CSC 6585

Midterm Exam 2024

Secure Software Development, Computer Science, Tennessee Tech University

Instructions

This is an open-book, open-notes exam. Answer questions to the best of your ability, working **on your own**. Each question is worth 10 percent of the exam grade. Some questions have multiple parts; all parts have the same weight. You may answer as many questions as you wish, but **only the 10 best answers** will be counted.

Handwritten and scanned or photographed documents are fine, but you are expected to write legibly and to use reasonably correct grammar and spelling; answers that cannot be read or evaluated will not receive any points. Please do not create ASCII art diagrams.

Be sure to state your assumptions explicitly.

Questions

1. Your company uses an online data-sharing system for communicating in real time with partners. The online system is pre-configured, provisioned, and maintained by a third party.

   Is the data-sharing system inside or outside your company's system boundary? Justify your answer.

2. You work for a logistics company and maintain your corporate database on a cloud provider's platform. Consider the following.

   - Serrice Technology provides telematics (telematics is hardware and software for monitoring a fleet of vehicles) and pushes information to your cloud database. This information includes vehicle location, weight, and health.
   - Ascension Financial Services has a data feed from your cloud database. This information includes contract fulfillment, payment processing, and accounts due.
   - You communicate with your drivers using a third-party communication provider, Sonax Industries.

   Draw the components of this system, identify trust boundaries, and label all flows. Be sure to state your assumptions.

3. A software system has a large set of configuration options, stored as JSON files with one JSON file per component. These files are stored in a git repository, allowing users to switch configurations (using branches) and to roll back to prior configurations if problems are detected.

   a. The git repository can be stored locally in the software's configuration folder (an application-specific folder under `AppData` on Windows, `Library/Preferences` on macOS, and `~/.config` on Linux). What, if any, concerns would you have with this arrangement?

   b. The git repository can track a remote cloud-hosted repository, enabling settings sync across multiple machines. What, if any, concerns would you have with this arrangement, distinct from those identified in part a?

4. Consider this proposed definition. "A software system is secure iff there is no series of inputs to the system that results in output of the content of a file unless the final input in the series originates from the file owner."

   Now consider the CIA triad. For each of the elements of the triad, does the above definition address that element? Explain.

5. Consider a simple web store. A back end server provides the business logic, while the front end runs in a web browser.

   a. Draw a block diagram of this scenario, labeling all information flows. You should include the end user, web browser, front end software, back end server software, server hosting environment, inventory database, authentication server, customer information database, payment processing contractor, the administrator interface (for simplicity we will assume it runs natively on the server), and the system administrator.

   b. Specify the system boundary for the server software by identifying its interfaces.

   c. For every control or information flow in your diagram for part a, determine if the flow crosses a trust boundary. Consider bidirectional flows to be a single flow for the purpose of this part.

6. A piece of financial software has the following logic. Here `RESULT` is a result code, `Acct` is an account structure, and the `LOCK` and `UNLOCK` macros prevent a race condition. Note that `static` functions in C are private to the containing module.

```
static RESULT _deduct(Acct * account, int value)
{
    if (account->balance < value) {
        return INSUFFICIENT_FUNDS;
    }
    account->balance -= value;
    return OK;
}

RESULT deduct(Acct * account, int value)
{
    LOCK(account);
    RESULT result = _deduct(account, value);
    UNLOCK(account);
    return result;
}
```

a. What happens when the value is a negative number and the account balance is positive?

b. How would you recommend addressing this problem in the given code? Be specific about how you would change the code.

7. Canada and the US decide to connect information processing systems based on the Bell-LaPadula model. Fortunately, the two countries have exactly the same sensitivity levels. To preserve state secrets, both countries introduce an additional security modifier: CAN-ONLY for Canada and US-ONLY for the US, and corresponding clearance tags: CAN for Canada and US for US. These sensitivity levels are independent of the usual TOP SECRED, SECRET, CONFIDENTIAL, UNCLASSIFIED forms.

