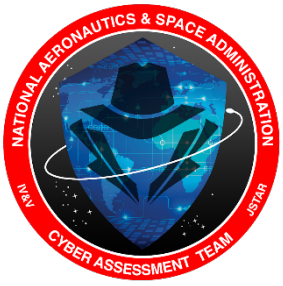
DSS Security Assessment



This assessment was performed by the Mission Protection Service (MPS) under the Safety and Mission Assurance (SMA) Office within NASA's Independent Verification and Validation (IV&V) Programin support of the Air Traffic Management - eXploration (ATM-X) Project.

Version: ***Project Release***

Date: 9/26/2024

Contents

1. Executive Summary

The Discovery and Synchronization service (DSS) has implemented some reasonable technical controls that help improve security and there are very few technical findings. The use of Docker and Kubernetes helps simplify the deployment process, and the DSS uses mutual TLS (mTLS) to connect. Much of the security risk across the DSS is a factor of its nature, a distributed environment that relies on all members to secure their parts correctly. As such, the DSS team should attempt to emphasize security controls that reduce complexity for securing entities' Cockroach DB (CRDB) instances properly and improve coordination among DSS members for things like security patching, incident response, detecting, and removing bad actors. The DSS team must recognize that operating a DSS instance securely will require a combination of technical and procedural controls. Each DSS entity must configure their instance properly and follow standard operating procedures to ensure that the DSS service is secure.

2. Introduction

This assessment was performed by the Mission Protection Service (MPS) under the Safety and Mission Assurance (SMA) Office within NASA's Independent Verification and Validation (IV&V) Program in support of the Air Traffic Management - eXploration (ATM-X) Project.

NASA has researched traffic management approaches for small unmanned aircraft systems (UAS) for a decade. That research is becoming operational, but there are cybersecurity questions that still need to be answered. The DSS cybersecurity analysis task seeks to document risks in the current approach to configuration and deployment of the DSS, which is a fundamental component of the UAS Traffic Management (UTM) ecosystem. The DSS relies on a database technology called Cockroach DB, which is a source-available distributed datastore that allows SQL queries but stores data as key-value pairs intelligently placed among multiple servers, allowing for high availability and resiliency. Multiple entities can provide the DSS as a logical service. Thus, the CRDB pool can have nodes that different organizations supply. These organizations must authorize each other's nodes so that CRDB can store and synchronize data properly. The nodes communicate via Transmission Control Protocol (TCP) on a specified port secured by Transport Layer Security (TLS). As such, the DSS analysis task seeks to identify potential security risks in using a CRDB deployment across organizational boundaries and identify security risks in the general DSS deployment strategy.

Overall, the report will define the team's approaches to analyzing the DSS and its place within the UTM ecosystem, document cybersecurity risks related to the current approach to the configuration and deployment of DSS, rank those risks relative to each other, and offer recommendations for risk reduction or remediation of the issues for the long-term security of the DSS and UTM ecosystem.

3. Approach

The Cyber Assessment Team's (CAT) approach to analyzing the DSS within the UTM underscores the critical role of this system. The DSS, a vital infrastructure enabling real-time data synchronization and communication between multiple UAS service providers (USS), was the focus of the analysis. CAT began by examining its architectural foundation, particularly its reliance on CRDB, a distributed, highly-available

SQL-compatible database. The distributed nature of CRDB and its deployment across organizational boundaries posed several unique security challenges that we sought to address.

Step 1: Conceptual Evaluation of DSS

CAT started by conceptually evaluating the DSS to understand its role in the UTM ecosystem. The primary function of the DSS is to enable the safe, real-time exchange of UAS operational data between USSs, which different organizations typically manage. Given the distributed nature of the service, ensuring secure, consistent, and trusted communication between nodes is critical. Each organization in a DSS pool operates its own CRDB node, meaning that multiple organizations must trust each other's infrastructure to maintain data integrity and synchronization. CAT's evaluation focused on identifying potential risks associated with managing trust, enforcing mTLS for secure communication, and ensuring data consistency across the distributed environment, but also evaluated proximate needs for the DSS such as monitoring, logging, and incident recovery. Additionally, the DSS conceptual analysis was partially driven by five questions asked within the original task order:

1. How is the approach to authorizing CRDBs with each other affected by deploying a single CRDB across organizational boundaries?
2. Are the CRDB configuration instructions provided in the DSS repository reasonable? Sufficient? Flawed?
3. Are there any concerns about how "TCP secured by TLS" is configured for this DSS application?
4. Are there best practices (or anti-patterns) for exchanging critical information between organizations that provide DSS?
5. Are there new cybersecurity requirements (i.e., not currently documented or not presently known) that should be levied on entities that join a DSS pool?

Note that the findings from the DSS conceptual analysis are presented in the main report, providing an overview of the evaluation. For those who seek a deeper understanding, ***the raw conceptual findings for DSS, framed in the context of the five questions, can be found in Appendix A***.

Step 2: Technical Analysis of the DSS Implementation

Next, CAT conducted a detailed technical analysis of the DSS implementation using mTLS for secure communication between CRDB nodes, emphasizing certificate management, key exchange, and trust establishment between different organizations. Each organization operates its certificate authority (CA), requiring secure mechanisms to exchange CA certificates and update trust stores. CAT reviewed the certificate sharing process, as well as the deployment scripts (such as make-certs.py and apply-certs.sh), DSS source code, and Docker / Kubernetes configurations, and evaluated the security risks associated with rotating, revoking, and distributing certificates across organizational boundaries. CAT also analyzed the role of the CRDB in ensuring data encryption at rest and in transit, emphasizing the importance of configuring TLS correctly to prevent vulnerabilities. ***All technical findings can be found in Appendix B.***

Step 3: Threat Modeling

Based on the conceptual and technical evaluation of the DSS implementation, the CAT team applied the MITRE ATT&CK framework, a knowledge base of adversarial techniques based on real-world observations on how adversaries interact with systems during an operation, to the DSS to help identify which security findings posed the most risk. In the MITRE ATT&CK framework, Tactics, Techniques, and Procedures (TTPs) are a core concept used to describe adversaries' behavior during cyberattacks. Tactics are the why behind an attack. Tactics represent the overall objectives or goals adversaries aim to achieve, such as gaining initial access, executing malicious code, or exfiltrating data. Each tactic reflects a specific phase of an attacker's behavior in the cyber kill chain. Examples of tactics include Initial Access, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, and Exfiltration. Techniques are the how an attacker accomplishes their objectives and detail the methods or actions adversaries use to achieve specific tactics. For example, under the tactic "Credential Access," techniques might include methods such as keylogging, brute-forcing passwords, or credential dumping. Procedures are the specific implementations of techniques used by adversaries. These are detailed actions based on a particular threat actor or campaign. Procedures describe how a threat actor might apply a technique in a real-world attack, including specific tools, configurations, or strategies they employ. For example, while many adversaries may use credential dumping (under the Credential Access tactic), different groups may use various procedures (specific malware or tools) to achieve it.

The CAT team attempted to identify relevant Tactics and Techniques related to capabilities within the DSS implementation to help flesh out the core security pertinent capabilities of the DSS. In addition, the development team can use the TTPs in this report to help perform a threat modeling exercise if desired. Finally, MITRE ATT&CK TTPs can be related to ISO 27001, to which the DSS claims compliance.

Step 4: Identifying Key Security Concepts and Themes

Through CAT's conceptual, technical, and threat evaluations, several key security concepts and themes emerged that are vital to the secure operation of the DSS. These include:

1. Mutual TLS and Certificate Management
    a. Mutual TLS Encryption: Ensuring secure communication between CRDB nodes using mutual TLS involves generating and managing certificates for each node.
    b. Certificate Sharing and Trust: Establishing mutual trust by sharing CA certificates among organizations, concatenating them into a single file, and distributing them to all nodes.
    c. Certificate Rotation and Revocation: Implementing regular certificate rotation and an effective revocation process to maintain security.
2. Cluster Initialization and Node Joining
    a. Cluster Initialization: Configure and initialize new nodes to join the existing cluster, ensuring they trust all necessary CA certificates.
    b. Dynamic Trust Management: Mechanisms to dynamically update trust stores with new CA certificates as organizations join or leave the DSS instance.
3. Data Encryption
    a. Encryption at Rest and in Transit: Ensuring that data stored on CRDB nodes is encrypted and all inter-node communication is secured using TLS.
4. Access Control and Authorization
    a. RBAC: Implementing RBAC ensures only authorized personnel and services can access specific data and functions within the CRDB cluster.

    b. Granular Permissions: Setting granular permissions to control access based on the principle of least privilege.
5. Compliance and Data Privacy
    a. Regulatory Compliance: Ensuring the deployment complies with relevant regulations such as GDPR, HIPAA, and CCPA.
    b. Data Segregation and Audits: Properly segregate data and conduct regular audits to ensure ongoing compliance and data privacy.
6. Monitoring and Logging
    a. Centralized Monitoring: Implementing a centralized monitoring system to collect and analyze logs and metrics from all nodes.
    b. Anomaly Detection: Using tools to promptly detect and respond to anomalies or potential security incidents.
7. Incident Response and Recovery
    a. Incident Coordination: Developing clear procedures for coordinating incident response efforts across multiple organizations.
    b. Disaster Recovery: Ensuring all organizations have consistent disaster recovery plans, regular backups, and tested recovery procedures.
8. Configuration Management
    a. Consistent Configurations: Ensuring consistent configurations across all nodes to prevent misconfigurations and maintain security.
    b. Secure Network Segmentation: Segmenting networks to isolate CRDB nodes from other organizational networks and prevent unauthorized access.
9. Automation and Scripting
    a. Automated Deployment and Maintenance: Using automated scripts to handle deployment, certificate management, and regular maintenance tasks.
    b. Consistency Checks: Implementing automated consistency checks to ensure all nodes are properly configured and up-to-date.
10. Enhanced Documentation and Support
    a. Beginner-Friendly Guides: Providing detailed documentation and guides to help new users understand and follow the deployment process.
    b. Troubleshooting and Examples: Expanding troubleshooting sections and providing more examples and use cases to address potential issues and complex scenarios.

The Key Security Concepts and Themes will be used to frame findings, recommendations, and risk assessments throughout the rest of this report.

4. Summary of Findings

The findings section is organized with respect to DSS key security concepts / themes as identified by the CAT team. Each concept / theme will have findings, recommendations, related TTPs, and relevant ISO 27001 controls. The intent of this section is to provide a reasonable set of concerns and recommendations without overwhelming the DSS team. The related MIRRE ATT&CK TTPs and ISO 27001-2022 are intended to provide the DSS stakeholders a reference in case they want to do further threat modeling or are interested in the ISO controls associated with CAT findings. The TTPs and controls were also used in a risk assessment of the concerns identified by CAT, described in the Risk Assessment section.

Mutual TLS and Certificate Management

Capabilities and Concerns

1. Mutual TLS Encryption
    a. Key Capability: Each CRDB node in the cluster communicates securely via mTLS. This ensures that all data transmitted between nodes is encrypted and authenticated. Each node requires a certificate signed by its organization's internal CA.
    b. Issue Identified: Establishing and maintaining mTLS across multiple organizations is complex, especially when each organization manages its own CA. The challenge is ensuring that all nodes trust certificates issued by other organizations.
    c. Concern: Incorrect or incomplete certificate configurations can result in failed communications between nodes, potentially leading to data synchronization issues or security vulnerabilities.
2. Certificate Sharing and Trust
    a. Key Capability: To establish mutual trust between nodes from different organizations, organizations must share their CA certificates. Sharing is done by concatenating the CA certificates of all participating organizations into a single file, which is distributed to all nodes.
    b. Issue Identified: The process of sharing and updating CA certificates must be revised and automated, creating potential gaps in trust establishment, particularly when organizations join or leave the DSS instance.
    c. Concern: If any CA certificate is missing or outdated, trust between nodes can break down, preventing secure communication and potentially exposing the system to man-in-the-middle attacks.

3. Certificate Rotation and Revocation
    a. Key Capability: Regular rotation of certificates is necessary to maintain security and prevent the risk of compromised certificates. Revocation processes are also crucial in the event of a key compromise.
    b. Issue Identified: Currently, no automated mechanism for rotating or revoking certificates across the entire DSS instance involves multiple organizations. This lack of coordination increases the risk of using outdated or compromised certificates.
    c. Concern: Without an effective process for certificate revocation and timely updates across all nodes, the system could remain vulnerable to compromised credentials, leading to data breaches or unauthorized access.

Recommendations

1. Mutual TLS Encryption
    a. Automated Certificate Management: Automate the process of generating, distributing, and configuring certificates for each node. Use a centralized or distributed tool to manage certificates and ensure consistency across all organizations.
    b. Strong TLS Configurations: Ensure that TLS configurations use the latest versions (TLS 1.2 or 1.3) and that only robust and secure cipher suites are enabled. Periodically review encryption settings and perform automated checks to prevent misconfigurations.
2. Certificate Sharing and Trust
    a. Centralized or Federated CA Management: Establish a more structured process for sharing CA certificates between organizations. The process could include using a centralized or federated service to exchange CA certificates securely and ensure that all nodes receive the latest CA certificates promptly.
    b. Automated Trust Updates: Implement automated scripts to concatenate and distribute the CA certificates to all nodes across the DSS instance. Configuration management tools (e.g., Ansible, Puppet) could be used to ensure that all trust stores are updated consistently.
3. Certificate Rotation and Revocation

a. Automated Certificate Rotation: Develop an automated system for rotating certificates regularly. This system should coordinate between all participating organizations to ensure that all nodes trust new certificates before the old ones expire.
b. Revocation Process: Implement a shared revocation mechanism to propagate certificate revocation information quickly across all nodes. The revocation mechanism could involve regularly distributing Certificate Revocation Lists (CRLs) or utilizing the Online Certificate Status Protocol (OCSP) to ensure compromised certificates are promptly invalidated.
c. Monitoring for Certificate Validity: Continuously monitor certificates across all nodes to ensure they are valid and trusted. Set up alerts for expiring or revoked certificates to prompt timely action.

Applicable Threats / TTPs

1. Mutual TLS Encryption
    a. Tactic: Defense Evasion (TA0005), Credential Access (TA0006)
        i. Technique: Adversary-in-the-Middle (T1557)
            1. Description: Adversaries may position themselves between two or more networked devices to intercept or manipulate communications. Using weak or improperly configured TLS can lead to an adversary-in-the-middle attack. Proper mTLS configuration prevents such attacks.
            2. Applicability: Mutual TLS ensures both parties are authenticated, reducing the risk of communication interception or manipulation by attackers.
        ii. Technique: Exploitation for Credential Access (T1212)
            1. Description: Weak or misconfigured encryption mechanisms (including TLS) can be exploited to capture credentials or sensitive information.
            2. Applicability: Ensuring strong encryption (TLS 1.2 or 1.3) and proper mutual TLS configuration helps prevent the exploitation of communication channels for credential access.
    b. Tactic: Initial Access (TA0001)
        i. Technique: Exploit Public-Facing Application (T1190)
            1. Description: Adversaries may target publicly exposed services vulnerable due to misconfiguration or poor encryption practices.
            2. Applicability: Ensuring secure mutual TLS encryption protects public-facing services from being exploited by attackers.
2. Certificate Sharing and Trust
    a. Tactic: Credential Access (TA0006)
        i. Technique: Steal Application Access Token (T1528)
            1. Description: Adversaries may attempt to steal application access tokens, which could allow them to impersonate legitimate users or services.
            2. Applicability: Managing trust between nodes via certificates reduces the risk of token or certificate theft, as mTLS ensures identity validation between nodes.
    b. Tactic: Defense Evasion (TA0005)
        i. Technique: Valid Accounts (T1078)
            1. Description: Adversaries may attempt to use valid credentials to gain unauthorized access to systems.
            2. Applicability: Secure certificate sharing ensures that only valid and trusted certificates are used, reducing the risk of unauthorized access using stolen or forged certificates.
    c. Tactic: Lateral Movement (TA0008)
        i. Technique: Use Alternate Authentication Material (T1550)

1. Description: Adversaries may attempt to use alternate authentication material, such as keys or certificates, to authenticate and move laterally within a network.
2. Applicability: Proper trust establishment through certificate sharing reduces the risk of adversaries using alternate certificates or keys to access the system.

3. Certificate Rotation and Revocation
    a. Tactic: Credential Access (TA0006), Defense Evasion (TA0005)
        i. Technique: Brute Force (T1110)
            1. Description: Adversaries may attempt to guess or brute-force keys or passwords. If certificates are not rotated regularly, an adversary who obtains one can maintain persistent access.
            2. Applicability: Regular certificate rotation minimizes the window of opportunity for attackers to use compromised certificates.
    b. Tactic: Persistence (TA0003)
        i. Technique: Valid Accounts (T1078)
            1. Description: Adversaries may maintain access using valid credentials, including certificates, long after the initial compromise. Failure to rotate or revoke certificates can allow attackers to persist in the environment.
            2. Applicability: Certificate rotation and revocation are essential for preventing attackers from using stolen or compromised certificates to maintain unauthorized access.
    c. Tactic: Defense Evasion (TA0005)
        i. Technique: Indicator Removal on Host (T1070)
            1. Description: Adversaries may attempt to remove artifacts or indicators of their activity from systems to evade detection. Regular certificate rotation ensures that attackers cannot continue to use the same access points unnoticed.
            2. Applicability: Timely revocation and rotation of certificates eliminate avenues for attackers to evade detection by using old or compromised certificates.

ISO 27001 Controls

1. A.5.19 - Secure Communication: Ensures that communication channels are secured with appropriate cryptographic mechanisms, including mTLS. This control requires the use of encryption to protect data in transit, which directly addresses the need to prevent man-in-the-middle attacks.
2. A.5.18 - Cryptographic Key Management: Manages the lifecycle of cryptographic keys and certificates, ensuring that certificates used for mTLS are securely generated, distributed, rotated, and revoked.
3. A.5.16 - Identity Management: Ensures that all entities (e.g., nodes) using mTLS for communication are properly identified and authenticated using certificates. This control helps ensure that nodes are authenticated and trusted during mTLS exchanges.
4. A.5.17 - Authentication Information Management: This control covers the secure handling of authentication credentials, including certificates. It ensures that certificates are shared securely, and that trust is established between entities using certificates, preventing the use of compromised or stolen certificates.
5. A.5.10 - Access Control: This control requires that access to information and systems is restricted based on established trust, such as certificates issued by trusted CAs. It ensures that certificate-based authentication is properly enforced across systems and entities.

