COMPSCI 314 S2 T 07 Assignment 3

Department of Computer Science
The University of Auckland

Sample Answers

Q1. According to Shay, "A virus is a collection of instructions attached to an executable file that does something the original executable file was not designed to do."

a. What security property of the executable file is violated when it is infected by a virus (according to Shay's definition)? Explain briefly. **[2 marks]**

The integrity of the executable file is violated, because the person who wrote the virus is not authorised to modify an executable on someone else's machine.

Grading notes: 2 marks for any answer that is well-supported by an explanation; 0 marks for any answer without an explanation.

b. Which security properties of a computer system might be violated, if it executes a file which has been infected by a virus? Explain briefly. To receive credit your answer must discuss all three of the basic "CIA" properties. **[3 marks]**

All three of the security properties of the system are at risk when an infected file is executed. The execution of this file might violate system confidentiality, for example by transmitting sensitive information from the system through its network connection. This execution might violate system integrity by replicating itself in other system files. Finally, this execution might violate system availability by deleting a system file required for the next system boot to succeed.

Grading notes: 1 mark for the discussion of each property.

Q2. Shay describes the Internet worm as follows.

> "In November 1988 a Cornell graduate student released a worm into the Internet, which invaded thousands of Sun 3 and VAX computers ... This worm was of the so-called harmless variety; it did not damage any information or give away any of the secret passwords it uncovered.
>
> "On the other hand, it was a serious breach of security. It replicated quickly throughout the Internet, clogging communications and forcing many systems to be shut down. ...
>
> "The worm itself was written in C and attacked UNIX systems through flaws in the software. ... In one approach, it used a utility called **fingerd** that allows one user to obtain information about other users. ... The flaw that was exploited was that the fingerd program's input command (the C language `gets` command) did not check for buffer overflow. Consequently, a worm running on a remote machine could connect to the fingerd program and send a specially constructed message that overflowed the fingerd program's input buffer.
>
> "... Because of the overflow, ... when fingerd finished, ... the worm was now connected to the shell. From that point the worm communicated with the shell and eventually sent a copy of itself, thus infecting the new machine. The worm then proceeded to inspect system files, looking for connections to other machines it could infect.

> "It also attacked a password file trying to decipher user passwords. Deciphering a password allowed the worm to attack other computers where that user had accounts. ..."

a. Did the author of the internet worm (as described in the quoted material above) violate the confidentiality, integrity, or availability of the fingerd utility? Explain briefly. **[2 marks]**

I will assume that the author of the internet worm did *not* have authorisation to use fingerd on every computer that was infected by this worm. I will also assume that an unauthorised execution of a file is a confidentiality violation, although this is arguable because it is actually the CPU (and not the author of the worm) which is reading the file. Under these assumptions, there was a confidentiality violation.

I do not see any integrity violation of the fingerd utility itself, for this executable file wasn't modified by the attack. However the attack *did* cause an overflow in the input buffer of the fingerd utility, which is a violation of the integrity of its dynamic state. So I'd say there was an integrity violation of the dynamic (running) state of the fingerd utility, but no integrity violation of its static (stored) state.

The author violated the availability of the fingerd utility on every system that was shut down as a direct result of their worm.

Marking notes: This was a difficult question and should be marked rather generously. 2 marks for any coherent discussion which considers all three properties in the context of the complete attack.

The question of whether an execution is a confidentiality violation is vexed, especially on a Unix file system where the "x" (execute) permission bit is distinct from the "r" (read) permission bit. (See http://www.cs.auckland.ac.nz/~cthombor/Pubs/TC/introSecurity.ppt at slide 14 for one possible resolution of this taxonomic confusion.)

b. Did the author of the internet worm (as described in the quoted material above) violate the confidentiality, integrity, or availability of any file? Explain briefly. **[2 marks]**

As argued above, the author violated all three CIA properties of the fingerd file.

