CS 161 Computer Security
Spring 2010 Paxson/Wagner

MT1 Solns

Midterm 1 exam solutions with annotations

The annotations are set in *italic type*. They are meant to clarify how problems were graded and illuminate some common errors or misconceptions seen in the exams.

Problem 1. [True or false] (14 points)

Each correct answer was worth 2 points. No credit was given for incorrect answers even if accompanied by an explanation.

(a) TRUE or FALSE: SQL injection vulnerabilities can be avoided by applying the following algorithm to every untrusted input: replace every instance of `&` with `&`, replace every instance of `<` with `<`, replace every instance of `>` with `>`, and replace every instance of `"` with `&quot;`.

Comment: That might be part of a defense against XSS, but it doesn't help with SQL injection.

(b) TRUE or FALSE: The same-origin policy generally allows JavaScript from `berkeley.edu` to read cookies associated with `berkeley.edu`.

(c) TRUE or FALSE: The same-origin policy generally allows JavaScript from `berkeley.edu` to read cookies associated with `stanford.edu`.

(d) TRUE or FALSE: A benefit of privilege separation is that it can provide an opportunity to reduce the size of the TCB.

(e) TRUE or FALSE: If we ensure that the attacker does not have permission to read the value of session cookies stored by the victim's browser, session fixation attacks become impossible.

Comment: Session fixation attacks rely upon the attacker's ability to *write* the value of session cookies; reading is irrelevant.

(f) TRUE or FALSE: It is easier to passively eavesdrop on UDP traffic than on TCP traffic.

Comment: *It is more difficult to* spoof *TCP traffic, but to simply listen in on it works the same as does listening in on UDP traffic.*

There is a separate question of how much work it is to then analyze the captured traffic. While TCP is more complex than UDP in this regard, the difficulty will also depend on the application protocol; some UDP-based protocols are very complex, such as file system protocols like SMB and NFS.

(g) TRUE or FALSE: Ingress filtering refers to carefully escaping meta-characters in URLs.

Comment: Ingress filtering refers to filtering of network packets to avoid forged source addresses.

Problem 2. [Short answer] (22 points)

(a) BankOBits is a local bank that offers its customers access to a number of conveniently located ATMs. Normally, when a customer inserts his/her ATM card into a BankOBits ATM, the ATM will contact the BankOBits central server to validate the ATM card inserted into it and check that the corresponding account has sufficient funds before allowing the user to withdraw money. However, if the server does not respond, the network connection is down, or something else goes wrong with this query, the BankOBits ATM will assume all is well, allow the customer to withdraw up to $300, keep a record of the transaction, and upload that information to the BankOBits server whenever connectivity is restored. As a result of this design decision, a gang of criminals are able to steal from the bank by cutting the network connection on BankOBits ATMs and withdrawing $300 from them using a fake ATM card.

In this story, which security principle was violated? Circle one (the best answer), and briefly explain.

(i) Least privilege. (ii) Fail-safe defaults.

(iii) Separation of responsibilities. (iv) Human factors matter.

Justification: The default is to give the customer $300 in case of doubt, which is unsafe because it does not check whether the card is valid or the account has sufficient funds.

Comment: *Nearly all of the exams answered this problem correctly.*

(b) David Wagner once heard about a kiosk at one airport that let you access the web, for a fee. To use the kiosk, you had to enter your credit card information at a welcome screen before the kiosk would give you access to a web browser. However, some hacker discovered that if you press F1 to invoke the "help" screen, the Windows help subsystem would pop up a window with generic help information about the login screen. The help text happened to contain a link to an external web site with more help information, and if you click on that link, the kiosk would open the Internet Explorer web browser to display that web page. At that point, one could change the URL in the Internet Explorer address bar and gain full access to the web, without paying.

In this story, which security principle was violated? Circle one (the best answer), and briefly explain.

(i) Fail-safe defaults. (ii) Separation of responsibilities.
(iii) Complete mediation. (iv) Psychological acceptability.

Justification: The system does not control all ways to obtain access to the web.

Comment: *Nearly all of the exams answered this problem correctly.*

(c) The Fortune 500 company FooCorp has an internal web application that its employees can use to fill out travel vouchers. Unfortunately, FooCorp's system administrators have recently discovered that the voucher web application has cross-site request forgery (CSRF) vulnerabilities. FooCorp has a firewall that blocks all inbound connections from the external world to FooCorp's internal network, but allows all web connections initiated from machines on FooCorp's internal network.

Does FooCorp's firewall prevent exploitation of the CSRF vulnerabilities in its travel voucher application? Circle yes or no, then briefly explain (in one line or less).

