disassemble and add sequencing numbers to individual TCP/IP segments  12 (2). If its bunch of data *going in*, reassemble  - Reassemble all the data using sequencing number  13. Pass complete chunks of data *and* port numbers to next layer  **Layer 5-7/4**  14. Session layer POV (layer 5): designed to connect a server to a client on a remote system (try to connect to something)  - E.g. MS Word isn’t network aware (only local), so needed a distinct session layer to connect to a remote system  15. Presentation layer POV (layer 6): gets data to a particular application but not in the form that the application itself could use  16. Application layer POV: looks at port numbers  - Keeps return port numberin memory somewhere  17. Sends data to right application using destination port number | | | | |  | | |
| Frames | | - Devices on a network send and receive data (in discreet chunks called frames or packets)  - Frames are created and destroyed inside the network interface card (NIC)  - If frame not meant for that network, network card will just consume it (data will not go past network frame) | | | | |  | | |
| - Packetized data  - A single frame can be up to 1 500 bytes long/in size (10 000 ones and zeroes)  - Generated inside the network card, data comes from application  - Eaten up by network cards and sent up to whatever software that needs it, and wiped out on the card  - Have a discrete beginning and end | | | | |  | | |
| MAC address | | - Media Access Control (MAC)  - How frames know to get to the right computer  - Frame payload does not identify the destination  - 48-bit address  - 6 pairs, 12 hexadecimal values (each representing 4 binary characters)  - In total, 48 binary characters  - First 3 pairs, represents a unique OEM (Original Equipment Manufacturer), issued to the maker of the network cards  - Last 3 pairs, represents a unique ID  - Every network card in existence has a unique MAC address (built into Network Interface Card - NIC) | | | | | Terminal command:  *ipconfig /all*  - See all information about all network cards  - MAC address is under “physical address” | | |
| Cyclic redundancy check (CRC) | | A way to verify that the data is good  - If data is bad, network knows to resend it | | | | |  | | |
| Hardware | | Hub | - Allows multiple computers to connect to each other to share resources  - E.g. webpages, documents  - Where LAN came from  - Receives one signal from source computer, and repeats signal to send it out to all other connected computers in hub | | | | - IP packets never change but never travel by themselves but within a frame | | |
| Router | - Have two connections or more  - Connects multiple local area networks  - Usually includes a switch  - MAC addresses from source computers (especially over the internet) usually points to the router’s MAC address  - Flow:  1. Router receives frame  2. Identifies MAC address as its own (correct delivery)  3. Strips away its MAC address from frame  4. Uses routing table to find the computer IP packets is meant for  5. Adds back to frame MAC address of the destination computer (found by routing table)  - Has a routing table - tells based on network information, where to send data | | | |
| Switch | - Local Area Connection (LAC) | | | |
|  |  | | | |
|  |  | | | |
|  |  | | | |
| Data | | Source MAC + Dest MAC  + Source IP + Dest IP  + Source port no. + Dest port  + Frame  + CRC | | | | |  | | |
| **Broadcast** | | | | **Unicast** | | | | | |
| - A broadcast transmission is sent to every device in a broadcast domain | | | | - Unicast transmission is addressed to a single device on a network | | | | | |
| - Computer does not know who it is sending out to  **How it works**  - One computer sends out a frame that propagated to everyone else  - But for the destination MAC address, it puts **“FF-FF-FF-FF-FF-FF”** (broadcast address)  - Every computer that receives this information will accept it can pass it up through the layers  - Very important for a local area network (LAN)  - Broadcast domain  - When a group of computers are in a broadcast domain, they can hear each other’s broadcast  - When plugged into a hub, all connected computers are in the same single broadcast domain  - E.g. Broadcast for “Tom’s” computer to send back his MAC address | | | |  | | | | | |
| - Problem of broadcast domain and MAC addresses, sending out too many broadcast messages that nothing gets done | | | |  | | | | | |
