75Dineshchandra / Google-IT-Support-Professional-Cert-Notes Public
forked from Clifton893/Google-IT-Support-Professional-Cert-Notes

Code Pull requests Actions Projects Wiki Security Insights Settings

main Google-IT-Support-Professional-Cert-Notes / Course-2 / c2-week-4.md Go to file ...

Clifton893 Update c2-week-4.md Latest commit d263a78 on Nov 28, 2020 History

1 contributor

Week 4: Networking Services

The main purpose of computer networking is so network services can be available to answer requests for the data from clients.

Name Resolution

Why do we need DNS?

Remembering every single website by its IP address would be impossible for most people. Since humans tend to remember words better than numbers, the domain name system was created to fix this.

Domain Name System (DNS) is a global and highly-distributed network service, that resolves strings of letters into IP addresses for you.

- The IP address for a domain name can change all the time, for many different reasons.
- A **domain name** is the term we use for something that can be resolved by DNS.
- By using DNS, an organization can just change what IP a domain name resolves to, and the end user would never even know.
  - So not only does DNS make it easier for humans to remember sites; it allows admin changes to happen behind the scenes (without the user knowing).

DNS also helps global organizations resolving domains to local data centers (which use different IPs).

The many Steps of Name Resolution

DNS converts domain names into IP addresses.

- It's a system based on the way humans think of things, resolved into the way computers think of things.
- The process of turning a domain name into an IP address is known as *domain resolution.*
- DNS servers need to be specifically configured at a node on a network.

For a computer to operate on a modern network, they need to have a certain number of things configured:

1. IP address
2. Subnet mask
3. Gateway for a host
4. DNS server

- Technically, a computer can *operate* without DNS or a DNS server; but it wouldn't be *usable* for a human.

Five Primary Types of DNS Servers

1. Caching name servers
2. Recursive name servers
3. Root name servers
4. TLD name servers
5. Authoritative name servers

Note: Any given DNS server can fulfill many of these roles at once.

Caching name servers

- Are generally provided by an ISP, or your local network

- Their purpose is to store known domain name lookups for a certain amount of time
  - This prevents the lengthy domain name lookup process from happening upon each TCP connection
- Most caching name servers are also recursive name servers.

Recursive name servers

- Perform full DNS resolution requests

DNS Time to live (TTL)

A value, in seconds, that can be configured by the owner of a domain name for how long a name server is allowed to cache an entry before it should discard it and perform a full resolution again.

- A length TTL means it can take up to the length of a total TTL for a change in DNS record to be known to the entire Internet.
- Short answer: *TTL stands for "Time to Live" and determines how long a DNS entry can be cached.*

A full recursive resolution

1. The first step is [for a user -> caching/recursive name server] always to contact a root name server.

- There are 13 total root name servers; originally actual physical servers, but now lookup authorities distributed across the globe via *anycast*.

2. The root servers will respond to a DNS lookup with the TLD name server that should be queried.
3. The TLD name servers will respond with a redirect, informing the computer performing the name lookup with which authoritative name server to contact.

- Authoritative name servers are responsible for the last two parts of any domain name (which is the resolution at which a single organization may be responsible for a DNS lookup).
- Example: For `www.weather.com` , the TLD name server would point a lookup at the authoritative server for `Weather.com` , which would likely be controlled by The Weather Channel, the organization itself that runs the site.
  - Finally, the DNS lookup could be redirected at the authoritative server for `weather.com` , which would finally provide the actual IP of the server in question.

This strict hierarchy is critical to the stability of the Internet, making sure that all full DNS resolutions go through a strictly-regulated and controlled series of lookups to get the correct responses.

- It's also the best way to protect against malicious parties redirecting traffic.

*__Top level domain (TLD)__ is the last part of any domain name (like `.com`), and represents the top of the hierarchical DNS name resolution system.

- For each TLD in existence, there's a TLD name server.
  - This doesn't mean one physical server; but instead probably a global distribution of anycast-accessible servers responsible for each TLD.

Anycast is a technique that's used to route traffic to different destinations depending on factors like location, congestion, or link health.

- Using any cast, a computer can send a datagram to a specific IP, but could see it routed to one of many different actual destinations (depending on a few factors).

DNS and UDP

DNS is an *application layer* service, that uses UDP for the transport layer instead of TCP.

A single DNS request and its response can usually fit inside of a single UDP datagram, making it an ideal candidate for a connectionless protocol.

Using TCP would bog down the resolution process with nearly four times as much traffic.

- DNS listens on `port 53` .

DNS over TCP does exist, and is in use.

- With the Web growing more complex, sometimes a DNS lookup response can't fit within a single UDP datagram.
- In this case, a DNS name server would respond with a packet, explaining the response is too large.
- The DNS client would then establish a TCP connection to perform the lookup.

