Address Resolution

- Address resolution problem
  
The address resolution problem is to map high-level addresses (e.g., IPv4 addresses) to physical addresses. There are two basic types of physical addresses, one type has large fixed addresses (like those in Ethernet) and the other has small reconfigurable addresses (like those in proNET).

  For small reconfigurable addresses, the resolution can be done by direct mapping: we choose a mapping function $f$ and the physical address is computed by $P_A = f(I_A)$, where $P_A$ is the physical address and $I_A$ is the IP address of host $A$.

  For the large fixed physical addresses, the resolution in IPv4 is realized by dynamic binding. Address Resolution Protocol (ARP) provides a mechanism to realize the resolution.

  In IPv6, the fixed physical address is embedded in the interface identifier field of an IPv6 address, no resolution is needed.

- Address Resolution Protocol (ARP)

  ARP has two major functions. One function is to map an IPv4 address to a physical address for sending IPv4 packets. The other function is to reply ARP requests from other hosts. ARP is a low-level protocol which binds the physical addresses and IPv4 addresses dynamically for the network with broadcast capability. The idea used in ARP is as follows: When host $A$ wants to find the physical address $P_B$ of host $B$ with IPv4 address $I_B$, $A$ broadcasts a special packet called ARP request which asks the host with IPv4 address $I_B$ to respond with its physical address $P_B$.

  All hosts in the network receive the packet but only $B$ recognizes $I_B$ and replies with a packet called ARP reply containing $P_B$.

  Actually, $A$ can send data to $B$ by simply broadcasting the data. However, there is a big difference on broadcasting and sending data with a specific destination address. For the broadcast message, each host must process it, while for the message with a specific destination, the message will be ignored at network interface of a host which is not the destination.

  For efficiency, hosts using ARP maintain a cache of IPv4-to-physical address bindings. The host first looks up the cache to resolve the address problem. The ARP cache uses a soft-state technique which sets-up a timer (usually 20 mins.) for each IPv4-to-physical address binding. When the timer for a binding expires (called ARP cache timeout) the binding in the cache is deleted.

  To reduce the broadcast, when a source sends an ARP request for the IPv4-to-physical address binding, it also includes the IPv4-to-physical address binding for itself in the ARP request. Hosts receiving the ARP request can pick-up the IPv4-to-physical address binding for the source.

  ARP message is carried by data frame. To identify the Ethernet data frame which is carrying an ARP message, the type field in the frame header is set as $0806_{16}$ (in hexadecimal).

  ARP can be used for testing duplicated IPv4 addresses. For this purpose, a host sends an ARP request for the IPv4 address assigned to itself. If the host receives the reply from a different machine then the IPv4 address has been assigned to at least two machines and a duplication is detected.

- ARP message format
The message used in Ethernet has the following fields.

Hardware type: specifies a hardware interface type for which the sender seeks an answer; 1 for Ethernet.

Protocol type: specifies the protocol type of the address the sender provides; 0800\textsubscript{16} for IPv4.

HLEN: specifies the length of hardware address.

PLEN: specifies the length of protocol address.

Operation: specifies ARP request (1), ARP response (2), RARP request (3), and RARP response (4).

Sender hardware address
Sender IPv4 address
Target hardware address
Target IPv4 address

- **Proxy ARP**
  
  This is a method to allow multiple networks to share a single IPv4 network address. Here is an example: Two networks $A$ and $B$ are connected by a router $R$ and are assigned the same network address. If a host $s$ in $A$ wants to send a message to a host $t$ in $B$, it uses ARP to find $t$’s hardware address in $A$. Router $R$ receives this ARP request and knows that $t$ is in $B$ not in $A$. $R$ responses to $s$ with its own hardware address. Then $s$ sends the message with $t$’s IPv4 address but $R$’s hardware address. $R$ forwards this message to $t$ in $B$.

