ECE 458/750: Computer Security Final			Marks obtained ↓
Date: August 17, 2026,	Total questions: **17**	Total points: **71**	
ID:	Name:		Time: 2.5 hours

Instructions

No aids allowed. All you are allowed is a pen and pencil.

Use space provided. Answer the questions in the spaces provided. If you run out of room for an answer extra pages are provided at the end of the test booklet starting on page 22. They are clearly marked as EXTRA ANSWER SPACE. Please state in the original answer space if the extra pages are being used so that the grader knows to look there.

Point value in right-hand margin. The number of points each question is worth is listed in the right hand margin. The number of points and the space provided are roughly indicative of how long of an answer is expected.

Pencils and pens allowed. Both pens and pencils are allowed. Pencil marks need to be clearly present or erased. If it is unclear if a mark is or is not present then it is up to the discretion of the grader and this point cannot be contested later.

Fill in or circle multiple choice answers. Multiple choice and multi answer questions will have boxes or circles next to the answer options. You may fill these in, clearly mark them, or circle the whole answer. It needs to be clear which answer option(s) have been selected.

Figures provided. Some content has been provided from the lecture slides. This content appears as figures, and is directly referenced by the question were we expect that you will need the figure information. For example: "The rules on Figure 1 may be helpful in this question."

Grad only questions. The last question is only for ECE750 students. It is very clearly marked as grad only. Undergraduates taking ECE458 do not need to complete this question, and any answers they provide will be ignored when assigning final grades.

Solution: This version of the exam has solutions to the problems. The solutions presented here are one possible solution but for open-ended questions they may not be the only possible solution. The solutions also represent how the instructor originally intended the question to be interpreted. When marking, the instructor takes into account that students may interpret questions differently than intended which may alter the correct answers.

POST MARKING NOTES: While marking the exam we take note of common student confusions and unanticipated issues with how students interpreted exam content. Boxes marked "POST MARKING NOTES" contain comments from the Instructor or TAs about how we handled some of the more common cases.

General Security Questions

1. A malicious actor launches a ransomware attack which encrypts all the files on a system while ensuring that only the malicious actor has the private key. Which CIAAA principle is undermined when a user is unable to open a file? Name the principle and provide at most one sentence of justification. (2)

Solution: Availability - The ransomware attacker is using encryption to block the access to the file for users that should have read access.

Solution: POST MARKING NOTES
Accessability also accepted.

2. Imagine that you are playing a YouTube video and an advertisement comes up. You do not want to watch the advertisement. How might you use the URL below to avoid the advertisement? (2)
This is a real older attack that has now been fixed by YouTube and therefore no longer works.

https://www.youtube.com/watch?v=GZniJBygnX8&t=133s

Solution: Modify the "t=133s" to a slightly larger number like "t=134s". A better approach would be to change it to a much larger number and then rewind since rewinding does not cause ads.

This is a "how" question, so simply stating something like "t=150s" is not enough.

Solution: POST MARKING NOTES

This question is a nearly direct copy out of the "Modify URLs" activity, though it is intended to test understanding of URLs which was taught in lecture.
`https://ece.uwaterloo.ca/~kvaniea/teaching/ece458/S2026/activities/modify-urls/index.html`

Notes:

- **Students clearly did not do the activity** - Since the activities are not directly required, this question was graded based on accurate understanding of how URLs are read, modified, and how web requests work. It was not graded based on an understanding of how YouTube in particular works. Hence some full mark answers are technically wrong for YouTube but are reasonable first things to try on an unknown video website.
- **Interpret URL as correct answer** - Some students interpreted the question as "How might this specific URL cause ads to not display?" If this interpretation was clear from the explanation, then we accepted reasonable answers that explained why this particular URL might cause an ad to be skipped.