Since UDP doesn't have the concept of error recovery, the DNS resolver simply asks again if it doesn't get a response.

Name Resolution in Practice

Resource Record Types

DNS operates with a set of defined resource record types. They allow for different kinds of DNS resolutions to take place. (There are many, but here are a few of the most basic ones.)

A Record

- An **A record** is used to point a certain domain name at a certain IPv4 IP address.
  - At its most basic use, a single A record is configured for a single domain name.
  - A single domain name can have multiple A records, too.
    - This allows for *DNS round robin* to balance traffic across multiple IPs.

Round robin is a concept that involves iterating over a list of items, one by one, in an orderly fashion.

- The goal is to ensure an equal balance of each entry on the list that's selected.

- The **AAAA (Quad A)** is an emerging record type that's becoming more popular.

  - It's different from an A record in that it returns an IPv6 IP address.

CNAME Record

- A **CNAME record** is used to redirect traffic from one domain name to another.
- CNAMEs are useful, because they ensure you only have to change the canonical IP address of a server in one place.
  - CNAME is just shorthand for *canonical name*.
- An example of a CNAME is `microsoft.com` pointing to `www.microsoft.com`

MX Record

- **MX** stands for *mail exchange*.
- This resource record is used to deliver email to the correct server.
- Company web and mail servers often run on different machines (with different IPs);
  - The MX record ensures that email gets delivered to a company's mail server, while web traffic gets delivered to the web server.

SRV Record

- **SRV** stands for *service record*.
- It's used to define the location of various specific services.
- It serves the exact same purpose as the MX record;
  - But while the MX is only for mail services, the SRV can return the specifics of many different service types.

TXT Record

- **TXT** stands for *text*.
- It was originally intended for providing descriptive text for human consumption.
- Now, it's used to convey additional data intended for other computers to process.
  - It is free form, so clever engineers can use it to communicate data not originally intended to be communicated by a system like DNS.
- It's often used to communicate configuration preferences about network services, that you've entrusted other organizations to handle for your domain.

Anatomy of a Domain Name

Any given domain name has 3 primary parts, and they all serve specific purposes.

Top Level Domain

- The **top level domain (TLD)** is the last part of a domain name--or top level domain.
- Administration and definition of TLDs is handled by the **Internet Corporation for Assigned Names and Numbers (ICANN)**.
  - ICANN is a sister organization to the IANA.
  - Together, they help define and control both the global IP spaces, along with the global DNS system.

Domain

- The **domain** is the second (or middle) part of a domain name.
- Domains are used to demarcate where control moves from a TLD name server, to an authoritative name server.

Subdomain

- The first (such as "www") portion is known as the **subdomain**.
- It's also known as the *host name*, if it's been assigned to only one host.
- Subdomains can be freely chosen and assigned by anyone who controls such a registered domain.

Fully Qualified Domain Name

- Combined, the subdomain, domain, and top level domain form a **fully qualified domain name (FQDN)**.
- DNS can support up to 127 levels of domain in total for a single fully qualified domain name.
- Each individual part of the domain name (sub, domain, top) can only be 63 characters long.
- A complete FQDN is limited to a total of 255 characters.

DNS Zones

Authoritative name servers are responsible for responding to name resolution requests for specific domains; but they do more than that.

An authoritative name server is actually responsible for a specific DNS zone.

DNS zones are a hierarchical concept.

- The root name servers are responsible for the root zone;
- Each TLD name server is responsible for the zone covering its specific TLD;
- Authoritative name servers are responsible for fine-grained zones under that.

The root and TLD name servers are actually just authoritative name servers, too.

Zone's *don't* overlap.

The purpose of **DNS zones** is to allow for easier control over multiple levels of a domain.

- Zones are configured through **zone files**
  - These are simple configuration files, that declare all resource records for a particular zone.
- A zone file has to contain a **start of authority (SOA)** resource declaration.
  - This declares the zone and the name of the name server that is authoritative for it.
- Also often found is **NS records**, which indicate other name servers that might also be responsible for this zone.
- You'll also find A, AAAA, and CNAME records; along with default TTL values for the records served by this zone.
- Zones can be configured to go many layers deep; but do have a limit.
- You may also see **reverse lookup zone files**.
  - These let DNS resolvers ask for an IP and get the FQDN associated with it returned.
  - In these files, you'll find mostly pointer resource record declarations.
- A **pointer resource record (PTR)** resolves an IP to a name.

Dynamic Host Configuration Protocol

Overview of DHCP

Every single computer on a modern TCP/IP network needs to have at least four things configured:

1. IP address
2. Subnet mask
3. Primary gateway
4. Name server

- The subnet mask, primary gateway, and DNS server are likely the same on every node on the network.
  - But an IP address must be different for every single node on the network.