- **Find internet address at start-up**
  
  Usually, the IPv4 address of a computer is kept on its secondary storage where the OS finds it at startup. However, some computer may need to find its IPv4 address from a remote file server. The Reverse Address Resolution Protocol (RARP) is the protocol in TCP/IP for this purpose. To find the IPv4 address, a host broadcast an RARP message, specifying itself as the target, the server responses with the IPv4 address for the sender. RARP uses the same message format as ARP. The type field of the data frame which carries the RARP message is set to 8035\textsubscript{16}.

- **IPv6 neighbor discovery**
  
  In IPv6, a direct mapping or a Neighbor Discovery Protocol (NDP) (replacing ARP) is used to map an IPv6 address to a physical address. NDP also includes the following functions: find all routers in a network; decide if a specific network connection is reachable; get the prefix of a network; find the maximum packet size of a network, etc. Functions of NDP are realized by ICMPv6 (Internet Control Message Protocol)

  ARP is reactive, finding physical address when needed, while NDP is proactive, finding physical address at start-up.

**Bootstrap and Autoconfiguration**

- **Client-server model**
  
  The term *server* applies to any program which provides a service that can be reached over a network. The term *client* is a program which sends a request to a server and waits for a response. Client-server interaction forms the basis of most network communications.
• **BOOTstrap protocol (BOOTP)**

Bootstrap protocol (BOOTP) is a protocol used to find the IP address. It is developed to overcome some drawbacks of RARP and uses UDP message based on the client-server model.

BOOTP uses IP to determine an IP address. To realize this, a client can use the limited broadcast IPv4 address (255.255.255.255) to broadcast a UDP datagram on a local network before IP has found the IP address of the network or the client’s IP address. The server, on receiving the request of the client, broadcasts its answer to the request on the network and the client will get it.

To ensure the reliable communication, BOOTP requires UDP use checksum, does not allow fragmentation, uses timeout and retransmission to handle datagram loss.

• **Two-step bootstrap**

The boot process uses a two-step operation. In the 1st step, the client sends a BOOTP message and the server replies with a BOOTP message providing client the IP address and location of the boot file. In the 2nd step, the client uses a protocol tftp (or a similar protocol) to download the boot file.

• **Dynamic host configuration protocol (DHCP)**

BOOTP is designed for relatively static environment in which each host has a permanent network connection. BOOTP only provides a static mapping from a host identifier to parameters for the host but does not include a way to dynamically assign values to individual machines. Static parameter assignment works well if computers have fixed locations and the network manager has enough IP addresses to assign each computer a unique IP address.

Dynamic Host Configuration Protocol (DHCP) is developed to handle automated address assignment. DHCP extends BOOTP in two ways. First, DHCP allows a computer to get all the configuration information in one message. Second, DHCP allows a computer to get an IP address dynamically. By these extension, DHCP has three types of address assignment.

The manual configuration used in BOOTP.

The automatic configuration assigns a permanent IP address when a computer first connected to the network.

The dynamic configuration leases an IP address to a computer for a limited time.

• **Dynamic IP address assignment**

There is a database similar to that used in static configuration. But an IP address is not permanently tied to a client. An IP address is selected from a pool and assigned to a client for a limited time. If the time is infinite, it becomes automatic configuration.

• **DHCP message format**

DHCP messages have fixed-length fields, and replies have the same format as requests. The following fields are included in the BOOTP message.

OP: specifies the message is a request (with value 1) or a reply (with value 2).

HTYPE: the network hardware type (Ethernet has type 1).

HLEN: the length of the hardware address in bytes (Ethernet address has length 6).
HOPS: the number of hops the message can travel. Client sets the field to 0 and a DHCP server can increase it if there is a necessary to pass the request to another network.

Transaction id: used by client to match response to request.

Seconds: the time (in seconds) since the client sent its first bootrequest.

Flag: the most significant bit is used to indicate if the message should be sent by broadcast (set to 1) or unicast (set to 0).

Client IP address: set to 0.0.0.0 if the client is willing to accept any IP address from the server. The client can also specify an IP address here.