   Is the resulting system still Bell-LaPadula secure? Consider each of the required properties and whether it holds for the composite system.

8. Recall the five functions for physical security and classify each of the following security measures to those functions. Identify which function or functions are addressed by each.

   a. A login screen states that the system is for official use only by US government employees, and that misuse of the system could result in fines or jail time.

   b. After three successive login failures the system logs you out and blocks your IP address.

   c. After a two login failures the system forces you to wait a full minute before trying again.

   d. The system detects when a specific file is read and, if that is detected, the stored disk encryption keys are randomized and the system is shut down.

9. Recall the NIST five functions and classify each of the following security measures as one of those functions. Identify which function or functions are addressed by each.

   a. You hire a lawyer to advise you about the relevant laws governing the specific data you intend to collect and store.

   b. Your system periodically autosaves its current state to a git repository.

   c. The system implements a "watchdog timer." If the timer ever counts down to zero, an alarm is reported through the endpoint monitoring software.

   d. Your IT team runs a "tabletop" exercise with the scenario that you detect an intrusion in the network.

10. A software system checks the host key of a connecting machine and requires that the user identify themselves with both a password and a 2FA token. For as long as the user is logged in, the software periodically sends a challenge to the remote host and then checks the reply to re-authenticate the connecting machine. While the user is interacting with the system, biometrics are collected and compared to the user's baseline.

Is this an example of defense in depth? Why or why not?

11. Consider the CIA model of Confidentiality, Integrity, and Availability. Different contexts require different security measures.

    a. Give an example of a system where confidentiality is more important that the other two properties. Explain.

    b. Give an example of a system where integrity is more important that the other two properties. Explain.

    c. Give an example of a system where availability is more important that the other two properties. Explain.

12. Consider the HRU command definitions and the safety rule, and let A be the access control matrix.

    a. Create a command foo that accepts two subjects, a and b, and an object name X. The command checks that A(a,b) contains FOO that that A(b,a) does not contain FOO and, if both are true, it adds object X with owner a.

    b. Consider the command you wrote for part a. Does the command ever leak the right FOO from the system? That is, is it safe with respect to the right FOO?

13. The HRU paper shows that, given a Turing machine, we can convert the question "Does this Turing machine halt on all inputs?" to a protection system and the question of whether it is safe. Thus if we could answer the latter, we would also be able to answer the former, and thus solve the Halting Problem. Since we are confident the Halting Problem cannot be solved in general, it follows that it is undecidable whether a given protection system is safe.

    A friend shows you a design for a protection system and tells you that they can prove it is safe. Is that possible? Why or why not?

14. Your goal is to gain access to my iLearn account for this class. Build an attack tree starting with this goal. The leaves of the tree should be specific actions, but you do not need to be more detailed that the example given in class.

15. An attack has an 10% chance of succeeding and, when unsuccessful, there is a 25% chance of being detected. Assume detection results in a change to the system preventing the attack in the future. Create an attack graph that represents this situation and label the edges with actions (success, failure, detection, no detection) and probabilities. Assume the attack will be repeated until it either succeeds or is detected, and capture that in your graph.

16. [Extra 10 points] What is the overall probability that the attack in question 15 eventually succeeds?

CSC 6585 Midterm Exam

Blaine Swieder

10 October 2024

1 Solutions

1.1 Question 1 Answer

We can say that the data-sharing system is ***inside*** your company's system boundary because the organization has significant control, management, and accountability over the system's security, operations, and data protection, even when external vendors are involved:

