Distributed Systems - Assignment 2

Markus Roth

October 11, 2016

1 Exercise 1: Ethernet Switches

1.1 Task of an Ethernet Switch

The Ethernet switch links multiple Ethernets at the Ethernet network protocol level. It operates at the level of Ethernet frames, which it forwards to different networks as required. This enables higher-level protocols to see the combined Ethernets as a single network.[1]

1.2 Layer 2 Switch: MAC-address table

A switch has multiple ports, each with a MAC-address. Unlike normal ports, these ports run in promiscuous mode, meaning that they real all arriving messages - even the ones not addressed to their own MAC-address.

For each packet the Switch receives, it looks at the source address contained in the Ethernet frame. It then adds the source address of the frame to a table, thereby mapping the source MAC-address to the MAC-address of the Switch port that received the message. In this way, the switch can learn which MAC-addresses are reachable through which of its ports.

This then enables to forward Ethernet frames with certain destinations over certain ports, a process called adaptive filtering.[2]

1.3 Layer 3 Switch: IGMP snooping

Strictly following the OSI layer model, an Ethernet switch only knows about communication on a MAC level. However, this leads to inefficiency in IP-multicasting a layer higher up. Since a normal Layer 2 Switch does not know about IP at all, it must forward IP multicast packets to all its ports. After all, any of the devices on any of the ports might be on the IP multicast address list.

Using a process called *IGMP snooping*, a Layer 3 Ethernet switch can listen in on the IP conversations happening in the Ethernet frames it receives and transmits. It can then manage a table with information about which of its ports it needs to forward which frames of IP multicast steams to so that IP multicasts work without needing to flood the network witch each frame.

[1] Coulouris, Chapter 3.5.1

[2] https://www.safaribooksonline.com/library/view/ethernet-switches/9781449367299/ch01.html

This functionality is normally performed by a router. Doing it on the level of a switch shows that the OSI-layer is more of a guideline for understanding network layers and not actually followed in reality that precicely. [3]

2 Exercise 2: Encapsulation

Communication on a computer network is divided into layers. At the top layer, a computer program on one computer might want to send some information to a computer program on another computer. At the bottom-most layer, individual bits are transferred over some physical medium.

Information always starts at the top layer, moves down to the bottom-most layer, then moves back up again to the top layer at the destination. Each layer presents an interface the the layers above it. When a layer gets some information it should transmit, it will create new protocol data unit for the layer below it, embedding the information from the upper layer inside it.

The process of taking a protocol data unit from a layer n + 1 and wrapping it inside a protocl data unit from layer n is called *encapsulation*.

Encapsulation is used in communications systems because it enables a clear separation of concerns between the layers. It enables the lower layers to ignore the contents of the payload they are transmittig, and allows them to only focus on their own protocol. This also enables the lower layers to work with arbitrary upper-layer protocols, without knowing them. [4]

3 Exercise 3: Standards

Communication between different computers or other automated devices can only work if the various devices speak a common language. If all devices that need to communicate are manufactured by the same organization at the same time, it is possible to define their communication as any arbitrary well-defined protocol.

However, this is mostly not the case. Systems from different vendors, and systems built at different times, are expected to communicatie with each other. With the Internet, devices as diverse as a fire alarm sensor and a high-performance cloud backend system need be able to communicate.

The role of standards in communication systems is to provide a common language for all systems to communicate. These standards, if they are followed by everyone, guarantee interoperability between various systems.

In distributed systems, we require the network to work between systems of different vendors, so we need the standards to work.

[3] `https://en.wikipedia.org/wiki/Network_switch`

[4] Coulouris Chapter 3.4

4 Exercise 4: TCP/IP

4.1 1a

According to Stackoverflow[5], this is because it is much faster to only create a checksum for the header fields. In large-scale routers, this adds substantial speed.

Also, the protocol layers above have their own mechanism for ensuring that packets were not corrupted.

IPv6 completely removed checksums for those reasons.

4.2 1b

This is because the data offset, that determines the size of the header, has exactly four bits and counts 32 bit words. So if we set all the data offset bits to 1, we have a total header size of 15 words (60 bytes). The minimal TCP header is 20 bytes long, so at most we get the remaining 40 bytes for the options field.

4.3 1c

TCP is reliable in that is guarantees that messages arrive exactly as they were sent, in the correct order, or not at all. To guarantee this, TCP uses a technique called *positive acknowledgement with re-transmission*. This means that the receiver of a TCP packet must respond with an acknowledgement when it receives data. If the sender does not receive this acknowledgement within a certain timeframe, it will try to send the packet again.