| Logical addressing | | - Needed especially in a very big network (using only MAC addresses isn’t sufficient)  - E.g. IP addressing  - Not fixed with card  - Can be used to identify a particular network  - First three octets identifies computers on the same network, last octet uniquely identifies the computer on the network  - Now MAC address refers to the router | | | | |  | | |
| Default gateway | | Connection router itself | | | | |  | | |
| Packets | | - Packets have sequence numbers so the network software can reassemble the file correctly  - TCP (Transmission Control Protocol)  - Connection-oriented conversation between 2 computers  - Sequencing numbers  - Receivers acknowledge receipt of data  - UDP (Use Datagram Protocol)  - Connectionless, sends and hopes you’re ready for it  - If not, send again  - Does not verify the receipt of data | | | | |  | | |
| Ports | | - Port numbers help direct packet traffic between the source and destination  - How computer knows which web page to send IP packet to  - Port range is 0 - 65535  - First 1024 (TCP, UDP) port numbers are reserved  - Port 80: HTTP  - Port 20, 21: FTP | | | | |  | | |
| **TCP/IP Basics** | | | | | | | | | |
| IP addresses | | - Each computer on a TCP/IP network must have a unique IP address  - IPv4 addresses are written as four octets  - E.g. 192.168.4.12  - Each octet represents a binary string  - E.g. 192 is 11000000 | | | | |  | | |
| - Usually given in dotted decimal notation  - Dots in an address has no meaning  - Actually made up of 32-long ones&zeros  - Number of combinations: 28 = 256  - If separate bits into octets (groups of 8, 4 groups), each octet is valued between 0 and 255  - How to convert binary (1 octet) to decimal  1. Start with binary  - E.g. 11000101  2. Use code = **“128 64 32 16 8 4 2 1”**  - Which is 128 divided by 2 repeatedly  3. To get decimal,  - decimal = sum(code[bit] for bit in binary)  - i.e. sum up numbers in code with index corresponding to 1s’ index in code  4. Therefore, decimal = 128 + 64 + 4 + 1 = 197  5. Conclusion, octet 11000101 = 197  - How to convert decimal to binary  1. Using code, start from “128” and continue towards “1”  2. If decimal is larger than current number in code, set bit 1 for that index and substract from decimal  3. Else if decimal is smaller than number, set bit 0 for that index  4. Continue to next number until 8 bits are filled | | | | | Examples:  00000000 = 0  11111111 = 255  10000000 = 128  00000001 = 1 | | |
| Address Resolution Protocol (ARP) | | - Resolves MAC addresses from IP address  - When: used when computer knows IP address but needs MAC address of another computer to send data to  - What: sends a broadcast over the network (requests computer with IP address to send back its MAC address)  - Computer keeps a list of IP address-to-MAC address mappings (for efficiency, avoid repeated asking) | | | | | On Terminal:  *arp -a*  - Shows ARP cache (the list of IP-MAC addresses stored in computer) | | |
| Classful Addressing | | - Given a scenario, configure the appropriate IP addressing components  - Internet Assigned Numbers Authority (IANA) keeps track of all IP addresses and assigns them to computer  - Does not do so directly, passes IP addresses to regional Internet registries (RIRs)  - RIRs passes them to Internet Service Providers (ISPs)  - **Class licenses**  - Determines the first octet of network ID  - Class A gives the largest group of addresses  - **Class A** (0-126 **/8**)  - First octet goes from 1 to 126  - Pronounced “*whack 8*”  - **Class B** (128-191 **/16**)  - First octet goes from 128 to 191  - First 2 octets assigned by ISP, any combination of remaining 2 octets are up to you to decide  - **Class C** (192-223 **/24**)  - First octet goes from 192 to 223  - E.g. 254 possible hosts | | | | | - Part of IP address that changes for every individual computer | | |
| IDs | | **Network ID** | | | **Host ID** | | | | |
| - Part of network numbering system  - Has to be identical for every computer in the same network | | |  | | | | |
| - IP address ends with either 0 or 255 (reserved for network IDs) | | | - Cannot use 0/255 for Host ID | | | | |
| - Use with subnet mask to determine the range of IP addresses in subnet | | |  | | | | |