Your IP address: the IP address assigned by the server for the client. It can be different from client IP address.

Server IP address: the IP address of boot file server which will be used by the client to download the boot file.

Router IP address: (with HOPS field) provide a way for a client to boot from a server in another network.

Client hardware address: the hardware address of the client.

Server host name: the name of the BOOTP server; this can be used by the client to specify server.

Boot file name: used by client to specify boot file.

Options: provide optional information including subnet mask, static routing information, client’s host name, DNS and NIS (Network Information Service) domain names and addresses, etc. Items in the options area use a Type-Length-Value style encoding, each item has a type byte, a length byte, and a value field with the length specified by length field. The option field is used to identified different types of DHCP messages used for address lease process. The address leasing and those messages will be discussed later.

- **Address acquisition states**

To use DHCP, a host becomes a client by broadcasting a message to all servers on the local network. The host then collects offers from servers, selects one of the offers, and verifies acceptance with the server. The process of using DHCP can be described by a finite automaton with the following six states.

Initialize state: The boot process starts from here. The client broadcasts a DHCPDISCOVER message and moves to the select state.

Select state: The client waits for the offers from servers. After it receives the offers, it selects one, sends the server a DHCPREQUEST message, and moves to the request state.

Request state: The client waits for the message DHCPACK. When it receives the message it moves to the bound state.

Bound state: This is the normal state and the client remains in this state while it uses the IP address. If the client decides to release the IP address to the server, it sends a DHCPRELEASE message to the server and moves to the initialize state. If the client wants to renew its lease, it sends a DHCPREQUEST message to the server and moves to the renew state.

Renew state: The client waits for the DHCPACK message from the server. If it receives the message, it moves to the bound state. If the lease is closing to expiration, the client sends the DHCPREQUEST message and moves to the rebind state.
Rebind state: If the client receives the DHCPACK, it moves to the bound state. If the client receives a DHCPNAK message or the lease expires, it moves to the initialize state.

- **Managed and unmanaged configuration**
  
  A managed network system is configured by network administrators. In such a system, there is usually a server machine for the network. When a client computer joins the network, the client gets an IP address from the server. DHCP is widely used in managed networks for IP address assignment.

  An unmanaged network system does not require the configuration by administrators. When a computer joins the network, the computer randomly selects an IP address and confirms if the selected address has any conflicts with the computers already connected to the network. If no conflicts, the computer uses the selected IP address to join the network, otherwise the computer selects another address until a conflict-free address is found.

- **Configuration for IPv6**

  In IPv6, DHCPv6 is used for managed network systems and stateless autoconfiguration is used for unmanaged network systems. DHCPv6 is similar to DHCP for IPv4 but has more functions and is more complex. The stateless autoconfiguration relies on the IPv6 Neighbor Discovery Protocol (NDP).

- **IPv6 Neighbor Discovery Protocol**

  IPv6 NDP operates at Layer 3 and has the following major functions:

  - Router discovery, for a host to find the routers on a specific link.
  - Next-hop routers, for a host to find the next-hop router for a destination.
  - Neighbor unreachable detection (NUD), for a host to find a neighbor who becomes unreachable.
  - Address prefix discovery, for a host to learn the network prefix(es) used on a link.
  - Configuration parameter discovery, for a host to find parameters such as MTU used on a link.
  - Staleless autoconfiguration, for a host generate an IP address used on a link.
  - Duplicate address detection (DAD), for a host to find whether a generated address is already in use.
  - Address resolution, for a host to map an IPv6 address to an MAC address.
  - DNS (Domain Name System) server discovery, for a host to find the DNS servers on a link.
  - Redirect, for a router to inform a host on a preferred first-hop router.

  The above functions are realized by (combinations) of five ICMPv6 messages: router solicitation, router advertisement, neighbor solicitation, neighbor advertisement, and redirect.