A strong case could be made for a confidentiality violation, if the worm had gained elevated authority (e.g. "root") in order to read some file. We aren't given a complete list of the files which were inspected by the worm. I am not aware of any confidential file that was read by the Morris worm. In particular the /etc/hosts.equiv file (which is one of the ways the Morris worm discovered IP addresses of other machines) is readable by any authorised user.

So it seems difficult to make a very strong case for a confidentiality violation on any file.

It also seems difficult to make a very strong case for an integrity violation on any file. The Morris worm attacked the dynamic image of an executing file (the fingerd utility) but apparently it did not modify system files (as stored in the filesystem).

Marking notes: this is another difficult question which should be marked leniently. 2 marks for any coherent discussion. Partial credit may be assigned to somewhat weaker discussions.

Further notes: Some students may have assumed that the passwords are stored in a file that is readable only by processes with elevated authority. This is a reasonable assumption, and a student should be given full credit if they make this assumption and then assert that the Morris worm breached the confidentiality of the password file. However, the /etc/passwd file on a Unix system is readable by *any* authorised user. Because the passwords are not in plaintext they are moderately secure, but they are still vulnerable to dictionary attacks – see

http://dblp.uni-trier.de/rec/bibtex/journals/cacm/MorrisT79. Spafford's 1988 technical report http://homes.cerias.purdue.edu/~spaf/tech-reps/823.pdf is a good source for more information on the Morris worm. Apparently the Morris worm used a small dictionary, with just 432 words, to interpret the /etc/passwd file – this revealed the passwords of the accounts whose passwords were in its dictionary. Recent Unix systems have somewhat less vulnerability to dictionary attacks, because an elevated authority is required to read the "shadow" file in which the hashed passwords are stored.

c. Assume, for the moment, that the internet worm did *not* modify any file on any Unix filesystem, and that it read only files which could be legitimately read by any authorised user of that system. Under this assumption, are your answers to the preceding two questions (2a and 2b) still correct? Explain briefly.
[1 marks]

These assumptions have already been discussed above. The confidentiality of user's passwords has been violated by the Morris worm, but it would be possible to argue that this is *not* a breach of the confidentiality of any file in the filesystem. There is no clear evidence of a violation of integrity of any file in the filesystem. However the Morris worm has clearly violated the availability of all files in the filesystems which (according to Shay) it rendered unavailable by "clogging communications" and by "forcing systems to be shut down."

Grading note: 1 mark for any coherent discussion of security violations under these assumptions.

d. Did the internet worm (as described in the quoted material above) mount an interception, modification, interruption, or fabrication attack on the legitimate messages carried on the internet? When answering this question, you should assume that the Cornell graduate student who released this worm was authorised to send internet messages (such as email) from their computer to any other user of the internet. **[1 bonus mark]**

The Morris worm interrupted communications on the internet when it caused systems to be shut down. It also interrupted communications, temporarily, when its activities "clogged" the network. The Morris worm fabricated communications when it sent messages from other users' accounts. I see nothing in Shay's description to indicate that the Morris worm had either intercepted or modified messages.

Grading note: 1 mark for any clear discussion that does not contain any errors apparent to the marker. Because this is not a particularly difficult bonus question, it can be marked stiffly: 0 marks should be awarded if the marker thinks the answer is unclear.

Q3. Consider the following regulation. "Users shall... use only the login name(s) assigned to you by the University and shall not allow any other person to use your login name(s) to access one of the Universities' computer systems without the express permission of the Director of that system."

a. Which of the five "security functions" defined in set #5 of your lecture slides are addressed by this regulation? Your answer should briefly explain how each of these functions is addressed.

This regulation directly addresses identification, and it indirectly addresses authentication, authorisation, auditing, and non-repudiation. Anyone who uses someone else's name is mis-identifying themself to the system. When someone successfully uses another person's name to login to a computer system, they must have also used their password – this is a mis-authentication. When they actually access that other person's account, this is a mis-authorisation. This activity would compromise the accuracy of the system's auditing records, for these would not be an accurate record of who actually used the system. If the mis-

identification is done without permission (as is possible under the first clause of the regulation "Users shall... use onlythe login name(s) assigned to you"), then a non-repudiation violation might be covered by this regulation: someone might use someone else's login to access a non-repudiable service, and then destroy the receipt.

b. Which of the five "security functions" are *not* addressed by this regulation? Your answer should briefly explain a reasonable way in which each of these functions might be addressed, in a more extensive set of regulations regarding the use of login names.

