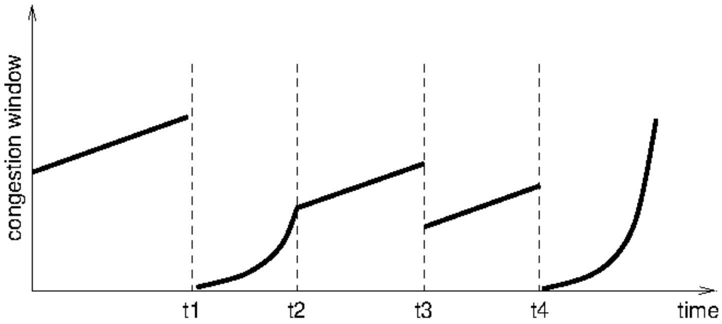
Network Systems (201600146/201600197), Test 3

October 5, 2020, 18:15–19:45

(Taken online on campus)

1. Congestion control



The above figure shows schematically how TCP's congestion window evolves over time. The following three questions are about this figure.

2 pt **Q 1.1** At which moment(s) did a packet-loss happen? (Mark all that apply.)

A. t1
B. t2
C. t3
D. t4
E. no packet loss happened
F. one can't deduce packet loss from this graph

→ A,C,D

1 pt **Q 1.2** In which phase is TCP in the interval between times t2 and t3?

A. Slow start
B. Additive increase
C. Fast recovery
D. CUBIC
E. This is incorrect behaviour for standard TCP congestion control

→ B

1 pt **Q 1.3** In which phase is TCP in the interval after time t4?

A. Slow start
B. Additive increase
C. Fast recovery
D. CUBIC
E. This is incorrect behaviour for standard TCP congestion control

→ E

At first glance, it looks like TCP is in Slow Start, with the exponential growth of the congestion window, as it would do after a packet loss detected by timeout. However, it should stop the slow start phase when the window reaches the slow start threshold, which should be set at half the congestion window just before the packet loss, i.e., just before t4. Compare this to what happened at t2: at t2, the exponential growth stopped when the congestion window reached half the value it had just before t1. It doesn't do this after t4, so this is incorrect behaviour.

1 pt **Q 1.4** Does the beginning of the diagram correspond to the start of the TCP connection, or must the TCP connection already have been active for a while?

A. We can't know based only on what is in this diagram.

B. The diagram starts at the start of the connection.
C. The connection must have been going on for a while, with no packet loss.
D. The connection must have been going on for a while, with at least one packet loss.
E. The connection must have been going on for a while, but we can't conclude whether there has been packet loss.

→ D

We can know that there has been packet loss because the congestion window growth is linear at the beginning of our diagram. If there had been no packet loss until then, TCP would still have been in its (initial) slow start phase, with exponential growth.

1 pt **Q 1.5** The book says that during "fast recovery" phase in TCP congestion control, TCP "use[s] the ACKs that are still in the pipe to clock the sending of packets". What does this mean?

A. The TCP sender can keep packets flowing while recovering from packet loss.
B. The TCP sender can quickly remove the remaining outdated ACKs from the network.
C. The TCP sender can precisely measure the round-trip time.
D. The TCP sender doesn't need to reduce its congestion window as the ACKs come in.

→ A

1 pt **Q 1.6** Some TCP variants measure the round-trip time (RTT) for congestion control. How does this work?

A. If the RTT increases, that's a sign buffers are filling up.
B. If the RTT decreases, that's a sign the network is overloaded.
C. If the RTT stays constant, that's a sign we're using the network optimally.
D. If the RTT varies a lot, that's a sign the network still has capacity to spare.

→ A

1 pt **Q 1.7** Why do we need advanced TCP congestion control algorithms like CUBIC on high-speed links with a large RTT?