6. A.5.21 - Information Exchange: Ensures that information, including CA certificates, is securely shared between systems and organizations. This control supports secure certificate sharing among organizations to establish mutual trust.
7. A.5.18 - Cryptographic Key Management: Manages the entire lifecycle of cryptographic keys, including regular rotation of certificates, timely revocation of compromised certificates, and ensuring that all nodes and systems are updated accordingly.
8. A.5.30 - Information Deletion: Ensures that revoked certificates are properly removed from trust stores and that outdated certificates are securely removed from systems to prevent unauthorized access using invalid certificates.
9. A.5.9 - Access Rights Management: Ensures that access rights (including certificates) are regularly reviewed and adjusted as necessary. This control helps ensure that compromised or outdated certificates are revoked and that valid certificates are rotated periodically to maintain security.

Cluster Initialization and Node Joining

Capabilities and Concerns

1. Cluster Initialization
    a. Key Capability: When initializing new nodes in the CRDB cluster, the process requires that all other nodes trust the node's CA certificate in the cluster. Each node must trust the CA certificates of all participating organizations.
    b. Issue Identified: Configuring and initializing new nodes is complex, especially across multiple organizations, because each organization has its own CA. Trust must be established manually by sharing and updating the combined CA certificate file on every node.
    c. Concern: Inconsistencies in how nodes are initialized and whether they correctly trust the necessary CA certificates can lead to failed node communication, synchronization issues, and potential security vulnerabilities.
2. Dynamic Trust Management
    a. Key Capability: Trust stores must be dynamically updated with new CA certificates as organizations join or leave the DSS instance. This requires that nodes are configured to trust new CA certificates and outdated or revoked certificates are removed promptly.
    b. Issue Identified: There is no automated process for dynamically updating trust stores across all nodes. Adding or revoking CA certificates is manual and prone to human error, delays, and inconsistencies across the cluster.
    c. Concern: Delays or failures in updating trust stores can result in nodes failing to communicate securely. Additionally, remaining outdated or compromised CA certificates in trust stores increase the risk of unauthorized access or man-in-the-middle attacks.

Recommendations

1. Cluster Initialization
    a. Automated Node Initialization: Develop and implement computerized scripts or tools to streamline initializing new nodes. Automation would ensure all nodes trust the necessary CA certificates without manual intervention.
    b. Consistency Across Nodes: Ensure all nodes are initialized with a standardized configuration that includes the most up-to-date CA certificates from all participating organizations. Use configuration management tools (e.g., Ansible, Puppet) to enforce consistency across the cluster.
    c. Node Discovery: Implement a reliable mechanism for nodes to discover each other and join the cluster. Consider using Kubernetes services or other service discovery mechanisms to ensure that new nodes can securely connect to the existing nodes.
2. Dynamic Trust Management

a. Automated Trust Store Updates: Implement an automated process for dynamically updating trust stores on all nodes when new CA certificates are added or revoked. Consider using configuration management tools to ensure that all nodes receive the updated combined CA certificate file.
b. Certificate Revocation Mechanism: Develop a shared mechanism to propagate certificate revocations quickly across all nodes in the cluster. The mechanism should include regular distribution of CRLsor use of OCSP responses to prevent compromised certificates from being used.
c. Monitoring and Alerts: Set up monitoring tools to track the state of CA certificates across all nodes. Alerts should be configured to notify administrators when certificates are about to expire, when new CA certificates need to be added, or when revocations have occurred.

Applicable Threats / TTPs

1. Cluster Initialization
    a. Tactic: Initial Access (TA0001)
        i. Technique: Exploitation of Remote Services (T1210)
            1. Description: Adversaries may attempt to exploit vulnerabilities in remote services or systems during node initialization to gain initial access or control over a system. Properly configuring and initializing nodes ensures that they are resistant to such attacks.
            2. Applicability: Ensuring secure initialization of new nodes with trusted CA certificates can prevent adversaries from exploiting remote services and gaining unauthorized access during the cluster setup phase.
    b. Tactic: Defense Evasion (TA0005)
        i. Technique: Impair Defenses (T1562)
            1. Description: Adversaries may attempt to disable or misconfigure defenses, such as certificate validation mechanisms, to evade detection or gain unauthorized access.
            2. Applicability: If nodes are not properly initialized to trust the correct CA certificates, adversaries could impersonate legitimate services by bypassing trust mechanisms.
2. Dynamic Trust Management
    a. Tactic: Defense Evasion (TA0005)
        i. Technique: Hijack Execution Flow (T1574)
            1. Description: Adversaries may attempt to hijack execution flows by compromising certificates or trust mechanisms, allowing them to alter or intercept legitimate communications between nodes.
            2. Applicability: Failure to dynamically update trust stores with new CA certificates as organizations join or leave can allow adversaries to hijack trust mechanisms and compromise the cluster.
    b. Tactic: Privilege Escalation (TA0004)
        i. Technique: Valid Accounts (T1078)
            1. Description: Adversaries may leverage valid credentials, including certificates, to escalate privileges and access additional resources.
            2. Applicability: Proper management of CA certificates and trust relationships ensures that valid certificates are rotated and revoked to prevent adversaries from using them to escalate privileges.
    c. Tactic: Credential Access (TA0006)
        i. Technique: Steal Application Access Token (T1528)
            1. Description: Adversaries may attempt to steal access tokens or certificates to impersonate legitimate users or services.

2. Applicability: Dynamically updating trust stores and properly managing certificate revocation can reduce the risk of adversaries using stolen or outdated tokens or certificates.

ISO 27001 Controls

1. A.5.16 - Identity Management: This control ensures that all entities (e.g., CRDB nodes) joining the system are securely identified and authorized. In the context of cluster initialization, this involves properly authenticating new nodes using certificates and establishing trust via CA validation.
2. A.5.10 - Access Control: This control mandates that access to information and services is controlled based on authorization levels. For CRDB nodes, it ensures that the proper mechanisms, such as mutual TLS and CA certificate validation, are in place during initialization to prevent unauthorized access.
3. A.5.13 - Secure Configuration of Information Systems: Ensures that the system is securely configured during initialization, including all security settings such as mutual TLS, key management, and trust establishment with CA certificates.
4. A.5.18 - Cryptographic Key Management: This control manages the lifecycle of cryptographic keys, including the creation, distribution, and revocation of CA certificates. It ensures that as organizations join or leave the DSS, their certificates are properly managed and updated across all trust stores dynamically.
5. A.5.17 - Authentication Information Management: Ensures that credentials and certificates used for authentication are securely managed, rotated, and revoked when necessary. It also ensures that trust relationships are dynamically maintained as new certificates are issued or revoked.
6. A.5.19 - Secure Communication: Protects communication channels between nodes by ensuring that mutual TLS is correctly implemented, and that trust stores are updated dynamically with the latest CA certificates.
7. A.5.30 - Information Deletion: Ensures that revoked or invalid certificates and authentication tokens are securely removed from trust stores, preventing unauthorized access to the cluster. This is particularly important when dynamically updating trust stores as organizations leave the DSS instance.

Data Encryption

Capabilities and Concerns

1. Encryption at Rest
    a. Key Capability: Data stored on CRDB nodes must be encrypted at rest to protect sensitive information from unauthorized access, particularly in cases where physical security controls may differ across organizations.
    b. Issue Identified: There needs to be more clarity on whether encryption at rest is enabled across all participating CRDB nodes. Some organizations might need to have configured CRDB to encrypt stored data, leading to inconsistencies in security levels across the cluster.
    c. Concern: Failure to encrypt data at rest can lead to unauthorized access to sensitive data, especially if physical security controls are insufficient or if an attacker gains access to the storage media.
2. Encryption in Transit
    a. Key Capability: All inter-node communication in the CRDB cluster must be encrypted using Transport Layer Security (TLS) to prevent eavesdropping, data tampering, and man-in-the-middle attacks.
    b. Issue Identified: Inconsistent configurations across nodes may prevent some communications from being adequately encrypted. If TLS is not enforced or configured correctly, sensitive data exchanged between nodes may be exposed to attackers.

c. Concern: Any communication between CRDB nodes that is not encrypted in transit leaves data vulnerable to interception, modification, or unauthorized access by attackers.

Recommendations

1. Encryption at Rest
    a. Enable Data-at-Rest Encryption: Ensure data-at-rest encryption is enabled on all CRDB nodes across all organizations. CockroachDB supports encryption at rest, which should be configured consistently across the entire DSS instance.
    b. Use Strong Encryption Algorithms: When configuring encryption at rest, ensure that strong encryption algorithms are used (e.g., AES-256) to protect data from unauthorized access.
    c. Audit and Validate Encryption Settings: Regularly audit the configuration of all CRDB nodes to ensure that encryption at rest is correctly enabled and that no node is storing data in an unencrypted format.
    d. Backup Encryption: Ensure that backups of CRDB data are also encrypted. Encryption is crucial to prevent unauthorized access to backup data, which could contain sensitive information.
2. Encryption in Transit
    a. Enforce TLS for All Communications: Ensure that TLS is enabled and enforced for all inter-node communications in the CRDB cluster, including both internal communications between nodes and any external communication channels.
    b. Use Secure TLS Versions: Configure all nodes to use the latest, secure versions of TLS (e.g., TLS 1.2 or TLS 1.3). Disable older, insecure versions such as TLS 1.0 and TLS 1.1 to protect against vulnerabilities.
    c. Automated Encryption Audits: Implement automated tools that regularly audit TLS configurations on all nodes to ensure that encryption in transit is consistently enforced and properly configured across the DSS instance.

Applicable Threats / TTPs

1. Encryption at Rest and in Transit
    a. Tactic: Defense Evasion (TA0005), Credential Access (TA0006), Exfiltration (TA0010), Defense Evasion (TA0005)
        i. Technique: Masquerading (T1036)
            1. Description: Adversaries may attempt to avoid detection by using techniques like data manipulation or modifying system artifacts to avoid encryption at rest.
            2. Applicability: Proper encryption at rest ensures that even if adversaries compromise the storage system, they cannot access sensitive data without the corresponding decryption keys.
        ii. Technique: Impair Defenses (T1562)
            1. Description: Adversaries may attempt to disable encryption or reduce its effectiveness (e.g., forcing the use of weak encryption algorithms or removing encryption in transit).
            2. Applicability: Ensuring that strong encryption algorithms are enforced for data-at-rest and in-transit ensures that defenses like TLS are not disabled or downgraded during attacks.
    b. Tactic: Credential Access (TA0006)
        i. Technique: Unsecured Credentials (T1552)
            1. Description: Adversaries may attempt to extract unencrypted credentials or sensitive information from systems where encryption is improperly implemented.

2. Applicability: Encryption at rest protects sensitive data, including credentials, preventing attackers from accessing unencrypted data or credentials stored on compromised nodes.

ii. Technique: Steal Application Access Token (T1528)

1. Description: Adversaries may attempt to intercept or steal application tokens or session credentials during communication between nodes if TLS is not properly enforced.
2. Applicability: Encryption in transit using TLS protects against man-in-the-middle attacks where attackers might attempt to steal application tokens or other credentials during inter-node communication.

c. Tactic Exfiltration (TA0010)

i. Technique: Exfiltration Over Alternative Protocol (T1048)

1. Description: Adversaries may exfiltrate data over alternative protocols or unsecured communication channels if TLS is not properly enforced.
2. Applicability: Ensuring that all communication between nodes is encrypted in transit (using TLS) prevents attackers from using unsecured channels to exfiltrate sensitive data.

ii. Technique: Exfiltration Over Encrypted/Obfuscated Channels (T1048.003)

1. Description: Adversaries may exfiltrate data over encrypted or obfuscated communication channels. However, if encryption in transit is not properly configured, attackers could leverage poorly secured channels.
2. Applicability: Proper TLS enforcement ensures that even if adversaries attempt to use encryption to exfiltrate data, secure configurations prevent unauthorized access or data manipulation.

ISO 27001 Controls

1. A.5.18 - Cryptographic Key Management: Control ensures that cryptographic keys for data-at-rest encryption are managed throughout their lifecycle, from generation to revocation. This is critical for protecting data from unauthorized access in case of compromise, addressing Impair Defenses (T1562) and preventing access to Unsecured Credentials (T1552).
2. A.8.23 - Secure Disposal or Reuse of Equipment: Ensures that when storage media or equipment is decommissioned, data is securely wiped or encrypted, preventing adversaries from accessing sensitive information. This mitigates the risks associated with Masquerading (T1036) and accessing unencrypted data.
3. A.5.9 - Access Rights Management: This control involves reviewing and adjusting access rights to encrypted data regularly. By limiting access to encryption keys and encrypted data, organizations can prevent unauthorized access, which helps mitigate Valid Accounts (T1078) and related Credential Access (T1552) techniques.
4. A.5.19 - Secure Communication: Ensures that communication channels are protected using encryption, such as TLS. This directly mitigates the risk of Man-in-the-Middle (T1557) attacks, as TLS protects the confidentiality and integrity of data in transit, preventing eavesdropping and data tampering. It also addresses risks such as Exfiltration Over Alternative Protocol (T1048).
5. A.5.21 - Information Exchange: Ensures secure and encrypted exchange of information, including between CRDB nodes, reducing the risk of data interception and theft during transmission. This control mitigates Credential Theft (T1528) and Exploitation of Public-Facing Applications (T1190) by enforcing encryption standards.
6. A.5.17 - Authentication Information Management: Focuses on the secure handling of authentication data, such as session tokens and credentials, especially when transferred over communication channels. This control protects against Stealing Application Tokens (T1528) and ensures that encrypted tokens are not misused during transit.

Access Control and Authorization

Capabilities and Concerns

1. Role-Based Access Control (RBAC)
    a. Key Capability: Implementing RBAC ensures only authorized personnel and services can access specific data and functions within the CRDB cluster. By segmenting roles and permissions, organizations can ensure that each entity only has the minimum access required to perform its role.
    b. Issue Identified: In multi-organizational environments, ensuring consistent RBAC implementation across all nodes can be challenging, especially when different organizations have their security policies. Lack of uniform RBAC enforcement could lead to unauthorized access to sensitive data.
    c. Concern: Without proper RBAC, unauthorized personnel or services may gain access to sensitive data or administrative functions within the cluster, increasing the risk of data breaches or accidental misconfigurations.
2. Granular Permissions
    a. Key Capability: Setting granular permissions based on the principle of least privilege ensures that access to sensitive data and functions is tightly controlled, minimizing the risk of unauthorized access and reducing the attack surface within the CRDB cluster.
    b. Issue Identified: Inconsistent or overly permissive access controls could allow users or services to access more resources than necessary, increasing the risk of insider threats or exploitation by malicious actors.
    c. Concern: Failure to enforce granular permissions across the DSS instance can result in users or services accessing data they are not authorized to view or modify, potentially leading to data leaks or system disruptions.

Recommendations

1. RBAC
    a. Implement Centralized Identity and Access Management (IAM): Use a centralized IAM solution that integrates with CRDB to enforce RBAC consistently across all nodes. RBAC will ensure that access policies are uniformly applied regardless of which organization manages the node.
    b. Automate RBAC Configuration: Automate the configuration and management of RBAC policies using tools like Ansible or Puppet. Automation will help prevent misconfigurations and ensure all nodes adhere to the same access control policies.
    c. Periodic Role Review: Regularly review roles and permissions to ensure that access rights are appropriate for each user or service's responsibilities. Reviews will help prevent privilege creep over time.
2. Granular Permissions
    a. Enforce the Principle of Least Privilege: Ensure that each user and service has only the minimum permissions necessary to perform their function. Least privilege can be achieved by carefully defining roles and using IAM tools to enforce granular permissions.
    b. Segment Data Access by Sensitivity: Classify data based on sensitivity and restrict access to critical data sets. Ensure that users and services that do not require access to sensitive data are denied by default.
    c. Regular Audits of Access Controls: Conduct regular audits of access permissions to verify that they are appropriate and have become manageable permissive. Implement automated checks where possible to detect and alert administrators to potential misconfigurations.

Applicable Threats / TTPs

1. RBAC
    Tactic: Privilege Escalation (TA0004)
        i. Technique: Valid Accounts (T1078)
            1. Description: Adversaries may leverage valid accounts (including service accounts or user accounts) with improperly configured RBAC to escalate their privileges and access unauthorized data or perform restricted operations within the CRDB cluster.
            2. Applicability: Ensuring that RBAC is properly implemented and strictly enforced helps prevent adversaries from exploiting valid accounts with excessive privileges.
    b. Tactic: Defense Evasion (TA0005)
        i. Technique: Impair Defenses (T1562)
            1. Description: Adversaries may attempt to disable or bypass security controls like RBAC by exploiting misconfigurations, especially if administrative privileges are granted to unauthorized users or services.
            2. Applicability: Proper RBAC implementation prevents users with unauthorized access from impairing or disabling security controls within the cluster.
    c. Tactic: Initial Access (TA0001)
        i. Technique: Exploit Public-Facing Application (T1190)
            1. Description: Adversaries may exploit a public-facing application with poorly configured RBAC to gain access to sensitive areas of the system or escalate privileges once inside.
            2. Applicability: Enforcing RBAC ensures that public-facing applications have restricted access, minimizing the attack surface for exploitation.
2. Granular Permissions
    a. Tactic: Privilege Escalation (TA0004)
        i. Technique: Access Token Manipulation (T1134)
            1. Description: Adversaries may attempt to manipulate access tokens, using granular permissions or insufficient access controls to escalate privileges or gain unauthorized access to additional resources.
            2. Applicability: Enforcing granular permissions ensures that even if tokens or credentials are compromised, the attacker's ability to escalate privileges is limited by the principle of least privilege.
    b. Tactic: Lateral Movement (TA0008)
        i. Technique: Pass the Hash (T1550.002)
            1. Description: Adversaries can leverage stolen credentials and abuse improper permission settings to move laterally within the system and access additional data or nodes in the cluster.
            2. Applicability: Granular permissions ensure that even if credentials are compromised, lateral movement within the system is restricted, and access to additional resources is minimized. Note: Windows only.
    c. Tactic: Credential Access (TA0006)
        i. Technique: Steal Application Access Token (T1528)
            1. Description: Adversaries may steal application access tokens and use them to bypass RBAC restrictions if granular permissions are not enforced properly.
            2. Applicability: Properly setting granular permissions ensures that access tokens have limited capabilities, minimizing the risk of unauthorized access even if a token is stolen.