3. Which of the following access control structures we learned about best approximates how Linux manages access control of files? (2)

- ◯ Access Control Directory
- ◯ Access Control Matrix
- ◯ Access Control Triples
- ◯ Access Control Lists

Solution: Access Control Lists

4. A software vendor discovers a buffer overflow vulnerability in its web server and assigns it the identifier CVE-2026-12345. (2)

   What does this CVE identifier represent?

   - ◯ A category of programming mistakes that can occur in many applications
   - ◯ A specific publicly documented vulnerability in a particular product
   - ◯ A security patch released by the vendor
   - ◯ A security risk score assigned to the vulnerability

   Solution: A specific publicly documented vulnerability in a particular product

Public/private key cryptography

5. Alice and Bob decide to use the third party chat app *ChatApp* to communicate which is developed and run by Eve. *ChatApp* uses a central server where it stores all messages. Because they do not fully trust *ChatApp*, they decide to use the Linux GPG program to securely do public/private key encryption and decryption of messages. They then copy/paste the encrypted messages in/out of *ChatApp*. Alice and Bob only ever communicate over *ChatApp* and do not independently verify or sign their keys in advance. (6)

   Imagine that the following scenario occurred where Eve was successfully using her control over *ChatApp* to perform a Man-in-the-Middle attack on Alice and Bob's communication.

   - Alice and Bob attempt to exchange public keys using ChatApp. Eve intercepts both requests and sends her own public key to each user while pretending to be the other person.
   - Alice and Bob then begin using GPG and *ChatApp* send and encrypt/decrypt messages. Both can successfully encrypt and decrypt messages and neither notices anything unusual. Eve can read all messages exchanged between them.
   - Eve is able to read all messages sent between Alice and Bob.

   In the following table, place a mark (✓or ■) in each box corresponding to a key (row) that the person (column) possesses at the end of the attack described above. Assume that no keys were exchanged except through ChatApp in the scenario described above.

Key ↓	Alice	Bob	Eve
Alice's Public Key	☐	☐	☐
Alice's Private Key	☐	☐	☐
Bob's Public Key	☐	☐	☐
Bob's Private Key	☐	☐	☐
Eve's Public Key	☐	☐	☐
Eve's Private Key	☐	☐	☐

Solution:

Key ↓	Alice	Bob	Eve
Alice's Public Key	✓		✓
Alice's Private Key	✓		
Bob's Public Key		✓	✓
Bob's Private Key		✓	
Eve's Public Key	✓	✓	✓
Eve's Private Key			✓

Because Eve switches the keys during the MITM attack, Alice never gets Bob's public key and Bob never gets Alice's public key. But Eve gets both keys.

Solution: POST MARKING NOTES

Grading was done based on misconceptions. 1-1.25 points were lost for a violation of each of the following:

- Each user, and only that user, has their own private key
- Each user has their own public key
- Eve has Alice and Bob's public keys (they were sent to her)
- Alice and Bob both have Eve's public key (she sent it to them)
- Bob does not have Alice's public key, and Alice does not have Bob's public key (Eve blocked the transmission)

6. The attack in question 5 works because both Alice and Bob are making an inaccurate assumption. What inaccurate assumption are they making? (2)

Solution: Alice encrypted her messages using Eve's public key because she incorrectly believed it belonged to Bob. The same is also true for Bob, as he incorrectly believes that Eve's public key is Alice's public key.

OR

Alice and Bob assume that the public keys cannot be changed in transit, so they are not considering the possibility that Eve could change the files.

7. A legitimate company named Acme contracts with a root-level Certificate Authority (CA) to obtain a new certificate. The root CA generates a new key pair and sends both keys back for Acme to install. (3)

The above described scenario is a very bad way to do security, it breaks a key assumption in cryptography.

What attack becomes possible if Certificate Authorities worked this way?

Solution: The assumption is that private keys are private and only possessed by one person or one entity.

Now the CA has the private key of the company. They can now pretend to be that company. They can send the certificate (which they, and everyone else, have) and they can prove that they are the correct company by signing content with the private key. This lets them setup a TLS connection that looks like it really is from that company.