  ICMPv6 is developed for IPv6. The functions of ICMP, ARP, and IGMP are included in ICMPv6. RARP is dropped in the TCP/IP protocol suite with IPv6. The function of RARP is realized by BOOTP (DHCP). The messages of ICMPv6 can be classified into two groups: the messages for error reporting and the messages for query. All ICMPv6 messages have a same general format:

  - Type field (one byte), define the message types.
  - Code field (one byte), further define sub-types of a message.
– Checksum (two bytes)
– Other information (four bytes).
– Data (variable length).

Router solicitation and advertisement messages: type=133 for solicitation, type=134 for advertisement, and code=0.

Neighbor solicitation and advertisement messages: type=135 for solicitation, type=136 for advertisement, and code=0.

Redirection message: type=137 and code=0.

**Domain Name System (DNS)**

• **Name vs IP address for a host**

It is better to refer a machine by a name instead of an IP address, e.g., it is easier to use `seasons` than `172.16.1.1` for the server in our Networking Lab. Domain name system is a scheme for

1. assigning meaningful high-level names for machines and
2. mapping between the names and the IP addresses of the machines. The scheme handles both the translation from names to IP addresses and from IP addresses to names.

• **Naming a machine**

When the Internet was small, the name-address business was handled by a central site, the Network Information Center (NIC). When the Internet grew and became large, a distributed system was developed for the name-address mapping. The system partitions the name space to give hierarchical machine names and delegate authorities over the partitions.

In a TCP/IP internet, hierarchical machine names are assigned based on the structure of organizations that have the authority for parts of the namespace, not necessarily based on the structure of the physical network interconnection.

The Domain Name System (DNS) is the mechanism which implements a machine name hierarchy for a TCP/IP internet. DNS specifies the name syntax and rules for delegating authorities over names and the implementation of a distributed computing system that maps the names to addresses.

• **Domain name space**

To have a hierarchical name space, a *domain name space* is designed. In the domain name space, the names for machines are defined in an inverse-tree structure with the root at the top. The tree can have 128 levels at most, from level 0 (the root) to level 127. The root glues the whole tree together. Each level of the tree defines a hierarchical level.

Each node in the tree has a *label* which is a string of characters (at most 63). The label is called a *domain name*. The root has a null label.

A machine is assigned a *full domain name* which is a sequence of domain names separated by dots (·). The domain names are always read from the node up to the root.

A *domain* is a subtree of the domain name space. The name of the domain is the domain name of the root of the subtree.
• **Internet domain names**

Geographic domains: The top-level domains are grouped by countries. Each country is given a 2-letter top-level domain name. For example, machines in Canada fall in the top-level domain `ca`.

Organizational domains: The top-level domains are grouped by organizational type. For example `com` for commercial organizations, `edu` for educational institutions (4 years), `gov` for government institutions, etc.

Inverse domain: The inverse domain is used to map an address to a name. The 1st level node in the domain has label `arpa` and the 2nd level node has label `in-addr`. The rest of the domain defines the IP addresses. For example, the IP address `142.58.101.25` is defined by the domain `25.101.58.142.in-addr.arpa`.

• **Map domain name to address**

In addition to the rules for assigning domain names to machines, the domain name scheme includes mapping names to addresses. To do so, the information in the domain name space should be stored. But it is inefficient and unreliable to store the information at a single site (machine). The solution for these problems is to distribute the information among many machines called DNS (or name) servers which provide name-to-address translation, mapping domain names to IP addresses. Each name server can be responsible for a specific domain (subtree). A client uses a name resolver (a client program) which uses one or more name servers to find the name-to-address mapping.

A DNS domain is a complete subtree in the DNS name tree. A zone is a domain managed by one server (one authority), minus any delegated subtrees. The server for a zone keeps a database called *zone file* to store all the information in the zone and some partial information on the delegated subdomains. An administrative authority may have control of several zones, e.g., a zone for the domain the authority controls and a zone for the corresponding `.in-addr.arpa` domain.

