End Sem Rubric

Q1)

1(a) K-Anonymity for k = 2 and k = 3 [6 marks]

There is no single correct anonymised table. Any table is acceptable as long as it satisfies k-anonymity (for the chosen k) and only uses valid operations (generalisation/suppression). Students are free to combine any rows they want.

For k = 2 table – 3 marks

- Use of correct QIs & valid transformations – 1 mark
  - Treat Pin Code, Age, and Gender as quasi-identifiers.
  - Uses only generalisation/suppression (e.g., age ranges, truncated pin, “*”), not changing Acc Balance.

- 2-anonymity actually satisfied – 2 marks
    - All QI-patterns occur in at least two rows.
    - 2 marks: every equivalence class has size ≥ 2.
    - 1 mark: only 1–2 minor violations.
    - 0.5 mark: serious attempt but several violations.

  If they make everything "*" (trivial k-anon): max 1.5/3 (correct but minimal utility).

For k = 3 table – 3 marks

- Further generalisation for k=3 – 1 mark
    - Shows additional generalisation/row-grouping beyond the k=2 table (wider age ranges, more suppressed digits, maybe drop gender, etc.).
- 3-anonymity actually satisfied – 2 marks
    - All QI-patterns occur in at least three rows.
    - Same partial-credit scheme as above.

1(b) Increasing utility of anonymised data [5 marks]

They should realise that utility increases when we minimise suppression/generalisation by grouping "similar" rows, and that this can be achieved via a similarity metric/clustering (k-means, distance metrics, etc.).

- States the goal: reduce suppression/generalisation to improve utility – 1 mark
    - Explicitly notes that combining records with similar values keeps more specific information.
- Technique 1 – similarity/clustering–based grouping – 2 marks
    - 1 mark: Mentions a similarity metric or clustering (e.g., k-means, distance on [Age, Balance], or other similarity measure) to form groups before anonymisation.
    - 1 mark: Explains how this leads to less generalisation/suppression (e.g., ages already close → narrower ranges, pin codes already similar → fewer

digits suppressed).

- Technique 2 – any sensible refinement building on similarity – 1.5 marks
  Examples (any one, well explained):
    - Micro-aggregation (replace each cluster by the mean/median)
    - Local recoding / different generalisation for different clusters
    - Using domain-specific hierarchies (e.g., age bands, regional pin groupings) after clustering.
    - 1.5 marks for a straightforward, correct technique that again aims to keep similar rows together; 0.5–1 mark for vague but relevant ideas.
- Concrete demonstration on the table – 0.5 mark
    - Shows a small example of two or three similar rows being grouped and generalised.

Other generic answers (e.g., "publish only aggregates," "make synthetic data" without linking to similarity) can earn up to 2–3 marks in total, but full marks require the idea of similarity/clustering.

1(c) Survey + GDPR – must use Differential Privacy [8 marks]

Expected direction:
Students should explicitly choose Differential Privacy (DP), justify *why* (re-identification resistance, better than classical anonymisation), and explain what DP is and how it would be applied (e.g., DP synthetic data or noisy aggregates).

- Explicitly identifies Differential Privacy as the primary technique – 1 mark.
- Explains what Differential Privacy is – 2 marks
    - 1 mark: Correct intuitive idea: output distribution is (almost) the same whether any one person's data is included or not.
    - 1 mark: Mentions ε-DP or the notion of neighbouringdatasets/stabilityy under individual changes.
- Why DP (vs traditional anonymisation like k-anonymity) – 2 marks
  Possible points to mention (any two, well explained):

    - Resistant to re-identification even with auxiliary/background data.
    - Provides a formal mathematical guarantee, not ad hoc.
    - Publishing DP synthetic data or noisy aggregates avoids sharing raw records.
    - Better aligned with GDPR's "data protection by design" than simple de-identification.
    - Full two marks for a clear comparative explanation; 1 mark if high-level only.
- Application to this survey scenario – 2 marks
    - Describes how to use DP in practice:
        - E.g., release only DP-noisy statistics to the third party (counts, averages, histograms), OR
        - Train a DP generative model and share DP synthetic data whose distribution matches the real data but does not contain actual records.
    - 1 mark for a basic description; 2 marks if they walk through a concrete mini-example (dummy data, adding noise to counts, or describing synthetic data generation).
- Links to GDPR principles – 1 mark
    - Mentions at least one relevant principle (data minimisation, purpose limitation, integrity & confidentiality, privacy by design) and ties it briefly to using DP.