Solution: POST MARKING NOTES:
Some answers assumed that the private key that was shared belonged to the CA, as in similar to the Lenovo Superfish attack, which is not what is described in the question. The question says *a* key-pair was shared so it just means that the private key is shared between the CA and the company.

Networking

8. In the Networking module we discussed an attack done by Archive.today against a blogger (`gyrovague.com`) who wrote something the Archive.today site owner did not like. Archive.today is a website that was heavily used by Wikipedia at the time to reference paywalled citations. To accomplish the attack, Archive.today inserted the following code into the Archive.today website:

```
setInterval(function() {
  fetch("https://gyrovague.com/?s=" +
    Math.random().toString(36).substring(2, 3 + Math.random() * 8), {
      referrerPolicy: "no-referrer",
      mode: "no-cors"
    });
  }, 300);
```

(a) Name the type of attack Archive.today is doing. (1)

Solution: Distributed Denial of Service Attack (DDoS)

This is not a XSS because the attack is coming from the Archive.today web server itself, there is no "cross site".

This is a browser-based DDoS attack. Archive.today intentionally served JavaScript to its visitors. Those visitors' browsers repeatedly generated requests to gyrovague.com, creating a distributed flood of traffic. It is not XSS because no code is being injected into or executed within gyrovague.com's pages. The malicious code is hosted by Archive.today itself and was added to the website by its own developers.

(b) Explain what happens when a visitor loads the Archive.today webpage. Describe how the JavaScript affects the visitor's browser, what requests are generated, and how those requests may affect `gyrovague.com`. (5)

Solution: Every visitor to the Archive.today website will run this code. The code causes the client browser to request that a random page be loaded from the gyrovague.com website every 300 miliseconds. Archive.today is a popular website used by Wikipedia, it gets many thousands of visitors. If each visitor starts requesting a small blog's web page every 3.3 times a second, that can overload the web server. It is distributed because the "attack" is coming from individual web clients which are located world wide.

Key points in the solution:

- Visitors execute code on their browsers
- Repeated requests are made by each browser
- Requests originate from many visitors who are located in different parts of the world (distributed)
- Traffic may overwhelm the target server

Extra non-required points:

- Random URLs defeat cacheing and generate unique requests

- `mode: "no-cors"` causes the browser to send the request without doing CORS checks (which we did not learn about in class). CORS checks help enforce same origin policy. But the setting here has no real impact because it protects against reading other pages, but the goal is only to fetch the page, not to read the content. So CORS is unhelpful in this case and does not prevent the DDoS.

Solution: POST MARKING NOTES

- **Redirect to gyrovague.com** - some student clearly thought that the fetch command caused a redirect. This is not the case, especially since the attacker wanted to silence the blog, not direct people to go there. But even if this was what happened, the more serious result would still be a DDOS of `gyrovague.com`.

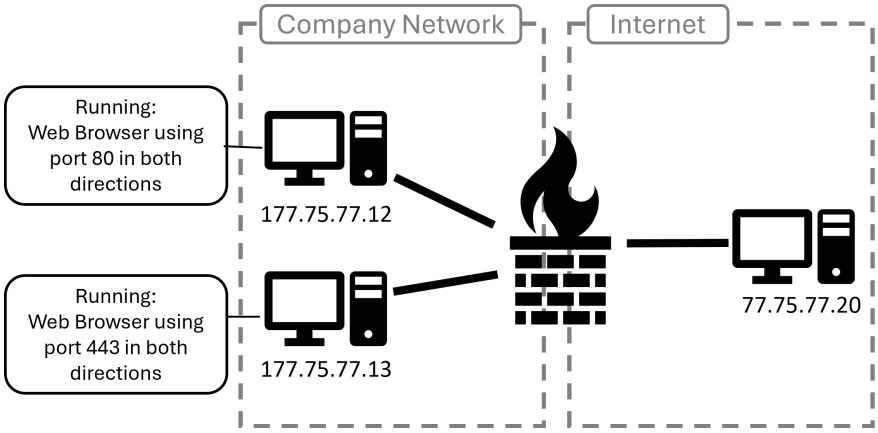

Figure 1: Network setup depicted similarly to the BlueTeam game.

