Resit Advanced Network Security

Jaap-Henk Hoepman, Harald Vranken

July 8, 2019

Name: _____

Student number: S _____

Please answer your questions using the space provided on the exam sheet. Write legibly, and use proper sentences; an unreadable answer is a wrong answer. . . Answer questions concisely, but with sufficient detail and precision. Always explain your answers. Please leave space in the margin for correction marks. Use a separate piece of scratch paper for draft answers, private computations or remarks.

Write your name and student number on every page.

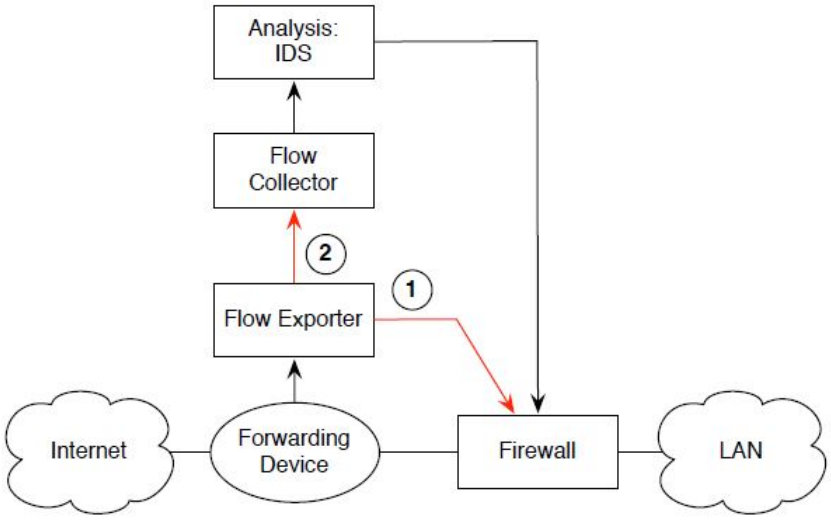
You are not allowed to use books, notes, tablets, PC's and (smart)phones during the exam.

The total number of points that can be scored is 60, as specified alongside the questions. The final grade equals $1 + 9 * \frac{points}{60}$, rounded to the nearest half grade (except for grades between 5 and 6 which are rounded to nearest full grade).

Good luck!

Question 1 (5 points): Preventing and detecting network attacks

a. *(2 points)* The figure below shows a typical deployment of a flow monitoring system, in which intrusion detection is integrated in the flow metering process. How can this help in the mitigation of DDoS flooding attacks? Consider both the path from the flow exporter to the firewall (1) and the path from the flow exporter to the flow collector (2).



b. *(3 points)* A flow exporter can monitor the following three metrics: number of flow records created per second, the average number of bytes per flow, and the average number of packets per flow. Which one is most suited to detect DDoS flooding attacks, and why?

Answer:

a. This helps in the mitigation of DDoS flooding attacks by (1) configuring the firewall to block malicious traffic before it reaches the LAN, and (2) filtering malicious flow data before it reaches and potentially overloads the flow collector.

b. The number of flow record creations appears to be the most suitable metric. DDoS flooding attacks typically generate a large amount of short flows, which is rather uncommon compared to other applications. The average numbers of bytes or packets per flow are less distinguishing, as also other applications may generate short flows. In practice, the number of flow record creations shows the least amount of noise and attacks are clearly identifiable as peaks. (It also is easiest to compute, as only a single counter is needed that has to be reset after every measurement interval.)

Question 2 (5 points): Economics of network security

a. *(2 points)* E-commerce providers and banks have conflicting incentives when it comes to enhancing security. Give an example of such conflicting incentives.

b. *(3 points)* What are the externalities of sending spam?

Answer:

a. Incentives to enhance security are for instance that this supports growth and trust in online transactions and reduces the risk of reputation loss and brand damage. Incentives to not enhance security are that security measures are costly and may reduce the usability of the service.

b. The spammer's goal is to make money with a spam compaign, for which little investment or infrastructure is required. The negative externalities are that this puts a burden on the infrastructure and bandwidth offered by ISPs, and effort or time waisted by recepients (for filtering spam, cleaning up mailboxes, negative consequence when replying to spam messages).

Question 3 (5 points): WiFi security

a. *(3 points)* When connecting to eduroam, security is provided by outer authentication and inner authentication. How are outer and inner authentication achieved?

b. *(2 points)* Why is a 4-way handshake still required after successfull completion of the outer and inner authentication?

Answer:

a. Outer authentication takes place by means of a TLS tunnel between the client and the authentication server. The client can use an anonymous identity. The server is authenticated by a certificate. Inner authentication typically takes place next by a username/password (as with MS-CHAPv2) inside the TLS tunnel.

b. The outer and inner authentication are carried out as 802.1x authentication in which the client and access point are mutually authenticated (they both have the PMK). In the 4-way handshake a session key (PTK) is negogiated. (The PTK is split into three keys: KCK for message authentication in the 4-way handshake, KEK for encryption of keys, and TK for data confidentiality and integrity.)