1(d) Guarantees of DP + noise mechanisms & failure cases [8 marks]

Expected direction:
This section should delve deeper into DP's mathematical guarantees, specifically how adding noise (Laplace/ Binomial /Binomial-related mechanisms) protects against re-identification, and then discuss where these guarantees fail or are weakened.

1. Formal DP guarantee – 3 marks

- States the ε-DP (or (ε,δ)-DP) definition:

    - For any two neighbouring datasets differing in one person and any output set S

- Or an equivalent, clearly correct explanation that the probability of any output does not change much when one person's data is added/removed.

Scoring:

- 3 marks: reasonably formal, essentially correct definition.
- 2 marks: mostly correct but slightly informal/incomplete.
- 1 mark: only high-level intuition.

2. Noise mechanisms (Laplace / Binomial, etc.) and why they give the guarantee – 3 marks

- Identifies appropriate mechanisms – 1 mark
    - Mentions Laplace mechanism (for numeric queries), possibly Gaussian/Binomial/discrete Laplace for counts.
- Explains how noise is calibrated – 1 mark
    - Talks about sensitivity (max change by one person) and noise scale proportional to sensitivity/ε.
- Connects noise to re-identification guarantee – 1 mark
    - Explains that because of this noise, an attacker cannot tell with high confidence whether a particular individual's data was included (probabilities differ by at most e^ε).

If they name Laplace/Binomial noise but don't link to sensitivity/ /εε, give at most 1.5/3 here.

3. Failure modes/limitations – 2 marks

- At least two specific scenarios where DP protection is weakened/inadequate, with explanation, e.g.:
    - ε chosen too large → noise too small → weak privacy.

- Multiple queries (composition): repeated releases gradually leak more information (privacy budget exhaustion).
- Mis-specified adjacency or highly correlated data (e.g., whole families), so protection for groups is weaker.
- Implementation issues: non-DP pre/post-processing, side channels, or incorrectly implemented mechanisms.

Scoring:

- 2 marks: two clear, relevant scenarios with explanation.
- 1 mark: one good scenario or two very vaguely described.

Answers that discuss only k-anonymity/encryption here, without DP/noise, should receive at most 2–3/8 points, depending on quality.

1(e) Privacy (Total Marks - 8)

(e) Consider a scenario where the healthcare provider wants to implement a permission model for accessing sensitive data. How would you design a usable mediated authentication system for such a healthcare provider? [8]

A healthcare environment contains multiple classes of sensitive data, including patient medical history, diagnostic images, prescriptions, lab reports, and various user types (doctors, nurses, lab technicians, administrators), who require different levels of access. To enforce this safely and with minimal burden on users, a mediated authentication system similar to a Kerberos-style architecture provides a practical and usable solution.

1. Rationale for a Mediated (Third-Party) Authentication Model

Direct authentication between every user and every service is dchallengingto manage in healthcare settings because:

- services (EHR systems, lab systems, pharmacy systems) are distributed,
- users frequently move between terminals and departments,
- Passwords or session keys stored across multiple systems increase the risk of compromise.
- There must be strong logging, delegation, and revocation support for regulatory compliance.

2. System Architecture

The design uses three entities:

1. **User (U):** doctor, nurse, or technician
2. **Authentication Server (AS):** validates identities and issues time-bound credentials
3. **Healthcare Services (S):** EHR system, lab server, imaging repository, etc.

3. Authentication & Authorization Workflow

Step 1: User Authentication

The user logs in once per session using strong authentication (password + intelligent card/biometrics). The AS verifies the identity and consults the hospital's Role-Based Access Control (RBAC) policy.

Step 2: Issuance of a Ticket-Granting Token

The AS issues a Ticket-Granting Token (TGT) encrypted with a secret known only to the AS. This token does *not* reveal the user's password to any service.

Step 3: Accessing Healthcare Services

When the user requests access to a specific service (e.g., radiology server), the client presents the TGT to the AS.

The AS checks the permission model:

- Doctors → full access to patient diagnosis & medical history
- Nurses → access to treatment plans, medication orders
- Lab staff → access to lab requests/results only
- Admin staff → administrative metadata, no clinical details

If access is permitted, the AS issues a Service Ticket containing: user identity, role/permissions, session key, short expiry, and audit identifiers (for compliance logging).