ISO 27001 Controls

1. A.5.9 - Access Rights Management: This control ensures that access rights are properly assigned and regularly reviewed, guaranteeing that only authorized personnel have access to specific data or functions. Implementing RBAC ensures that users and services are granted appropriate access based on their roles and responsibilities, addressing the risks of Valid Accounts (T1078) and Impairing Defenses (T1562).
2. A.5.10 - Access Control: This control mandates strict access control mechanisms to protect sensitive information. It enforces RBAC and ensures that only authorized users can access specific functions or data within the system. This addresses Exploit Public-Facing Applications (T1190) by restricting unauthorized access to public-facing services.
3. A.5.11 - Secure Authentication: Ensures that authentication mechanisms are securely configured, requiring proper credentials or tokens for role-based access. This control addresses risks associated with Valid Accounts (T1078) by enforcing proper authentication for RBAC-controlled resources.
4. A.5.14 - Access Control to Source Code: This control ensures that source code access is restricted to authorized personnel, mitigating the risk of adversaries with elevated privileges accessing or modifying critical application components, addressing Impair Defenses (T1562).
5. A.5.9 - Access Rights Management: This control ensures that access permissions are assigned based on the principle of least privilege. It addresses Access Token Manipulation (T1134) and Pass the Hash (T1550.002) by ensuring granular permissions are enforced and that tokens do not grant excessive privileges.
6. A.5.10 - Access Control: Enforces granular permissions across systems, ensuring that only specific users or services have access to specific functions or datasets. This mitigates the risk of Steal Application Access Token (T1528) by ensuring that stolen tokens have limited privileges.
7. A.5.17 - Authentication Information Management: This control ensures the secure management and handling of authentication information (e.g., tokens, credentials), reducing the risk of stolen tokens being used for unauthorized access, addressing Steal Application Access Token (T1528) and limiting the capabilities of compromised credentials.
8. A.5.12 - User Authentication for External Connections: This control enforces secure authentication for external access to systems, ensuring that access tokens are properly authenticated and monitored. This mitigates the risk of unauthorized external connections using Pass the Hash (T1550.002) techniques.

Compliance and Data Privacy

Capabilities and Concerns

1. Regulatory Compliance
    a. Key Capability: Ensuring that the DSS deployment complies with relevant regulations such as GDPR, HIPAA, and CCPA is critical for maintaining the security and privacy of sensitive information, particularly given the sensitive nature of the data managed by the DSS.
    b. Issue Identified: Due to the decentralized nature of the DSS, with multiple organizations managing different nodes, ensuring consistent compliance across the entire system can take time and effort. Each organization may have different standards, security controls, and data protection policies, leading to consistency in maintaining compliance.
    c. Concern: Failure to comply with data protection regulations across all nodes could lead to significant legal penalties, data breaches, and loss of trust. If any organization fails to

comply with regulatory requirements, the entire DSS instance may be affected, compromising the privacy of individuals' data and the system's security.

2. Data Segregation and Audits
    a. Key Capability: Properly segregating data and conducting regular audits are essential for ensuring ongoing compliance with data privacy regulations and maintaining data integrity. Segregation ensures that data from one organization is not accessible to another without proper authorization.
    b. Issue Identified: Data segregation may not be implemented consistently across all nodes of the CockroachDB cluster, especially when managed by different organizations with varying levels of security controls. Additionally, regular audits of data segregation and access controls may not be enforced uniformly across all organizations, increasing the risk of unauthorized access to sensitive data.
    c. Concern: Insufficient data segregation or lack of regular audits could result in data breaches, regulatory violations, and unauthorized access to personal or sensitive information, increasing the potential for data leaks or misuse, undermining compliance with regulations such as GDPR and HIPAA.

Recommendations

1. Regulatory Compliance
    a. Standardize Compliance Practices: Implement centralized compliance guidelines across all organizations involved in the DSS to ensure uniformity in meeting regulatory requirements such as GDPR, HIPAA, and CCPA. Establish consistent data handling, retention, encryption, and breach notification policies.
    b. Compliance Automation Tools: Automate compliance monitoring tools to ensure that all nodes in the CRDB cluster adhere to the same regulatory requirements. These tools should automatically check each organization's compliance status and flag any discrepancies for timely remediation.
    c. Documentation and Audits: Provide detailed documentation and templates to help organizations demonstrate compliance with relevant regulations. Documentation should include audit trails, consent management practices, data handling procedures, and proof of compliance with regulatory requirements.
    d. Centralized Privacy Impact Assessments (PIAs): Require all organizations within the DSS to conduct privacy impact assessments regularly to ensure compliance with data privacy regulations. Each organization should identify potential risks to data privacy and take corrective measures to address them.
2. Data Segregation and Audits
    a. Implement Strict Data Segregation Policies: Enforce strict data segregation rules to ensure that data from one organization is not accessible by another without explicit permission. Implement RBAC and multi-tenant isolation mechanisms to prevent unauthorized data access.
    b. Regular Compliance Audits: Conduct regular audits of data segregation policies, encryption configurations, and access controls across all organizations involved in the DSS. Automate audit tools are used to ensure consistent monitoring and to identify any violations or potential risks early.
    c. Granular Access Control Enforcement: Ensure that granular access control mechanisms are implemented at each node to prevent unauthorized access to sensitive data. Data should be classified based on sensitivity, and access should be restricted based on the principle of least privilege.
    d. Encryption of Segregated Data: Ensure that segregated data is encrypted at rest and in transit, particularly for sensitive or personal information. Regularly audit encryption

mechanisms to ensure they meet regulatory standards and provide adequate protection against unauthorized access.

e. Comprehensive Audit Trails and Logging: Implement audit trails and logging for all data access and modification activities, ensuring transparency and accountability. Audit trails should be regularly reviewed to detect unauthorized data access or modifications, supporting ongoing compliance efforts.

Applicable Threats / TTPs

1. Regulatory Compliance
    a. Tactic: Data Destruction (TA0040)
        i. Technique: Data Destruction (T1485)
            1. Description: Adversaries may destroy data to prevent its retrieval. If compliance data (like personal or sensitive information subject to regulations) is destroyed, this could result in non-compliance with regulations like GDPR or HIPAA.
            2. Applicability: Ensuring compliance means that proper controls should be in place to prevent unauthorized data destruction, especially when dealing with regulations that require data retention.
    b. Tactic: Credential Access (TA0006)
        i. Technique: Credentials from Password Stores (T1555)
            1. Description: Adversaries may steal credentials from password stores to access sensitive data. This could result in the compromise of personally identifiable information (PII), or healthcare data protected under regulations.
            2. Applicability: Compliance requirements dictate strong password management policies and encryption standards to prevent credential theft.
2. Data Segregation and Audits
    a. Tactic: Defense Evasion (TA0005)
        i. Technique: Masquerading (T1036)
            1. Description: Adversaries may alter their activities to avoid detection during audits, bypassing segregation policies, or appearing as authorized users to access restricted data.
            2. Applicability: Segregating data properly ensures that even if adversaries masquerade as legitimate users, they cannot access data that is not meant for them. Proper auditing ensures detection of such activities.
    b. Tactic: Collection (TA0009)
        i. Technique: Data from Information Repositories (T1213)
            1. Description: Adversaries may collect sensitive information from databases, file systems, or information repositories. This is particularly relevant for GDPR or HIPAA-regulated data, where unauthorized collection is a significant compliance violation.
            2. Applicability: Regular audits can help detect unauthorized collection attempts, ensuring compliance with data privacy regulations.

ISO 27001 Controls

1. A.5.18 - Cryptographic Key Management: Ensures that cryptographic keys used for encrypting sensitive data are properly managed, protecting against unauthorized access or data breaches. This control helps maintain compliance with GDPR and HIPAA by ensuring that encryption keys are securely stored, and that data cannot be accessed even if credentials are compromised.
2. A.5.30 - Secure Disposal of Information: Ensures that data is securely deleted or disposed of when no longer needed, which aligns with the requirement to prevent unauthorized data destruction. This

control addresses T1485 (Data Destruction) by ensuring that only authorized individuals can delete or destroy data.

3. A.5.9 - Access Rights Management: Ensures that access to sensitive data is controlled and that only authorized individuals have access to protected information. This control addresses T1555 (Credentials from Password Stores) by ensuring that access controls and password management practices prevent unauthorized access to personal data.
4. A.5.10 - Access Control: This control enforces proper access control mechanisms, ensuring that only authorized users can access specific data and functions. By implementing this control, organizations can prevent unauthorized users from bypassing segregation policies, addressing T1036 (Masquerading).
5. A.5.21 - Information Security for Use of Cloud Services: This control ensures that data stored in cloud services is properly secured, segregated, and accessed only by authorized individuals. It addresses T1213 (Data from Information Repositories) by ensuring that sensitive information repositories are securely managed, preventing unauthorized data collection.
6. A.5.17 - Authentication Information Management: Ensures that authentication information, including credentials and access tokens, is securely managed to prevent unauthorized access to segregated data. This control supports T1036 (Masquerading) by ensuring that only valid credentials are used to access sensitive information.
7. A.5.29 - Secure Logging and Monitoring: This control mandates the logging and monitoring of activities to detect unauthorized access attempts or data collection. It helps address T1213 (Data from Information Repositories) by ensuring that all data access and modification activities are logged and audited regularly.
8. A.5.9 - Access Rights Management: Ensures that access rights are reviewed and adjusted as necessary, based on the principle of least privilege. This control supports T1036 (Masquerading) by ensuring that even if adversaries gain credentials, they are restricted to minimal access.

Monitoring and Logging

Capabilities and Concerns

1. Centralized Monitoring
    a. Key Capability: Implementing a centralized monitoring system that can collect and analyze logs and metrics from all nodes in the CRDB cluster.
    b. Issue Identified: The DSS deployment involves multiple organizations managing CRDB nodes, which makes centralized monitoring across all nodes challenging. Lack of centralized log collection and analysis could lead to delays in detecting node failures, security incidents, or performance bottlenecks.
    c. Concern: Without a centralized monitoring system, it may be difficult to maintain real-time insights into node health, detect security breaches, or ensure timely responses to critical events such as node failures or high latency.
2. Anomaly Detection
    a. Key Capability: Using tools to promptly detect and respond to anomalies or potential security incidents, ensuring that irregularities in data, performance, or security are caught in time.
    b. Issue Identified: Current mechanisms for detecting anomalies, unusual activities, or potential security breaches are insufficient. Without automated anomaly detection, organizations are at risk of not identifying potential incidents promptly.
    c. Concern: Failure to detect anomalies in a timely manner could lead to prolonged exposure to security incidents, unauthorized access, or service disruptions.

Recommendations

1. Centralized Monitoring
    a. Centralized Logging and Monitoring Tools: Implement a centralized system to aggregate and visualize logs and metrics from all nodes in the cluster. Tools like Grafana and Prometheus should be used to monitor system performance and availability.
    b. Configure Alerts: Set up automated alerts for critical events such as node failures, high latency, or security breaches.
    c. Real-Time Dashboards: Implement dashboards to provide real-time insights into cluster performance and system health.
2. Anomaly Detection
    a. Anomaly Detection Tools: Use anomaly detection tools and machine learning models to identify unusual patterns of activity that may indicate security incidents or system anomalies.
    b. Security Information and Event Management (SIEM): Implement a SIEM system to correlate events and detect potential security incidents across the CRDB cluster.
    c. Incident Response Procedures: Develop and test incident response procedures to ensure prompt and effective responses to anomalies. Ensure coordination across all organizations in the DSS pool.

Applicable Threats / TTPs

1. Centralized Monitoring
    a. Tactic: Collection (TA0009)
        i. Technique: Data from Information Repositories (T1213)
            1. Description: Adversaries may collect sensitive information from databases or information repositories that log system and security events. In a centralized monitoring system, these logs may contain valuable information that adversaries seek to access.
            2. Applicability: A centralized logging system that collects and analyzes data from multiple nodes can become a target for adversaries seeking to gather intelligence on the system, particularly if logs contain sensitive information or security configurations.
    b. Tactic: Defense Evasion (TA0005)
        i. Technique: Indicator Removal on Host (T1070)
            1. Description: Adversaries may delete or manipulate logs to avoid detection. Log manipulation is particularly relevant in centralized monitoring systems where logs from multiple nodes are aggregated for analysis.
            2. Applicability: Centralized monitoring systems need to protect logs from tampering, ensuring the integrity and availability of logs for audit and investigation purposes.
    c. Tactic: Execution (TA0002)
        i. Technique: System Services: Service Execution (T1569.002)
            1. Description: Adversaries may attempt to execute system services that can generate new logs or modify existing monitoring services.
            2. Applicability: Centralized monitoring systems must detect any unauthorized services being executed to alter monitoring outputs.
2. Anomaly Detection
    a. Tactic: Discovery (TA0007)
        i. Technique: File and Directory Discovery (T1083)

1. Description: Adversaries may explore the file system to identify logs or data points that are relevant to their objectives. Anomaly detection systems must be able to detect unusual patterns of file and directory access across monitored nodes.
2. Applicability: Anomaly detection tools should identify abnormal access to system logs and configurations that could indicate an adversary is attempting to explore or tamper with the system.

b. Tactic: Command and Control (TA0011)
   i. Technique: Encrypted Channel (T1573)
      1. Description: Adversaries may use encrypted channels to communicate with compromised nodes to evade detection. Anomaly detection tools can monitor for unusual encrypted traffic that deviates from the normal baseline.
      2. Applicability: Anomaly detection systems must be capable of identifying suspicious encrypted traffic patterns that indicate possible command and control activity.

c. Tactic: Collection (TA0009)
   i. Technique: Automated Collection (T1119)
      1. Description: Adversaries may automate the collection of log data or sensitive system information. Anomaly detection can help identify and alert when excessive or unusual collection of logs occurs.
      2. Applicability: Anomaly detection systems should flag unusual levels of data collection activity, which may indicate adversary attempts to extract valuable logs or security data.

ISO 27001 Controls

1. A.5.29 - Secure Logging and Monitoring: This control mandates the implementation of secure logging practices and monitoring of events across systems. Logs must be protected from tampering and must provide sufficient detail for forensic investigations, addressing T1070 (Indicator Removal on Host) by ensuring the integrity of logs in a centralized monitoring system.
2. A.5.28 - Protection of Information at Rest: This control ensures that sensitive information, such as logs from centralized monitoring systems, is adequately protected against unauthorized access. It addresses T1213 (Data from Information Repositories) by safeguarding log repositories from unauthorized access or exploitation.
3. A.5.30 - Event Monitoring: Requires that system activities are continuously monitored for security events, helping detect and respond to potential incidents. This control directly addresses T1569.002 (System Services: Service Execution) by ensuring that any unusual services or processes that may interfere with the centralized monitoring system are identified and investigated promptly.
4. A.5.18 - Cryptographic Key Management: This control ensures that cryptographic keys used to secure logs and monitoring data are properly managed, reducing the risk of log tampering or unauthorized access to sensitive log data, relevant to T1213 and T1070.
5. A.5.30 - Event Monitoring: This control requires continuous monitoring of system events to detect anomalies or suspicious activity. It directly addresses T1083 (File and Directory Discovery) and T1119 (Automated Collection) by identifying unusual access patterns or excessive data collection activities within the anomaly detection framework.
6. A.5.29 - Secure Logging and Monitoring: Logs must be monitored for abnormal activity, and anomaly detection tools should be integrated to detect irregular patterns in system behavior, such as unauthorized access or attempts to bypass security controls. This control supports anomaly detection relevant to T1083 and T1119.
7. A.5.10 - Access Control: This control enforces restrictions on access to logs, directories, and files, ensuring that only authorized personnel have the ability to view or modify sensitive data. It

addresses T1083 (File and Directory Discovery) by ensuring that access controls are in place to prevent unauthorized discovery or manipulation of log files.

8. A.5.21 - Information Security for Use of Cloud Services: Ensures that logs and monitoring data stored in cloud environments are protected against unauthorized access or tampering. This control helps secure log repositories, which are essential for effective anomaly detection, particularly in cloud-based DSS environments, addressing T1573 (Encrypted Channel) and T1083.

Incident Response and Recovery

Capabilities and Concerns

1. Incident Coordination
    a. Key Capability: Developing clear procedures for coordinating incident response efforts across multiple organizations is crucial in a decentralized DSS deployment. Each organization must align on communication channels, roles, and responsibilities during an incident.
    b. Issue Identified: In the DSS environment, multiple organizations manage different nodes, creating potential challenges in coordinating incident responses. Misaligned procedures or unclear responsibilities may lead to delays in incident responses, escalating security threats.
    c. Concern: The absence of a well-coordinated incident response plan across all organizations is a significant concern. Without this, security incidents may not be addressed in a timely or unified manner, leading to prolonged exposure to threats and potential data breaches. It's crucial that all organizations in the DSS ecosystem act in unison during such incidents.
2. Disaster Recovery
    a. Key Capability: Ensuring all organizations and the DSS itself have consistent disaster recovery plans, regularly perform backups, and have tested recovery procedures is critical to the DSS's overall resilience. Each organization and the DSS needs a robust recovery strategy to ensure continuity of services.
    b. Issue Identified: There may be inconsistencies in disaster recovery plans across different organizations participating in the DSS pool. Some organizations may not perform regular backups or may not have tested recovery procedures, which increases the risk of prolonged outages following a disaster.
    c. Concern: Inconsistent or absent disaster recovery strategies across the DSS ecosystem can result in data loss, prolonged downtime, or an inability to restore services after a major incident. E.g. if everyone's DSS instance for a particular area is hosted on us-east-2 and it goes down, how does the DSS continue to support UTM? However, the risks can be mitigated with regular testing and backup validation. All organizations in the DSS ecosystem should ensure these measures are in place.