Total for these two questions: **[5 marks]**

Note: our department's complete regulations are published at http://www.cs.auckland.ac.nz/administration/policies/ComputerScienceComputingServices.pdf.

As argued above, all five functions are addressed.

Grading notes: Coverage of non-repudiation by this regulation is quite tangential, so full credit may be awarded if the student's answer is strong on the other four points.

Q4. Shay describes the TLS and SSL protocols at pages 320-322 as a seven-step process.

1. The client sends information to the server. The information includes ... a list of key exchange algorithms ... (including) RSA, Diffie-Hellman, and Fortezza. ...
2. The server sends a ... key exchange specification ... chosen from among those the client suggested. ... The server also sends some randomly generated data and its certificate. ...
3. ... the client ... must validate the certificate and authenticate the server... The client performs the following steps. A problem in any step causes an alert to be issued to the user. All steps must be completed to go on.
    i. Compares today's date with the issue and expiration dates in the certificate. If today's date is not in that range, the certificate is not valid.
    ii. Checks to determine whether the CA that issued the certificate is in the list of trusted CAs.
    iii. ... [verifies the] digital signature [on the certificate]. Two digest algorithms, SHA-1 and MD5, are used. If a security breach is found in one, the other provides an extra measure of security. ... [To verify these signatures, the client accesses] the CA's public key and applies it to the digital signature to get the original digest value ... and determines whether it is the correct digest value. ... these keys are public knowledge and, in fact, are stored along with the list of CAs. ... [In most browsers, you can view] a list of CAs, ... [if you] select any one of them and click View, ... details tab, ... [you will see its] Public Key ...
    iv. Compares the domain name in the certificate with the domain name of the server. ...
4. The client creates a *pre-master secret* (a 48-byte sequence), encrypts it using the servers' public key, and sends it to the server. The client will use the pre-master secret to generate a symmetric encryption key for the secure session. The server receives the pre-master secret, decrypts it using its private key, and does similar calculations to generate the key.

5. If required, the server may authenticate the client. It's a process similar to authenticating the server, and we won't go into detail here.
6. Both client and server use the pre-master secret to generate a *master secret.* To calculate the master secret, the client feeds its randomly generated data, the randomly generated data it received from the server (recall steps 1 and 2), and the pre-master secret into hash routines that generate a 48-byte sequence. The server proceeds analogously. Both client and server then feed the master secret into hash algorithms to eventually generate session keys used to encrypted ata that they exchange later in the session.
7. The client sends the server another message confirming the creation of the session key and indicates that all further messages will be encrypted using that key ... The server sends analogous information to the client. Once both the client and the server receive these last messages, the secure session is established and secure communications ... begin.

a. What message(s) are sent in step 4 on the previous page, if the preceding messages were

   1. $C \rightarrow S : (E_1, E_2, E_3, ...)$ ... [a list of key-exchange algorithms]
   2. $S \rightarrow C : (E_S, R, Cert_S)$ [a key-exchange algorithm, a random number, and the server's certificate. The certificate contains the server's public key S_P.]
   3. [The client validates $Cert_S$. If the certificate is invalid, the protocol aborts.]
   4. ??

   To receive full marks, your notation for the message(s) of step 4 should be consistent with what I have written for steps 1-3. You should also provide a brief comment in square brackets. **[2 marks]**

4. $C \rightarrow S : S_P(M)$ [M is a pre-master secret which was chosen at random by C, and which has been encrypted under the server's public key S_P. Both the client C and the server S know some function *f()* which will allow them to compute the session key *f(*M*)* from M.]