Step 4: Service Grants Access

The user presents the Service Ticket to the healthcare service, which decrypts it using its shared key with the AS. If valid, the session is established using the session key.

4. Usability Features

a. Single Sign-On (SSO)

The user authenticates only once, reducing friction in clinical environments where delays impact patient care.

b. Automatic Timeout & Renewal

Short-lived tickets expire automatically, minimizing user burden while preventing long-term misuse.

c. Device-Independent Access

Clinicians frequently switch terminals. The TGT allows seamless, secure access without repeated logins.

d. Built-In Auditing

Service tickets include traceable audit fields to meet regulatory requirements (similar to HIPAA logging).

5. Why This Works Well for Healthcare

- Centralized trust creates a single, well-protected point for identity management.
- Least privilege enforcement is straightforward through role-driven service tickets.
- Compromise containment: short-lived keys minimize damage if a workstation is exposed.
- Regulatory compliance: mediated models offer excellent support for logging, access review, and revocation.
- Improved usability minimizes workflow disruption in high-pressure environments.

Full-Marks Criteria (8/8)

1. Correct identification of a mediated authentication model (2 marks)

- Kerberos-style, Needham-Schroeder, third-party authentication, or any equivalent centralized mediator.
- Must clearly describe the role of an Authentication Server.

2. Well-reasoned justification for using a mediated system in healthcare (2 marks)

Examples include: complexity of distributed services, avoiding repeated passwords, strong auditing requirements, role-based permissions, and sensitive data access control.

3. Clear description of the authentication flow (2 marks)

A strong answer includes: initial authentication to AS, issuance of a ticket or token, service-specific ticket generation, and verification by the service.

4. Integration of usability principles (1 mark)

Such as: single sign-on, reduced cognitive load, minimal workflow disruption, and fast access for clinicians.

5. Alignment with permission model (1 mark)

Students must explicitly link: **tickets/tokens → encoded permissions / RBAC → controlled data access**.

Partial Credit (4–6 marks)

Award when the student:

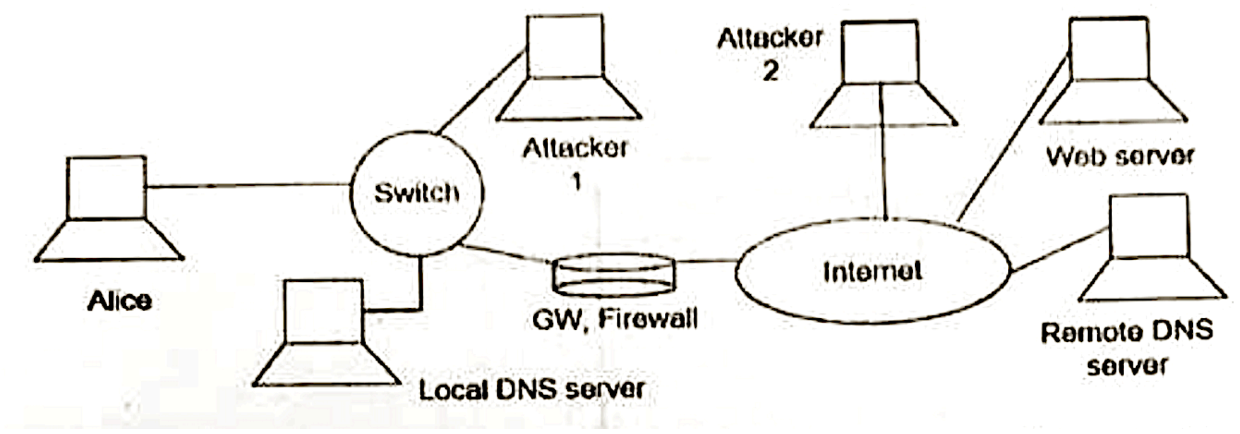
- Describes mediated authentication but with incomplete detail,
- Mentions Kerberos but does not explain why it suits healthcare,
- Gives an authentication flow without connecting it to permissions,
- Mentions usability but superficially.

Minimal Credit (1–3 marks)

Award when the student:

- Describes generic authentication (e.g., passwords, MFA) without mediation,
- Mentions access control but not the authentication architecture,
- Gives high-level ideas without technical grounding.