| Subnetting | | - Divides network IDs into two or more networks  - Use case: avoid plugging in all computers into one switch (especially in a large organization)  - Classful subnetting  - First effort to divide network IDs  - E.g. if given Class B license (160.25.X.X)  - Can have 160.25.**1-255**.X (network ID of each subnet has a different third octet)  - Moving from /16 to /24  - Subnets do not have to be on the dots  - Commonly done by ISPs  - To give a user an appropriate number of IP addresses (no need give so many) | | | | | *“whack 24”*  - Gives 254 hosts/addresses  - 28 - 2 = 254  - Explanation:  - 8 (bits that can be variable in address: 32 - 24 = 8)  - 2 (either 0 or 1)  - minus 2 (cannot end with 0 or 1)  “*whack 16*”  - Means only the first two octets have to be the same for all computers  - Indicates a subnet mask of 255.255.0.0 since first 16 binary digits are 1 (i.e. 11111111 11111111 0 0) | | |
| **Q**: Given network address class type (e.g. Class C network address 217.105.2.0), which subnet mask will yield how many  - subnetworks  - hosts per network? | | | | **A:**  Number of subnetworks=  **2^(*x* - *y*)**  where *x* is whack of class type (e.g. /24) and *y* is whack of subnet mask (e.g. /26)  Number of hosts per network=  **2^(32 - y)**  where minus two since a host IP cannot end with 0 nor 1 | | | |
| Subnet masks | | - Each host needs a subnet mask  - Uses it to know if the destination is on the local network or a remote network  - Each host knows the default gateway so that it can forward traffic to remote networks  - Built into every network that identifies it as its own LAN  - If local address, do a local ARP  - If address not within same LAN, go through default gateway through router (to the internet)  - Made up of 1s and 0s (has to start with a string of 1s and followed by a string of 0s)  - IP address number with corresponding to 0s are allows to vary within the subnet  - Gets smaller for very huge networks  - Gets longer for longer whack | | | | |  | | |
| **How to use subnet mask**  - Compare IP addresses of local computer with destination computer  - To determine if they are in the same network  - If subnet mask of local computer is “whack 24”, then first 24 bits of binary IP address has to be the same for the two addressed for them to be on the same network  - Subnet masks are used by a computer to determine the network ID  - They are never sent out of a host | | | | | Subnet mask:  11111111 11111111 11111111 00000000  Local IP:  11101000 00011001 11010000 00010110  Destination IP (to talk to):  11101000 00011001 11010000 00001001  - Where subnet mask has 1s, IP address/numbers is the same for both computers  - Hence they are in the same local network | | |
| Classless Inter-Domain Routing (CIDR) | | - Granular way on how to take a subnet and chop it up into many smaller subnets  - E.g. from whack 24 to whack 25, gives 2 network IDs  - E..g. from whack 24 to whack 26, gives 4 network IDs/subnets and 62 hosts per subnet  - The more you subnet (i.e. increase whack *x*), the less hosts are available | | | | |  | | |
| What you should know when setting up a computer’s network | | 1) IP address  2) subnet mask  3) default gateway | | | | |  | | |
| Dynamic Host Configuration Protocol (DHCP) | | - Dynamic Host Configuration Protocol (DHCP) or Bootstrap Protocol (BOOTP) (usually for Linux)  - DHCP  - Usually manifests itself as a server, or can be a special software sitting on certain computers (e.g routers)  - Individual computers as DHCP clients | | | | | | | |
| - When computer firsts boots up, has no IP settings  - Will broadcast a “DHCP Discover” message  - DHCP server will respond with a “DHCP Offer” unicast traffic  - Computer accepts and sends a “DHCP Request”  - DHCP server receives the request and sends an acknowledgment  - Stores information and keeps track of all the clients | | | | | | | |
| - Each broadcast domain must only have one DHCP server  - DHCP server has to be run within the broadcast domain  - Cannot configure any routers and servers across a wireless network, for security reasons  - Every modern operating system comes with DHCP enabled by default | | | | | | | |
| **DHCP Relay**  - When router is selected to act as DHCP Relay, it will forward any DHCP request to a DHCP server located elsewhere  - Enables a single DHCP server to service more than one broadcast domain | | | | | | | |