Grading notes: To obtain full credit, the student must have both this protocol step (but of course any variable name may be used in place of M) and at least a brief comment.

b. In slide #19 of lecture set #5, a protocol for "user authentication" was shown. Would that protocol be appropriate for use in step 5 of the SSL protocol (as described by Shay)? Explain. **[1 bonus mark]**

There are only 17 slides in set #5; the relevant slide is #13.

The client C knows the server's public key, so it would be possible to run the protocol. In the first step, the client sends their name (U), public key (C_p), and a timestamp t_c. This message is encrypted under the server's public key.

In the second step of the user-authentication protocol, the client C receives the message (U, t_c, t_s) encrypted under their public key. In the third step of the user-authentication protocol, the client sends their identifier U and the server's timestamp t_s back to the server. At this point the server is assured that the client knows the private key corresponding to the public key (C_p) sent in the first step of the user-authentication protocol. This assures the server that the client is the owner of key C_p, and this is a form of authentication. However if the server has not (in some previous transaction) associated C_p as the public key for the client whose name is U,

then there has been no authentication of U. The most we could say is that the client has identified themselves as the owner of $C_{p.}$ If, in future transactions, this client uses the same public key, they could be authenticated as the "same client" who has communicated with the server on a prior transaction.

A more appropriate method of client authentication would be for the client to send a certificate $Cert_U$ to the server in the first step, along with a timestamp. The remaining steps would be the same as shown in Shay's Figure 10.10. The client's certificate would contain the client's name (U) and their public key (C_p), and it would be signed by a trusted third party who has confirmed that the person who owns this certificate really is named U. This is a "bug" in the textbook Figure 10.10, for the protocol in that figure does *not* provide any meaningful authentication.

Grading notes: full credit (1 point) if the answer is understandable, reasonably correct, and points out that the protocol establishes only that the client "U" knows the private key corresponding to the public key Cp they sent in the first step. This authenticates the client as the owner of Cp but it does not prove that the client's name is "U".

Further notes: When client authentication is requested by the server in SSL, the client must send a certificate, not just a public key and their (claimed) name U. To learn a bit more about SSL, you could read Sun's 1998 technical report "Introduction to SSL", which is available at http://docs.sun.com/source/816-6156-10/contents.htm.

c. Use a browser to examine the certificate offered by the SSL server at https://adb.ec.auckland.ac.nz/adb/. What is the date on this certificate? What CA signed this certificate? What is the domain name on the certificate? Did your browser advise you to trust this certificate? Do you think this advice is appropriate? Explain. **[2 marks]**

There are two dates on this certificate. It is "valid from" 28 Aug 07 4:21:22pm, and it is "valid to" 28 August 2008 11:35:49am.

The certificate is signed by its "issuer": Thawte Premium Server CA. The certificate identifies the issuer by their physical address (Cape Town, ZA) and by their email address premium-server@thawte.com.

The domain name of the certificate is specified in its "Subject" field: ".adb.ec.auckland.ac.nz". This domain is associated with the physical address "Dept. of Computer Science, The University of Auckland, Auckland, Auckland NZ".

My browser (IE7, fully patched, running on XP) advised me to trust this certificate. This seems appropriate, as today's date is within the validity range of the certificate (step 3i), the issuer is a well-known certificate authority (Thawte was the fifth on the list when I googled for "certificate authority"), the domain on the certificate corresponds to the URL on which it was found (step 3iv), and the physical address corresponds to what I would expect for my department's assignment dropbox.

Grading notes: full credit if the student is able to interpret the certificate accurately, and if they check the certificate's date- and domain-validity.

d. Use a browser to examine the certificate offered by the SSL server at https://jobhound.cs.auckland.ac.nz/index-s.php. Did your browser advise you to trust this certificate? Do you think this advice is appropriate? Explain. **[1 mark]**

My browser (fully patched IE7 on XP) advised me *not* to trust this certificate. It issued the following warning message

The security certificate presented by this website was not issued by a trusted certificate authority.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server. We recommend that you close this webpage and do not continue to this website.

- Click here to close this webpage.
- Continue to this website (not recommended).
- More information

If you arrived at this page by clicking a link, check the website address in the address bar to be sure that it is the address you were expecting.