Firewalls

9. Using the network diagram in Figure 1, determine what the firewall rules should be and fill in the tables below. This question is heavily based on the assigned game BlueTeam. Similar to that game, you should create a least-privilege setup that will allow the depicted computers to function. (6)

   Fill in only the parts of the table you need, leave unused rows blank.

INPUT Rules

Rule #	Policy (Reject)	Port	IP Address
1			
2			
3			

OUTPUT Rules

Rule #	Policy (Reject)	Port	IP Address
1			
2			
3			

Solution: This question is a near direct copy of level 3 of the BlueTeam game.

INPUT Rules

Rule #	Policy (Reject)	Port	IP Address
1	Accept	443	77.75.77.20
2	Accept	80	77.75.77.20
3			

OUTPUT Rules

Rule #	Policy (Reject)	Port	IP Address
1	Accept	443	177.75.77.13
2	Accept	80	177.75.77.12
3			

Additional possible solution elements which are ok:

- Adding a default Rejection or Drop rule to the end. It is implied that this is a default Reject similar to BlueTeam. But there is no harm to making that explicit.
- Alternative rule orderings within INPUT and OUTPUT. In this question the order of rules does not matter because they are disjoint. In a real firewall configuration it is possible for the order to matter, especially if Reject decisions are being used.
- Using DROP instead of REJECT

Solution: POST MARKING NOTES

- The *principle of least privilege* states that entities should have the least number of permissions that they need. That means that we should not give full read and edit access to someone who only needs to read. Similarly, the firewall should give access to IP addresses and ports that are needed, but not give access to all IP addresses and ports as that would be overly permissive.

10. Why are both INPUT and OUTPUT rules needed in the scenario in Figure 1? (2)

Solution: Both INPUT and OUTPUT rules are needed because communication is bidirectional. Clients on the company network send requests to the web server (OUTPUT), and the web server sends responses back to the clients (INPUT). If only one direction were allowed through the firewall, the communication would not work correctly. TLS, for example, requires content to be sent and received between server and client to correctly setup the connection.

Spectre

11. Spectre is an example of a side-channel attack. What is the side channel used by Spectre? (2)

Solution: Either "timing attack" or "cache state".

Spectre uses changes in CPU cache state as a side channel. The attacker causes sensitive data to influence which memory locations are loaded into the cache and then measures memory access times to determine which locations are cached, allowing the attacker to infer secret information.

Solution: POST MARKING NOTES

I allowed references to code execution time measurements if the surrounding text was otherwise correct and making it clear what was meant. Technically this is wrong. Spectre measures cache access time, which is different than code execution time.

12. Which of the following statements about the Spectre attack is **true**? (2)

- ◯ Spectre allows an attacker process to directly read arbitrary memory belonging to another process using a normal memory load instruction.
- ◯ Spectre tricks a victim process into performing a buffer overread on behalf of the attacker and then uses a side channel to infer the leaked data.
- ◯ Spectre exploits a flaw in TLS certificate validation to impersonate websites.
- ◯ Spectre requires physical access to the victim's device in order to extract secrets from memory.

Solution: *Spectre tricks a victim process into performing a buffer overread on behalf of the attacker and then uses a side channel to infer the leaked data.*

Website

Figure 2 shows a section of the server-side code of a website.