Recommendations

1. Incident Coordination
    a. Develop a Unified Incident Response Plan: Create a joint incident response procedure that clearly defines the roles, responsibilities, and communication protocols for each organization involved in the DSS. The plan should ensure that all organizations are aligned on coordinating efforts during a security event. When a DSS provider signs up, they should be provided the plan to ensure that they can reasonably respond to incidents.
    b. Establish Secure Communication Channels: Implement secure and reliable communication channels for incident response coordination, ensuring that all organizations can quickly share information during an incident. This could include encrypted messaging platforms or dedicated incident response portals.

c. Conduct Regular Incident Drills: Perform joint incident response drills and simulations to test and refine the response process. Drills will help ensure that all organizations are familiar with their roles and can coordinate effectively during real incidents.

2. Disaster Recovery
    a. Create Consistent Disaster Recovery Plans: Ensure that each organization in the DSS pool develops a disaster recovery plan that aligns with the overall DSS strategy. Each plan should include regular backups, defined recovery time objectives (RTOs), and recovery point objectives (RPOs) for critical data and systems.
    b. Perform Regular Backups and Test Restorations: Mandatory regular, automated backups of critical data across all organizations, ensuring that they are encrypted and stored securely. Periodically test the restoration of these backups to verify that they can be successfully recovered during a disaster.
    c. Develop a Shared Recovery Strategy: Establish a shared recovery strategy to ensure that all organizations can restore services in a coordinated manner following a disaster. This may include mutual assistance agreements, shared recovery sites, or centralized disaster recovery management.
    d. Monitor and Review Disaster Recovery Plans: Periodically review disaster recovery strategies across the DSS. Perform audits to ensure that plans are up to date and effective, and that backup data is secure and accessible when needed.

Applicable Threats / TTPs

1. Incident Coordination
    a. Tactic: Execution (TA0002)
        i. Technique: Shared Modules (T1129)
            1. Description: Adversaries might take advantage of shared or decentralized resources to execute malicious code. Coordination between organizations helps prevent the misuse of shared resources.
            2. Applicability: Proper coordination between organizations ensures that incidents involving shared modules or dependencies are handled efficiently, minimizing the risk of exploitation.
    b. Tactic: Impact (TA0040)
        i. Technique: Service Stop (T1489)
            1. Description: Adversaries may stop services to disrupt operations. Incident response coordination is critical to restore services across all nodes in the DSS environment.
            2. Applicability: Effective incident coordination ensures that services affected by attacks, such as service stops, are quickly restored across all organizations.
    c. Tactic: Defense Evasion (TA0005)
        i. Technique: Indicator Removal on Host (T1070)
            1. Description: Adversaries may attempt to delete or modify logs to cover their tracks. Effective incident coordination ensures that logs and forensic data are preserved across organizations.
            2. Applicability: Coordinating the incident response ensures that tampering with logs or evidence across nodes is identified and addressed quickly, preventing further evasion.
2. Disaster Recovery
    a. Tactic: Impact (TA0040)
        i. Technique: Data Destruction (T1485)

1. Description: Adversaries may destroy data to hinder recovery or response efforts. Disaster recovery plans and backups are essential to mitigate this risk.
2. Applicability: Ensuring consistent disaster recovery plans and regular backups across all organizations in the DSS minimizes the impact of data destruction.

b. Tactic: Impact (TA0040)
   i. Technique: Disk Wipe (T1561.001)
      1. Description: Adversaries may wipe disks as part of a destructive attack. Regular, tested backups and disaster recovery procedures are essential to recover quickly from such events.
      2. Applicability: Effective disaster recovery and consistent backup testing ensure that disk-wiping incidents do not lead to irrecoverable data loss.

c. Tactic: Persistence (TA0003)
   i. Technique: Boot or Logon Initialization Scripts (T1037)
      1. Description: Adversaries may modify boot or logon scripts to maintain persistence. Disaster recovery plans should include procedures to detect and remove malicious persistence mechanisms.
      2. Applicability: Coordinated recovery efforts must ensure that modified or malicious initialization scripts are identified and restored to secure versions.

ISO 27001 Controls

1. A.5.22 - Incident Management: This control ensures that information security events are promptly reported and that response procedures are in place to manage incidents effectively across multiple organizations. It aligns with T1129 by enabling coordinated response efforts when shared modules or dependencies are targeted.
2. A.5.31 - Logging and Monitoring:
3. This control mandates that logging is properly implemented and monitored, ensuring that tampering with logs is detected. This control supports the preservation of logs and addresses T1070 (Indicator Removal on Host) by ensuring that log integrity is maintained and that incidents involving log tampering are investigated.
4. A.5.30 - Event Monitoring: Requires continuous monitoring of systems and services to detect disruptions, including T1489 (Service Stop) attacks. This control ensures that service disruptions are promptly detected and that response actions can be coordinated across organizations to restore service continuity.
5. A.5.36 - Communication Security: This control ensures the secure communication of sensitive information during incident response. It facilitates incident coordination across multiple organizations by protecting the confidentiality and integrity of incident response communications.
6. A.5.23 - Information Backup: This control ensures that regular backups of critical information are performed and stored securely. It directly mitigates T1485 (Data Destruction) and T1561.001 (Disk Wipe) by ensuring that data can be restored following an incident, minimizing the impact of data destruction and disk wiping.
7. A.5.24 - Redundancies: Ensures that systems and services have redundancy mechanisms in place to continue operating during or after an attack. This control is critical for mitigating T1489 (Service Stop) by ensuring services can be restored even after a disruption.
8. A.5.35 - Business Continuity Management: Focuses on establishing and maintaining business continuity and disaster recovery plans, including coordination among multiple organizations. This control ensures that each organization in the DSS environment has consistent disaster recovery strategies, addressing T1485 (Data Destruction) and T1561.001 (Disk Wipe).

9. A.5.32 - Information System Continuity: Ensures the availability of information systems in the face of a disruptive event. This control relates to T1037 (Boot or Logon Initialization Scripts) by ensuring that critical systems can be restored to a secure state after an attack that modifies initialization scripts.

Configuration Management

Capabilities and Concerns

1. Consistent Configurations
    a. Key Capability: Consistent configurations across all nodes within the CRDB cluster are essential to prevent misconfigurations and maintain security, especially when multiple organizations manage different nodes.
    b. Issue Identified: Due to the decentralized nature of the DSS environment, inconsistent configurations across nodes may arise, leading to vulnerabilities, operational issues, and security gaps, particularly in how certificates, network settings, or security policies are implemented across nodes.
    c. Concern: Misconfigured nodes can introduce attack vectors that adversaries could exploit. Additionally, inconsistent configurations may result in operational inefficiencies, secure communication failures, or node discovery and network segmentation challenges.
2. Secure Network Segmentation
    a. Key Capability: Segmenting networks to isolate CRDB nodes from other organizational networks is critical to preventing unauthorized access and reducing the risk of lateral movement in the event of a breach. Proper segmentation ensures that CRDB nodes are adequately protected from attacks originating from less secure areas of the network.
    b. Issue Identified: Inadequate network segmentation between CRDB nodes and other parts of an organization's infrastructure increases the risk of unauthorized access and makes it easier for attackers to move laterally through the network. Weak segmentation practices could allow an attacker who compromises one node to access others or even spread to other parts of the organization's network.
    c. Concern: Failure to properly segment CRDB nodes from the broader organizational network can lead to unauthorized access, data breaches, and challenges in maintaining security boundaries between organizations participating in the same CRDB cluster.

Recommendations

1. Consistent Configurations
    a. Standardize Configuration Procedures: Develop standardized configuration templates or scripts that ensure consistency across all CRDB nodes in the DSS pool. Use configuration management tools like Ansible, Puppet, or Chef to automate and enforce configuration consistency, reducing the likelihood of misconfigurations.
    b. Automate Configuration Audits: Implement automated configuration audits across the CRDB nodes to ensure all nodes adhere to the established baseline configuration. Audits can help detect deviations from expected configurations and flag potential security risks early on.
    c. Document and Review Configuration Settings: Ensure configuration settings for certificates, network policies, security protocols, and service parameters are clearly documented. Regularly review and update these settings to ensure they remain secure and appropriate for the evolving DSS environment.
    d. Use Version Control for Configuration Files: Implement version control for configuration files across all nodes to track changes and consistently apply updates. Tools like Git can

help maintain a consistent configuration history and ensure rollback capability in case of issues.

2. Secure Network Segmentation
    a. Implement Network Segmentation Policies: Use Virtual Private Networks (VPNs), Virtual Private Clouds (VPCs), and subnets to isolate CRDB nodes from the rest of the organization's networks. Ensure each organization segments its CRDB infrastructure from other parts of its network to prevent unauthorized access or lateral movement.
    b. Deploy Firewalls and Access Control Lists (ACLs): Configure firewalls and ACLs to restrict access to CRDB nodes. Only allow traffic from authorized IP ranges or services. To segment and secure communication between nodes, Kubernetes Network Policies, cloud-based firewalls, or dedicated firewalls can be used.
    c. Encrypt Network Traffic: Ensure all network traffic between CRDB nodes is encrypted using TLS or other strong encryption mechanisms. Encryption is essential to prevent attackers from intercepting sensitive information or manipulating inter-node communication.
    d. Regularly Test Segmentation Effectiveness: Conduct penetration and network audits periodically to ensure that segmentation policies effectively isolate CRDB nodes from unauthorized access. Testing ensures that segmentation strategies remain effective as the network evolves.
    e. Isolate Critical Components: Ensure that the most sensitive parts of the CRDB infrastructure, such as the management interfaces and administrative tools, are isolated on separate networks from the production environment to reduce the risk of privilege escalation or compromise.

Applicable Threats / TTPs

1. Consistent Configurations
    a. Tactic: Defense Evasion (TA0005)
        i. Technique: Modify System Image (T1601)
            1. Description: Adversaries may modify system images to maintain persistence or evade detection. Ensuring consistent configurations across all CRDB nodes can mitigate these risks by preventing unauthorized configuration changes.
            2. Applicability: Inconsistent configurations across nodes can be exploited by adversaries to introduce persistent mechanisms or evade detection, thus highlighting the importance of strict configuration management.
    b. Tactic: Initial Access (TA0001)
        i. Technique: Exploitation of Public-Facing Application (T1190)
            1. Description: Misconfigured nodes or inconsistent security settings in public-facing applications (e.g., CRDB nodes accessible via network interfaces) can lead to vulnerabilities being exploited for initial access.
            2. Applicability: Misconfigurations, if not consistently managed, can open up vulnerabilities that allow attackers to gain unauthorized access to CRDB nodes.
    c. Tactic: Execution (TA0002)
        i. Technique: System Services (T1569)
            1. Description: Adversaries may exploit system services that are improperly configured or inconsistently secured. Consistent configuration helps mitigate the risk of service exploitation.
            2. Applicability: Misconfigurations in services can be exploited to gain execution privileges or unauthorized access.
2. Secure Network Segmentation

a. Tactic: Privilege Escalation (TA0004)
    i. Technique: Exploitation for Privilege Escalation (T1068)
        1. Description: Adversaries may exploit vulnerabilities within networked environments to escalate privileges. Proper network segmentation reduces the attack surface by isolating critical CRDB nodes from other parts of the network.
        2. Applicability: Failure to segment the network properly may enable attackers to move laterally and escalate privileges across the infrastructure.
b. Tactic: Lateral Movement (TA0008)
    i. Technique: Exploitation of Remote Services (T1210)
        1. Description: Adversaries may use remote services to move laterally across network segments. Proper network segmentation ensures that compromised services are isolated, preventing lateral movement.
        2. Applicability: Secure segmentation limits the ability of attackers to exploit remote services and move laterally between CRDB nodes and other network segments.
c. Tactic: Defense Evasion (TA0005)
    i. Technique: Network Segmentation (T1049)
        1. Description: Segregating network traffic can prevent adversaries from evading detection or manipulating communication between nodes.
        2. Applicability: Effective segmentation helps in isolating threats, ensuring that adversaries cannot move freely between CRDB nodes and other parts of the infrastructure.
d. Tactic: Credential Access (TA0006)
    i. Technique: Network Sniffing (T1040)
        1. Description: Adversaries may use network sniffing to capture credentials in transit. Proper network segmentation, especially in combination with encrypted traffic (e.g., TLS), reduces the risk of credential exposure.
        2. Applicability: Segmentation, in combination with encryption, prevents adversaries from intercepting sensitive information such as credentials.

ISO 27001 Controls

1. A.5.15 - Configuration Management: This control ensures that configurations are consistently applied across systems, minimizing the risk of misconfigurations that can be exploited. It directly addresses T1190 (Exploitation of Public-Facing Application) by enforcing proper configuration management procedures and T1601 (Modify System Image) by preventing unauthorized configuration changes.
2. A.5.9 - Access Rights Management: Ensures that access to configuration settings and files is restricted to authorized personnel only, reducing the likelihood of unauthorized modifications to system images, addressing T1601 (Modify System Image) and T1569 (System Services).
3. A.5.14 - Access Control to Source Code: Protects access to source code and configuration files to ensure that only authorized personnel can modify critical configuration settings, mitigating T1601 (Modify System Image).
4. A.5.12 - User Authentication for External Connections: Ensures that users connecting to systems (including through public-facing applications) are properly authenticated, reducing the risk of T1190 (Exploitation of Public-Facing Application).
5. A.5.34 - Segregation of Networks: This control ensures that network segments are properly segregated based on the sensitivity and purpose of information. It addresses T1049 (Network Segmentation) by ensuring that critical systems, such as CRDB nodes, are isolated from other parts of the organization's infrastructure, reducing the risk of lateral movement.

6. A.5.18 - Cryptographic Key Management: Ensures the proper management of encryption keys used to secure network traffic. This control mitigates T1040 (Network Sniffing) by ensuring that encrypted traffic between segmented networks is protected from interception.
7. A.5.19 - Secure Communication: Ensures that all communication between network segments is encrypted and secure, protecting sensitive data in transit and preventing interception or manipulation, directly addressing T1040 (Network Sniffing).
8. A.5.11 - Secure Authentication: This control ensures that authentication mechanisms are in place for remote services and network access, mitigating T1210 (Exploitation of Remote Services) by preventing unauthorized access to critical network segments.
9. A.5.24 - Redundancies: Ensures that redundancies in network infrastructure are maintained, helping to prevent attackers from exploiting privilege escalation vulnerabilities, such as T1068 (Exploitation for Privilege Escalation), by isolating key network components.

Automation and Scripting

Capabilities and Concerns

1. Automated Deployment and Maintenance
    a. Key Capability: Using automated scripts for deployment, certificate management, and routine maintenance tasks ensures consistency, reduces human error, and speeds up operational processes across the DSS environment, especially when multiple organizations manage different nodes.
    b. Issue Identified: Manual deployment and maintenance processes introduce a higher risk of misconfiguration, inconsistencies, and delays in certificate management or service updates. This is particularly challenging in a decentralized DSS environment, where organizations must ensure that configurations are synchronized across nodes.
    c. Concern: Without automation, there is a higher risk of misconfigurations and errors during deployment, leading to potential security vulnerabilities, delays in operational tasks, and inconsistent configurations that can affect the system's overall performance and security.
2. Consistency Checks
    a. Key Capability: Implementing automated consistency checks to verify that all nodes are correctly configured, have the necessary certificates, and are up-to-date with security patches and other critical configurations.
    b. Issue Identified: In a distributed, multi-organizational environment, ensuring consistency across all nodes is crucial but can be challenging without automated tools. Manual checks for configuration consistency are error-prone and may not scale well, especially as more nodes and organizations join the DSS environment.
    c. Concern: Inconsistent configurations across nodes can lead to security vulnerabilities, operational failures, and troubleshooting difficulties. Regular automated checks ensure that all nodes are aligned with security policies and configuration standards, reducing risks associated with inconsistent settings.

Recommendations

1. Automated Deployment and Maintenance
    a. Develop Automated Deployment Scripts: Create scripts to automate the deployment of CRDB nodes, configuration settings, and certificate management. Automation will help ensure that each node is deployed with the correct configurations and that certificates are handled securely and consistently across the DSS environment.

b. Automate Certificate Management: Use automated scripts to generate, distribute, and rotate certificates across nodes. This ensures that nodes can securely communicate with each other and that certificate updates are handled seamlessly without requiring manual intervention.
c. Schedule Regular Maintenance Tasks: Automate routine maintenance tasks, such as security patching, performance tuning, and service restarts, to reduce the need for manual intervention and ensure systems remain secure and up-to-date.

2. Consistency Checks
    a. Implement Automated Configuration Audits: Develop automated tools to regularly check that all nodes have consistent configurations, certificates, and security settings. Automation will help ensure that no node deviates from the standard configuration, reducing the risk of security vulnerabilities.
    b. Monitor Certificate Validity and Expiration: Automate the monitoring of certificate validity and expiration to ensure that all certificates are valid and that nodes are not using expired or revoked certificates. Automated monitoring reduces the risk of communication failures or security incidents due to invalid certificates.
    c. Run Regular Consistency Reports: Generate regular reports on the consistency of configurations, certificates, and network settings across all nodes. The reports allow for continuous monitoring of configuration drift and provide insights into areas where further adjustments may be needed.
    d. Alert on Configuration Deviations: Set up automated alerts for any deviations from the expected configuration or inconsistencies in node settings. This will allow for quick remediation of issues before they affect the system's overall security and performance.

Applicable Threats / TTPs

1. Automated Deployment and Maintenance
    a. Tactic: Defense Evasion (TA0005)
        i. Technique: Deobfuscate/Decode Files or Information (T1140)
            1. Description: Adversaries may attempt to alter or tamper with deployment scripts or automated processes to deobfuscate sensitive information or gain unauthorized access.
            2. Applicability: Using secure, automated scripts for deployment and maintenance tasks is crucial to prevent adversaries from tampering with or injecting malicious code into the deployment process.
    b. Tactic: Initial Access (TA0001)
        i. Technique: Exploitation of Vulnerability (T1190)
            1. Description: If automated deployment scripts or services are misconfigured or lack security controls, adversaries may exploit these vulnerabilities to gain unauthorized access during deployment or maintenance.
            2. Applicability: Proper automation of deployment ensures secure and consistent configuration, reducing the risk of vulnerabilities during initial access.
    c. Tactic: Persistence (TA0003)
        i. Technique: Scheduled Task/Job (T1053)
            1. Description: Adversaries may use scheduled tasks or jobs to maintain persistence. Secure automation can prevent adversaries from injecting malicious tasks or misusing automated processes to remain persistent in the environment.
            2. Applicability: Regular automated maintenance should detect and mitigate unauthorized scheduled tasks that adversaries might use for persistence.