When going to a website with an address such as https://example.com, try adding the 'www' to the address, https://www.example.com.

If you choose to ignore this error and continue, do not enter private information into the website.

For more information, see "Certificate Errors" in Internet Explorer Help.

The certificate is within its valid date range (17 Sept 07 to 3 Oct 08). It is issued by the same authority, Thawte Premium Server CA, as the certificate analysed in the previous problem. However the certificate was issued for the domain “unisign.auckland.ac.nz”, and this SSL server is at “jobhound.cs.auckland.ac.nz”. The discrepancy in domains is probably what triggered IE7’s warning message – certainly it is dangerous to rely on a certificate issued for some other domain! However in this case I think it is ok to trust the certificate, as the SSL server at jobhound.cs.auckland.ac.nz is in a subdomain of the University of Auckland. Probably our department is just saving some money, by using an “almost-right” certificate for this server.

Grading notes: full credit (1 mark) for noticing the discrepancy in the domain.

Q5. Some specifications for the USRobotics 24 Port 10/100 Mbps Switch and the USRobotics 8 Port 10/100 Ethernet Switching Hub were given in lecture set #6.

a. Consider an application where minimum-length 100 Mbps Ethernet packets are being sent continuously from one station to another station. If these two stations are on the same link, and if no other stations were transmitting on that link, approximately how many packets per second would be transferred between these two stations? **[2 marks]**

According to Figure 9.13 of Shay, there are 7+1+6+6+2+46+4 = 72 bytes in a minimum-length Ethernet packet. If a station were transmitting such packets continuously at 100 Mb/sec, there would be (100 Mb/sec)/(72 B/packet)/(8 b/B) = 173,600 packets/sec on that link. Note: I have rounded to the nearest 100 packets/sec, but full credit can be given to students who don’t round. I believe the clock on an Ethernet transmitter must be accurate to +/- 0.01%, so rounding to 4 digits would be appropriate – but I haven’t read this part of the 802.3 standard carefully!

It is reasonable to assume that the station is adhering to the 802.3 standard, in which case it must allow a 12B interframe gap between packets. This would lower the packet rate to (100 Mb/s)/(84 B/packet)/(8 b/B) = 148,800 packets/sec.

Marking notes: There are several plausible lengths for a “minimum” Ethernet packet, such as the 64B minimum length of the packet without its preamble. No marks should be deducted for students who “forget” about the preamble or who don’t know about the interframe gap; but

students must explain their reasoning to get one mark for asserting that the minimum packet length is 64B, 72B or 84B. They must explain their packets/sec calculation (and it must be consistent with their minimum packet length assertion), in order to get the second mark on this question.

b. If a station on one port of the USRobotics 24-port switch were transmitting minimum-length packets continuously to a station on another port of this switch, and if both stations have 100 Mbps Ethernet cards, approximately how many packets per second would be transferred? **[2 marks]**

The switch has a maximum throughput of 148,800 packets/sec on an 100 Mb/s port. This is exactly the maximum throughput we calculated for a continuous transmission of 72 B packets with 12 B gaps, so it seems that the USRobotics 24-port switch will not introduce a bottleneck even when a station is continuously sending minimum-length packets.

Grading notes: Students who computed some larger packet rate (e.g. the 173,600 packet/sec which would be possible if there were no interframe gaps) should get full credit if they note that the148,800 packets/sec rate of the USRobotics switch will be a bottleneck. Students who computed some smaller packet rate should get full credit if they note that the USRobotics switch will not be a bottleneck.

c. If a station on one port of the USRobotics 8-port switching hub were transmitting minimum-length packets continuously to a station on another port of this switching hub, and if both stations have 100 Mbps Ethernet cards, approximately how many packets per second would be transferred?**[1 mark]**

An Ethernet hub is a very simple device. It merely reshapes the incoming signal (or signals, in the case of full-duplex links), and would not create a bottleneck – even if a station were sending packets of less than the minimum size or without the interframe gaps specified in the 802.3 standard. However it is reasonable to assume that the stations *are* fully 802.3-compliant, in which case the maximum transfer rate would be 148,800 packets/sec.