| **Rogue DHCP Servers**  - Can assign incompatible IP addresses to hosts on a network making them unable to communicate with other hosts or the Internet  - Cause IP address incompatibilities or worse  - But does ensure that hosts will not generate APIPA addresses  **Automatic Private IP Addressing (APIPA)**  - Built into all DHCP clients  - A fallback if DHCP client can’t be found (an DHCP lease expires), so hosts gives itself an APIPA  - APIPA address always starts with **169.254** (indication that client cannot connect to a DHCP server)  - Next two octets are randomly generated  - Troubleshooting  1) If get a APIPA address, check to see if connected to a DHCP server  - Check if cable and switches are connected  2) If connected to DHCP server, and still get an APIPA address, make sure that the DHCP server is working  3) If get an IP address other than your correct network ID, there may be a rogue DHCP server  - Check if there is more than one DHCP server (especially wireless) | | | | | | | |
| Static IP addresses | | - Sets a constant, unchanging IP address to a host | | | | | | | |
| Special IP addresses | | **APIPA**  - 169.254.X.X | | | | |  | | |
| **Private IP addresses**  - Only used in private networks/internal networks that do not share outside of LAN  - Will not be found on the internet  - Can get out to internet but other people cannot assess you  - 3 possible groups  1) 10.X.X.X  2) 172.16.X.X to 172.31.X.X  3) 192.168.X.X | | | | |  | | |
| **Loopback addresses**  - Loopback adapter  - Allow you to address yourself  - Example: ping yourself  - Example:  - **127.0.0.1** (IPv4 loopback)  - 127.0.X.X (entire network ID of 127 is the loopback)  - ::1 (IPv6 loopback) | | | | |  | | |
| IP Addressing Scenarios | | Duplicate IP address | Possible reasons:   1. Rogue DHCP server passing out identical IP addresses 2. Statically setup the same IP address on two different system servers    * Windows 10: Automatically detects there’s a problem, will try to turn off static IP and try to set to DHCP    * Ubuntu: Will not give an easy way to detect same IP address (will cause 2 systems to have same IP address)   What happens:   1. Router will not know who to return response to (duplicate IP address) 2. Can cause one or both hosts with the duplicated address to lose communication   How to solve:   1. Manually set different IP addresses statically 2. Store static IP addresses in a database so there is no confusion | | | | | | |
| Duplicate MAC address | Possible reasons:   1. Accidentally set same MAC address when cloning virtual machines   What happens:   1. Virtualised switch or device trying to send ethernet frames to individual virtual machines will be confused 2. When switches find that they have to store 2 identical MAC addresses for different ports, they will default into hub mode 3. Affected computer cannot receive data, cannot connect to internet, cannot connect locally to individual computers 4. Really hard to detect 5. Can cause one or both hosts with the duplicated address to lose communication | | | | | | |
| Incorrect gateway | Possible reasons:   1. Man-in-the-middle attack 2. Wrong gateway set (since manually entered)   What happens:   1. Cannot connect beyond LAN | | | | | | |
| Incorrect subnet mask | Possible reasons:   1. Wrong subnet mask set (since manually entered)   What happens:   1. Example (unilaterally ping, different subnet mask):    1. Computer A has subnet mask 255.255.0.0 and Computer B has subnet mask 255.255.255.0    2. Computer A can ping Computer B cannot ping Computer A    3. Computer B sees Computer A as in a different subnet   How to solve:   1. Check that all computers in the same broadcast domain have the same subnet mask | | | | | | |
| Expired IP address | How things work:   1. Computer goes to DHCP server, gets a lease    * Lease period is measured depending on DHCP server    * E.g. 8 days in Windows 2. After half the lease period, computer will automatically re-establish the lease by going back to the DHCP server 3. DHCP server will always give back the same IP address if possible   Possible reasons:   1. DHCP no longer exists    * DHCP lease expires (or when computer resets) | | | | | | |
| **Routing** | | | | | | | | | |
| - Explain the characteristics and concepts of routing and switching | | | | | | | | | |
| Router | | - Box that interconnect network IDs  - Does not care about:  1) Ethernet information on a frame  2) Where the frame came from (i.e. the interface)  - Only cares about:  1) Destination  - Has a routing table  - Never almost change IP packet  - Can have all kinds of connections/any network medium  - Ethernet, DSL, Optical, etc  - Main role:  - Read destination IP address and change the MAC address in frame depending on where they want to send it to | | | | | Note: Different from switches, filters and forwards based on MAC addresses | | |