The code is written in PHP, you may assume the following:

- variables start with a `$`
- `.` means string concatenation
- `isset()` returns TRUE if the variable has a value, FALSE otherwise
- `echo()` prints a string on the resulting website
- Comments start with a `//` and can be read as accurate general descriptions of the code (obviously code is more precise).

```php
// Global session management variables
// (in reality these would be in a database or file)
$SESSION_TABLE = [];        // Maps session_id to user data
$NEXT_SESSION_ID = 1;       // Session ID counter

// --- Session Setup Logic ---

// Check if there is a sid in the GET string
if (isset($_GET['sid'])) {
    // Sanitize sid and ensure it is an integer
    $sid = intval($_GET['sid']);

    // Check if the sid is a real session ID
    if (is_valid_session($sid)) {
        // Existing valid session
        $current_session = $SESSION_TABLE[$sid];
        echo "Welcome back! Your session ID is: " . $sid;
    } else {
        // Invalid session ID provided
        echo "Invalid session ID!";
        exit();
    }

} else {
    // No session ID provided - create a new one
    $sid = $NEXT_SESSION_ID;
    $NEXT_SESSION_ID++;

    // Create new session record indexed at position $sid
    $SESSION_TABLE[$sid] = ["created" => time()];

    echo "New session created. Your session ID is: " . $sid;
}
```

Figure 2: Snippet of server code. This code has a security flaw. You may treat the comments as accurate descriptions of the code's intent. You may also ask course staff about function meanings.

13. The code in Figure 2 has a flaw (whiteboard-level design issue). Explain how an attacker could exploit the session management design shown in Figure 2. (6)

Solution: In Figure 2 the server is assigning sequential session IDs. If the user does not have a session ID, then they are assigned the next session ID (line 26) and the next session ID number is incremented (line 27).

This is an opportunity for the attacker to conduct a *session hijacking* attack. To succeed in this attack all the attacker needs is to know the session ID of the victim. Normally if the session ID was a long random string, they would need to steal it. But because this session ID is assigned sequentially, it is possible to guess it instead. If they sign up with the website, they can get their own new session ID and then choose any number less than that, odds are they will get a real session of another user.

Solution: ALTERNATIVE ANSWER
This code allows for *session fixation* where the session ID is fixed and referred to in the GET string. So an attacker might setup a session and then create a web link:
`https://example.com/login?sid=1234`
They send this link to the victim who then starts using the session ID in the URL. But the attacker already knows this session ID, so they can perform *session hijacking*.

Solution: POST MARKING NOTES

Key to this question is the hint that the is a *flaw* as opposed to a bug or a typo. Flaws are poor design decisions by the developer that they did on purpose. The code in Figure 2 is very intentionally designed to have sequential session ID numbers. A student need only read the comments and line 27 which incriments the session ID number to find the flaw.

Here are some common student answers and an explanation for them:

- **Input sanitization** - The code does sanitize the input. It checks if the "sid" GET parameter is set at all (line 9) and then casts it to a variable using a function (line 11). The comment on line 10 even states that it is sanitizing the sid.
- **Printing the sid** - Printing the sid out is a minor issue, especially given that it is part of the GET string of the URL already and therefore already visible and editable by users. Even if the sid was stored in a cookie, the user could still technically edit it if they wanted to. A web developer should *always* assume that data provided by the client may have been tampered with.
- **Stating the sid is wrong** - The website prints out a statement that sid is wrong when it is. This does provide some minor information to an attacker, but is is very minor. Most systems will tell you if the session ID is wrong or expired. In a well designed system the information loss here is very minor.
- **Only checks the sid** - This is a more problematic issue, though still far less serious than the sequential sid. It was given partial marks on the exam. This issue is quite common on websites,

especially those that do not consider their content to be particularly important. Other checks like IP address or user-agent strings are well known to be unreliable due to things like NAT. Because it only checks the sid number it is possible for an attacker to steal it and use it without detection. It would be better though to make it hard to steal the sid to start with. And the question is asking about an exploit, rather than the lack of a defense.

- **No auth** - No authentication is done before creating a new sid. This is actually not an issue for most pages. Consider amazon.ca for example, it will happily let a user who is not logged in see its pages and put things in a shopping cart. It only asks for login when the user wants to pay for the items. Many sites do this to make interacting with them easier for users.
- **Buffer overflow** - This code is not vulnerable to a buffer overflow for multiple reasons. The most simple is that input is sanitized before use (line 11) and is therefore guaranteed to be an integer within PHP's normal integer value ranges.