2. Consistency Checks
    a. Tactic: Discovery (TA0007)
        i. Technique: System Information Discovery (T1082)
            1. Description: Adversaries may attempt to gather information about the system configuration. Automated consistency checks can detect configuration anomalies that adversaries might try to exploit.
            2. Applicability: By regularly checking system information and configurations, automated consistency checks can prevent adversaries from exploiting misconfigurations to gain unauthorized access.
    b. Tactic: Privilege Escalation (TA0004)
        i. Technique: Exploitation for Privilege Escalation (T1068)
            1. Description: Misconfigured nodes or systems can be exploited for privilege escalation. Automated consistency checks ensure that all nodes are securely configured, reducing the risk of privilege escalation.
            2. Applicability: Consistency checks ensure that misconfigurations are identified and corrected, preventing adversaries from exploiting vulnerable configurations for privilege escalation.
    c. Tactic: Defense Evasion (TA0005)
        i. Technique: Impair Defenses (T1562)
            1. Description: Adversaries may attempt to impair defenses by modifying configurations or disabling security tools. Automated consistency checks can detect and alert on any attempts to modify or disable critical defenses.
            2. Applicability: Regular consistency checks help detect and prevent any defense impairments caused by adversarial modifications to the system configuration.

ISO 27001 Controls

1. A.5.15 - Configuration Management: Ensures the consistent and secure configuration of systems, including automated deployment scripts and infrastructure as code (IaC). This control mitigates the risk of misconfigurations or exploitation of vulnerabilities during deployment, addressing T1190 (Exploitation of Vulnerability).
2. A.5.23 - Information Backup: Ensures that automated backups and recovery scripts are securely configured to protect data from unauthorized access or modification. This also applies to automated deployment processes, reducing the risk of tampering by adversaries, as seen in T1140 (Deobfuscate/Decode Files or Information).
3. A.5.31 - Logging and Monitoring: Ensures that scheduled jobs and maintenance tasks are monitored for unauthorized changes or suspicious activity, addressing T1053 (Scheduled Task/Job) by detecting any unauthorized scheduled tasks or jobs used for persistence.
4. A.5.17 - Authentication Information Management: Protects the integrity of authentication mechanisms used in automated deployments. It ensures that credentials or sensitive information within deployment scripts are protected, preventing unauthorized access through tampered scripts, aligned with T1140 (Deobfuscate/Decode Files or Information).
5. A.5.15 - Configuration Management: Ensures that all nodes are consistently configured and that automated consistency checks are in place to prevent misconfigurations. This control helps detect configuration anomalies, addressing T1082 (System Information Discovery) and T1068 (Exploitation for Privilege Escalation).
6. A.5.12 - User Authentication for External Connections: Ensures that authentication information is securely managed, which is crucial for automated consistency checks, preventing unauthorized access that may lead to misconfigurations or exploitation of vulnerabilities, related to T1068 (Exploitation for Privilege Escalation).

7. A.5.29 - Secure Logging and Monitoring: Provides mechanisms to log and monitor system activity, including changes to configurations. This control is crucial in detecting and preventing T1562 (Impair Defenses), ensuring that all nodes and configurations are protected from adversarial tampering.
8. A.5.30 - Event Monitoring: This control ensures that automated consistency checks are actively monitored for suspicious or unauthorized activity. It can help identify attempts to disable or modify defenses, addressing T1562 (Impair Defenses) by detecting anomalous behavior early.

Enhanced Documentation and Support

Capabilities and Concerns

1. Beginner-Friendly Guides
    a. Key Capability: Providing detailed documentation and guides to help new users understand and follow the deployment process of DSS and CRDB.
    b. Issue Identified: The current documentation lacks beginner-friendly guidance, which may make it difficult for new users or organizations unfamiliar with the DSS deployment process to properly configure and maintain their systems. This can lead to misconfigurations or security gaps due to users not following proper procedures.
    c. Concern: Without clear and accessible documentation, new organizations might struggle with deployment, increasing the risk of configuration errors, operational delays, or security missteps, which could ultimately compromise the security and functionality of the DSS.
2. Troubleshooting and Examples
    a. Key Capability: Expanding troubleshooting sections and providing more examples and use cases to help users resolve issues and address potential complexities during deployment and operations.
    b. Issue Identified: The current troubleshooting documentation is limited and does not cover a wide range of potential issues or misconfigurations that users might encounter. Additionally, the examples provided are insufficient for handling complex or real-world scenarios, especially in multi-organization deployments.
    c. Concern: Without comprehensive troubleshooting guides and examples, users may find it difficult to resolve issues, leading to prolonged outages or misconfigurations, potentially affecting the availability and security of the DSS deployment.

Recommendations

1. Beginner-Friendly Guides
    a. Create Beginner-Friendly Guides: Develop beginner-friendly, step-by-step guides with screenshots and explanations to help new organizations or users get up and running quickly with the DSS. This should include quick start sections, explanations of key concepts, and best practices. Include a section on best practices for maintaining a secure and efficient DSS deployment. This covers regular updates, security patches, performance tuning, and backup strategies. Provide guidelines for scaling the CRDB cluster, including adding or removing nodes, balancing load, and optimizing performance. Considerations for geographic distribution and network latency should also be included. Provide guidance and best practices for tuning the performance of the CRDB cluster and DSS instances, such as database optimization, query tuning, and resource management.
    b. Expand Introductory Documentation: Break the deployment process into smaller, manageable steps, catering to users with varying levels of technical expertise.
    c. Include Examples and Use Cases: Add use cases that cover common and complex deployment scenarios, such as multi-region deployments, to provide users with more comprehensive guidance.

2. Troubleshooting and Examples
    a. Expand the Troubleshooting Section: Provide a detailed troubleshooting guide covering various potential issues, from network failures to certificate management problems, and their resolutions.
    b. Offer Detailed Examples: Include more detailed deployment examples for complex scenarios, such as multi-organization setups, scaling, and integration with other systems. These examples should be tied to specific use cases that reflect common challenges in DSS deployments.
    c. Provide Logs and Diagnostics Commands: Include log examples, diagnostic commands, and possible resolutions to help users quickly identify and fix common problems.

Applicable Threats / TTPs

1. Beginner-Friendly Guides
    a. Tactic: Initial Access (TA0001)
        i. Technique: Exploit Public-Facing Application (T1190)
            1. Description: Public-facing services such as CockroachDB nodes or DSS APIs might be deployed insecurely by new users if the documentation is not clear or comprehensive. This could leave the deployment vulnerable to exploitation.
            2. Applicability: Providing detailed, beginner-friendly deployment guides helps new users secure public-facing services, reducing the risk of exploitation during setup.
    b. Tactic: Persistence (TA0003)
        i. Technique: Account Manipulation (T1098)
            1. Description: Without clear guidance on account management, beginners may misconfigure user roles or access rights, leaving accounts vulnerable to manipulation by attackers who can gain persistence in the system.
            2. Applicability: Comprehensive documentation on user and account management can help new users configure access rights properly, mitigating the risk of adversaries manipulating accounts to maintain persistence.
2. Troubleshooting and Examples
    a. Tactic: Execution (TA0002)
        i. Technique: Command-Line Interface (T1059)
            1. Description: New users without sufficient troubleshooting documentation may incorrectly execute commands in the command-line interface (CLI), potentially introducing misconfigurations that could be exploited by adversaries.
            2. Applicability: Expanding the troubleshooting sections with detailed command-line examples and use cases helps users avoid errors during deployment, reducing the risk of improper execution.
    b. Tactic: Defense Evasion (TA0005)
        i. Technique: Exploitation for Defense Evasion (T1210)
            1. Description: Inadequate troubleshooting guidance or unclear examples may result in misconfigurations that adversaries can exploit to evade detection or bypass security mechanisms.
            2. Applicability: Providing clear examples and troubleshooting scenarios helps prevent misconfigurations that could lead to defense evasion, ensuring that security controls remain intact.

ISO 27001 Controls

1. A.5.15 - Configuration Management: Ensures that system configurations, including for public-facing applications, are consistently applied and securely managed. Beginner-friendly guides are essential to help users follow proper configuration standards, reducing the risk of insecure deployment, which addresses T1190 (Exploit Public-Facing Application).
2. A.5.9 - Access Rights Management: Ensures that access rights are properly assigned, reviewed, and managed. Clear documentation on managing user roles and access rights prevents unauthorized account manipulation, addressing T1098 (Account Manipulation).
3. A.5.10 - Access Control: Ensures that controls are in place to manage and restrict access to critical systems. Providing clear guides to set up proper access control mechanisms helps reduce the risk of misconfigurations that could allow unauthorized access.
4. A.5.31 - Secure Logging and Monitoring: This control ensures that logging mechanisms are in place and monitored for any suspicious or unauthorized activity. Providing proper troubleshooting guides, including logs and diagnostics, ensures that users can detect and respond to potential security issues, addressing T1210 (Exploitation for Defense Evasion).
5. A.5.30 - Event Monitoring: Ensures that all systems are monitored for potential issues or security incidents. With appropriate troubleshooting examples, users can prevent misconfigurations during command-line usage, helping address T1059 (Command-Line Interface).
6. A.5.15 - Configuration Management: Ensures consistent and secure configurations across systems. Detailed examples and troubleshooting guides prevent improper configurations, which can lead to exploitation, aligning with T1210 (Exploitation for Defense Evasion).

5. Risk Assessment

The risk assessment for DSS is based on the nineteen concerns within the Summary of Findings section. For the assessment, the CAT team used a 5x5 likelihood and impact score where likelihood is essentially "How likely is this concern to be exploited?" and impact is "What would be the impact on the system if the concern was exploited?".

Likelihood scores range from 1-5 and are represented as:
1. Rare (Almost never happens)
2. Unlikely (Has happened but very infrequent)
3. Possible (Could happen)
4. Likely (Has happened in similar systems)
5. Very Likely (Almost certain)

Impact Scores range from 1-5 and are represented as:
1. Insignificant (Minor effect on the system, no significant harm)
2. Minor (Some operational issues but no serious consequences)
3. Moderate (Noticeable impact on security, availability, or data integrity)
4. Major (Significant damage to the system, data breach or serious downtime)
5. Catastrophic (Critical failure, complete loss of control or severe data breach)

Risk scores are calculated with the formula Risk Score = Likelihood x Impact.

The rationale for each score is ultimately based on CAT's engineering judgment but includes weighting based on MITRE ATT&CK TTPs and ISO 27001. Concerns linked to tactics, techniques, and procedures in MITRE ATT&CK should be given higher likelihood considerations, mainly if they correspond to widely used methods by adversaries. Concerns tied to critical ISO 27001 controls were given higher impact considerations, especially those related to key security principles such as access control, encryption, and

incident management. Finally, the CAT team attempted to take into account the characteristics of the DSS itself, precisely its distributed nature and need for trust across multiple parties. Practically, the nature of DSS increased likelihood by a point, as the concerns being evaluated apply to numerous entities instead of just one, and a failure by any entity could lead to an impact. The concerns are ordered within the concepts identified in the Summary of Findings section.

Mutual TLS and Certificate Management

Concern 1: Incorrect or Incomplete Certificate Configurations

Likelihood: 4 (Likely)
Rationale: Incorrect certificate configurations are a known issue in many distributed systems. Based on T1078 (Valid Accounts) and T1190 (Exploitation of Public-Facing Application), improper certificate configurations can lead to unauthorized access or compromised authentication. It is common for organizations to misconfigure certificates due to complex management, leading to a relatively high likelihood of occurrence.

Impact: 4 (Major)
Rationale: If certificates are incorrectly configured or incomplete, communication between nodes may fail, exposing the system to data breaches or man-in-the-middle attacks. The impact is significant because it directly affects secure communication between CRDB nodes, which could result in service disruption or compromised data confidentiality, integrity, and availability. ISO 27001 controls like A.5.19 (Secure Communication) and A.5.15 (Configuration Management) emphasize this importance, making it a major concern.

Risk Score=4×4=16

Concern 2: Failure to Update CA Certificates

Likelihood: 3 (Possible)
Rationale: Updating CA certificates is a coordinated effort, and failure to do so may happen due to oversight or technical issues, especially in a multi-organization context like DSS. The likelihood is moderate since, although the process may fail, it can be automated and monitored. Based on T1562 (Impair Defenses) and T1070 (Indicator Removal on Host), an attacker could exploit unexpired certificates to maintain access or disrupt communication.

Impact: 4 (Major)
Rationale: If CA certificates are not updated, trust between CRDB nodes could break down, leading to service failures or insecure communications. The impact is high because outdated certificates might allow unauthorized access or cause network communications to stop functioning securely. ISO 27001 control A.5.18 (Cryptographic Key Management) underlines the criticality of certificate updates.

Risk Score=3×4=12

Concern 3: Certificate Rotation and Revocation

Likelihood: 3 (Possible)
Rationale: Regular certificate rotation and revocation can be challenging, but failure to rotate certificates is not uncommon. Based on T1070 (Indicator Removal on Host) and T1078 (Valid Accounts), attackers may exploit outdated or revoked certificates to maintain access. Therefore, the likelihood is possible but not as frequent as configuration errors.

Impact: 5 (Catastrophic)

Rationale: Failure to rotate or revoke compromised certificates can result in a catastrophic security breach. Attackers could exploit old or invalid certificates to bypass authentication, resulting in unauthorized access to data or services. Compromised certificates can severely undermine the entire system's security. ISO 27001 controls like A.5.18 (Cryptographic Key Management) stress the importance of timely certificate revocation.

Risk Score=3×5=15

Cluster Initialization and Node Joining

Concern 1: Cluster Initialization

Likelihood: 4 (Likely)
Rationale: Cluster initialization issues are common in distributed systems like CRDB due to the complexity of establishing trust between nodes. Based on T1078 (Valid Accounts), failure to correctly initialize new nodes can lead to unauthorized access or denial of service. Organizations may often struggle with configuration during initialization, making this a likely occurrence.

Impact: 4 (Major)
Rationale: Incorrect initialization of nodes could prevent nodes from joining the cluster, affecting service availability and leading to potential data inconsistencies. If trust is not established between nodes, the entire CRDB cluster might fail to operate securely, significantly impacting data integrity and operational continuity. ISO 27001 controls like A.5.15 (Configuration Management) ensure the importance of correct initialization.

Risk Score=4×4=16

Concern 2: Dynamic Trust Management

Likelihood: 3 (Possible)
Rationale: Updating trust stores dynamically as nodes join or leave the cluster requires coordination, and organizations may fail to update certificates in real-time. Failure to update upon leaving may be less frequent than initialization issues but still possible, especially when multiple organizations are involved. Based on T1556 (Modify Authentication Process), adversaries could exploit improper trust management to maintain unauthorized access.

Impact: 5 (Catastrophic)
Rationale: If dynamic trust management fails, new nodes may not be trusted, or unauthorized nodes may be able to communicate within the system, resulting in unauthorized data access or a breach in the security of the entire cluster. The impact is catastrophic because it compromises the whole cluster's integrity, violating ISO 27001 controls like A.5.19 (Secure Communication) and A.5.18 (Cryptographic Key Management).

Risk Score=3×5=15

Data Encryption

Concern 1: Encryption at Rest

Likelihood: 3 (Possible)

Rationale: While data encryption at rest is standard, failures can occur due to misconfigurations or lack of awareness. Based on T1078 (Valid Accounts) and T1003 (Credential Dumping), attackers may attempt to access data stored on the disk if encryption is not properly implemented. Encryption at rest is typically handled correctly, but there are cases where it is missed or misconfigured, making this issue possible.

Impact: 4 (Major)
Rationale: If data is not encrypted at rest, attackers with access to the storage medium could retrieve sensitive information without protection. Lack of encryption at rest could lead to data breaches or personal, sensitive, or regulatory-controlled data exposure. ISO 27001 controls such as A.5.23 (Information Backup) and A.5.18 (Cryptographic Key Management) emphasize the importance of encrypting sensitive data, making the potential impact of this concern major.

Risk Score=3×4=12

Concern 2: Encryption in Transit

Likelihood: 4 (Likely)
Rationale: Encryption of data in transit is crucial in distributed systems like DSS, but misconfigurations in TLSare common. Attackers may intercept unencrypted or improperly encrypted communications based on T1071 (Application Layer Protocol) and T1040 (Network Sniffing). This is a likely concern, especially in multi-organization environments where communication protocols must be more consistently applied.

Impact: 5 (Catastrophic)
Rationale: Failure to encrypt data in transit opens the system to man-in-the-middle attacks, allowing attackers to intercept or manipulate data as it is transmitted between nodes. This could result in severe data breaches, unauthorized access, or critical data manipulation. ISO 27001 controls like A.5.19 (Secure Communication) stress the importance of ensuring that data in transit is always encrypted, making the impact catastrophic.

Risk Score=4×5=20

Access Control and Authorization

Concern 1: Role Based Access Control

Likelihood: 3 (Possible)
Rationale: RBAC is widely adopted in secure environments, but improper configurations or the absence of strict policies could lead to excessive privileges. Based on T1078 (Valid Accounts) and T1087 (Account Discovery), attackers often target misconfigured RBAC systems to gain unauthorized access. While well-managed in many organizations, RBAC can be improperly configured in multi-tenant systems like DSS, making this concern moderately likely.

Impact: 4 (Major)
Rationale: If RBAC is improperly implemented, users or systems may gain unauthorized access to sensitive data or services. This could result in data breaches, privilege escalation, or unauthorized access to critical systems. ISO 27001 controls like A.5.9 (Access Rights Management) and A.5.10 (Access Control) emphasize the importance of RBAC to ensure proper segregation of duties, leading to a major impact.

Risk Score=3×4=12

Concern 2: Granular Permissions and Least Privilege

Likelihood: 4 (Likely)
Rationale: Enforcing least privilege is challenging, especially in environments with many users or services. Based on T1055 (Process Injection) and T1078 (Valid Accounts), adversaries often exploit excessive privileges to escalate their control over a system. Organizations likely struggle to implement granular permissions properly, making this a frequent issue.