- **Data Ownership and Responsibility**: The company owns the data shared through the system and is responsible for its protection and regulatory compliance (e.g., GDPR for EU data privacy, HIPAA for U.S. health information), regardless of third-party maintenance.
- **Control Over Access and Usage**: You manage user access and enforce security policies (e.g., password requirements, MFA), indicating administrative control over the system's use.
- **Integration with Internal Systems**: The system uses the company's authentication methods and exchanges data with internal applications, showing its integration with your IT environment.
- **Contractual Obligations and Shared Responsibility**: Service agreements with the third-party provider outline security responsibilities, including a shared responsibility model where the company may be responsible for managing access and encryption, while the provider handles hardware maintenance and server uptime.
- **Risk Management and Incident Response**: The system is included in your risk assessments and incident response plans, aligning with standards such as ISO/IEC 27001 and NIST SP 800-53, which require managing risks for all systems handling company data. Risk assessments may include evaluating potential data breaches or system downtime.
- **Operational Dependence**: The system supports critical business functions like real-time communication with partners and is part of your business continuity planning, making it integral to your operations.
- **Policy Enforcement and Compliance**: Employees are trained on secure usage per your company's policies, and system usage is monitored to ensure compliance and detect unauthorized activities.

To conclude, although the system is maintained by a third party, it is still considered ***inside*** your company's system boundary because the company retains control over the data, access, integration with internal systems, contractual obligations, and responsibility for security and compliance.

1.2 Question 2 Answer

See pdf attached from Threat Dragon.

1.3 Question 3 Answer

1.3.1 Part A

Example 1.1. Concerns with Storing the Git Repository Locally in the Configuration Folder:

Storing the Git repository locally within the software's configuration folder raises several concerns. First, security risks are paramount because configuration files often contain sensitive information like API keys, passwords, or personal user settings. If these repositories are stored without stringent access controls, unauthorized users or malicious software could access and exploit this sensitive data. Additionally, other applications or users with access to the configuration folder might unintentionally modify or corrupt the repository, leading to further security issues. Moreover, there is a risk of data integrity issues since Git repositories can become corrupted due to improper shutdowns, disk errors, or software bugs. This corruption can lead to a loss of configuration history or the inability to roll back changes when necessary. Next, user complexity is another concern, as not all users are acquainted with Git operations. Misusing commands such as `commit`, `branch`, or `rollback` could result in accidental data loss or misconfiguration. Another concern is disk space consumption, which might become problematic over time. Repositories can grow significantly if configuration files change frequently or are large in size. Finally, there could be compatibility issues across different operating systems. Variations in how Windows, macOS, and Linux handle hidden files and permissions could lead to synchronization problems or conflicts when configurations are shared across platforms.

1.3.2 Part B

Example 1.2. Additional Concerns with Tracking a Remote Cloud-Hosted Repository:

Tracking a remote cloud-hosted repository to sync settings across multiple machines introduces further concerns beyond those of local storage. First, security and privacy risks are heightened; transmitting configuration files over the internet could expose sensitive data to interception if proper encryption protocols are not used. Secondly, there is the risk of unauthorized access if authentication credentials are compromised or if the cloud service itself is vulnerable to attacks. Thirdly, compliance with legal and regulatory standards becomes critical. Storing potentially sensitive configuration data in the cloud may violate data protection regulations like GDPR or HIPAA, especially if the data includes personal or confidential information. Fourthly, reliance on network connectivity and cloud service availability can affect the reliability of configuration management. Network outages or service disruptions may prevent synchronization, leading to inconsistent configurations across machines. Moreover, users may face challenges with data consistency and merge conflicts. Concurrent changes made on different machines can result in conflicts that are difficult to resolve, particularly for users not proficient with Git, potentially leading to data loss or misconfiguration. Additionally, there is a risk of vendor lock-in or dependency on third-party services. Relying on a specific cloud provider for critical configuration management can make it difficult to switch services in the future or adapt if the provider changes its terms of service or experiences extended outages.

1.4 Question 4 Answer

Let us consider how this definition relates to the CIA triad:

1.4.1 Confidentiality

Is confidentiality addressed? **Yes**, since confidentiality involves protecting information from unauthorized access and disclosure and the provided definition specifically aims to prevent the unauthorized output of a file's content. Moreover, the definition states that only when the final input comes from the file owner can the content be outputted; hence, this implies that even if unauthorized users attempt to access the file through a series of inputs, they will fail, unless the file owner initiates the final input. Therefore, the definition directly addresses the Confidentiality aspect by ensuring that only the file owner can cause the file's content to be revealed.