| Gateway routers | | - Routers with only two connections | | | | |  | | |
| Routing table | | - Has 4 columns  - Address column identifies IP addresses used by routers (all the addresses ends with 0)  Example:   |  |  |  |  | | --- | --- | --- | --- | | Address | Subnet | Gateway | Interface | | 192.168.15.0 | 255.255.255.0 | 0.0.0.0 | 192.168.15.1 | | 232.25.201.0 | 255.255.255.0 | 0.0.0.0 | 232.25.201.1 |   Interpreting a row, given a frame with a destination IP address:  - Router identifies anything that says it is a 192.168.15 network with a whack 24 subnet  - Gateway 0.0.0.0 means that router is directly connected to the network  - Router can ARP that system directly (i.e. at address)  - Will send out through the interface (whatever it might be) | | | | | | | |
| Default route | | - Built into a router (in the routing table)  - Tells of what is the IP address of a upstream router  - If destination IP address does not meet any criteria in the routing table, router does not know where to send, then by default, send it to the default route | | | | | |  |  | | --- | --- | | Address | 0.0.0.0 | | Subnet | 0.0.0.0 | | Gateway | 232.25.201.11 | | Interface | 232.25.201.1 |   Example:  - Does not care what the address or subnet is  - Since not directly connected to gateway (0.0.0.0), router will ARP gateway for MAC address of upstream router  - Router sends frame to upstream router | | |
| - When there are more than two default routes (i.e., multiple ISPs connected to router), routing table has an extra table “*Metric*”  - “*Metric*”: a relative value that gives router an idea which destination address to ultimately connect and send information to  Example:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Address | Subnet | Gateway | Interface | Metric | | 192.168.15.0 | 255.255.255.0 | 0.0.0.0 | 192.168.15.1 | 100 | | 232.25.201.0 | 255.255.255.0 | 0.0.0.0 | 232.25.201.1 | 100 | | 0.0.0.0 | 0.0.0.0 | 232.25.201.11 | 232.25.201.1 | 10 | | 0.0.0.0 | 0.0.0.0 | 75.29.6.1 | 75.29.6.144 | 11 |   How it works:  - Default gateway (ISP) with lower metric has higher priority.  - If ISP connection goes down, router automatically switches over to secondary default route | | | | | | | |
| Static routes | | - A fixed route that is manually configured and persistent  - Routing tables contain address information for destination, subnet mask, gateway and NIC  - More common on internal networks  Example:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Network dest | Netmask | Gateway | Interface | Metric | | 0.0.0.0 | 0.0.0.0 | 192.168.1.1 | 192.168.15.1 | 100 | | 127.0.0.0 | 255.0.0.0 | On-link | 232.25.201.1 | 100 | | 127.255.255.255 | 255.255.255.255 | On-link | 232.25.201.1 | 10 | | 255.255.255.255 | 255.255.255.255 | On-link | 75.29.6.144 | 11 |  |  |  | | --- | --- | | Interface | Metric | | 192.168.1.13 | 50 | | 127.0.0.1 | 331 | | 127.0.0.1 | 331 | | 192.168.1.13 | 306 |   How to read:  - First row: Any address, any subnet mask - send it out to the default gateway through the network card UNLESS  - Second row: any address starting with 127 with subnet mark starting with 255 - send it *locally* (i.e. loopback)  - Etc … | | | | | # View routing table (display current known routes)  route print  netstat -r | | |
| Dynamic routes | | - Metric value is an arbitrary value  - Different dynamic protocols use metric values in different ways to decide the final destination  - Two kinds of metrics - distance vector or link state  **- Distance vector**  - Uses hop count  - Sends data in intervals  - If router down, need to wait a duration (depending on the protocol) before going back to a state of convergence  **- Link state**  - More modern  - Uses advertising  - When router down, affected routers dynamically notify other routers of change in routing table and asks them if they want to update theirs  - *Convergence:* where all router tables reflect all routes  **Dynamic routing protocols**  - Internal and external gateway protocols (i.e. IGP, EGP)  - Autonomous system (AS) communicate outside of their AS  - Border Gateway Protocol (BGP)  - The only EGP protocol used for Inter-Autonomouse System routing  - Assigned an AS number to use within routers  - To use AS numbers (ASN) to send data between routers | | | | | **Possible metric evaluations**  - *Hop count*: the number of routers it took to get to a particular network ID  - *Maximum transmission loss (MTU)*: how much data there is in a particular frame  - E.g. Ethernet has MTU size of 1500 bytes  - *Bandwidth*  - *Cost*  - *Latency* | | |