Impact: 4 (Major)
Rationale: Failure to enforce least privilege can result in unauthorized users gaining excessive access to critical systems, increasing the likelihood of privilege escalation or lateral movement across systems. This poses a significant risk to the overall security of the DSS, as sensitive data or critical services could be compromised. ISO 27001 controls like A.5.10 (Access Control) emphasize restricting access to the minimum necessary for operational tasks.

Risk Score=4×4=16

Compliance and Data Privacy

Concern 1: Regulatory Compliance

Likelihood: 4 (Likely)
Rationale: Organizations operating under regulatory frameworks such as GDPR, HIPAA, or CCPA must comply with strict privacy and data protection requirements. Non-compliance is likely in complex environments like DSS, where multiple organizations share data, each with potentially different policies or interpretations of compliance standards. Based on T1485 (Data Destruction) or T1078 (Valid Accounts), attackers may exploit poor compliance practices to cause significant damage. Non-compliance is common in multi-tenant systems, making this issue likely.

Impact: 5 (Catastrophic)
Rationale: Failure to meet regulatory compliance can result in severe consequences, including heavy fines, loss of business, reputational damage, and legal action. Unauthorized exposure of personal data or failure to adhere to regulatory mandates can severely impact the organization and its stakeholders. ISO 27001 controls like A.5.1 (Information Security Policies) and A.5.29 (Compliance with Legal and Contractual Requirements) stress the importance of meeting regulatory obligations, making the potential impact catastrophic.

Risk Score=4×5=20

Concern 2: Data Segregation and Audits

Likelihood: 3 (Possible)
Rationale: While data segregation between organizations is typically enforced, technical complexities in shared infrastructures can sometimes lead to gaps. Regular audits may not catch every issue, especially when systems evolve rapidly. Based on T1484 (Domain Trust Discovery) and T1531 (Account Access Removal), attackers may exploit poor segregation practices to access unauthorized data. The possibility of misconfiguration or oversight is moderately likely.

Impact: 4 (Major)

Rationale: Poor data segregation or lack of regular audits can lead to unauthorized access to sensitive or regulated data, resulting in data breaches, privacy violations, and financial penalties. Effective data segregation is critical to ensuring the confidentiality of each organization's data in a multi-tenant system. ISO 27001 controls like A.5.30 (Event Monitoring) and A.5.15 (Configuration Management) emphasize regular audits and proper data segregation to prevent breaches, making this concern a major one.

Risk Score=3×4=12

Monitoring and Logging

Concern 1: Centralized Monitoring

Likelihood: 4 (Likely)
Rationale: Monitoring multiple nodes across organizations can be difficult, and failures in centralized monitoring systems are common due to technical challenges and coordination issues. Based on T1020 (Automated Exfiltration) and T1071 (Application Layer Protocol), attackers may leverage inadequate monitoring to exfiltrate data or communicate covertly within systems. Given the complexity of DSS, centralized monitoring failures are likely, especially if the monitoring system is not consistently maintained.

Impact: 4 (Major)
Rationale: Without proper centralized monitoring, security incidents could go undetected, leading to prolonged data exfiltration, unauthorized access, or operational disruptions. The inability to promptly monitor and respond to anomalies could result in significant financial and reputational damage. ISO 27001 controls like A.5.31 (Secure Logging and Monitoring) underscore the importance of centralized monitoring, making the impact of this concern major.

Risk Score=4×4=16

Concern 2: Anomaly Detection

Likelihood: 3 (Possible)
Rationale: While anomaly detection tools are commonly deployed in security environments, they can be misconfigured or may not be fine-tuned to detect all relevant threats. Based on T1057 (Process Discovery) and T1049 (System Network Connections Discovery), adversaries may evade detection by utilizing legitimate system functions or blending into normal operations. Misconfigurations or lack of proactive updates to detection tools make this a possible issue.

Impact: 5 (Catastrophic)
Rationale: Failure to detect anomalies promptly can result in undetected breaches or malicious activities continuing for extended periods. This could lead to significant data loss, system corruption, or a complete compromise of the DSS environment. The catastrophic nature of a prolonged undetected breach, combined with the emphasis on timely detection from ISO 27001 controls such as A.5.31 (Secure Logging and Monitoring), makes this a high-impact concern.

Risk Score=3×5=15

Incident Response and Recovery

Concern 1: Incident Coordination

Likelihood: 3 (Possible)
Rationale: In a distributed, multi-organizational system like DSS, coordination between entities during an incident can be challenging. Organizations may have different incident response processes, and misalignment could occur. Based on T1078 (Valid Accounts) and T1105 (Ingress Tool Transfer), attackers might exploit a lack of coordinated response to maintain access or escalate an attack. This makes incident coordination a moderately likely issue, as different organizations may only sometimes collaborate efficiently during incidents.

Impact: 5 (Catastrophic)
Rationale: Poor incident coordination could result in a slow or ineffective response, potentially allowing an attack to spread across multiple nodes and organizations. The impact is catastrophic, as uncoordinated responses could lead to extensive data breaches, operational downtime, or regulatory non-compliance. ISO 27001 controls like A.5.32 (Information Security Incident Management) emphasize the importance of coordinated incident response, making this a high-impact concern.

Risk Score=3×5=15

Concern 2: Disaster Recovery

Likelihood: 2 (Unlikely)
Rationale: While disaster recovery plans are typically in place for most organizations, ensuring recovery strategies are aligned across multiple entities can be challenging. Based on T1490 (Inhibit System Recovery) and T1070 (Indicator Removal on Host), attackers may attempt to inhibit disaster recovery procedures. However, most organizations regularly test and implement disaster recovery plans, making failure to recover unlikely, although possible.

Impact: 5 (Catastrophic)
Rationale: A failed disaster recovery process could result in prolonged downtime, permanent data loss, or failure to restore critical services. This could have severe operational, financial, and reputational consequences. ISO 27001 controls like A.5.27 (Information Security Continuity) emphasize disaster recovery, making the potential impact catastrophic.
Risk Score=2×5=10

Configuration Management

Concern 1: Consistent Configurations

Likelihood: 4 (Likely)
Rationale: Misconfigurations are a common security concern, especially in distributed environments where different teams or organizations manage nodes. Based on T1078 (Valid Accounts) and T1068 (Exploitation for Privilege Escalation), adversaries often exploit misconfigured systems to gain access or escalate privileges. Given the complexity of maintaining consistency across CRDB nodes and the potential for human error, the likelihood of inconsistent configurations is relatively high.

Impact: 4 (Major)
Rationale: Inconsistent configurations can result in security vulnerabilities, such as unauthorized access, data breaches, or operational failures. The risk is significant in environments like DSS, where multiple organizations rely on each other's configurations to maintain security. ISO 27001 controls like A.5.15 (Configuration Management) ensure that system configurations are properly managed, and failures in this area can have major consequences.

Risk Score=4×4=16

Concern 2: Secure Network Segmentation

Likelihood: 3 (Possible)
Rationale: Network segmentation is a key defense mechanism for identifying critical systems. However, failures in segmentation, particularly in shared environments like DSS, are possible due to misconfigurations or changes in network architecture. Based on T1071 (Application Layer Protocol) and T1078 (Valid Accounts), attackers often exploit weak network segmentation to move laterally through systems. While segmentation is typically enforced, gaps can exist, making this a possible issue.

Impact: 5 (Catastrophic)
Rationale: If network segmentation fails, attackers may gain access to critical systems and move laterally across the network, potentially affecting multiple CRDB nodes and compromising the entire DSS environment. This could result in significant data loss, operational disruption, or security breaches. ISO 27001 controls like A.5.34 (Segregation of Networks) stress the importance of secure network segmentation, making the impact of this concern catastrophic.

Risk Score=3×5=15

Automation and Scripting

Concern 1: Automated Deployment and Maintenance

Likelihood: 3 (Possible)
Rationale: Automated deployment and maintenance scripts are widely used in cloud-native systems like DSS. However, errors in scripting or incorrect automation configurations can lead to security misconfigurations or failures. Based on T1059 (Command-Line Interface) and T1064 (Scripting), attackers can exploit misconfigured or vulnerable scripts to execute unauthorized actions. Given the widespread use of automation, this concern is possible but can be mitigated through script testing and validation.
Impact: 4 (Major)
Rationale: A failure in automated deployment or maintenance scripts can result in major system misconfigurations, leading to security gaps, service downtime, or data loss. If adversaries exploit vulnerable scripts, they may gain unauthorized access or manipulate system settings. ISO 27001 controls like A.5.15 (Configuration Management) and A.5.18 (Cryptographic Key Management) emphasize secure and consistent automation processes, making the potential impact major.

Risk Score=3×4=12

Concern 2: Consistency Checks

Likelihood: 3 (Possible)
Rationale: While automated consistency checks can reduce human error, they can fail due to misconfigurations, out-of-date scripts, or overlooked edge cases. Based on T1057 (Process Discovery) and T1485 (Data Destruction), attackers may exploit poorly implemented checks to introduce inconsistencies or disrupt system integrity. While such issues are possible, organizations typically have mechanisms in place to catch major inconsistencies, making the likelihood moderate.

Impact: 4 (Major)
Rationale: If consistency checks fail, critical security configurations may be missed, leading to potential security vulnerabilities, system instability, or data corruption. This could result in data breaches, service

outages, or degraded performance. ISO 27001 controls like A.5.15 (Configuration Management) and A.5.29 (Secure Monitoring and Logging) underline the importance of automated checks, making the impact major.

Risk Score=3×4=12

Enhanced Documentation and Support

Concern 1: Beginner-Friendly Guides

Likelihood: 3 (Possible)
Rationale: The lack of beginner-friendly guides or proper documentation can lead to misconfigurations or improper setups, especially for new users or organizations with less experience in deploying distributed systems like DSS. Based on T1078 (Valid Accounts) and T1190 (Exploit Public-Facing Application), adversaries can exploit weak deployments due to incorrect setups. While comprehensive documentation is often prioritized, gaps still exist, making the likelihood possible.

Impact: 4 (Major)
Rationale: Misconfigurations or improper setups due to insufficient documentation can result in major security vulnerabilities, including improper access controls, lack of encryption, or insecure communication between CRDB nodes. This could lead to service downtime, data breaches, or unauthorized access. ISO 27001 controls like A.5.15 (Configuration Management) and A.5.19 (Secure Communication) highlight the need for clear and comprehensive documentation, making the potential impact major.

Risk Score=3×4=12

Concern 2: Troubleshooting and Examples

Likelihood: 3 (Possible)
Rationale: The absence of detailed troubleshooting guides or examples can result in users struggling to resolve common or complex issues during deployment and operations. Based on T1071 (Application Layer Protocol) and T1059 (Command-Line Interface), misconfigurations or mistakes during troubleshooting could be exploited by adversaries to gain unauthorized access or disrupt operations. This is possible, especially for less experienced users, making the likelihood moderate.

Impact: 3 (Moderate)
Rationale: Insufficient troubleshooting documentation can lead to moderate issues such as prolonged downtime, misconfigurations, or missed security patches. While these issues are not catastrophic, they could degrade system performance or introduce security vulnerabilities. ISO 27001 controls like A.5.29 (Secure Monitoring and Logging) and A.5.32 (Information Security Incident Management) emphasize proper documentation to resolve issues, making the impact moderate.

Risk Score=3×3=9

Ordered Risks and Risk Summary

1. Encryption in Transit (Risk Score: 20)
    a. Failure to encrypt data in transit could lead to catastrophic security breaches through man-in-the-middle attacks.
2. Regulatory Compliance (Risk Score: 20)
    a. Non-compliance with regulations such as GDPR, HIPAA, or CCPA can result in severe legal, financial, and reputational consequences.
3. Consistent Configurations (Risk Score: 16)
    a. Misconfigurations can lead to significant security vulnerabilities and operational issues in a distributed, multi-organization DSS setup.
4. Granular Permissions and Least Privilege (Risk Score: 16)
    a. Failure to enforce least privilege could allow unauthorized users to access sensitive systems and data, leading to major security risks.
5. Centralized Monitoring (Risk Score: 16)
    a. A lack of proper centralized monitoring could allow security incidents to go undetected, resulting in major data breaches or operational disruptions.
6. Incorrect or Incomplete Certificate Configurations (Risk Score: 16)
    a. Improper certificate configurations could lead to security vulnerabilities and data synchronization issues in the DSS.
7. Dynamic Trust Management (Risk Score: 15)
    a. Failing to update trust stores dynamically could result in unauthorized access or breakdowns in secure communication, leading to catastrophic breaches.
8. Incident Coordination (Risk Score: 15)
    a. Uncoordinated incident response across multiple organizations could allow an attack to spread, resulting in catastrophic damage.
9. Secure Network Segmentation (Risk Score: 15)
    a. Poor network segmentation could enable lateral movement across the system, resulting in catastrophic security breaches.
10. Anomaly Detection (Risk Score: 15)
    a. Failure in anomaly detection could result in prolonged undetected breaches, causing catastrophic damage to the system and its data.
11. Certificate Rotation and Revocation (Risk Score: 15)
    a. Outdated or compromised certificates could allow unauthorized access, leading to catastrophic security risks.
12. Automated Deployment and Maintenance (Risk Score: 12)
    a. Failures in automation could lead to system misconfigurations and security gaps, posing a major risk to system security.
13. Consistency Checks (Risk Score: 12)
    a. Poorly implemented consistency checks can result in misconfigurations, security vulnerabilities, and operational instability.
14. Beginner-Friendly Guides (Risk Score: 12)
    a. Insufficient documentation can lead to misconfigurations and security vulnerabilities for new users or organizations unfamiliar with DSS deployment.
15. Failure to Update CA Certificates (Risk Score: 12)
    a. If CA certificates are not updated, it could cause communication breakdowns between nodes, leading to major security risks.
16. RBAC (Risk Score: 12)
    a. Misconfigured RBAC can allow unauthorized access to sensitive systems and data, causing major security issues.

17. Data Segregation and Audits (Risk Score: 12)
    a. Poor data segregation or lack of regular audits can result in unauthorized access to sensitive data, posing a major risk.
18. Disaster Recovery (Risk Score: 10)
    a. A failed disaster recovery process could result in prolonged downtime or data loss, causing catastrophic damage.
19. Troubleshooting and Examples (Risk Score: 9)
    a. Lack of detailed troubleshooting guides may lead to operational delays and moderate security risks, but the impact is not as severe.

Overall, two out of the top three risks come from misconfigurations, as a mistake by a single DSS entity could compromise the entire cluster. The DSS uses TLS, but automation of TLS deployment, and critical security configurations during deployment should be a priority. If automation is not possible, identifying key security configurations and providing a guide to their recommended settings could significantly reduce risk. Consider that DSS security essentially relies on any DSS entity's 'worst' configuration. Providing reasonable security guidance and expanding automated configuration/auditing of key security settings should significantly reduce risk. Additionally, several other high-ranking concerns deal with coordination problems. Creating clear guidance on expectations for incident response and certificate revocation could be a low-hanging fruit in reducing risk within the DSS.

Risks and Recommendations Table

Risk/Finding	Risk Score	Recommendation
Encryption in Transit	20	Enforce TLS for All Communications: Ensure that TLS is enabled and enforced for all inter-node communications in the CRDB cluster, including both internal communications between nodes and any external communication channels. Use Secure TLS Versions: Configure all nodes to use the latest, secure versions of TLS (e.g., TLS 1.2 or TLS 1.3). Disable older, insecure versions such as TLS 1.0 and TLS 1.1 to protect against vulnerabilities. Automated Encryption Audits: Implement automated tools that regularly audit TLS configurations on all nodes to ensure that encryption in transit is consistently enforced and properly configured across the DSS instance.
Regulatory Compliance	20	Standardize Compliance Practices: Implement centralized compliance guidelines across all organizations involved in the DSS to ensure uniformity in meeting regulatory requirements such as GDPR, HIPAA, and CCPA. Establish consistent data handling, retention, encryption, and breach notification policies. Compliance Automation Tools: Automate compliance monitoring tools to ensure that all nodes in the CRDB cluster adhere to the same regulatory requirements. These tools should automatically check each organization's compliance status and flag any discrepancies for timely remediation. Documentation and Audits: Provide detailed documentation and templates to help organizations demonstrate compliance with relevant regulations. Documentation should include audit trails, consent management practices, data handling procedures, and proof of compliance with regulatory requirements. Centralized Privacy Impact Assessments (PIAs): Require all organizations within the DSS to conduct privacy impact assessments regularly to ensure compliance with data privacy regulations. Each organization should identify potential risks to data privacy and take corrective measures to address them.

Risk/Finding	Risk Score	Recommendation
Consistent Configurations	16	Standardize Configuration Procedures: Develop standardized configuration templates or scripts that ensure consistency across all CRDB nodes in the DSS pool. Use configuration management tools like Ansible, Puppet, or Chef to automate and enforce configuration consistency, reducing the likelihood of misconfigurations. Automate Configuration Audits: Implement automated configuration audits across the CRDB nodes to ensure all nodes adhere to the established baseline configuration. Audits can help detect deviations from expected configurations and flag potential security risks early on. Document and Review Configuration Settings: Ensure configuration settings for certificates, network policies, security protocols, and service parameters are clearly documented. Regularly review and update these settings to ensure they remain secure and appropriate for the evolving DSS environment. Use Version Control for Configuration Files: Implement version control for configuration files across all nodes to track changes and consistently apply updates. Tools like Git can help maintain a consistent configuration history and ensure rollback capability in case of issues.
Granular Permissions and Least Privilege	16	Enforce the Principle of Least Privilege: Ensure that each user and service has only the minimum permissions necessary to perform their function. Least privilege can be achieved by carefully defining roles and using IAM tools to enforce granular permissions. Segment Data Access by Sensitivity: Classify data based on sensitivity and restrict access to critical data sets. Ensure that users and services that do not require access to sensitive data are denied by default. Regular Audits of Access Controls: Conduct regular audits of access permissions to verify that they are appropriate and have become manageable permissive. Implement automated checks where possible to detect and alert administrators to potential misconfigurations.
Centralized Monitoring	16	Centralized Logging and Monitoring Tools: Implement a centralized system to aggregate and visualize logs and metrics from all nodes in the cluster. Tools like Grafana and Prometheus should be used to monitor system performance and availability . Configure Alerts: Set up automated alerts for critical events such as node failures, high latency, or security breaches . Real-Time Dashboards: Implement dashboards to provide real-time insights into cluster performance and system health .