Question 4 (5 points): Mobile telephony security

a. *(3 points)* In a one-way MitM attack on the GSM wireless link, the attacker acts as a cell tower towards the victim's phone, but not as the victim's phone towards the network. Can the MitM listen in on ingoing and/or outgoing calls of the victim's phone?

b. *(2 points)* Can a one-way MitM attack be detected by the caller and/or callee (ie., the phone that makes the call and the phone that is being called)?

Answer:

a. Once the victim's phone starts a call, it notifies the MitM. The MitM behaves as a normal cell tower, but instructs the victim's phone not to use encryption. The MitM can now listen to the unencrypted data stream sent by the phone. The MitM will forward the data stream to the actual cell tower using a (separate) encrypted connection in which the MitM encrypts the data stream of the victim's phone and decrypts the data stream of the callee. The MitM can only listen in on outgoing calls.

b. In this attack the callee can notice the incoming call originating from a different number than normal. The attacker may use a hidden number. This could still deviate from what the callee expects, but would be less suspicious.

Question 5 (5 points): Routing security

a. *(2 points)* The SCION architecture separates the control plane from the data plane. What is each plane responsible for?

b. *(3 points)* How is origin validation (ie., identifying the sender of a packet) done in the SCION architecture?

Answer:

a. The control plane is responsible for discovering paths and making those end-to-end paths available to end hosts. The data plane ensures packet forwarding using the provided paths.

b. Sources share a separate symmetric key with every AS on the path. The packet header contains a list of OV values, which is a list of MACs over DataHash (hash over payload). Each MAC is made with the key shared between the source and AS or destination. Every intermediate AS and the destination can verify its corresponding OV value.

Question 6 (5 points): Botnets

a. *(2 points)* What are the advantages and disadvantages of hardcoding either the IP address or the domain name of the C&C server in the bot binary?

b. *(3 points)* What non-technical measures can be taken to combat botnets?

Answer:

a. Reverse engineering of the bot binary will reveal the IP address or domain name of the C&C server. Advantage of hardcoding the IP address is that no DNS-lookup is required; advantage of hardcoding the domain name is that it can be resolved to different IP addresses in time.

b. Attacker dissuasion (follow the money, targetting the business model employed by botnets); legal accountability; user education.

Question 7 (15 points): Leader election

a. *(7 points)* This is LeLann's leader election protocol (code shown for node i) to elect a leader on a ring of n nodes.

```
I_i ← ∅
send right C[i].id
while C[i].id ∉ I_i
do begin receive left id
         I_i ← I_i ∪ {id}
         send right id
   end
C[i].leader ← (C[i].id = min_j j ∈ I_i)
```

In this protocol it is assumed that message passing is FIFO. Explain what this means exactly, and explain why this is necessary.

b. *(5 points)* What is the worst case message complexity of this protocol? Prove this.

c. *(3 points)* How can this algorithm be fixed to work even if message passing is not FIFO?

Answer:

a. FIFO message passing means that messages on a single transmission link between two nodes are always received in the same order as they are sent (2 points).

Assume message passing is not FIFO, and that we run LeLann's leader election algorithm. Recall that in LeLann's algorithm nodes start by sending their own identifier to their clockwise neighbour, and that nodes j store identifiers they receive in a set I_j, and that they forward identifiers they receive in a clockwise direction, until they receive their own identifier back. If this identifier is the minimum among all identifiers in I_j, then node j is leader.

Now assume the following situation: there are three nodes, a, b and c ordered like that in clockwise fashion as a ring of three nodes. Node a has identifier 1, node b has identifier 0, and node c has identifier 2. All nodes start by sending their identifiers to their clockwise neighbour. Node b is fast and immediately forwards the identifier 1 to its clockwise neighbour c. If the link between b and c is not FIFO, node c may receive the value 1 before the value 0. Node c then sends value 1 to node a before sending value 0 to node a. This means node a will stop looking for a leader, decide it is leader (as among the set $\{1,2\}$ of values it received, 1 is the lowest value), and *stop forwarding values*! This means the value 0 is not forwarded, which means node 0 waits forever (and does not know whether it is leader or not). (5 points)

Note: it doesn't matter that a node that is not the minimal id gets elected leader, as long as there is a single leader! (-2 point).

b. This is more tricky than one thinks, because a node (unnecessarily!) sends out its own identifier once more right before it terminates. This means we need to explicitly invoke the FIFO property in the proof (to argue that a clockwise neighbour does not forward this message once more)!

The worst case message complexity is $n(n+1) = O(n^2)$ where n is the size of the ring.