(i) Yes. (ii) No.

Justification: If an internal employee visits a malicious external website, that website can exploit the CSRF vulnerability. The request to the voucher system will come from the employee's internal machine and therefore will not be blocked (or even seen) by the firewall.

Comment: *To receive full credit, an answer had to clearly convey that the attack was still possible due to visiting an external site via an outbound conneection. An answer that the CSRF vulnerability could still be exploited by an insider misses the point that it is still also exploitable by an* outsider.

(d) Suppose we are building a web application that asks the user for their email address and stores it in a variable `m`. We want to invoke the shell to send an email message to the email address `m`, like this:

```
void sendemail(char *m) {
  char cmd[1024];
  snprintf(cmd, sizeof(cmd), "mail %s", m);
  f = popen(cmd, "w");
  ...
}
```

However before we invoke `sendemail(m)`, we want to ensure that `m` is safe to use with this code. Which of the following would be the best way to do that? Circle the best (safest) answer.

(a) Check that `m` does not contain any of the following characters: `*|‘()`.

(b) Remove all instances of the following characters from `m`: `*|‘()`.

(c) Check that `m` starts with a letter (a-z or A-Z) and is composed solely of the following characters: `abc...zABC...Z0123...9@+-_.`.

(d) None of the above: This code cannot be made safe, no matter what checks you do on `m`.

Explain why briefly (one line or less):
Whitelisting is safer than blacklisting. The whitelist doesn't contain any shell metacharacters, so any string that passes the check will be interpreted by the shell as a single argument to mail and will not affect the flow of execution.

Comment: *Answers indicating (d), that code cannot be made safe regardless of checks on `m`, were only acceptable if the accompanied by a discussion of a specific remaining vulnerability. For example, it was not enough to state that "`popen` is inherently unsafe" or "`mail` could have a vulnerability." It did however suffice to frame that `mail` might allocate less than 1,024 bytes for holding its first argument, and thus a long value of `m` could overflow it, if accompanied by a statement that executable code could be encoded using only those characters allowed by the filter in (c).*

We also accepted the quite different vulnerability that an attacker could repeatedly abuse `sendemail` to send spam.

Problem 3. [Printer discovery] (16 points)

A consortium of printer vendors have come up with a great new protocol to help users automatically discover the set of printers on their local network. In this protocol, when the user wants to print something, the user's computer automatically broadcasts a Printer Discovery packet. A Printer Discovery packet is a UDP packet whose destination address is the broadcast address, and whose source and destination port is 56184. Because this is a broadcast packet, every host on the local network will receive it.

Printers constantly listen for Printer Discovery packets. Any time that they receive one, they immediately respond with a Printer Announcement packet. A Printer Announcement packet is a UDP packet whose destination address is the broadcast address, and whose source and destination port is 56185; its payload identifies the name of the printer, the printer's IP address, and any special options supported by the printer (e.g., 2-sided printing, color printing). The Printer Announcement packet is broadcast to the entire network, so that other hosts on the local network can also learn about this printer.

Whenever a machine receives a Printer Announcement packet, it checks that the source address of the packet matches the printer's IP address found in the payload. In case of a mismatch, it ignores the packet. Otherwise, it accepts the packet and adds this printer to its list of known printers. If the machine's list of known printers already contains a printer with the same name, the machine overwrites the previous entry in its list with the information found in the newly received packet.

Vicky the Victim is about to connect her laptop to a local switched Ethernet network. Her laptop will use this printer discovery protocol to look for a printer, and then Vicky will connect to one of the printers found in this way and send it a sensitive corporate document to be printed. Meanwhile, Attila the Attacker's computer is attached to this same network. Attila has the ability to inject packets onto this network and to receive all broadcast packets, but he cannot eavesdrop on other traffic. The printers are in locked rooms that Attila does not have access to, and Attila has not been able to hack or access any of the machines or printers attached to this network, so his only hope is to attack the printer discovery protocol.

(a) Can Attila arrange to learn the contents of Vicky's document, without physically accessing any of the printers? Circle either "yes" or "no", then briefly justify your answer. If you circle "yes", describe the attack; if you circle "no", explain why this kind of attack is not possible.

(i) Yes. (ii) No.

Justification: Attila can observe Vicky's Printer Discovery packet and the real printers' Printer Announcement packets, then (before Vicky prints the document) broadcast Printer Announcement packets containing Attila's IP address but the name of the other printers. When Vicky prints her document, she will send it to Attila, and Attila can see the contents of the document. Attila can then optionally forward the document on to the printer so Vicky doesn't notice anything amiss.