Risk/Finding	Risk Score	Recommendation
Incorrect or Incomplete Certificate Configurations	16	Automated Certificate Management: Automate the process of generating, distributing, and configuring certificates for each node. Use a centralized or distributed tool to manage certificates and ensure consistency across all organizations. Strong TLS Configurations: Ensure that TLS configurations use the latest versions (TLS 1.2 or 1.3) and that only robust and secure cipher suites are enabled. Periodically review encryption settings and perform automated checks to prevent misconfigurations.
Dynamic Trust Management	15	Automated Trust Store Updates: Implement an automated process for dynamically updating trust stores on all nodes when new CA certificates are added or revoked. Consider using configuration management tools to ensure that all nodes receive the updated combined CA certificate file. Certificate Revocation Mechanism: Develop a shared mechanism to propagate certificate revocations quickly across all nodes in the cluster. The mechanism should include regular distribution of CRLsor use of OCSP responses to prevent compromised certificates from being used. Monitoring and Alerts: Set up monitoring tools to track the state of CA certificates across all nodes. Alerts should be configured to notify administrators when certificates are about to expire, when new CA certificates need to be added, or when revocations have occurred.
Incident Coordination	15	Develop a Unified Incident Response Plan: Create a joint incident response procedure that clearly defines the roles, responsibilities, and communication protocols for each organization involved in the DSS. The plan should ensure that all organizations are aligned on coordinating efforts during a security event. When a DSS provider signs up, they should be provided the plan to ensure that they can reasonably respond to incidents. Establish Secure Communication Channels: Implement secure and reliable communication channels for incident response coordination, ensuring that all organizations can quickly share information during an incident. This could include encrypted messaging platforms or dedicated incident response portals. Conduct Regular Incident Drills: Perform joint incident response drills and simulations to test and refine the response process. Drills will help ensure that all organizations are familiar with their roles and can coordinate effectively during real incidents.

Risk/Finding	Risk Score	Recommendation
Secure Network Segmentation	15	Implement Network Segmentation Policies: Use Virtual Private Networks (VPNs), Virtual Private Clouds (VPCs), and subnets to isolate CRDB nodes from the rest of the organization's networks. Ensure each organization segments its CRDB infrastructure from other parts of its network to prevent unauthorized access or lateral movement. Deploy Firewalls and Access Control Lists (ACLs): Configure firewalls and ACLs to restrict access to CRDB nodes. Only allow traffic from authorized IP ranges or services. To segment and secure communication between nodes, Kubernetes Network Policies, cloud-based firewalls, or dedicated firewalls can be used. Encrypt Network Traffic: Ensure all network traffic between CRDB nodes is encrypted using TLS or other strong encryption mechanisms. Encryption is essential to prevent attackers from intercepting sensitive information or manipulating inter-node communication. Regularly Test Segmentation Effectiveness: Conduct penetration and network audits periodically to ensure that segmentation policies effectively isolate CRDB nodes from unauthorized access. Testing ensures that segmentation strategies remain effective as the network evolves. Isolate Critical Components: Ensure that the most sensitive parts of the CRDB infrastructure, such as the management interfaces and administrative tools, are isolated on separate networks from the production environment to reduce the risk of privilege escalation or compromise.
Anomaly Detection	15	Anomaly Detection Tools: Use anomaly detection tools and machine learning models to identify unusual patterns of activity that may indicate security incidents or system anomalies . Security Information and Event Management (SIEM): Implement a SIEM system to correlate events and detect potential security incidents across the CRDB cluster . Incident Response Procedures: Develop and test incident response procedures to ensure prompt and effective responses to anomalies. Ensure coordination across all organizations in the DSS pool .

Risk/Finding	Risk Score	Recommendation
Certificate Rotation and Revocation	15	Automated Certificate Rotation: Develop an automated system for rotating certificates regularly. This system should coordinate between all participating organizations to ensure that all nodes trust new certificates before the old ones expire. Revocation Process: Implement a shared revocation mechanism to propagate certificate revocation information quickly across all nodes. The revocation mechanism could involve regularly distributing Certificate Revocation Lists (CRLs) or utilizing the Online Certificate Status Protocol (OCSP) to ensure compromised certificates are promptly invalidated. Monitoring for Certificate Validity: Continuously monitor certificates across all nodes to ensure they are valid and trusted. Set up alerts for expiring or revoked certificates to prompt timely action.
Automated Deployment and Maintenance	12	Develop Automated Deployment Scripts: Create scripts to automate the deployment of CRDB nodes, configuration settings, and certificate management. Automation will help ensure that each node is deployed with the correct configurations and that certificates are handled securely and consistently across the DSS environment. Automate Certificate Management: Use automated scripts to generate, distribute, and rotate certificates across nodes. This ensures that nodes can securely communicate with each other and that certificate updates are handled seamlessly without requiring manual intervention. Schedule Regular Maintenance Tasks: Automate routine maintenance tasks, such as security patching, performance tuning, and service restarts, to reduce the need for manual intervention and ensure systems remain secure and up-to-date.

Risk/Finding	Risk Score	Recommendation
Consistency Checks	12	Implement Automated Configuration Audits: Develop automated tools to regularly check that all nodes have consistent configurations, certificates, and security settings. Automation will help ensure that no node deviates from the standard configuration, reducing the risk of security vulnerabilities. Monitor Certificate Validity and Expiration: Automate the monitoring of certificate validity and expiration to ensure that all certificates are valid and that nodes are not using expired or revoked certificates. Automated monitoring reduces the risk of communication failures or security incidents due to invalid certificates. Run Regular Consistency Reports: Generate regular reports on the consistency of configurations, certificates, and network settings across all nodes. The reports allow for continuous monitoring of configuration drift and provide insights into areas where further adjustments may be needed. Alert on Configuration Deviations: Set up automated alerts for any deviations from the expected configuration or inconsistencies in node settings. This will allow for quick remediation of issues before they affect the system's overall security and performance.
Beginner-Friendly Guides	12	Create Beginner-Friendly Guides: Develop beginner-friendly, step-by-step guides with screenshots and explanations to help new organizations or users get up and running quickly with the DSS. This should include quick start sections, explanations of key concepts, and best practices. Include a section on best practices for maintaining a secure and efficient DSS deployment. This covers regular updates, security patches, performance tuning, and backup strategies. Provide guidelines for scaling the CRDB cluster, including adding or removing nodes, balancing load, and optimizing performance. Considerations for geographic distribution and network latency should also be included. Provide guidance and best practices for tuning the performance of the CRDB cluster and DSS instances, such as database optimization, query tuning, and resource management. Expand Introductory Documentation: Break the deployment process into smaller, manageable steps, catering to users with varying levels of technical expertise. Include Examples and Use Cases: Add use cases that cover common and complex deployment scenarios, such as multi-region deployments, to provide users with more comprehensive guidance.

Risk/Finding	Risk Score	Recommendation
Failure to Update CA Certificates	12	Centralized or Federated CA Management: Establish a more structured process for sharing CA certificates between organizations. The process could include using a centralized or federated service to exchange CA certificates securely and ensure that all nodes receive the latest CA certificates promptly. Automated Trust Updates: Implement automated scripts to concatenate and distribute the CA certificates to all nodes across the DSS instance. Configuration management tools (e.g., Ansible, Puppet) could be used to ensure that all trust stores are updated consistently.
RBAC	12	Implement Centralized Identity and Access Management (IAM): Use a centralized IAM solution that integrates with CRDB to enforce RBAC consistently across all nodes. RBAC will ensure that access policies are uniformly applied regardless of which organization manages the node. Automate RBAC Configuration: Automate the configuration and management of RBAC policies using tools like Ansible or Puppet. Automation will help prevent misconfigurations and ensure all nodes adhere to the same access control policies. Periodic Role Review: Regularly review roles and permissions to ensure that access rights are appropriate for each user or service's responsibilities. Reviews will help prevent privilege creep over time.
Data Segregation and Audits	12	Implement Strict Data Segregation Policies: Enforce strict data segregation rules to ensure that data from one organization is not accessible by another without explicit permission. Implement RBAC and multi-tenant isolation mechanisms to prevent unauthorized data access. Regular Compliance Audits: Conduct regular audits of data segregation policies, encryption configurations, and access controls across all organizations involved in the DSS. Automate audit tools are used to ensure consistent monitoring and to identify any violations or potential risks early. Granular Access Control Enforcement: Ensure that granular access control mechanisms are implemented at each node to prevent unauthorized access to sensitive data. Data should be classified based on sensitivity, and access should be restricted based on the principle of least privilege. Encryption of Segregated Data: Ensure that segregated data is encrypted at rest and in transit, particularly for sensitive or personal information. Regularly audit encryption mechanisms to ensure they meet regulatory standards and provide adequate protection against unauthorized access.

Risk/Finding	Risk Score	Recommendation
		Comprehensive Audit Trails and Logging: Implement audit trails and logging for all data access and modification activities, ensuring transparency and accountability. Audit trails should be regularly reviewed to detect unauthorized data access or modifications, supporting ongoing compliance efforts.
Disaster Recovery	10	Create Consistent Disaster Recovery Plans: Ensure that each organization in the DSS pool develops a disaster recovery plan that aligns with the overall DSS strategy. Each plan should include regular backups, defined recovery time objectives (RTOs), and recovery point objectives (RPOs) for critical data and systems. Perform Regular Backups and Test Restorations: Mandatory regular, automated backups of critical data across all organizations, ensuring that they are encrypted and stored securely. Periodically test the restoration of these backups to verify that they can be successfully recovered during a disaster. Develop a Shared Recovery Strategy: Establish a shared recovery strategy to ensure that all organizations can restore services in a coordinated manner following a disaster. This may include mutual assistance agreements, shared recovery sites, or centralized disaster recovery management. Monitor and Review Disaster Recovery Plans: Periodically review disaster recovery strategies across the DSS. Perform audits to ensure that plans are up to date and effective, and that backup data is secure and accessible when needed.

Risk/Finding	Risk Score	Recommendation
Troubleshooting and Examples	9	Expand the Troubleshooting Section: Provide a detailed troubleshooting guide covering various potential issues, from network failures to certificate management problems, and their resolutions. Offer Detailed Examples: Include more detailed deployment examples for complex scenarios, such as multi-organization setups, scaling, and integration with other systems. These examples should be tied to specific use cases that reflect common challenges in DSS deployments. Provide Logs and Diagnostics Commands: Include log examples, diagnostic commands, and possible resolutions to help users quickly identify and fix common problems.

Final Thoughts and Future Work

The analysis and findings in this document illustrate some of the risks associated with a multi-organizational, decentralized system like DSS. However, the key takeaway is that maintaining a secure DSS environment is not just a task but a responsibility that requires an active and ongoing effort from all participating entities, including the DSS development team, which plays a crucial role in this process.
Next Steps

1. Enforce Consistent Security Practices Across All Entities: A fundamental lesson from this analysis is the importance of standardized security measures across all entities in the DSS pool. Consistent security means ensuring that every organization adheres to the same standards for patch management, certificate rotation, incident response, and access control. Inconsistent practices can create security gaps, leading to vulnerabilities that compromise not just one organization but the entire DSS ecosystem. Establish and regularly review compliance with shared security standards and audit them across organizations.
2. Prioritize Risk Areas Identified in the Report: The report identifies several high-risk areas, including encryption in transit, regulatory compliance, and consistent configurations. Addressing these should be a top priority. Organizations need to ensure data is encrypted in transit between nodes, remain compliant with any relevant regulations (such as GDPR and HIPAA), and enforce consistent configurations across all participating entities to avoid misconfigurations and security loopholes. Focus immediate attention on the highest-risk areas by implementing specific security controls and conducting vulnerability assessments.
3. Strengthen Incident Response and Coordination: One key concern in a multi-entity environment like DSS is the coordination of incident response across different organizations. A swift, well-coordinated incident response plan can significantly reduce the damage caused by malicious actors. Implement and test a coordinated incident response plan that includes communication protocols, response timelines, and recovery strategies between all entities in the DSS pool.
4. Ensure Regular Certificate Management and Updates: The risk of outdated or poorly managed certificates is a significant vulnerability. Regular certificate rotation, revocation, and renewal are essential to secure communications between DSS nodes. An expired certificate can disrupt secure communication and expose the system to attacks. Automate certificate management processes

where possible and establish an explicit schedule for certificate updates, with shared accountability across entities.

5. Invest in Enhanced Monitoring and Logging: Continuous monitoring and logging are crucial for detecting and responding to anomalies or potential security incidents. Implementing a centralized, standardized monitoring system across the DSS pool can help identify threats early before they escalate. Ensure robust logging practices and centralized monitoring across all nodes to capture real-time data that can be used to detect potential security threats.
6. Foster a Culture of Proactive Security: Beyond technical controls, a proactive security mindset must be cultivated across all participating organizations. Security is not a one-time activity but a continuous process that evolves with emerging threats and technological advancements. Encourage regular training, awareness, and communication about security protocols and updates, ensuring that all stakeholders remain vigilant and engaged in maintaining DSS security.

Securing the DSS is a collective effort. Each entity involved in the DSS pool must actively follow best practices and address identified risks. By focusing on the actions outlined in this report — enforcing consistent security measures, prioritizing key risks, coordinating incident response, ensuring regular certificate updates, investing in monitoring, and fostering a culture of security — the DSS can be safeguarded against potential threats and vulnerabilities. It's not just about responding to security risks; it's about staying ahead of them.

For future work, the DSS team should perform a threat modeling exercise using a reference like MITRE ATT&CK to help better understand how malicious actors could attack various parts of the DSS architecture. Consider that compromising any DSS entities or a malicious DSS entity could likely harm the system. Use the modeling exercise to help inform security guidelines to apply to DSS participants. It would also be a good idea to perform a full-fledged penetration test of an operational DSS / UTM / USS system (configured as securely as possible by the DSS development team), in order to better evaluate the technical controls that have been implemented.

Appendix A

Five questions

How is the approach to authorizing CRDB with each other affected by the deployment of a single CRDB across organization boundaries?

Key Capabilities

1. Mutual TLS Encryption - Each node in the CockroachDB cluster uses mTLS for secure communication, which requires generating and distributing certificates to each node. Each organization may have its own CA, but nodes need to trust certificates issued by other organizations' CAs.+
2. Certificate Management - Nodes need the CA certificates of all participating organizations to validate incoming mTLS connections. Each node has its certificate signed by its organization's CA.
3. Cluster Initialization and Node Joining - During initialization, the cluster's CA certificate is distributed and used to authenticate nodes joining the cluster. Nodes need to dynamically trust certificates from other organizations, requiring a mechanism to share and update CA certificates across the cluster.

Current Understanding of DSS Implementation of Capabilities

1. Mutual TLS Encryption - Each organization generates its certificates using its internal CA, including a CA certificate, the root certificate used to sign node certificates, and the certificates for each CockroachDB node. The organization's CA signs these certificates.
2. Certificate Management - To enable mutual trust, organizations must share their CA certificates, which is done by concatenating CA certificates and combining all participating organizations' CA certificates into a single file that is distributed to all nodes. Trust stores are updated, ensuring each node's trust store is updated with the combined CA certificates file. Nodes are configured with the necessary certificates and secrets using Helm charts, or Kubernetes manifests. The values.yaml and other configuration files are updated to include paths to these certificates.
3. Cluster Initialization and Node Joining - New nodes joining the cluster must be configured to trust all CAs using the combined CA certificates file. New nodes present their node certificates signed by their organization's CA. New nodes discover other nodes using the Kubernetes services or other discovery mechanisms.

Security Challenges and Potential Solutions

1. Mutual Trust and Certificate Management- Establishing and maintaining mutual trust between nodes from different organizations.
    a. Establishing Trust: Establishing mutual trust between CRDB nodes from different organizations requires sharing and managing CA certificates. Each organization must generate and distribute its CA certificate, and nodes must be configured to trust the CA certificates of all participating organizations.
        1. Potential Solution: Independent repository or service for distributing and managing CA certificates. It might go against the spirit of DSS.
    b. Certificate Rotation: Regularly rotating certificates is necessary to maintain security, but certificate rotation must be coordinated to ensure that all nodes trust new certificates.
        1. Potential Solution: Automate the process of certificate rotation and distribution using scripts or tools.

    c. Certificate Revocation: Implementing an effective certificate revocation process is critical in compromise but requires timely communication and updates across all nodes. Revocation lists must be shared and updated promptly to revoke compromised certificates.
        1. Potential Solution: Implement a shared revocation mechanism to propagate certificate revocations quickly.
2. Data Encryption - Ensuring data is encrypted at rest and in transit.
    a. Encryption at Rest: Ensuring that data stored on CRDB nodes is encrypted at rest is crucial to protect against unauthorized access, especially if physical security varies across organizations.
        1. Potential Solution: Determine data-at-rest encryption needs and configure CRDB to use encryption for data-at-rest (if applicable).
    b. Encryption in Transit: To prevent eavesdropping and tampering, all inter-node communication must be encrypted using TLS. Ensuring that all nodes are properly configured for TLS is essential.
        1. Potential Solution: Use automated tools to check and enforce encryption settings.
    c. Encryption Configuration Across DSS: Ensure that TLS is enabled and properly configured for all inter-node communication. Regularly review and update encryption configurations to adhere to best practices.
        1. Potential Solution: Regularly audit encryption configurations to ensure compliance with security policies.
3. Access Control and Authorization - Implementing and enforcing access control policies across multiple organizations.
    a. RBAC: Implementing RBAC to ensure that only authorized personnel and services have access to specific data and functions within the CRDB cluster. E.g., The admin UI for CockroachDB
        1. Potential Solution: Consider using centralized IAM solutions that integrate with CRDB to manage access control policies or a secure container hosting RBAC configs.
    b. Granular Permissions: Setting granular permissions to control access to sensitive data and operations based on the principle of least privilege.
4. Compliance and Data Privacy - Ensuring compliance with regulatory requirements and protecting data privacy.
    a. Regulatory Compliance: Organizations may be subject to regulatory requirements (e.g., HIPAA, GDPR, CCPA). Ensuring the shared CRDB deployment complies with all relevant regulations could be challenging.
        1. Potential Solution: Implement data classification and segregation policies within CRDB.
    b. Data Segregation: Properly segregating data to ensure that each organization's data is only accessible by authorized users from that organization.
    c. Data Audits: Regular audits and assessments may be needed to ensure ongoing compliance with applicable regulations.
        1. Potential Solution: Use automated compliance tools to monitor and enforce regulatory requirements.
5. Monitoring and Logging - Implementing a centralized monitoring system that can collect and analyze logs and metrics from all nodes in the cluster. Monitoring is likely needed to detect bad actors.
    a. Centralized Monitoring: Implementing a monitoring system that can collect and analyze logs and metrics from all nodes in the cluster.
        1. Potential Solution: Use centralized logging and monitoring tools to aggregate and visualize logs and metrics.