| Home routers | | - All home routers have a default IP address, username and password  - Almost all home routers are DHCP servers  - Router WAN connections are commonly DHCP clients by default  - Typically Small Office/Home Office (SOHO) routers  - How to do a hard reset (30-30-30 rule)  1) Unplug all cables (except power source)  2) Hold down reset button on router for 30 seconds  3) While still holding down reset button, unplug power source for 30 seconds  4) While still holding down reset button, plug in power source and keep button held down for another 30 seconds  - All configurations set back to default | | | | |  | | |
| SOHO Routers | | - Are for small groupos (5-6 devices) and can have built-in capability for switches, firewalls and WAPs  - Often have web-based interfaces | | | | |  | | |
| Enterpise Routers | | - Have expanded connection capability to other devices (i.e. routers, switches, WAPs)  - Typically have their own OS interfaces  - E.g. Need to putty in | | | | |  | | |
| **Network Address Translation (NAT)** | | - Built into routers  - Allows many more devices on the internet without an legitimate IP address  - Translates internal IP addresses to an Internet address and tracks the packets  **Static NAT/Port Address Translation (SNAT/PAT)**  - All incoming addresses for one particular IP address goes to one particular device  - Sends specific traffic to one internal IP address  **Dynamic NAT/Pooled NAT (DNAT)**  - Fixed number of IP addresses shared among devices in the same LAN  - If number of IP addresses < number of devices requesting to connect beyond LAN (e.g. internet), surplus demand will have to wait until an IP address is no longer in use  - Has a limited pool of internet addresses to give to a number of internal devices | | | | |  | | |
| **Implementing NAT**  - Given a scenario, configure the appropriate IP addressing components  - Most home routers have NAT turned on, by default  - SOHO routers ship with NAT enabled  - Most industrial routers have NAT turned off, by default  - NAT on a SOHO router can be disabled from the router’s configuration page  - Some older routers call this setting gateway/router mode | | | | |  | | |
| **Ports** | | | | | | | | | |
| Port numbers | | - Destination and source port numbers will always be included in a TCP packet (2 port no./packet)  - Well known ports (no. 0 to 1023) have fixed applications, pretty much set in stone  - Source port number (1024 to 65535) is incrementally generated by computer and is transient (lasts a short time)  - Depending on OS, source port number can be up to 65 000  - When returning a response, source/destination IP addresses and port numbers are reversed | | | | |  | | |
| Port forwarding | | - Port forwarding allows external devices to have internal communication through a router  - NAT router needs to be capable enough to handle  - Use a unique port for security  - Example:  1) WAN: 1.1.1.1  2) Camera: 192.168.5.13 (listens on port 80)  - Set up port forwarding from port 8181 to port 80  - Access camera from internet by entering WAN:port (i.e. 1.1.1.1:80) | | | | | **Dest port no.** | |  |
| 80 | | web client |
| 21 | | FTP server |
| 110 | | email server |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
| **Port Range Forwarding**  - Port forwarding but for a range of ports  - Example:  - Setting up server of games, uses a lot of different ports  **Port Triggering**  - Will open an alternative assigned port when the initial port is contacted (e.g. FTP)  - Type of forwarding ports but for a different reason  - Set triggered port number/range, and forwarded port number/range  - For any request going out these triggered ports, allow response to come back through specified forwarded ports (i.e. data go out port 20, response can come back through other ports, not just 20)  - Example:  - For File Transfer Protocol (FTP) to work  **DMZ (SOHO DMZ)**  - Enabling DMZ when setting up port forwarding places that device outside the protection of that router | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |
|  | |  | | | | |  | | |