Q2)



- [8] Consider a scenario where an adversary sets up a large number of malicious nodes on the TOR network in an attempt to deanonymize users. How can the network detect and prevent this attack? What are the potential drawbacks of these measures?
- [8] Discuss the role of exit nodes in the TOR network and the potential risks associated with using them. How can users mitigate thesé dangers and protect their privacy with exit nodes?
- [9] Consider a scenario where Attacker 1 and Attacker 2 want to deanonymize your Tor traffic. Compare and illustrate how both can perform such deanonymization attacks. Justify.

Bonus [5]: Assume we are using DNSSEC. Explain how attacker 1 is able to deanonymize Alice's TOR traffic.

Q2)

A. Malicious Nodes Detection and Prevention in TOR Network (8 Points)

Expected Answer: The network can detect and prevent malicious nodes through the detection of timing attacks, statistical analysis measures, and monitoring of node behavior patterns. Potential drawbacks include increased network overhead, reduced performance, false positives, and potential privacy concerns from monitoring.

Marking Scheme:

- **2 Marks:** Mention detection methods (timing attacks, statistical analysis)
- **2 Marks:** Explain prevention measures (node reputation systems, traffic analysis detection)
- **2 Marks:** Identify potential drawbacks (performance overhead, false positives, reduced anonymity from monitoring)
- **2 Marks:** Clear explanation linking detection/prevention to the Sybil attack scenario

B. Exit Nodes: Roles, Risks, and Mitigation (8 Points)

Expected Answer: Exit nodes connect TOR traffic to the destination server and see unencrypted traffic. A compromised exit node alone does NOT deanonymize users because it doesn't know the user's identity. Deanonymization requires collusion between the exit node and the entry node/server. Mitigation includes using HTTPS, avoiding the transmission of sensitive data, and utilizing trusted exit nodes.

Marking Scheme:

- **2 Marks:** Explain the role of exit nodes (final relay, connection to server)
- **2 Marks:** Identify that a compromised exit node ALONE cannot deanonymize (the exit node doesn't know the user's identity)
- **2 Marks:** Explain collusion risks (exit node + entry node/server collaboration enables deanonymization)
- **2 Marks:** Suggest mitigation strategies (HTTPS, end-to-end encryption, trusted exits, avoiding sensitive data)

C. Comparative Analysis: Attacker 1 vs Attacker 2 Deanonymization (9 Points)

Expected Answer: Attacker 1 (internal) generally has advantages, including the ability to identify Alice's IP directly, perform ARP spoofing, intercept traffic at the gateway/firewall, and redirect traffic. Attacker 2 (external) has limitations but may have advantages in specific scenarios (e.g., controlling exit nodes or destination servers). The answer should present a justified argument for which attacker has advantages in different scenarios.

Marking Scheme:

- **3 Marks:** Explain Attacker 1's advantages (internal network position, ARP spoofing, gateway/firewall access, can identify Alice's IP)
- **2 Marks:** Explain Attacker 2's position and limitations (external to the network, must rely on traffic analysis or compromised nodes)
- **2 Marks:** Present specific scenarios where Attacker 2 might have advantages (controlling exit nodes, destination server compromise)
- **2 Marks:** Clear justification and argument quality (no binary right/wrong answer; grading based on logical argumentation)

D. DNSSEC and Deanonymization by Attacker 1 (5 Points)

Expected Answer: DNSSEC provides DNS integrity but does NOT prevent DNS leaks. Attacker 1 (internal) can exploit DNS leaks by monitoring DNS requests that escape the TOR tunnel. Even with DNSSEC ensuring authenticity, if DNS queries leak outside TOR to the local DNS server, Attacker 1 can observe which domains Alice is querying, enabling correlation attacks to deanonymize her traffic.

Marking Scheme:

- **2 Marks:** Explain that DNSSEC provides integrity/authenticity but NOT confidentiality
- **2 Marks:** Explain DNS leak vulnerability (DNS requests escaping the TOR tunnel, visible to the local network)
- **1 Mark:** Connect DNS leak observation to deanonymization (Attacker 1 monitors local DNS queries to correlate with TOR traffic)

Key Grading Notes:

- **No binary right/wrong answers** for Q3 - grade based on quality of argumentation and logical reasoning
- Students should reference **three principal TOR vulnerabilities**: DNS leaks, timing attacks, and statistical analysis
- Emphasis on understanding that **single compromised nodes (especially exit) don't automatically deanonymize** - requires collusion or additional information.
- Bonus question requires a **deep understanding** of DNS leak vs DNSSEC functionality.