To handle the tricky configuration of these unique IP addresses, DHCP comes into play.

Dynamic Host Configuration Protocol (DHCP)

- DHCP is an application layer protocol that automates the configuration process of hosts on a network.
- Through DHCP, a machine can query a DHCP server when the computer connects to the network, and receive all the networking configuration in one go.
- DHCP reduces administrative overhead;
- It also solves the problem of having to choose what IP to assign to what machine.
  - This can help if local DNS servers are malfunctioning, since network administrators would still need to connect to devices (like routers) through IP address.

There are several standard ways DHCP can operate:

Dynamic Allocation

- This is the most common way.
- A range of IP addresses is set aside for client devices and one of these IPs is issued to these devices when they request one.
- Under dynamic allocation, the IP of a computer could be different almost every time it connects to the network.

Automatic Allocation

- Works similar to dynamic allocation (a range of IP addresses is set aside for assignment purposes).
- Big difference is the DHCP server is asked to keep track of which IPs it's assigned to certain devices in the past.
    - Using that info, the DHCP sever will assign the same IP to the same machine each time, if possible.

Fixed Allocation

- Requires a manually-specified list of MAC address and their corresponding IPs.
- When a computer requests an IP, the DHCP server looks for its MAC address in a table and assigns the IP that corresponds to that MAC address.
- If the MAC address isn't found, the DHCP server could fall back to another allocation method; or refuse to assign an IP altogether.
    - This can be used as a security measure, to ensure that only devices with configured MAC addresses will be able to communicate on the network.

You can also use DCHP to assign NTP servers.

- **Network time protocol (NTP) servers** are used to keep all computers on a network synchronized in time.

DHCP in Action

The entire point of DHCP is to configure the network layer itself. This is despite DHCP being an application layer protocol, atop the TCP/IP hierarchy.

The process by which a client configured to use DHCP attempts to get network configuration information, is known as **DHCP discovery**.

DHCP Discovery

This process has four steps:

1. **Server discovery**

- The DHCP client sends a *DHCPDISCOVER* message out onto the network;
- This is a specially-crafted broadcast message, since the client lacks an IP address and doesn't know the IP address of the server.
- DHCP listens on UDP `port 67` , and DHCP discovery messages are always sent from UDP `port 68` .
- The message is encapsulated inside an IP datagram with a destination IP of `255.255.255.255` , and a source IP of `0.0.0.0`
- The message is broadcast to every node on the local area network.
- When the DHCP server receives the message, it examines its own configuration, and decides which (if any) IP address to offer to the client.
    - The response is sent as a **DHCPOFFER** message with a destination port of `68` , a source port of `67` , a destination broadcast IP of `255.255.255.255` , and its actual IP as the source.

2. Offer

- The DCHP offer is broadcast to every machine on the network; but the original client recognizes the message is for itself.
    - This is because DHCPOFFER specifies the MAC address of the client that sent the DHCPDISCOVER message.
- The client processes the offer to see what IP is being offered to it.
- In theory, a client could reject this offer;
    - Multiple DHCP servers could be running on the same network, and a client could be configured to only respond to an offer of an IP within a certain range.

3. Request

- But usually, the client responds to the offer with a **DHCPREQUEST** message.
- This is again sent from an IP of `0.0.0.0` to `255.255.255.255` (since the client still doesn't have an assigned IP).

4. Acknowledgement

- The DHCP server responds with a **DHCPACK** acknowledgement message.
    - The client recognizes the message by its MAC address in one of the message fields.
- Now the client computer's networking stack can use the configuration information presented to it by the DHCP server, to set up its own network layer configuration.
- At this point, the client should have all the info it needs to operate in a fully-fledged manner on the network it's connected too.

All of this configuration is known as the **DHCP lease**, as it includes an expiration time.

- This could last for days, or a short amount of time.
- Once a lease has expired, the DHCP client would need to negotiate a new lease by repeating the entire DHCP discovery process all over again.
- A client can also release its lease to the DHCP server, which it would do when it disconnects from the network.
    - This would also the DHCP server to return the IP address assigned back into the pool of available IPs to lease out.

Network Address Translation

NAT is not a protocol, but a technique.

Basics of NAT

A technology that allows a gateway--usually a router or firewall--to rewrite the source IP of an outgoing IP datagram, while retaining the original IP in order to rewrite it into the response.

Normally, a router would simply take an IP datagram, inspect the contents, decrement the TTL by 1, recalculate the checksum, and forward the rest of the data at the network layer (without touching it).

But with NAT, the router will also rewrite the source IP address. It will make the message look like it originated from the *router*, not the computer that sent it.