A. We don't really need them, but engineers like to play with them.
B. The linear congestion window growth of standard TCP is too slow for such links.
C. The exponential congestion window growth of standard TCP is too fast for such links.
D. The fast recovery phase of standard TCP would overload such links.
E. CUBIC is not for high-speed links but for low-speed links.
F. CUBIC is not for links with a large RTT but for those with a small RTT.

→ B

1 pt **Q 1.8** RED is often used to drop some packets even when there is still space in the router's buffer. Why is this a good idea?

A. Dropping the packet at this router reduces congestion at the next router.
B. Better drop one packet now, than be forced to drop many later due to buffer overflow.
C. By dropping the packet, the router makes space for sending a congestion probe packet.
D. Dropping the packet saves CPU time, which is needed for calculating the drop probability.

→ B

---

2. Quality of Service

Consider a flow described by a token bucket with r = 100 000 tokens/second and B = 60 000 tokens (assuming one token per byte, as usual). Each packet is 4000 bytes large. Suppose this flow transmits a burst of 100 000 bytes every 2.5 s. Is this allowed?

2 pt **Q 2.1**

A. Yes, this is allowed.
B. No, for this to be allowed r should be higher.

C. No, for this to be allowed r should be lower.
D. No, for this to be allowed B should be higher.
E. No, for this to be allowed B should be lower.
F. No, for this to be allowed both r and B should be changed.

→ D

Sending a burst (that means, lots of data all at once) of 100 000 bytes is not allowed, as the bucket can contain at most 60 000 tokens, so you can't extract 100 000 tokens out of it all at once.

Q 2.2 If you answered that r and/or B should be different, what should they be?

→ $B \geq 100000$

Suppose this flow transmits every 0.5 s a burst of 40 000 bytes. Is this allowed?

1 pt **Q 2.3**

A. Yes, this is allowed.
B. No, for this to be allowed r should be higher.
C. No, for this to be allowed r should be lower.
D. No, for this to be allowed B should be higher.
E. No, for this to be allowed B should be lower.
F. No, for this to be allowed both r and B should be changed.

→ A

This is allowed. The burst size is not larger than B, and also the average rate, 40000 bytes per 0.5 s = 80000 bytes per second is not larger than r.

Q 2.4 If you answered that r and/or B should be different, what should they be?

→ (not applicable)

2 pt **Q 2.5** Suppose we have 2 flows, each obeying the above token bucket model, sharing a single link of 400 000 bytes/second. If Fair Queueing is used, what is the maximum delay (queueing delay plus transmission delay) a packet can experience at this link?

Value (in seconds):

→ 0.3

Q 2.6 Explanation:

→ Worst case, both flows send a maximum-sized burst at the same time, after being idle for a while so the buckets are full and the queue is empty. Each such burst is 60 000 bytes, or 15 packets of 4000 bytes each. So the queue will be filled with 30 such packets, or 120 000 bytes. Since both flows have the same packet size, FQ will simply alternately send a packet from each flow. Draining this queue will thus take 120 000/400 000 = 0.3 s.

Shorter argument: FQ gives each flow at least 200 000 bytes/s, regardless of the other flow's activity, so tranmitting a maximum size burst won't take more than 60 000 / 200 000 = 0.3 s.

The second argument shows how easily one can reason about delays in FQ.
It is slightly less precise though in the sense that it only proves an upper bound for the delay, not that this upper bound can be reached. Also, it doesn't take into account possible minor extra delays due to packets not being able to interrupt each other.
We gave full points already for the second argument.

1 pt **Q 2.7** Again consider 2 flows, each obeying the above token bucket model, sharing a single link using Fair Queueing, but now that shared link has a speed of only 150000 bytes/second. Is it possible that, averaged over a long time, one flow gets to transmit 100000 bytes/second, i.e., gets 2/3 of the capacity of the link?