1.4.2 Integrity

No, this definition does not address Integrity. Integrity is defined as safeguarding the accuracy and completeness of information and processing methods. It ensures that data is not altered or tampered by unauthorized individuals. On the contrary, the proposed definition focuses only on the unauthorized output of file contents and does not explicitly mention any mechanisms to prevent the unauthorized modification or deletion of files. Additionally, there is no provision to ensure that the data remains unaltered or that any changes are authorized and verifiable. Thus, the given definition does not address this aspect of the CIA triad.

1.4.3 Availability

No, the given definition does not address Availability. Availability ensures that authorized users have reliable and timely access to resources and information when needed. It involves maintaining system uptime and preventing denial-of-service attacks to ensure that resources are accessible. The proposed definition does not consider whether the system or the files are accessible to authorized users other than the file owner, nor does it address system performance or uptime. By neglecting availability, the system may fail to meet the needs of legitimate users, hence impacting the overall security posture of the system. Therefore, the definition does not consider the Availability aspect of the CIA triad.

1.5 Question 6 Answer

1.5.1 Part A

When the `value` is negative and the account balance is positive, the function behaves incorrectly. The issue lies in the condition that checks if the account balance is less than `value`. Since the `value` is negative, this condition will always evaluate as `false` because a positive number (the account balance) will always be greater than any negative number. As a result, the function proceeds to subtract the `value` from the balance.

On the contrary, subtracting a negative number is the same as adding its absolute value, which leads to an unintended increase in the account balance. For example, if the balance is 1000 and the `value` is -100, the function calculates `1000 - (-100)`, which results in 1100. This is not the desired behavior, as the balance should decrease, not increase, when performing a deduction.

1.5.2 Part B

To address this issue, the code should include input validation to ensure that the `value` passed to the `deduct` function is positive. Specifically, before locking the account and calling the `_deduct` function, a check should be added to confirm that the `value` is greater than zero. If the `value` is less than or equal to zero, the function should return an appropriate error code, such as `INVALID_VALUE`, and avoid proceeding with the deduction logic.

This input validation prevents negative values from being processed and ensures that the account balance does not increase when it should decrease. By implementing this check, the function will only proceed with valid, positive inputs, which prevents the unintended behavior and maintains the correctness of the financial software. This modification ensures that the function behaves as expected and helps to prevent potential errors in handling account balances.

1.6 Question 7 Answer

Yes, the composite system is Bell-LaPadula secure. By treating the additional security modifiers CAN-ONLY and US-ONLY as categories within the Bell-LaPadula model, the system maintains its security properties. According to the Simple Security Property (no read up) and the Star-Property (no write down), these properties hold with the inclusion of these categories. The Simple Security Property ensures that subjects cannot read data at a higher classification level, and the Star-Property ensures that subjects cannot write data to a lower classification level.

In this case, access decisions must consider both the standard sensitivity levels (i.e., TOP SECRET, SECRET, CONFIDENTIAL, UNCLASSIFIED) as well as these new country-specific categories. The CAN-ONLY and US-ONLY modifiers ensure that subjects from one country cannot read or write objects restricted to the other country. This enforces the required security policies between Canada and the US, while still adhering to the core principles of the Bell-LaPadula model.

1.7 Question 8 Answer

1.7.1 Part A

Security Measure: A login screen states that the system is for official use only by US government employees, and that misuse of the system could result in fines or jail time.

Classification: - Function: **Deterrence**

Explanation: This measure serves as a deterrent by informing users of authorized use and the legal consequences of misuse. Clear communication aims to discourage unauthorized access and deter potential malicious activities.

1.7.2 Part B

Security Measure: After three successive login failures, the system logs you out and blocks your IP address.