Q3)

3. Security & Usability Testing (Total Marks - 25)

A. Consider a scenario where a company wants to conduct usability testing for a new e-commerce website. What are some of the challenges that might arise during usability testing, and how can these be addressed to ensure that the results are accurate and actionable? [5]

B. What is the role of Mental Models in Usability Testing, and how would you account for its effect? [5]

C. During the testing phase, explain how you went about identifying and exploiting Shell code Injection and SQL vulnerabilities in another group's tool. Mention specific exploit . Specifically, describe how you would execute such an attac ? [8]

D. Can such attacks be prevented? How would you defend against such attacks? [7]

E. **Bonus** Can you execute the Shell Code injection without having access to the source code and NOPs? [10]

A. Challenges - (2.5 marks) :

1. **Wrong participants –** not matching real shoppers
2. **Unclear tasks –** instructions not well understood,
3. **Observer effect –** users act differently when watched
4. **Technical issues –** bugs or device inconsistencies,
5. **Unrealistic environment –** testing ,unlike real shopping

Ensuring accurate & actionable results (2.5 marks) :

1. **Recruit real users –** proper screening for the target group
2. **Create clear tasks –** simple, realistic shopping flows
3. **Use neutral facilitation –** avoid leading or influencing,
4. **Pilot test setup –** fix technical or task problems early
5. **Capture solid metrics –** record time, errors, success rate

These are possible answers ;students may present different but valid reasoning. Award - **Full marks** (2.5 + 2.5): if Anany - 3 valid points per section

Partial marks (1 + 1): At least one valid point in/any of the section

0 marks: No valid points or **irrational**/**irrelevant** arguments

B. Role of Mental Models (2.5 marks)

1. **Guide user expectations** – users predict how the site should work.
2. **Shape user behaviour** – actions depend on what they think will happen.
3. **Reveal design mismatches** – confusion shows gaps between user thinking and system layout.
4. **Affect task success** – clearer mental models lead to faster, error-free tasks.
5. **Highlight usability issues** – incorrect assumptions signal areas needing redesign.

Accounting for Their Effect (2.5 marks)

1. **Use think-aloud** – capture how users interpret actions and features
2. **Ask pre-task questions** – understand prior experience and expectations. ,
3. **Observe misconceptions** – note where the user thinking differs from the actual workflow.
4. **Adjust test tasks** – ensure tasks reveal mental-model gaps clearly
5. **Refine design** – align interface flow with common user expectations.

These are possible answers; students may present different but valid reasoning. Award -

Full marks (2.5 + 2.5): if any 2 - 3 valid points per section

Partial marks (1 + 1): At least one valid point in/any of the section

0 marks: No valid points or **irrational**/**irrelevant** arguments

C.

i) Identification of Shell Code Injection (2 marks)

1. Injected command operators – Tried; &&, | in input fields.

2. Checked system responses – Looked for unexpected command outputs (e.g., whoami).
3. Tested oversized input – Long strings to expose buffer/command handling flaws.
4. Found unsafe command calls – Identified fields passing data to system functions.

ii) Identification of SQL Injection / Vulnerabilities (2 marks)

1. Tested special characters – Used ', ", -- to check for SQL errors.
2. Observed error messages – Looked for DB-related warnings or query failures.
3. Tried login bypass patterns – Entered inputs like admin' OR '1'='1.
4. Noted direct query usage – Spotted inputs directly embedded into SQL code.

iii) Exploitation of SQL Injection (2 marks)

1. Classic login bypass – Input like admin' OR '1'='1.
2. UNION-based dump – Used UNION SELECT username, password.
3. Boolean/time tests – Used AND 1=1 or SLEEP(3) to confirm SQLi.
4. Read sensitive info – Extracted table names or user records.

iv) Exploitation of Shell Code (2 marks)

1. Executed basic commands – Payload like && whoami.
2. Listed files – Used; ls or ;ir to confirm execution.
3. Chained commands –; cat config.txt to read files.
4. Safe demonstration only – Performed non-destructive, read-only commands ethically.

These are possible answers; students may present different but valid reasoning. Award :

Full Marks (4 + 4) - Student gives **2–3 correct points** in each section (i ii,i ii,i v).