A. No, that would require the use of Weighted Fair Queueing.
B. No, that would require the other flow to transmits at most 50 000 bytes/s on average, which contradicts r=100 000.
C. No, since both sources have the same token bucket parameters, they will get treated equally.
D. Yes, this can happen if the other flow transmits just 50 000 bytes/s on average, which it is allowed to do.
E. Yes, if one flow transmits bursts twice as frequently as the other flow.
F. Yes, but only if one flow at setup time has indicated that it wants 2/3 of the bandwidth.

→ D

FQ only gives each flow 50% of the capacity when both flows are constantly active, filling the queue. If the other flow offers no packets, then your flow can get 100% of the bandwidth. And anything in between: if the other flow doesn't use all of its fair share of the bandwidth, your flow can use the rest.
Furthermore, token-bucket models only describe an upper limit of how fast a flow can send and how large its bursts can be. It is allowed to send less than that. So the competing flow still satisfies the token-bucket model if it transmits at only half its allowed rate r.

1 pt **Q 2.8** What does it mean if an application is ”elastic”?

A. That it can adjust its packet size to network conditions.
B. That it can deal with variation in delay and bandwidth.
C. That its demand on the network can vary a lot.
D. That it transports live audio and/or video over the network.
E. That it does not use congestion control.

→ B

1 pt **Q 2.9** Why is scalability of network design a harder problem for real-time applications than for file transfer?

A. Data rates are higher for real-time applications.
B. Routers may need to handle packets differently based on what real-time flow they belong to.
C. Real-time data packets are larger, needing larger buffers.
D. Real-time applications don't do congestion control, so routers need to do this instead.
E. File transfers can tolerate occasional packet loss, while real-time applications can't.

→ B

Answer B is the correct one: in order to give low delays to packets from real-time applications, while there's also a lot of other traffic, routers need to schedule packets in more complicated ways that simply first-come first-serve, e.g., use WFQ. And in order to do so, they need to find out, somehow, which packet needs to get which treatment. This becomes more and more a scalability problem when there is much traffic and many different (types of) flows.

Answer E was intended to be a wrong answer: most real-time applications can make up for an occasional packet loss. A single packet loss may give a slight interruption of the audio or video, but need not be serious, or the application may put redundant data in other packets to completely mask a single missing packet. However, we still gave 0.5 point for answer E, because it is true in case of loss-intolerant real-time applications.

1 pt **Q 2.10** What does “Expedited Forwarding” in DiffServ mean?

A. Give absolute priority to real-time data.
B. Make sure real-time data always sees a non-overloaded network.
C. Reserve enough bandwidth for each real-time flow separately.
D. The traffic class used for non-real-time data in a DiffServ network.
E. Increase the transmission rate for real-time data.
F. Decrease the propagation time for real-time data.

→ B

Answer A was quite popular, but not correct. Expedited forwarding can indeed be implemented by giving this traffic absolute priority, but it can also be implemented using e.g. WFQ. In fact, this is literally stated in the third paragraph of section 6.5.3.1 of the book.

1 pt **Q 2.11** Real-time applications typically don't use TCP and thus don't have TCP's congestion control. So what?

A. This is wrong, all internet applications use TCP.
B. This is wrong, TCP congestion control also works for non-TCP applications.
C. Real-time applications have to implement their own congestion control.
D. Congestion control is not desirable for real-time applications, as it would reduce audio/video quality.
E. UDP-based real-time applications use QUIC for congestion control.

→ C

Answer D was quite popular, but not correct. If real-time applications don't do congestion control, they can overload the network and then their performance will again be bad. So typically, they do implement some kind of congestion control, and adapt the coding of the audio/video data to the available bandwidth, giving e.g. a less sharp picture but avoiding complete dropouts as would happen when the network is overloaded. (Note that there's an entire section in the book about congestion control for real-time applications: section 6.5.4.)

3. Security

1 pt **Q 3.1** In many systems, a public-key algorithm with at e.g. 2048 bit keys is used to negotiate a symmetric key of e.g. 128 bits for the actual data. Isn't this difference in key length very risky?