- The point of all this, is to hide the IP address of the computer sending the message.
- This is known as **IP masquerading**, an important security concept.
    - No one can establish a connection to your computer, if they don't know what IP address it has.
- Through NAT, you could have hundreds of computers on a network, with all of their IPs being translated by a router into its own IP address;
    - This would effectively render the entire address space of the network as invisible;
    - This technique is known as **one-to-many NAT**.

NAT and the Transport Layer

To solve the issue of one-to-many NAT, port preservation comes into play.

Port Preservation

- **Port preservation** is a technique where the source port chosen by a client is the same port used by the router.
- Remember that outbound connections choose a source port at random from the ephemeral ports.
- In the simplest setup, a router set up to NAT outbound traffic will just keep track of what this source port is, and use that to direct traffic back to the right computer.

If two different computers on the same network booth choose the same source port at the same time, the router (normally) selects an unused port at random to use instead.

Port Forwarding

- **Port forwarding** is a technique where specific destination ports can be configured to always be delivered to specific nodes.
- This technique allows for complete IP masquerading, while still having services that can respond to incoming traffic.
    - It also simplifies how external users might interact with many services, all run by the same organization.

NAT, Non-Routable Address Space and the Limits of IPv4

Address Blocks

The IANA has primarily been responsible for assigning address blocks to five **regional Internet registries (RIRs)**:

- AFRINIC, which serves the continent of Africa;
- ARIN, which serves the United States, Canada, and parts of the Caribbean;
- APNIC, which is responsible for most of Asia, Australia and New Zealand, and Pacific Island nations;
- LACNIC, which covers Central and South America, and Caribbean parts not covered by ARIN;
- RIPE, which serves Europe, Russia, the Middle East, and portions of Central Asia.

But they all ran out of IP addresses to distribute (all 4.2 billion of them).

Solution

Implementation of IPv6 will take time, so NAT and non-routable address space have been temporary solutions to the limits of IPv4.

- With NAT, you can have hundreds (even thousands) of machines using non-routable address space.
- Yet with just a single, public IP, all those computers can still send traffic to--and receive traffic from--the Internet.

VPNs and Proxies

Virtual Private networks

Virtual private networks (VPN) are a technology that allows for the extension of a private or local network to hosts that might not be on that local network.

The most common example of how VPNs are used is for employees to access their business's network, when they're not in the office.

- VPNs are a *tunneling protocol*.
  - This means they provision access to something not locally available.
  - When establishing a VPN connection, you'd refer to it as a VPN tunnel being established.

For example:

- An employee uses a VPN client to establish a VPN tunnel to their company network.
- This would provision their computer with what's known as a *virtual interface*, with an IP that matches the address space of the network they've established a VPN connection to.
- Most VPNs work by using the payload section of the transport layer to carry an encrypted payload that actually contains an entire second set of packets:
  - The network, the transport, and the application layers of a packet intended to traverse the remote network.
  - This process is inverse, relative to normal TCP/IP process.
- *"Basically, this payload is carried to the VPNs end point where all the other layers are stripped away and discarded. Then the payload is unencrypted, leaving the VPN server with the top three layers of a new packet. This gets encapsulated with the proper data link layer information and sent out across the network. This process is completed in the inverse, in the opposite direction.* "

VPNs usually require strict authentication procedures.

- VPNs were one of the first technologies where two-factor authentication became common.
- **Two-factor authentication** is a technique where more than just a username and password are required to authenticate.

VPNs can also be used to establish site-to-site connectivity.

- The concept isn't too different; one router establishes the VPN tunnel to the router on another network.
  - The two physically-separated offices are then able to act as one network, and access network resources across the tunnel.

Remember that like NAT, VPN is a technique, not a protocol. So the details can differ.

VPN Summary

VPNs are a technology that use encrypted tunnels to allow for a remote computer or network to act as if it's connected to a network that it's not actually physically connected to.

Proxy Services

A **proxy service** is a server that acts on behalf of a client in order to access another service.

Proxies sit between clients and servers, providing an additional benefit:

- Anonymity
- Security
- Content filtering
- Increased performance

An example is of a proxy a gateway router.

A proxy is just an *abstraction* or concept.

"Proxy" is usually referring to a *web proxy*.

- These are proxies specifically built for web traffic.
  - These originally were to increase performance, in the days when Internet connections were much slower.
- A common use of a modern web proxy would be to prevent sites (like Twitter) from being accessed on a company's network.

Another example is a **reverse proxy**:

- A service that might appear to be a single server to external clients, but actually represents many servers living behind it.
- Much like the concept of DNS Round Robin, this is a form of load balancing.
- Reverse proxies are also used to deal with decryption.
  - EX: Clients -> Decryption Hardware -> Reverse Proxy -> Application Servers

Proxy Summary

Proxies are any server that act as an intermediary between a client and another server.

Questions for review

Keywords

Concepts