Comment: *As explained on the newsgroup, depending on the specifics of how Vicky's printing works, the attacker might need to reply before a legitimate printer does, or after it does. Either type of answer was acceptable, as was spoofing an existing printer's identity or creating a fake, apparently additional printer. Full credit required clearly conveying an understanding that broadcast allows Attila to monitor the request and response traffic. We also allowed a quite different solution inspired by Homework #2, where Attila floods the forwarding table used by the network's switch so that Vicky's print job would be broadcast, enabling Attila to observe it.*

(b) Can Attila modify what is printed on the printer? In other words, Attila wants to *replace* Vicky's chosen document with something else Attila has chosen, hopefully without Vicky noticing. It's not acceptable if Vicky's original document gets printed in addition to Attila's replacement, because then Vicky might notice and get suspicious; Attila is only interested in an attack that causes his document to be printed *instead of* Vicky's. Can Attila mount such an attack, without physically accessing any of the printer? Circle either "yes" or "no", then briefly justify your answer. If you circle "yes", describe the attack; if you circle "no", explain why this kind of attack is not possible.

(i) Yes. (ii) No.

Justification: Do the same as in (a), except modify the document before forwarding it on to the printer.

Comment: *Full credit required both conveying the notion of a man-in-the-middle attack and also suppressing the printing of Vicky's document.*

Problem 4. [DDoS] (19 points)

This question asks you to consider a (hypothetical) anti-spam company called Turquoise Security Inc. Turquoise Security uses a vigilante approach to fighting spam: when one of Turquoise Security's users identifies an email they've received as spam, Turquoise Security's servers automatically visits all the websites advertised in the spam message and leaves generic complaints on those websites. Turquoise Security operates on the assumption that as their user base grows, the flow of complaints from hundreds of thousands of computers will apply enough pressure on spammers and their clients to convince them to stop spamming.

Yesterday, Turquoise Security's web site came under a massive DDoS attack using a variety of techniques. The attackers are using DNS amplification: the attackers identified several third-party DNS servers that will respond to any DNS query, and are sending many spoofed DNS queries to those DNS servers with a forged source address. In particular, each query is sent in a spoofed UDP packet, where the source address on each of these DNS queries is forged to be the IP address of Turquoise Security's web server. Also, each query has been chosen so that it will trigger a response that is much larger than the query itself, amplifying the effect of the attack. This attack has overloaded Turquoise Security's web server with huge amounts of traffic.

(a) Consider the packets that Turquoise Security's web server received as a result of this DNS amplification attack. For each of the following fields in the IP header, state whether you expect that field to be the same for all of these packets or to differ from packet to packet (circle one choice). If you select "same", describe the value of that field, in the space to the right. If you select "differs", briefly justify your answer, in the space to the right.

Source address:	(i) same:	
	[(ii) differs:]	varies among addresses of third-party DNS servers
Destination address:	[(i) same:]	address of Turquoise web server
	(ii) differs:	
Source port:	[(i) same:]	port 53 (the DNS port)
	(ii) differs:	
Destination port:	(i) same:	
	[(ii) differs:]	DNS queries likely use source port randomization

Comment: *Credit was only given for answers that both correctly identified which of* same/differs *applied and supplied a valid reason for why this was the case. Missing or invalid reasons received no credit even if the student did indicate the correct choice.*

The first three elements were worth 2 points each; the last was worth 1 point (due to ambiguity regarding the correct answer). For the first three, reasons that were only somewhat correct received half credit. The last required a fully correct reason.

Some answers assumed either the attacker used only a single third-party DNS server, or that for multiple servers some of them accepted queries on different ports. These received partial credit.

Per the discussion on the newsgroup, a case can be made for the Destination port *not changing* if *either the attacker decided to fix the source port in their spoofed requests (perhaps because the tool they used to generate these does not randomize the port) or because the way in which the attacker generates the request goes through an old-style resolver that uses the default DNS port (53) as the source port on its requests. Thus, answers for* Destination port *that indicated either of these cases received credit. However, answers that indicated the port would always be either Turquoise Security's DNS port, or their web server port, did not receive credit, because (1) there is no reason to assume that Turquoise Security's DNS server sends its requests using the old style of fixing its source port to 53, and (2) it does not help to flood Turquoise Security's web server with* UDP *traffic sent to a port with the same number as that used by the web server for* TCP *traffic.*

(b) Suppose that Turquoise Security's ISP, the company who provides Turquoise Security with their Internet connection, would like to help Turquoise Security survive this attack. Name one thing that their ISP could do in response to this attack, to relieve the load on Turquoise Security's servers for now.