A. No, because the 128-bit key will be used for encrypting much less data than the 2048 bit key, so an attacker has less material to work with.
B. No, because a symmetric key algorithm is much harder to crack by brute-force search than a public-key algorithm with the same key length.
C. Yes, but symmetric cryptography with 2048 bit keys would be too slow, so we have to take the risk.
D. Yes, that's why DES is slowly being replaced by AES.
E. The statement is the wrong way around; the short key is used to negotiate the longer key for added safety.

→ B

Apparently, this was found a difficult question, so let's discuss all options:
A: although indeed it is a good idea to give an attacker less material to work with, that's not happening here; the bulk of the data is encrypted using the 128-bit session key, while the 2048-bit public key is only used very little data, namely to setup the symmetric encryption keys.
B is correct: the algorithms are fundamentally different. To crack a 128-bit a symmetric-key, an attacker will have to try on average 2^{127} different keys, one by one. To crack a 128-bit RSA key, the attacker would have to factor a product of two 64-bit primes.
C: although 2048-bit symmetric cryptography would be slower than 128-bit, this difference is not the reason not to use it; the main reason is simply that isn't needed: 128-bit keys are safe enough for most purposes.
D: it is true that for symmetric cryptography DES is being replaced by AES, but that's a change from 56-bit keys to 128-bit (or up to 256-bit), so not related to the question.
E: real systems use e.g. RSA or Diffie-Hellman with long keys to negotiate shorter session keys, as the question states.

1 pt **Q 3.2** Suppose you're trying to connect to an SSH server, to which you've safely connected before, but now there's an eavesdropper monitoring your traffic. Are you still able to securely connect to your SSH server?

A. No, the attacker would see all the traffic so it is no longer secure.
B. Yes, because SSH uses encryption.
C. Yes, SSH is made to run over untrusted connections.
D. No, the handshake fails because keys have changed.
E. No, the encryption can be broken because the attacker sees the handshake.

→ B or C

We didn't formulate answers B and C precisely enough, so we accept either of them as correct, since both are true statements. Options A, D and E are really incorrect statements.

1 pt **Q 3.3** Following the previous question, this time the eavesdropper is a Man-in-the-middle that actively impersonates the SSH server you are connecting to. Are you able to securely connect to your SSH server?

A. No, the attacker would see all the traffic so it is no longer secure.
B. Yes, because SSH uses encryption.
C. Yes, SSH is made to run over untrusted connections.
D. No, the handshake fails because keys have changed.
E. No, the encryption can be broken because the attacker sees the handshake.

→ D

Note that it was given that you have connected to this server before. So you (well, your ssh program) already knows the server's public key. The MitM will substute a differect public key (otherwise, he/she can't impersonate the server as he/she wouldn't have the corresponding private key), so your ssh program should refuse to continue, and warn you that the keys have changed.

2 pt **Q 3.4** You are searching something on Google via an HTTPS connection. What can your internet provider (ISP) see from your traffic? Mark all that apply.

A. The fact that it is a TLS connection.
B. The source and destination IP addresses.
C. The HTTP headers.
D. The HTML content.
E. None of the above, since everything is encrypted.

→ A,B

TLS will encrypt the data sent over the connection (such as the HTTP headers and HTML content), but does not encrypt the IP addresses (that would make it impossible to deliver the packets), nor the portnumbers; since destination port 443 is the well-known port for HTTPS, your ISP can still see that you're using HTTPS and thus TLS.

1 pt **Q 3.5** Your friend is connected to an open WiFi network. He/she says he/she is protected against eavesdroppers on the network because he/she is using an IPSEC VPN in AH tunnel mode to his/her home network. Is your friend right?

A. Yes, IPSEC encrypts the connection and he/she is therefore protected.
B. No, he/she would be protected if IPSEC was running in AH transport mode.
C. No, he/she would be protected if IPSEC was running in ESP tunnel mode.
D. No, you cannot protect against eavesdroppers on an open WiFi network.