Grading notes: 0 marks unless the student gives a reasonable explanation for why a hub is unlikely to create a bottleneck.

Q6. Slide 18 of lecture set #6 shows a forwarding database for what Halsall calls a bridging hub.

a. What would Shay call this device? **[1 mark]**

According to Shay's discussion on p. 467, this is a "switch", because it is a layer-2 device with more than two ports. If it were a layer-2 device with just two ports, Shay would call it a "bridge". If it were a layer-1 device with more than two ports, Shay would call it a "hub".

Grading notes: 0 marks for an answer without an explanation. The explanation should include the level of the device (or at least its ability to route)

b. If station 16 was disconnected from this system, what would happen to its entry in this forwarding database? When answering this question, you should assume that this bridging hub uses the "transparent bridge" algorithm described in lecture set #6. **[1 mark]**

After about 3 minutes, the entry for station 16 would be purged from the forwarding database. See slide #14 of lecture set #6.

Grading notes: 0 marks for an answer without any explanation.

c. If stations 1 and 9 are swapped, so that station 1 is on the right-hand repeater hub and station 9 is on the left-hand repeater hub, how long might it take for the bridging hub to "learn" their new locations? Explain briefly. **[2 marks]**

The bridging hub will "learn" their new locations as soon as they send a packet. The source address field on the packet will identify the station, and the bridging hub will adjust their entry in its forwarding database to reflect the new port number.

Grading notes: This should be an easy two points, but some explanation is required to get any marks.

d. If stations 1 and 9 are swapped, so that station 1 is on the right-hand repeater hub and station 9 is on the left-hand repeater hub, would the repeater hubs "learn" their new locations? Explain briefly. **[1 mark]**

Repeater hubs do not have a forwarding database, and so they never "learn" anything about the location of the stations.

Grading notes: Some explanation is required to get credit for an answer.

e. Neither the USRobotics 24-port switch nor the 8-port switching hub are specified as being 802.1Q compliant. Referring to the last slide of lecture set #6, you'll find that an 802.1Q VLAN packet is distinguished from normal ethernet packets by having an 0x8100 in the type/length field of 802.3. What do you suppose would happen if an 802.1Q VLAN packet is received on the input port of a USRobotics 24-port switch? What if it were received on the input port of a USRobotics 8-port switching hub? **[1 bonus mark]**

I'll assume that the 24-port switch is *not* compliant – otherwise USRobotics would probably advertise this as a feature and charge more for the device!

When the switch receives an 802.1Q packet, the length field will contain 0x8100, which is an illegal length for a packet. The switch would probably discard the packet, because it won't be able to find its FCS. When the link goes idle, the switch will be able to detect this condition, and thus will be able to handle future incoming packets on that port.

Because the 8-port hub merely interprets at the bit level, it will not be able to distinguish 802.1Q packets from normal (non-VLAN) Ethernet packets. The VLAN packets will be broadcast by the hub.

Grading notes: Full credit could be awarded to any student who writes an understandable and plausible answer. For example, a student could be given full credit for arguing that the non-compliant switch *might* forward the VLAN packet correctly (if the destination address is in its forwarding database). The switch will certainly be able to interpret the source and destination addresses correctly in the 802.1Q packet, for these parts of the packet protocol is identical. There's no obvious reason why the switch couldn't work at the bit-level for the remainder of the packet. If the switch doesn't attempt to interpret the type/length field, and if it can accurately distinguish the "idle" state of the link from its signaling state, then it could accurately determine the end of an 802.1Q packet. Finding the end of a packet by idle-detection rather than by interpreting the length field would, I believe, be an inferior way to design a switch, because of the relatively high cost and error rate of idle detection in comparison to field interpretation (in a device that is interpreting other level-2 Ethernet fields). However we haven't discussed detailed design of switches in COMPSCI 314, and it would certainly not be impossible to design a switch which interprets only the address portions of an Ethernet packet.