Answer: They could block all incoming UDP packets, or all incoming UDP packets with source port 53 (assuming the Turquoise web server does not perform any DNS queries itself).

Comment: There are probably other reasonable answers.

*Full credit required specifying either one of the above answers, or: (1) filtering all of the DNS servers seen to be sending the requests (*not *simply "block malicious DNS servers", as the servers need to be identified and in fact are not malicious); (2) a stateful firewall could drop incoming replies for which it had not seen a corresponding request; (3) the ISP could change the address allocated to the web server; (4) the ISP could apply* per source *rate-limiting (not just general rate-limiting).*

A bit of partial credit was given for answers that were either vague ("filtering") or would entail significant "collateral damage", such as severing all Internet connectivity, rate-limiting all incoming connections, or requiring Turquoise Security to split their web service across multiple servers/IP addresses.

Solutions that specified multiple reasons received partial credit proportional to the number of correct reasons versus the number of total reasons.

(c) Today, DNS servers accept queries via the UDP protocol. But imagine that DNS had been designed differently, so that DNS used only TCP (not UDP) and DNS servers accepted queries only via TCP (ignoring all UDP packets). Would this make the DNS amplification attack described above easier, harder, or have no effect? Circle one answer, then briefly explain your answer (in one line or less).

(i) The attack would be easier. (ii) No effect. **(iii) The attack would be harder.**

Justification: The attackers would have to guess TCP Initial Sequence Numbers to complete the three-way handshake. These days TCP ISNs are usually random unguessable 32-bit numbers, so guessing them is hard.

Comment: It's true that if the attackers sent SYN packets to the third-party DNS servers with the source address forged to be that of Turquoise Security's web server, those DNS servers would respond with a SYN|ACK packet to Turquoise's web server. However, this would not provide any amplification, so it's not a DNS amplification attack (and it's not clear it has any benefit over just sending packets directly to Turquoise with a spoofed source address). Also, such an attack is not specific to DNS—it's an attack that can be mounted against any server that is listening on any known TCP port.

Full credit required mention of a reason why it is difficult for an attacker to spoof a TCP connection, or mention that use of TCP would result in loss of amplification coupled with an explanation of why. Answers such as "it requires more work on the attacker's part" or "the server will drop the TCP packets" received partial credit, as did answers that stated the change would have no effect coupled with *reasoning that clearly conveyed the view of the attacker targeting per-packet load on the victim rather than per-byte load (the latter of which benefits from amplification).*

(d) Setting aside the current DDoS attack on Turquoise Security, how could the Turquoise Security service itself be used to mount a DoS attack on others?

Answer 1: An attacker could send millions of spam messages, containing links to a target web site; when the recipients mark those as spam, Turquoise will mount a DoS attack against the target.

Answer 2: An attacker could send an email containing thousands of links to the same web site to a conspirator who is a Turquoise customer, and ask the conspirator to mark that email as spam. Then Turquoise servers will send one complaint per link in the email, providing an amplification effect.

Comment: *Full credit required conveying a means by which many URLs of a given target site would be falsely reported to Turquoise Security as spam. Answers that did not clearly state how the attacker would arrange for many such URLs to be reported received at best partial credit. Answers that assumed that Turquoise Security itself would act in a malicious manner did not receive any credit, as that is missing the basic point of the problem.*

Problem 5. [Firewalls and NATs] (14 points)

(a) Can a stateless firewall (such as a packet filter) enforce the following policy?

Policy: Block TCP connection initiation requests from any external host to any internal host. Allow TCP connection initation requests from any internal host to any external host, and also allow returning traffic on these connections initiated by internal hosts.

You may assume that the internal hosts (those on the inside of the firewall) all have IP addresses of the form 128.32.153.x, where the x can be anything in the range 0–255, and no external host has an IP address of this form. You may assume that the TCP/IP stack on every internal host operates correctly.

Circle "yes" or "no", depending on whether you think a stateless firewall (such as a packet filter) can enforce the policy above or not, then briefly explain your answer (in one line or less).

(i) Yes, it can enforce the policy. (ii) No, it cannot.

Justification: Block any inbound packet that does not have the ACK bit set. Allow everything else.