→ C

AH mode only provides authentication, but does not encrypt the data. Therefore, ESP mode is needed to protect against eavesdroppers.

2 pt **Q 3.6** Which of the following security services are provided by WPA2 in enterprise mode? Mark all that apply.

A. Protection against DDoS attacks
B. Authentication of the mobile device to the infrastructure
C. Authentication of the infrastructure to the mobile device
D. Confidentiality of the data being transported
E. Making the mobile device undetectable to eavesdroppers

→ B,C,D

WPA2 can't protect against DDoS, nor can it make the mobile device undetectable. It does give authentication in both directions, and confidentiality by encrypting the data; as stated in section 8.5.5 of the book.

2 pt **Q 3.7** You receive an email from, supposedly, Pieter-Tjerk de Boer. Fortunately, the email is signed with PGP. What steps should you (and/or your mail software) take to verify that the email actually came from Pieter-Tjerk? Mark all that apply. (Assume you currently only posses your own public and private keys.)

A. Validate the signature with his public PGP key.
B. Validate the signature with his private PGP key.
C. Check that the public PGP key is actually his, e.g. by comparing a hash in a physical meeting.
D. Check that the private PGP key is actually his, e.g. by comparing a hash in a physical meeting.
E. Download his public PGP key from a keyserver.
F. Download his private PGP key from a keyserver.
G. Decrypt the signature with his public PGP key.
H. Decrypt the signature with his private PGP key.

→ A,C,E

You would need to get my public key (option E), use it to validate the signature (option A), and make sure that the key is actually mine (option C), as otherwise a MitM might deceive you by giving you a key, saying that it's mine, while it's actually his.
Decrypting a signature (option G) does not make sense, since there's no plaintext in it.
And you surely don't need my private key, otherwise it wouldn't be a "private" key; so options B, D, F and H are wrong.

1 pt **Q 3.8** What is the characteristic difference between stateless and stateful firewalls?

A. Stateless firewalls do not track state, which gives the flexibility to deal with dynamically assigned ports.
B. Stateful firewalls do track state, which gives the flexibility to deal with dynamically assigned ports.
C. Stateful firewalls allow for multiple zones of trust, whereas stateless firewalls do not.
D. Stateless firewalls are isolated from the Internet, whereas stateful firewalls are not.

→ B

2 pt **Q 3.9**

Imagine a corporate network that knows three zones of trust: the Internet, your internal network, and a demilitarized zone (DMZ). You are in the process of deploying a web server to the DMZ and need to configure a stateful firewall to allow for the Internet as well as users on your internal network to communicate with the server using HTTPS, which uses port 443/TCP.

The firewall uses the following policies:
1. All packets from the Internet to the DMZ to port 443/TCP are allowed
2. All packets from the internal network to the other zones are allowed
3. All packets from the Internet to the other zones are allowed, provided they belong to an established connection
4. ...
5. All other packets are dropped

You can accomplish your goal with **one** additional rule (in the 4th position). What is it?

→ All packets from the DMZ to the other zones are allowed, provided they belong to an established

connection.
Alternatively: All packets from the DMZ from port 443/TCP to the other zones are allowed, provided they belong to an established connection.

1 pt **Q 3.10** How do DDoS and reflection relate?

A. Every DDoS attack is a reflection attack
B. Every reflection attack is a DDoS attack
C. Reflection can be used to make a DDoS more powerful
D. Reflection can be used to protect against DDoS

→ C

Reflection is often used to make a DDoS more powerful, by having a small packet elicit more and/or larger packets to be sent to the victim: answer C.
However, reflection could also be used in other kinds of attack, so answer B is not right, although many chose it.

Grade calculation

The grade was calculated using the following formula:

$$\text{grade} = \frac{\text{points} - 6.85}{33 - 6.85} \times 9 + 1$$

33 is the maximum number of points for this test.
6.85 is the "guessing factor": it's the number of points one would get on average from giving totally random answers at the multiple-choice questions.