Partial Marks (2–3 per section) - Student gives **1–2 valid points** in a section, or explanations are shallow.

Minimum Marks (1 per section) - Student attempts with **some logic**, but points are unclear or weak.

Zero Marks - Response is **irrelevant** or technically incorrect.

D.

Yes, such attacks can be prevented using standard security control . **(1Mark)**

SQL Injection Prevention (3 marks)

1. Use prepared statements or parameterized queries.
2. Apply strict input validation to block characters like ', ", --.
3. Use least-privilege DB accounts with limited rights.
4. Escape and encode user inputs before sending to SQL.
5. Use WAF rules to block standard SQLi payloads.

Shell / Command Injection Prevention (3 marks)

1. Avoid using system(), exec(), or popen() with user input.
2. Allow only whitelisted commands and arguments.
3. Sanitize dangerous characters like:, &&, |.
4. Run the application with minimal OS privileges.
5. Encode/escape command parameters to prevent shell interpretation.

Yes = 1 mark

If the answer starts with **"Yes"**. **If "No" 0 marks (stop checking).**

For each question :

Full Marks (3): If the student gives **3–4 valid SQL defence points**.

Partial (1–2): If the student gives **1–2 valid points**.

Zero: No meaningful SQL prevention.

E . Bonus

Yes, it's still possible to run shell code even if you don't have the source code or any NOP sleds. Attackers simply use other tricks to get the code to execute. Some examples are :

1. **Return-to-libc** – Instead of injecting code, just call existing functions like system("/bin/sh").

2. **ROP chain –** Join small instructions already in the program to perform the attack.
3. **Use memory leaks –** If the program reveals addresses, that's enough to place the payload correctly.
4. **Direct Buffer Overflow** - Inject shellcode into the buffer and overwrite the return address to jump to it (NOPs optional).
5. **Put shellcode in environment or heap –** Store the payload there and redirect execution to it.
6. **Guessing offsets –** Repeatedly try different offsets until one works.
7. **Use small jumps –** Replace NOPs with short jumps to reach the shellcode.
8. **Heap or script spraying –** Fill memory with repeated patterns so execution eventually lands on the payload.

These are possible answers; students may present different but valid reasoning. Award:
Full Marks (8 - 10): 5+ valid points with reasoning/examples
Partial Marks (4- 7): 3 - 4 valid points with basic explanation
Minimum Marks (1- 3): 1 - 2 valid points with limited logic
Zero Marks (0): Says **"not possible"** or gives "**irrelevant points even after writing Yes.**"

Q4)

4. Building Secure Communication (Total Marks - 15)

   In building your group project.

   A. How would you compromise the HTTPS connection? Describe in detail.

   B. Provide a reasonable solution to counter such an attack? Describe its feasibility and effectiveness.

4(a) How would you compromise the HTTPS connection?

A realistic way to compromise HTTPS in practice is through a man-in-the-middle (MITM) attack based on ARP spoofing combined with HTTPS stripping. This does not break TLS cryptography; instead, it exploits weaknesses *before* the secure channel is established.

1. **Establish MITM Position (ARP Spoofing)**
   On a local network, the attacker sends forged ARP replies so that the victim associates the attacker's MAC address with the gateway's IP. As a result, all of the victim's traffic is transparently routed through the attacker.

2. **Intercept the Initial HTTP Request**
   Many websites still begin with a plaintext HTTP request before redirecting to HTTPS. The attacker intercepts this request and the server's legitimate 301 redirect to https://...

3. **Perform HTTPS Stripping**
   The attacker modifies the redirect to point back to an HTTP page. Because the TLS handshake has not yet occurred, the browser has no way to verify authenticity and simply follows the modified redirect.

4. **Read and Modify Traffic**
   The victim now communicates over unencrypted HTTP, believing a secure session exists. The attacker can view credentials, manipulate the content, or inject a fraudulent HTTPS page with a spoofed certificate. If the user ignores certificate warnings, the attacker can maintain complete MITM visibility even after the switch to HTTPS.

This attack succeeds because the security guarantees of HTTPS activate *only after* the TLS handshake is completed. Before that moment, an attacker who controls the network path can downgrade the connection.

4(b) Provide a reasonable solution and explain its feasibility.

The most effective defenses correspond directly to the weaknesses exploited above.