14. Describe two ways in which the code in Figure 2 could be made more secure. We recommend using the line numbers to be precise about what would need to be changed. (4)

(a) First improvement

Solution: The code needs to use randomly generated globally unique identifiers (GUID) instead of sequential ID numbers. Just having long random numbers as the session keys would be helpful.

(b) Second improvement

Solution: Session IDs need to expire. It is possible that is_vaid_session() on line 14 is already doing this, but that is unclear. Session IDs should not persist past a few days at most.

OR

Check other facts beyond the session ID when logging the user in. For example, check if the type of browser has changed, or if the IP address is now in a different range.

Solution: POST MARKING NOTES

Some common answers with notes:

- **Do not print sid (line 17)** - A student correctly pointed out that by putting the sid on the page, it removes the same origin policy protections granted by the browser to things like cookies. If the webpage has other vulnerabilities, this would make it easier for an attacker to steal the sid. This is a minor issue compared to the highly guessable sid, but it is a real issue that can be addressed. Hence we graded solutions that addressed line 17 with higher grades than solutions focused on line 20.
- **Do not print that sid invalid (line 20)** - An attacker can easily see if they provide a sid and that sid does not persist across pages. If this sid is being used for a login, then they can easily see they are not logged in. If it is being used to persist settings or a shopping cart, they can similarly see that these are default or empty. Hiding if a sid is invalid is not particularly useful. Especially compared with fixing the sid numbers so that they are not easy to guess.
- **Make user login** - This is a 0 points answer as it shows no understanding of what a session id is or why it is used. If the user had to login on every single page load, it would be insanely annoying and they would leave.
- **Trigger login if sid invalid** - Partial marks. Better than login on every page load. But it does not solve the problem in the code in Figure 2. A valid sid still lets the attacker in. An invalid sid is not useful to the attacker anyway.
- **GET strings are visible to MITM** - GET strings are protected if https is being used. Moving the sid from a GET string to a cookie provides no meaningful security improvement from a MITM attacker. What it does do is prevent the data from being stored in places like logs, but that is a different threat model.
- **Add encryption/hash** - Encryption and hashing will not fix the problems because it does not solve the problem. The attacker here is guessing the sid. If the goal is to make the sid more random, then just picking a random number to start with is better.
  A hash is not more random than a random number.

Cross-Site Scripting

15. A website uses cookies to keep users logged in. But an attacker discovers a Cross-Site Scripting (XSS) vulnerability on the site.

    (a) Briefly describe what Cross-Site Scripting (XSS) using JavaScript is. (2)

    Solution: Cross-Site Scripting (XSS) is a vulnerability that allows an attacker to inject malicious JavaScript into a web page. The injection is often done by putting JavaScript in a user input box like a comment box. That data is then sent to a server where it is stored (persistent) or is one-time processed (non-persistent). When a victim user visits the website, the server sends the user's browser the website which includes the attacker's JavaScript, the attacker's JavaScript then executes in that user's browser.

    XSS works because the user's browser cannot distinguish between JavaScript intentionally put on the page by the good website that is being visited, and JavaScript placed there by the attacker.

    (b) The attacker puts JavaScript into their XSS attack. Where is that JavaScript meant to be executed? (2)

    Solution: The attacker's JavaScript executes in the victim's web browser when the victim visits the vulnerable webpage.

    Solution: POST MARKING NOTES

    - JavaScript is typically used client-side and a XSS is trying to get the victim browser to execute it. We did not accept answers involving server-side execution unless there was good justification (rare).
    - **On the website** - Answers of just "website" did not get full marks because they are unclear. A website is partially built by the server and it is partially built by the client.