Comment: *Full credit required a clear indication that the firewall drops incoming SYNs* without *the ACK bit set, and allows traffic with ACK set. Mention of SYN without mention of ACK, or mention of "flags" without specifying which, was only worth partial credit. Correct answers that in addition included incorrect reasoning received only partial credit.*

Numerous answers stated that a stateless firewall cannot have any memory of past connections and so cannot enforce the policy. However, we explicitly covered in lecture how for TCP such filtering is possible due to the protocol's use of the ACK bit.

(b) Name one security benefit that NAT provides.

Answer 1: A NAT prevents external hosts from initiating connections to internal hosts.

Answer 2: A NAT prevents external hosts from scanning of the internal network.

Comment: *Other answers that received full credit include: (1) prevents the Kaminsky attack by randomizing source ports; (2) ameliorates DoS floods by preventing most of the traffic from reaching the internal target; (3) use of a single public address prevents internal hosts from spoofing IP addresses; (4) the NAT hides the internal network's topology; or (5) an external eavesdropper has more difficulty discerning just who is communicating since multiple private addresses all appear as the same public address.*

Answers that only hinted at the above, but did not explicitly frame one of these benefits, received only partial credit.

Answers that simply stated that addresses are hidden, or that it's hard for an attacker to locate a given internal host, or that an external attacker cannot eavesdrop on internal hosts (which is the case regardless of whether a site uses a NAT) did not receive any credit. These lack the notion of attackers being unable to initiate *connections to internal hosts, coupled with attackers still being able to exploit hosts that wind up connecting to them.*

Answers that stated that the NAT provides a chokepoint for enforcing policy did not receive any credit. Any network element at a site, such as a simple router, provides this benefit.

Problem 6. [Secure coding] (15 points)

Consider the following C code:

```
/* Information about the current CD. */
struct cd {
  int numtracks; /* The number of tracks on this disc. */
  int tracklen[16]; /* The length of each track on the disc, in seconds. */
  void (*notify)(struct cd *); /* Call this whenever the CD info changes. */
};
```

```
struct cd *curcd = makestructcd();

/* Update the length of track number 'track'. */
void update_cdinfo(int track, int newtracklen) {
  if (track > 16)
    return;
  curcd->tracklen[track] = newtracklen;
  (curcd->notify)(curcd);
}
```

(Don't worry about `makestructcd()`; it just allocates and initializes a `struct cd`.) Assume the adversary can arrange for `update_cdinfo()` to be called with whatever values of `track` and `newtracklen` he likes (those values may have been read directly off the CD, for instance). Answer the following questions about this code, *concisely*:

(a) What is the security vulnerability in this code?

Answer 1: Buffer overrun (or array out-of-bounds error): if `track=16`, then this writes one past the end of the `curcd->tracklen` array.

Answer 2: Buffer overrun (or array out-of-bounds error): if `track` is negative, the array dereference `curcd->tracklen[track]` writes outside the bounds of the `curcd->tracklen` array (it writes before the start of the array).

Either answer is acceptable.

Comment: *The code does* not *exhibit an integer "conversion" or "overflow" error. The* `int` *type in C is signed, and thus values that slip by the test by being negative do not require any notion of an error in interpreting an* `int` *value.*

Full credit for this part of the problem required exhibiting an understanding (perhaps including the discussion in the writeup for the next part) of both the deficient bounds-checking in the code, coupled with the notion of memory safety. *In addition, such answers had to use terminology correctly. For example, there is no TOCTTOU vulnerablity in the given code.*

(b) How could an attacker exploit this vulnerability to trigger the execution of malicious code? Describe how the attacker should choose the values of `track` and `newtracklen`.

Answer 1: Set `track=16` and make `newtracklen` the address of malicious code loaded somewhere in the address space of this program. This will overwrite `curcd->notify` with a pointer to malicious code.

Answer 2: Set `track` to some large negative number chosen so that `curcd->tracklen[track]` references, e.g., a return address stored on the stack somewhere, and make `newtracklen` the address of malicious code loaded somewhere in the address space of this program. (There are many possible variations on this answer.)

Comment: *Full credit required three elements. First, indicating a specific value for how to set* `track` *in order to enable a memory-safety violation. (Answers of "set it to a negative value" were acceptable here if there was also mention of finding a* return address *somewhere on the stack, or an interpretation that the stack grows in a direction such that the specific value of* `-1` *would access the* `notify` *function pointer.) Second, discussing how to set* `newtracklen` *to the* address *of malicious code somewhere in*

memory. Third, discussing how control would be transferred *to the value loaded via* `newtracklen`*: either by modifying the* `notify` *structure element, or by altering a return address on the stack.*

Solutions that contained correct elements as discussed above, but also included incorrect information were given partial credit.