1. Enforce HSTS (HTTP Strict Transport Security)

With HSTS, browsers refuse to make plaintext HTTP requests to the domain. Even if an attacker alters redirects, the browser will not downgrade. This directly prevents HTTPS stripping.

- **Feasibility:**
  Simple to enable (single header).
  Limitation: The very first visit is vulnerable unless the domain is included in browser HSTS preload lists.

2. Certificate Pinning

Pinning requires the browser or application to remember the legitimate public key or certificate fingerprint of the server. If an attacker presents a forged certificate—even one signed by a compromised CA—the browser rejects it.

- **Feasibility:**
  Strong protection but operationally sensitive: key rotations must be planned to avoid locking out legitimate users.

3. Network-Level Controls (e.g., Dynamic ARP Inspection)

Deploying Dynamic ARP Inspection (DAI) or similar switch-level protections prevents attackers from forging ARP packets. Without ARP spoofing, the attacker cannot position themselves as a MITM in the first place.

- **Feasibility:**
  Highly effective in enterprise networks that support DAI.
  Consumer networks rarely have such protections.

Rubric for Question 4 (Total: 15 marks)
Part (a): Compromising HTTPS (7 marks)

To receive full credit, the student must:

1. Identify a valid, technically grounded attack (2 marks)

Acceptable examples include:

- HTTPS stripping
- MITM via ARP spoofing
- Forged/compromised certificates
- DNS-based redirection to attacker-controlled hosts

- Manipulation of the TLS handshake

Answers that propose "breaking encryption" or similarly unrealistic attacks should not receive credit.

2. Explain why the attack works (3 marks)

The student should show a clear understanding that the attack succeeds because:

- The TLS handshake has not yet taken place,
- HTTPS redirects can be modified unless HSTS is enforced,
- ARP spoofing or similar techniques allow network-level interception,
- The CA trust model is only as strong as user behavior and certificate validation.

The explanation should demonstrate comprehension of the underlying security principles.

3. Provide a coherent step-by-step sequence (2 marks)

The student should outline the attack flow (e.g., ARP spoofing → intercept HTTP → modify redirect → downgrade to HTTP → observe traffic).
The sequence does not need to be long, but it must be logically correct.

Partial Credit Guidance (4–6 marks)

Award partial credit if:

- The student mentions a real attack but describes it incompletely,
- The reasoning is technically correct but not well-structured,
- The attack is described abstractly without clear steps.

Minimal or No Credit (0–3 marks)

Award minimal credit if the answer:

- is vague or inaccurate,
- describes unrelated attacks (SQL injection, buffer overflow),
- assumes cryptographic primitives can be "broken,"
- lacks any mechanism for compromising HTTPS.

Part (b): Defending Against the Attack (8 marks)

To receive full credit, the student must:

1. Propose defenses appropriate to the attack described (4 marks)

Examples aligned with common attack types:

- **Against HTTPS stripping:** HSTS, preload lists, mandatory HTTPS
- **Against forged certificates:** certificate pinning, CT logs
- **Against ARP spoofing, MITM:** Dynamic ARP Inspection, static ARP, network segmentation
- **Against DNS spoofing:** DNSSEC, DoH/DoT

The defense must fit the student's own Part (a).
Misaligned defenses should reduce credit.

2. Explain how the defense mitigates the attack (2 marks)

The student should justify *why* the countermeasure is practical, rather than merely listing it.

Examples:

- HSTS prevents the browser from accepting HTTP, eliminating downgrade attacks.
- Pinning ensures that forged certificates are rejected, even from compromised CAs.
- DAI blocks malicious ARP replies, preventing the attacker from becoming a MITM.

3. Discuss feasibility and limitations (2 marks)

A complete answer should mention at least one practical consideration: first-visit vulnerability of HSTS, operational overhead of certificate pinning, enterprise-grade requirements for DAI, usability, or deployment challenges.

Partial Credit Guidance (4–6 marks)

Award partial credit if:

- Defenses are correct but not connected to the attack.
- Explanation is brief or lacks clarity,
- Feasibility is not discussed.

Minimal or No Credit (0–3 marks)

Award minimal credit if the answer:

- suggests irrelevant solutions (firewalls, antivirus),
- describes defenses that do not mitigate the stated attack,
- repeats content from Part (a) instead of providing defenses.

TA Mapping:

Q1: Kirti, Sankar
Q2: Dhruv, Anant
Q3: Gour, Apoorva
Q4: Rujhil, Kunal