    (c) How could the attacker use JavaScript to take over another user's account? (4)

    Solution: The attacker could use JavaScript to read the victim's session cookie and then send it via web-request to a site they control. Because the JavaScript is executed as if it was part of the first-party website, it has access to all the first-party website cookies, including the session cookie. JavaScript also easily allows for creating web requests to fetch new content.

(d) State one server-side defense against XSS. (2)

Solution: One defense is to sanitize or encode user-supplied input before storing it or displaying it on a webpage. This prevents browsers from treating attacker-supplied text as executable JavaScript.

Memory

16. Consider the following program.

```
void function(char *str) {
    int authenticated = 0;
    char buffer[16];
    strcpy(buffer, str);
}

int main() {
    char input[128] = "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA";

    function(input);
    return 0;
}
```

Similar to Assignment 3, assume that the program is compiled as a Set-UID root program and that stack protection and address space randomization has been disabled.

(a) Identify the vulnerability. (2)

> **Solution:** The vulnerability is a stack-based buffer overflow, sometimes also known as stack smashing or a buffer overwrite. The strcpy() function copies data into a 16-byte buffer without checking the size of the input.

(b) Explain why an attacker would want to overwrite the return address. (2)

> **Solution:** The return address determines where execution continues after function() returns. By overwriting it, an attacker can redirect program execution to an address of their choosing.

(c) Draw a stack diagram for `function` showing the locations of: (3)

- buffer[16],
- authenticated,
- return address.

Indicate the order in which memory locations are overwritten.

Solution:
Stack layout:

```
Lower Addresses

+--------------------+
| buffer[16]         |
+--------------------+
| authenticated      |
+--------------------+
| Return Address     |
+--------------------+

Higher Addresses
```

If the input is longer than 16 bytes, data first writes to the buffer, then overwrites the authenticated variable, and then the return address.

To get full marks on this question, the stack order and the answer to the overwritten memory locations need to match, in addition to other answer-elements being correct.

Grad Students Only (ECE750)

Only graduate students completing ECE750 need to answer the questions in this section. We will ignore answers by ECE458 students when assigning final grades.

17. In the graduate-only assignment you watched a video by Kamara in which he describes the Vula system used by the ANC. The Vula system was intentionally designed to operate asynchronously.

    (a) The system is purposely asynchronous. What security advantage exists to making it asynchronous? (3)

    Solution: In the Vula system, asynchronous communication means that the sender and receiver do not need to be communicating at the same time. A user could record an encrypted message, leave it at a public payphone, and the recipient could retrieve it later.

    (b) The encryption scheme used words copied from books as seed values in a pseudo random number generator. In the authentication part of the class we learned about passwords and the general advice to not write a password down. (4)

    Explain why using words from books for encryption safe but writing a password down is unsafe?

    Solution: This is an intentionally challenging question in that there is not a singular correct answer. What we are looking for is strong reasoning about threats, and how these two very important secrets are being protected.

    Potential Answer 1: They are different because, unlike with normal modern cryptography, the ANC was using a proprietary system, so possession of the key was not sufficient to break the encryption by itself. So while gaining access to a password is enough data to gain access to a system immediately, that is not necessarily true that knowing the words used for seeds could break the system, the encryption algorithm must also be found. Note that in cryotography we normally assume that the algorithm is known to the attacker.

    Potential Answer 2: Books are a common household item and if common popular books are used by the ANC then differentiating between a book that is being used for encryption and a book that is not becomes quite challenging. A strong password, by comparison, does not look like other items that are normally in a home. In other words, writing down secrets can be safe if the secret looks like other non-secret items in the nearby area.

Extra Answer Space

If you need extra space to give an answer, please state in the original answer space that you will be using the extra pages and continue or write your answer here. If you choose to use the extra space as scratch paper, please write "scratch" so that graders know to ignore anything you have written. You may also tear off an extra space page and use it for scratch paper.

EXTRA ANSWER SPACE

EXTRA ANSWER SPACE