Every nodes start by sending its own identity to its clockwise neighbour. Nodes receive and forward every identifier they receive, until they receive their own identifier back (which they send out once more). By induction one can show that, because of the FIFO property, the i-th message a node receives must be the identity of the i-th counterclockwise neighbour along the ring. This means each node receives the values of all other $n-1$ nodes before it receives its own value back after which it stops sending. Which implies each node sends exactly $n+1$ messages (which includes the initial message sent). There are n nodes, hence $n(n+1) = O(n^2)$ messages sent in total.

c. An easy fix is to *always* forward an incoming message along the ring (even if you already know you are or are not a leader), unless it is your own identity that you see returned to you.

Other option is to add hop counter to a message and only stop forwarding if you receive your own id *and* the number of id's received equals the hop counter in the message that contained your own id.

Peterson leader election is also a nice idea, but only works on bidirectional links (2 points only).

Question 8 (15 points): Consider a ring of n nodes, that communicate in the state reading model, and that are scheduled using a fair central daemon. The ring is oriented. Nodes can read the state of their counterclockwise neighbour. Every node has a unique identity which is part of its program code (and hence not affected by memory corruptions) and can be used in the code for the node itself. This identity has the special form $(x, h(x))$ where x arbitrarily taken from the set $\mathbb{N}$ and h is a public hash function. You may assume that a transient memory fault never creates a state such that $(y, h(y))$ holds. (In other

words: if you read a state that contains such a value, you can be sure it is correct!). This identity is *not* part of the state (and hence cannot be read by other nodes; it can only be used by the node itself in its program code). Apart from this knowledge of identity, the program code for every node is the same.

a. *(5 points)*

   Write a selfstabilising protocol for leader election for this setting. Clearly specify the state each node keeps, and describe how this state is updated when the node is given a turn by the central daemon.

b. *(3 points)* Specify the legitimate states.

c. *(7 points)* Prove the protocol correct

Answer:

a. Under this assumption we can distinguish real identities from ones created by memory faults, so LeLann's idea can be used to find the minimum as follows.

   The state $s[i]$ of a node is (v,w). Node i has identity $\mathrm{id}(i) = (x, h(x))$. The state of the counter-clockwise neighbour is denoted $s[i-1]$ (wrapping to $n-1$ when $i = 0$). The transition function is

$$s[i] \leftarrow \begin{cases} \mathrm{id}(i), & \text{if } s[i-1] \neq (x,h(x)) \text{ for some } x \\ s[i-1], & \text{if } s[i-1] = (x,h(x)) \text{ for some } x \text{ and } x \leq \mathrm{id}(i).v \\ \mathrm{id}(i), & \text{if } s[i-1] = (x,h(x)) \text{ for some } x \text{ and } x > \mathrm{id}(i).v \end{cases}$$

   Node i is a leader when $s[i].v = \mathrm{id}(i).v$.

b. Let *min* be the minimal value among all identities. We say node i is legitimate if $s[i] = (min, h(min))$. Then the legitimate states are those where all nodes are legitimate. From this leader election follows.

c. We have to prove closure and convergence.

   We start with closure. In the legitimate state, all nodes have a state $s(i) = (x,h(x)$ for some x (in fact, for $x = min$). This means the first case of the update function never applies. Moreover, because in fact $x = min$, the minimum identity in the system, the third condition also never applies. In other words, the update function sets $s[i] = s[i-1]$. Given the definition of the legitimate states this in fact does not change $s[i]$ hence closure trivially follows.

   Convergence is proven by induction on the number of legitimate nodes.

   Because by assumption no transient error can create a state $(x,h(x))$, any state $(x,h(x)$ must in fact be the result of an assignment $s[i] = \mathrm{id}(i)$ for some i. In other words, if the state is $(x,h(x))$ then $x = \mathrm{id}(i)$ for some i.

   Let node i have the minimum identifier, i.e. $\mathrm{id}(i) = (min, h(min))$. Let node i take a step for the first time. (This will happen eventually because the central daemon scheduler is fair.) Case analysis and the above observation allow us to conclude that either case one or three of the update function applies, so that after that step, $s[i] = \mathrm{id}(i) = (min, h(min))$. In other words node i is legitimate, and we have at least one legitimate node.

   Suppose nodes $i, \ldots, j-1$ are legitimate (wrapping around $n-1$ where needed). If $j-1 = i-1$ we are done: then all nodes are legitimate. Suppose not. Observe that if any of the nodes $i+1, \ldots, j-1$ take a step their state will not change (similar to the closure proof). Also observe that if node i

takes a step its state will also not change (similar to the argument in the previous paragraph). We conclude that the number of legitimate nodes never decreases.

So suppose node j takes a step (this will again happen eventually as the scheduler is fair). Then case two of the update function applies setting $s[j] = s[j-1]$ making node j legitimate as well. This increases the number of legitimate nodes by one.

(end of exam)