Classification: - Function: **Access Control**

Explanation: Implementing account lockout and IP blocking restricts access to authorized users only. This mechanism prevents unauthorized individuals from gaining access through repeated credential guessing, thereby enhancing system security.

1.7.3 Part C

Security Measure: After two login failures, the system forces you to wait a full minute before trying again.

Classification: - Function: **Access Control**

Explanation: Enforcing a delay after failed login attempts acts as an access control measure by slowing down potential brute force attacks. This rate-limiting strategy reduces the likelihood of unauthorized access through rapid, repeated login attempts.

1.7.4 Part D

Security Measure: The system detects when a specific file is read and, if that is detected, the stored disk encryption keys are randomized and the system is shut down.

Classification: - Functions: **Detection** and **Response - Categories**: **- Detection**: Monitoring unauthorized file access **- Response**: Encryption key randomization and system shutdown

Explanation: This measure encompasses:

- **Detection**: Monitoring for unauthorized access to specific files to identify potential security breaches.
- **Response**: Taking immediate action by randomizing encryption keys and shutting down the system to mitigate the impact of the detected breach.

This combination ensures that threats are both identified promptly and addressed effectively to protect sensitive data.

1.8 Question 9 Answer

1.8.1 Part A

The function addressed is **Identify**, specifically under the **Governance** category. Hiring a lawyer helps the organization understand legal and regulatory requirements, which is crucial for effective risk management and ensuring compliance. Additionally, it aids in developing and refining policies and procedures that align with relevant laws and regulations, thereby strengthening the organization's overall security posture.

1.8.2 Part B

The functions addressed are **Protect** and **Recover**.

- **Protect**: The implementation of periodic autosaves to a Git repository falls within the Data Security category under Protect. This measure safeguards the integrity and availability of data by ensuring that critical system states are regularly preserved, mitigating risks associated with data loss or corruption.
- **Recover**: Maintaining backups aligns with the **Recovery Planning** category under Recover. This ensures that the organization can effectively restore systems and data following a cybersecurity incident, thereby reinstating any capabilities that were compromised during the event.

1.8.3 Part C

The function addressed is **Detect**, specifically within the **Anomalies and Events** category. Implementing a watchdog timer serves as a monitoring mechanism that detects system failures or irregular activities. When the timer expires without the expected reset signal, it triggers an alarm, facilitating the timely identification of potential cybersecurity incidents or system malfunctions. This proactive detection enables the organization to respond swiftly to anomalies, thereby enhancing overall security and system reliability.

1.8.4 Part D

The functions addressed are **Respond** and **Identify**.

- **Respond**: Conducting tabletop exercises exemplifies the Respond function by simulating incident response scenarios. These exercises align with the **Response Planning** and **Communications** categories, enabling the team to practice and refine their strategies for handling cybersecurity incidents effectively.
- **Identify**: Such exercises also contribute to the Identify function by informing the **Risk Assessment** category. Through these simulations, organizations can uncover potential vulnerabilities and gaps in their current security posture, thereby enhancing future risk management and mitigation strategies.

1.9 Question 10 Answer

Yes, the described software system effectively exemplifies defense in depth, a comprehensive security strategy that employs multiple layers of defense to protect against a variety of threats. Initially, the system verifies the host key of connecting machines, ensuring that only trusted devices can establish a connection and mitigating risks such as Man-in-the-Middle (MITM) attacks. This foundational layer is complemented by multi-factor authentication (MFA), where users must provide both a password and a two-factor authentication (2FA) token, adding an extra layer of security that guards against credential theft even if one factor is compromised. To maintain ongoing security, the system periodically sends challenges to re-authenticate the connecting machine during active sessions, thereby preventing session hijacking and ensuring continuous verification. Moreover, the integration of biometric verification during user interactions provides continuous authentication, ensuring that the authorized user remains the one interacting with the system throughout the session. Each of these layers addresses different potential vulnerabilities, so if an attacker bypasses one security measure, subsequent layers still offer protection. Furthermore, incorporating logging and real-time monitoring allows for the detection and response to suspicious activities promptly, while data encryption both in transit and at rest safeguards against eavesdropping and data breaches. Regular security audits and updates ensure that the system remains resilient against emerging threats and vulnerabilities. Finally, user education and training reduce the risk of social engineering attacks and ensure proper usage of authentication mechanisms. This layered approach ensures comprehensive security by combining various methods to defend against a wide range of threats, thereby embodying the principles of defense in depth.