A root server is a server whose zones cover the whole DNS name tree. A root server usually does not keep any information about domains but delegates its authority to other servers and keeps references to those servers.

A primary server is a server which is responsible for creating, maintaining, and updating the zone file.

A secondary server is a server which transfers the complete information about a zone from another server. It neither creates nor updates the zone files.

• **Domain name resolution**

Domain name resolution includes mapping names to IP addresses and mapping IP addresses to names. The DNS is designed as a client-server application. A host which needs a name resolution calls the DNS client (resolver). The resolver sends the query to the nearest name server.

The resolution can be recursive and iterative. In *recursive resolution*, if the server has the answer, it replies with the answer. Otherwise, it forwards the query to another server (usually the parent) and waits for response.

In *iterative resolution*, if the server has the answer, it replies with the answer. Otherwise, it returns the client an IP address of a server that it thinks can resolve the query.

DNS requires each name server knows its parent server and at least one root server.
• Efficiency

To make the name resolution efficient, some mapping information are kept in cache memory. Each mapping in cache is associated with a time-to-live TTL timer to prevent the outdated mappings. If TTL expires then the mapping is deleted.

• DNS messages

DNS has two types of messages: query and response. The query message has a header and the question records; the response message has a header, question records, answer records, authoritative records, and additional records. Query and response messages have the same header format which contains the following fields:
Identification, used by client to match the query and response.
Flags (parameters), specifies the operation requested and response code.
Number of questions, number of queries in question section.
Number of answers, number of answers in answer section.
Number of authority, number of authority records in authority section.
Number of additional, number of records in additional information section.
Question section, queries.
Answer section, answers.
Authority section, gives information (domain name) on authority servers for the query.
Additional information section.

• Abbreviation of domain names

Names can be made short by abbreviation if the name resolving process can provide part of the name automatically. In general, the abbreviation is implemented with a domain suffix list. When a resolver queries a name, it appends a suffix in the list to the name for resolution.

• Dynamic DNS

It is an extension of DNS. A dynamic DNS protocol allows a DNS server to accept requests for adding, updating, and deleting entries in the database dynamically. The request is usually sent by DHCP. To guarantee the security of the system, authentication (e.g., using digital signature) for requests may be required.

• Configuring DNS

DNS is usually implemented by a client/server software system called BIND (Berkley Internet Name Daemon). The client side of BIND is called a resolver which generates queries and send them to server. The server side of BIND is a daemon called named which answers the queries.
The basic BIND configuration includes: Configuring the BIND resolver; configuring the BIND name server; and configuring the name server files, called zone files. In Linux:
The resolver is configured in the /etc/resolv.conf file.
The name server is configured by a number of files:

– Configuration file, sets parameters and points to the sources of DNS databases used by this server. This file is usually called named.conf and resides under /etc directory.
- **Root hints file**, points to the root servers.
- **Localhost file**, used to locally resolve the loopback address.
- **Forward-mapping zone file**, maps hostnames to IP addresses.
- **Reverse-mapping zone file**, maps IP addresses to names.

- **DNS security extensions (DNSSEC)**
  To protect a user from being fooled into believing an impostor web site or revealing confidential information, DNS Security (DNSSEC) was developed. The primary services of DNSSEC include **authentication** and **data integrity**. Using DNSSEC, a user can verify a DNS message did come from the server responsible for the name in the query and the message was not changed by a third party during the transmission. Digital signature is used for authentication and data integrity in DNSSEC. Public-key encryption scheme is used for digital signature. More details on encryption and digital signature will be discussed in Chapters on network security.

- **DHCP and domain names**
  Three possibilities. The 1st one is that the client does not receive a host name. The 2nd one is that a host name is permanently associated with an IP address. In this way, a client may have different host names at different times and locations. The 3rd one is to assign a client a permanent host name. But this approach requires the cooperations between DHCP and the domain name server, because one host name may be associated to different IP addresses depending on the time and location the client attaches to the network.