Since packets also contain checksums, we can assume that they arrive with the exact same contents as they were sent. However, checksums are not a completely failsafe mechanism, as different transmission errors can cancel each other out. Still, the probability of that happening is very low, so we can safely say that TCP is a reliable transport protocol.[6]

4.4 1d

An IP address is assigned to a MAC address, so switches can transfer the IP packet to the correct interface. A machine may have different network interface cards, and each network interface card has a different MAC address. It is not possible for a switch to know that these different MAC addresses belong to the same machine. Therefore, an IP address is associated with the network interface, not with the machine.

4.5 1e

In the UDP header, there is no variable length options field. If a part of the header is not used (for example the source port), it is simply set to zero. Therefore, the UDP header is always of the exact same length, so no header field is needed.

[5] http://stackoverflow.com/questions/9205169/why-in-a-ipv4-packet-checksum-is-calculated-against-the-ip-header

[6] https://en.wikipedia.org/wiki/Transmission_Control_Protocol#Reliable_transmission

4.6 2

IP packets have a time-to-live number associated with them. Whenever a router retransmits an IP packet, it decrements the time-to-live counter. When the counter reaches 0, the packet is discarded. The source of the packer, read from the source field, is notified that the packet was not delivered.

4.7 3

A *transport address* consists of a *transport address* as well as a port number. This is because there might be multiple services that are responding on a particular network interface. For example, a web server might offer HTTP oon port 80 and HTTPS on port 443. Having different ports enables the services to notice which packets are meant for them.

4.8 4a: Terminal Access

It is very important for the commands to arrive in the correct order. For example, if the orders of a directory change command and a delete command are changed, something bad might happen. Use TCP!

4.9 4b: File Transfer

Here we have to assume large files, so overhead might be an issue. The order of the packets is not that important, as we can handle the ordering at the file transfer layer by giving each packet a number to signify which part of the file to transfer it represents. Of course, it would be easier to use TCP, but in this case if we handle some things at the file transfer level it might be worth it to use UDP for the reduced overhead and higher speed.

4.10 4c: User Location

User location is well-suited for use with UDP. A UDP packet for the request, and one for the response. It is trivial for the client to try requesting again if no answer was found within a certain timeframe. However, Finger actually runs on TCP. I'm not sure why.

4.11 4d: Information browsing

In HTTP, packets must arrive in the correct order, otherwise the web page will not be displayed correctly. Also, the HTTP protocol assumes a reliable protocol, as it does not handle re-transmission in itself. Therefore, TCP is the appropriate protocol.

4.12 4e: Remote procedure call

In a remote procedure call, the order of the packages is of paramount importance. Because procedure calls might have side-effects, their calls must not be re-ordered. Also, in RCP we want to use a reliable protocol for sure, as we don't want to implement any logic that handles failures in our RCP code. TCP!

5 Exercise 5: Centralized Web

5.1 Registering and Resolving Domains

To manage the domains, there must be an organization that is responsible for assigning domain names to organizations who want to use them. Steps to make this less centralized have already been taken by enabling pretty much anyone to start their own top-level domain and license the domains for it however they want to. However, the ICANN still is in full control over all domains. For a truly decentralized domain name system, it is possible to use blockchain technology. Namecoin [7] is such an attempt. Using it, you can get a .bit domain without any dealings with ICANN.

5.2 Tier 1 Networks routing using settlement-free peering

The Internet is a collection if networks that exchange traffic between each other. However, at the topmost level, there are few Tier 1 networks that handle much of the data transfer at the very top level. These Tier 1 networks are few in number and control lots of critical infrastructure. If a Tier 1 provider failed, it is possible that parts of the Internet would no longer work as well as before.

It might be possible to use more of a peer-to-peer approach for networking, but a huge costs concerning speed.

5.3 DDOS attack mitigation requires expensive hardware

In an ideal Internet, every network participant enjoys the same rights and everyones packets are treated equaly. However, with great power comes great responsibility: Evil actors might send illegitimate packets to distrupt services. If these packets come from a wide array of servers, they are called distributed denial of service attacks. These attacks are hard to locate without specialized hardware that implements protocols such as netflow. With netflow-enabled routers, it is possible to do deep packet inspection of packets passing through routers, so that it is possible to locate the source of illegitimate packets participating in denial of service attacks. [8] However, routers that support netflow are very expensive, and if they are required to participate in the Internet the Internet will have large hurdles to entry and no longer be a decentralized network of peers.

[7] `https://namecoin.org/`

[8] `https://idea.popcount.org/2016-09-20-strange-loop---ip-spoofing/`