1.10 Question 11 Answer

1.10.1 Part A

Example 1.3. Medical Record Management System

In a Medical Record Management System, confidentiality is paramount compared to integrity and availability. This system handles highly sensitive patient information, including personal identification details, medical histories, diagnoses, treatment plans, and billing information. Ensuring confidentiality is critical to comply with privacy laws and regulations such as the Health Insurance Portability and Accountability Act (HIPAA) in the United States. Protecting this data from unauthorized access preserves patient privacy, maintains trust between patients and healthcare providers, and prevents identity theft and fraud. While integrity and availability are also important to ensure accurate records and timely access, the primary focus remains on safeguarding sensitive information from breaches and unauthorized disclosures through measures like encryption, strict access controls, and secure authentication mechanisms.

1.10.2 Part B

Example 1.4. Financial Transaction Processing System

In a Financial Transaction Processing System, integrity holds greater importance than confidentiality and availability. This system manages critical financial operations such as fund transfers, payments, account updates, and transaction records. Maintaining data integrity ensures that all financial transactions are accurate, consistent, and trustworthy, which is essential for preventing errors, fraud, and discrepancies in accounts. Accurate transaction data is crucial for regulatory compliance, financial reporting, and maintaining stakeholder trust. Compromised integrity could lead to significant financial losses, legal repercussions, and damage to the institution's reputation. To uphold integrity, the system employs mechanisms like checksums, digital signatures, robust auditing processes, and validation protocols to detect and prevent unauthorized modifications or data corruption. While confidentiality and availability are necessary to protect sensitive financial information and ensure system operability, the accuracy and reliability of transaction data take precedence in this context.

1.10.3 Part C

Example 1.5. Emergency Response Communication System

In an Emergency Response Communication System, *availability* is the most critical attribute over confidentiality and integrity. This system is used by first responders, including police, fire departments, and medical personnel, to coordinate actions during emergencies such as natural disasters, accidents, or public safety threats. Ensuring high availability is essential because timely and reliable communication can be a matter of life and death; moreover, during crises, the system must remain operational without interruption to facilitate swift responses, resource allocation, and coordination among various agencies. Downtime or unavailability could lead to delayed responses, exacerbating the emergency situation and resulting in severe consequences. To achieve high availability, the system incorporates redundancy, failover mechanisms, robust infrastructure, and real-time monitoring to detect and address potential issues promptly, on the other hand, while maintaining confidentiality and integrity is important to prevent unauthorized access and ensure accurate information transmission, the paramount focus is on ensuring that the communication channels are always accessible and resilient against disruptions.

1.11 Question 12 Answer

1.11.1 Part A

To define the command `foo` that accepts two subjects a and b, and an object name X, the command must satisfy the conditions specified. Below is the command written in terms of the HRU (Harrison-Ruzzo-Ullman) access control model:

```
foo(a, b, X):
    if ’FOO’  A(a, b) and ’FOO’  A(b, a):
        create object X
        enter ’owner’ into A(a, X)
```

Reasoning:

1. **Inputs**:
   - Subjects a and b.
   - Object name X.
2. **Conditions**:
   - The right `FOO` is in the access control matrix entry $A(a, b)$, meaning subject a has the `FOO` right over subject b.
   - The right `FOO` is not in $A(b, a)$, meaning subject b does not have the `FOO` right over subject a.
3. **Actions**:
   - **Create Object**: Using the HRU model's 'create' operation, a new object X is created.
   - **Assign Rights**: The 'enter' operation from the HRU model is used to assign the right `’owner’` into $A(a, X)$, granting subject a ownership over object X.

1.11.2 Part B

To determine if the command leaks the right `FOO`, we analyze whether the right `FOO` is introduced or propagated in a way that could compromise system safety.

Analysis:

- The command `foo` checks for the presence of the right `FOO` in the matrix entry $A(a, b)$ and ensures it is not present from b to a.
- The command does not create, copy, or enter the right `FOO` anywhere else in the access control matrix. It only checks its presence as part of a conditional check.
- The only modification to the access control matrix is the creation of a new object X using the 'create' operation, and assigning the right `'owner'` to subject a using the 'enter' operation. The right `FOO` is not entered into any new access matrix entries.

Conclusion:

Since the command does not introduce the right `FOO` into any new entries of the access control matrix or transfer it to other subjects or objects, it does **not leak** the right `FOO`. Therefore, the command is considered **safe** with respect to the right `FOO`, and there is no risk of propagation that would compromise system safety in the context of the HRU model.

1.12 Question 13 Answer

It is currently impossible to prove that a protection system is safe in the context of the HRU model, as determining whether a system is safe (i.e, whether a specific right can be leaked) is undecidable. As a result, this undecidability is associated with the Halting Problem, which means that if we could prove safety for a general protection system, we could also solve the Halting Problem, which is known to be unsolvable in general. Thus, if your friend's protection system follows the full HRU model with unrestricted commands like creating new subjects, objects, or rights, it is not possible to definitively prove its safety. However, in some specific and restricted cases of protection systems—such as systems that limit operations or restrict rights propagation—safety may be decidable, and it could be possible to prove that the system is safe. To determine whether your friend's claim holds, you would need to know if their protection system is subject to such constraints that make safety decidable.

Logistics Company Corporate Database Model

Owner:
Reviewer:
Contributors:
Date Generated: Thu Oct 10 2024

Executive Summary

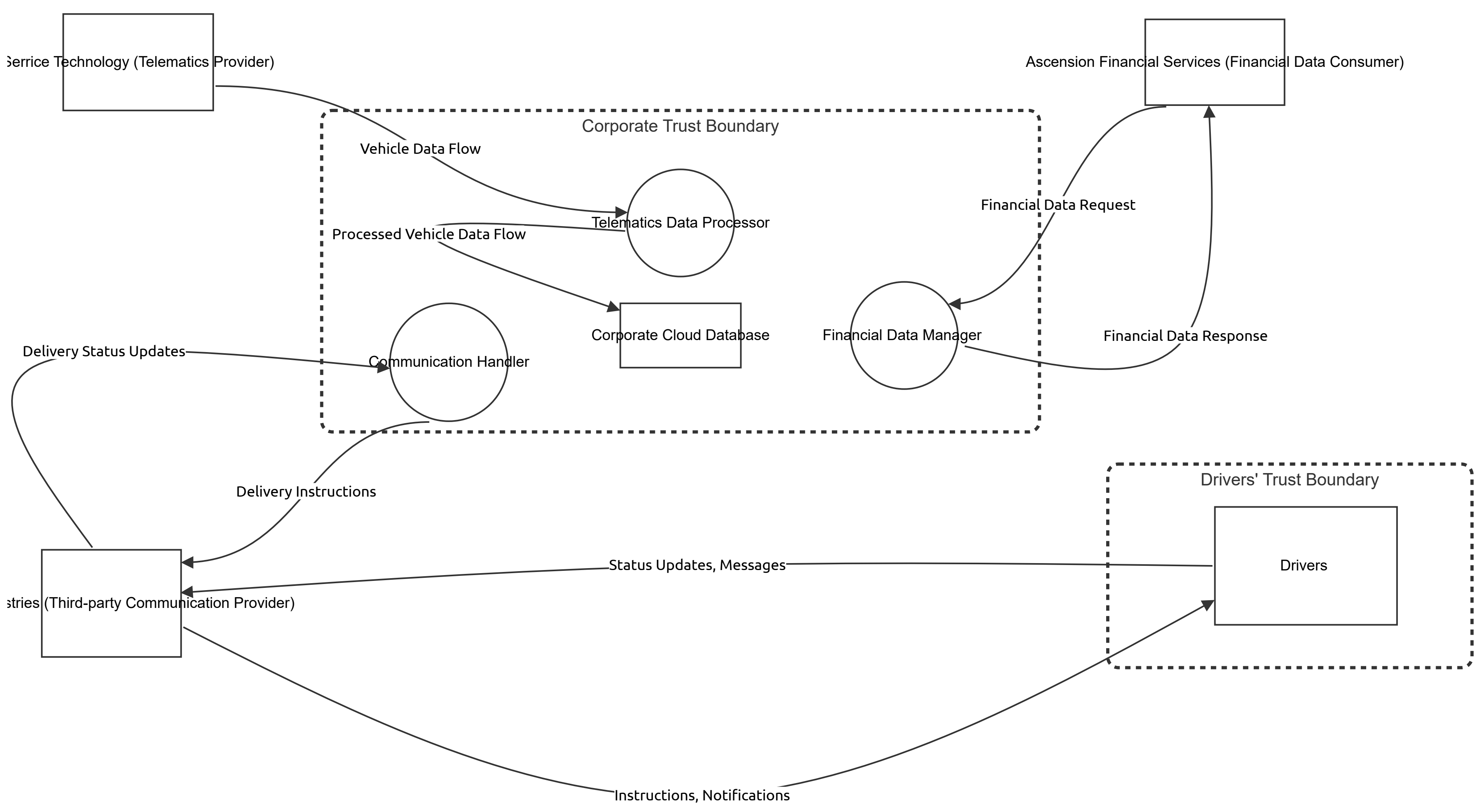
High level system description

You work for a logistics company and maintain your corporate database on a cloud provider's platform. Consider the following.
● Serrice Technology provides telematics (telematics is hardware and software for monitoring a fleet of vehicles) and pushes information to your cloud database. This information includes vehicle location, weight, and health.
● Ascension Financial Services has a data feed from your cloud database. This information includes contract fulfillment, payment processing, and accounts due.
● You communicate with your drivers using a third-party communication provider, Sonax Industries.

Summary

Total Threats	0
Total Mitigated	0
Not Mitigated	0
Open / High Priority	0
Open / Medium Priority	0
Open / Low Priority	0
Open / Unknown Priority	0

Logistics Company Corporate Database Model



Logistics Company Corporate Database Model

Corporate Cloud Database (Actor)

This stores sensitive corporate data related to vehicle telematics, financial information, and driver communications.

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Serrice Technology (Telematics Provider) (Actor)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Ascension Financial Services (Financial Data Consumer) (Actor)

Retrieves financial information from your cloud database for contract fulfillment, payment processing, and accounts receivable.

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Sonax Industries (Third-party Communication Provider) (Actor)

Facilitates communication between your system and your drivers, handling sensitive delivery instructions and status updates.

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Drivers (Actor)

Use devices (e.g., mobile apps) to receive instructions and send status updates via Sonax Industries to your system.

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Telematics Data Processor (Process)

Processes incoming vehicle data from serrice technology.

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Financial Data Manager (Process)

Handles financial data exchanges with Ascension Financial Services

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Communication Handler (Process)

Manages communications with Sonax Industries

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Vehicle Data Flow (Data Flow)

Vehicle Data(Location, Weight, Health)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Processed Vehicle Data Flow (Data Flow)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Financial Data Request (Data Flow)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Financial Data Response (Data Flow)

(Contracts, Payments)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Delivery Instructions (Data Flow)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Delivery Status Updates (Data Flow)

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Instructions, Notifications (Data Flow)

Instruction Flow

Number	Title	Type	Priority	Status	Score	Description	Mitigations

Status Updates, Messages (Data Flow)

Number	Title	Type	Priority	Status	Score	Description	Mitigations