        2. Potential Solution: Configure alerts for critical events such as node failures, high latency, or security breaches.
        3. Potential Solution: Implement dashboards to provide real-time insights into cluster performance and health.
    b. Anomaly Detection: Detecting and responding to anomalies or potential security incidents promptly.
        1. Potential Solution: Use anomaly detection tools and machine learning models to identify unusual activities.
        2. Potential Solution: Implement a Security Information and Event Management (SIEM) system to correlate events and identify potential security incidents.
        3. Potential Solution: Develop and test an incident response procedure to ensure quick and effective responses to anomalies.
6. Incident Response and Recovery - Coordinating incident response efforts across multiple organizations.
    a. Incident Coordination: Coordinating incident response efforts across multiple organizations requires clear communication channels and predefined processes.
        1. Potential Solution: Develop a joint incident response procedure that includes roles, responsibilities, and communication protocols for node owners.
        2. Potential Solution: Define a set of collaboration tools and secure communication channels to facilitate coordination during an incident.
        3. Potential Solution: Conduct incident response drills to test and refine the response procedure.
    b. Disaster Recovery: Ensuring all organizations have a consistent disaster recovery plan with regular backups and tested recovery procedures.
7. Configuration Management - Consistent configurations across organizations.
    a. Node Initialization and Joining: Properly initializing new nodes and ensuring they can securely join the existing cluster requires coordinated efforts and consistent configurations across organizations.
        1. Potential Solution: Create a capability for standardizing node configurations when joining to ensure consistency and correctness.
    b. Network Segmentation: Ensuring that network segments used by CRDB nodes are secure and properly configured to prevent unauthorized access and creating secure network segments for CRDB nodes isolated from other organizational networks.
        1. Potential Solution: Use Virtual Private Clouds (VPCs) and subnets to segment CRDB nodes from other networks.
        2. Potential Solution: Implement firewall rules and network ACLs to restrict access to CRDB nodes.
        3. Potential Solution: Use VPNs or dedicated network links to enhance security and inter-organizational communication.
        4. Potential Solution: Create a standard guide for security configuration for DSS operators.

Are the CRDB configuration instructions provided in the DSS repository reasonable? Sufficient? Flawed?

Key Elements Reviewed

1. General Instructions (READMEs, /build/README.md)
    a. Strengths:
        1. Good Catalogue of Prerequisites: The README lists all the necessary tools and installation steps, ensuring that users clearly understand what is needed.
        2. Detailed Deployment Steps: The instructions cover the requirements for the entire deployment process, including creating Kubernetes clusters, setting up static IPs, configuring DNS entries, and setting up CRDB nodes.
        3. Certificate Management: Instructions for joining an existing DSS pool and sharing certificates with other DSS instances are included, which is crucial for interoperability. The use of the make-certs.py script for generating and managing certificates is documented, ensuring secure communication between nodes.
        4. Validation and Testing: The README includes steps for validating the deployment and ensuring that services are running correctly.
        5. Docker and Kubernetes Integration: The README integrates Docker image building and pushing with Kubernetes deployment, a best practice for containerized applications.
    b. Areas for Improvement
        1. Technical Complexity: The detailed steps might overwhelm users unfamiliar with Kubernetes, Docker, and CRDB. It could be a barrier based on where DSS entrants sit regarding technical acumen.
        2. Error Handling and Troubleshooting: While the README provides some troubleshooting steps, it could be expanded to cover more potential issues and their resolutions. For example, what do you do if there are network failures during deployment? There are also some cases where the instructions for commands like "`tk apply workspace/$CLUSTER_CONTEXT`" say, "Don't do this until X." What happens if a user does it before X? What is the fix?
        3. Examples and Use Cases: More examples and specific use cases could help users understand the instructions better, especially in complex scenarios like multi-region deployments or single-region deployments with a significant number of providers.
        4. Certificate Rotation and Management: The README does not explicitly cover practical guidance for certificate rotation and management, which are crucial for maintaining security over time. It also does not mention certificate revocation.
        5. Automated Scripts: While make-certs.py and apply-certs.sh are helpful, more automated scripts or tools could simplify the overall deployment process and reduce the likelihood of human error.
        6. Consistency Checks: There are no explicit instructions for ensuring consistency across multiple DSS instances, which could lead to issues in a distributed deployment.
    c. Recommendations for Improvement
        1. Enhance Documentation for Beginners: Break down the instructions into smaller, more manageable steps with clear explanations and screenshots where applicable. Create separate guides for different stages: initial setup, joining an existing cluster, and ongoing maintenance. Add a beginner-friendly guide or a quick start section to help new users get up and running quickly.

2. Expand Troubleshooting Section: Expand the troubleshooting section with more detailed scenarios, joint issues, and their resolutions. Include logs and diagnostic commands to help identify and resolve problems quickly.
3. Automate Repetitive Tasks: Provide more scripts for repetitive tasks, such as environment setup, certificate rotation, and consistency checks across instances. More automation would also improve deployment consistency. Ideally, create comprehensive automation scripts to handle the entire deployment process, including setting up clusters, configuring DNS entries, and managing certificates.
4. Include Certificate Rotation and Revocation Instructions / Policy: Add a section on rotating and managing CA certificates over time. Include steps for generating new certificates, distributing them securely, revoking them, and updating existing nodes.
5. Coordination Procedures: Define clear procedures for coordinating between organizations, including communication channels, roles and responsibilities, and incident response plans. For example, ensure all parties know the processes for certificate updates, node additions, and other critical operations and have some guidance on managing said processes.
6. Provide More Examples: Include additional examples and detailed explanations for complex scenarios, such as multi-region deployments and integrating with existing infrastructure.
7. Best Practices: Include a section on best practices for maintaining a secure and efficient DSS deployment. This covers regular updates, security patches, performance tuning, and backup strategies. Provide guidelines for scaling the CRDB cluster, including adding or removing nodes, balancing load, and optimizing performance. Considerations for geographic distribution and network latency should also be included. Provide guidance and best practices for tuning the performance of the CRDB cluster and DSS instances, such as database optimization, query tuning, and resource management.
8. Data Consistency Procedures: Define procedures for ensuring data consistency across multiple DSS instances, such as conflict resolution mechanisms, synchronization strategies, and consistency checks.
9. Compliance Documentation: If applicable, include documentation and templates to help organizations demonstrate compliance with standards like GDPR, HIPAA, etc.

2. Certificate Management Scripts (apply-certs.sh, make-certs.py, etc.)
    a. Strengths:
        1. Ensures secure communication using mTLS.
        2. Manages client and node certificates.
        3. Deletes previous secrets to prevent stale configurations.
        4. Creates Kubernetes secrets for CA certificates, client certificates, and node certificates.
    b. Areas for Improvement:
        1. Certificate Rotation and Revocation: Instructions for rotating certificates are limited, and there are no instructions for revocation. Step-by-step guidance for regularly rotating certificates and handling revocations should be included.
        2. Multi-organization CA Management: The scripts do not clearly describe how to handle CA certificates from multiple organizations. The configuration instructions suggest "Reaching out to existing operators" to get their public cert, but no transaction method is established or suggested. Similarly, the process for providing the new cert to the existing operators is unclear. Better instructions on concatenating CA certificates and distributing them to all nodes would be helpful.

3. Helm Charts (values.yaml and other templates)
    a. Strengths:
        1. Using Helm to manage deployment configurations is a robust and scalable method.
        2. Specifies CockroachDB image and TLS configurations.
        3. Provides templates for load balancers to expose CockroachDB nodes.
    b. Areas for Improvement:
        1. Node Discovery and Joining: Explicit instructions on how nodes discover each other and join the cluster need to be included. The values.yaml file should include parameters or references to service discovery mechanisms.
        2. Resilience and Fault Tolerance: Documentation on handling node failures, reboots, and network partitions would enhance reliability.
4. Deployment Scripts (core_service.sh, migrate_local_db.sh, etc.)
    a. Strengths:
        1. Scripts automate core service startup and database migration processes.
        2. Parameters for CockroachDB host, logging, and security settings are well-defined.
    b. Areas for Improvement:
        1. Script Robustness: Ensure scripts handle edge cases, such as partially failed migrations or network interruptions during service startup.
        2. Documentation: Clearer documentation within the scripts would help operators understand and modify configurations as needed.

Any concerns with the way that "TCP secured by TLS" is configured for this DSS application?

No specific concerns for now, but the TCP secured by TLS capability for the DSS relies on various security topics covered in other sections. E.g.:

1. Certificate Expiration - Ensure that all certificates have appropriate expiration dates and that there is a robust process for renewing and replacing them before they expire.
2. Key Storage - Ensure that private keys are stored securely, using hardware security modules or other secure storage mechanisms to prevent unauthorized access.
3. Certificate Distribution - Verify that certificates are distributed securely to all nodes, ensuring that no unauthorized parties can intercept or modify them during transmission.
4. TLS Version - Ensure that only secure versions of TLS (e.g., TLS 1.2 or TLS 1.3) are used. Older versions, such as TLS 1.0 or TLS 1.1, are considered insecure and should be disabled.
5. Trust Store Management - Ensure all nodes have an updated and consistent trust store containing all necessary CA certificates from participating organizations.
6. Revocation Checking - Ensure that CRLs are checked to prevent the use of revoked certificates.
7. Mutual Authentication - Ensure that mTLS is configured correctly to authenticate the client and server, preventing unauthorized access.
8. TLS Vulnerabilities - Ensure the configuration is not susceptible to known TLS vulnerabilities.
9. Configuration Management - Ensure that TLS configurations are managed centrally and consistently across all nodes to prevent misconfigurations.
10. Lack of Visibility - Ensure that TLS connections are adequately monitored and logged to detect and respond to suspicious activities.
11. Auditing - Ensure audit logs are maintained and reviewed regularly to identify potential security incidents related to TLS connections.

Are there best practices (or anti-patterns) for exchanging key information between organizations that provide DSS?

1. Establish a Secure Communication Channel
    a. Best Practice: Ensure all organizational communications are encrypted using secure protocols and strong authentication mechanisms to authenticate the entities involved.
    b. Anti-pattern: Transferring CA certificates, keys, or sensitive information without encryption (e.g., via plain email or HTTP) is a significant security risk.
2. Centralized Key Management
    a. Best Practice: Use established protocols for secure key exchange, such as Diffie-Hellman or RSA, and utilize automated tools like HashiCorp Vault, AWS KMS, or Azure Key Vault for secure key management and distribution.
    b. Anti-pattern: Relying on manual processes to distribute certificates increases the risk of human error, exposes sensitive data during transmission, and can lead to inconsistencies.
3. Certificate Management
    a. Best Practice: Each organization should use strong cryptographic algorithms to generate its own CA certificate. Secure exchange of CA certificates between organizations can be done through encrypted emails, secure file transfer protocols (SFTP), or dedicated key exchange platforms. Combine all CA certificates into a single file to create a consolidated trust store. Distribute this file securely to all nodes.
    b. Anti-pattern: Generating or storing CA certificates and private keys on insecure devices or locations.
4. Secure Distribution of CA Certificates
    a. Best Practice: Encrypt the CA certificate bundle before distribution. Use encryption tools like GPG or OpenSSL and cryptographic hashes (e.g., SHA-256) to verify the integrity of the CA certificate bundle after distribution. Distribute the CA certificate bundle using secure channels (e.g., SFTP, encrypted emails).
    b. Anti-pattern: Distributing CA certificates through untrusted or insecure channels and failing to verify the integrity of CA certificates during distribution.
5. Trust Store Management
    a. Best Practice: Automate updating trust stores on all nodes with the combined CA certificate bundle. Use configuration management tools like Ansible, Puppet, or Chef to ensure consistency. Regularly update the trust stores to include new CA certificates as organizations join the DSS instance.
    b. Anti-pattern: Manually updating trust stores on nodes, leading to inconsistencies and potential trust issues.
    c. Anti-pattern: Failing to remove or update expired or revoked certificates in the trust store.
6. Key Rotation and Revocation
    a. Best Practice: Establish a regular schedule for rotating keys and certificates. Automate this process to minimize human error and implement a mechanism for distributing and managing certificate revocation lists (CRLs). Ensure all nodes are updated with the latest CRLs to prevent the use of compromised certificates.
    b. Anti-pattern: Not automating the key and certificate rotation process can result in outdated or compromised credentials remaining in use. Manually updating trust stores on each node is error-prone and can lead to inconsistencies.
    c. Anti-pattern: Not implementing a robust certificate revocation process allows compromised certificates to remain valid. Additionally, failing to promptly revoke compromised keys or certificates and update the trust stores when compromise is detected.
7. Access Control and Auditing

a. Best Practice: Implement RBAC to control access to key management systems. Ensure that only authorized personnel can generate, distribute, or update keys and certificates. Enable detailed logging and auditing of all key management activities. Regularly review audit logs to detect and respond to unauthorized access or anomalies.
b. Anti-pattern: Granting broad access to key management systems or certificates can lead to unauthorized use or exposure. Failing to implement RBAC results in inadequate separation of duties and increased risk of insider threats.
c. Anti-pattern: Failing to log key management activities makes it difficult to detect and investigate security incidents, which can lead to undetected vulnerabilities and compliance issues.

8. Incident Response and Recovery
    a. Best Practice: Develop an incident response plan for handling key compromise or security breaches. Ensure all organizations are familiar with the plan and know their roles and responsibilities. Regularly back up all keys and certificates. Ensure that backups are encrypted and stored securely. Test recovery procedures to ensure data can be restored in the event of a compromise.
    b. Anti-pattern: Not having a formal incident response plan for handling key compromises or security breaches and not having regular backups of keys and certificates or failing to test recovery procedures.
9. Clear Documentation and Policies
    a. Best Practice: Document all key generation, distribution, rotation, and revocation procedures. Ensure all participating organizations have access to this documentation. Develop and enforce security policies that govern the management and exchange of keys and certificates. Ensure all organizations adhere to these policies.
    b. Anti-pattern: Not documenting key management procedures?

Are there new requirements (i.e., not currently documented or not currently known) related to cybersecurity that should be levied on entities that join a DSS pool?

The answer to this question depends on the intent of DSS, as there is a trade-off between flexibility and security. That being said, some common agreement on core security capabilities within the DSS, such as key distribution, key rotation, key revocation, incident response, recovery, will likely require joining entities to comply with them. In a proximate way, DSS entities may need to be aware of privacy requirements or have some ability to segregate information in certain cases. For example, if a drone is delivering medication there could be a HIPPA issue if there is information leaked. Similarly, if the DSS is intended to support sensitive government functions, there may be a need to ensure that DSS pool participants can properly secure any relevant information.

Overall, it's critical to recognize that the DSS's security relies on strict adherence to established operational procedures by all participating entities. While existing cybersecurity frameworks and guidance for patching, certificate management, and incident response form a strong foundation, the DSS team should decide on many of the requirements that need to be levied on entities joining a DSS pool to ensure cohesive security.

Given the distributed and multi-tenant nature of the DSS, where multiple independent organizations manage individual nodes of the system, it's crucial to emphasize that securing the DSS as a whole is a collective responsibility. Each entity's strict adherence to the standard operating procedures (SOPs) and security guidelines set out by the DSS team is integral. Any deviation from these guidelines by a single entity can

introduce vulnerabilities into the shared infrastructure, compromising the security posture of the entire DSS pool. Each entity's role in this collective effort is vital.

For example, suppose a single entity neglects promptly applying patches or updates. In that case, it leaves their node vulnerable to known exploits, which attackers could use as a foothold to compromise the broader system. Similarly, if certificate rotation schedules are not adhered to, expired or compromised certificates could undermine mutual trust between nodes, disrupting secure communication channels and exposing data to unauthorized access. Moreover, incident response presents another area where unified action is essential. Suppose one entity fails to report a security incident promptly or does not coordinate its response efforts with the broader DSS pool. In that case, the incident may escalate, impacting the entire network's operational integrity and data security. For this reason, standardized and enforced incident response protocols must be followed by all entities, with clear communication channels established for cross-organization incident coordination.

In addition to existing guidelines, new mandates on the pool may be necessary to ensure the security of the DSS infrastructure. These could include more stringent auditing mechanisms to verify compliance with patching, certificate management, and incident response procedures. Enhanced threat intelligence sharing between entities could also be required, enabling proactive identification and mitigation of emerging threats across the pool. Once again, the DSS development team's critical role in determining these mandates and guidelines should reassure you of the system's robust management.

Securing the DSS pool requires that all participating entities adhere rigorously to the DSS team's operational guidance. Consistently applying things like patch management, certificate rotation, and incident response protocols will make it easier to maintain the DSS's security integrity. Therefore, reinforcing these standards and considering additional requirements will be key to safeguarding the entire system against both known and emerging cybersecurity threats.

Appendix B

Technical Findings

Technical findings are broken down by file using the DSS GitHub directory structure.

deploy\infrastructure\dependencies\terraform-google-kubernetes\cluster.tf

Concerning the resource "google_container_cluster" and "kubernetes_cluster," the specified Google terraform configuration does not define authorized networks to restrict GKE control planes' access. In general, limit the IP address range access to the cluster control planes and block other IP addresses. If possible, add a generic cidr block appropriate for the deployment or point users to configure the cluster file for their specific deployment. For example:

```
  cidr_blocks {
   cidr_block = "10.10.128.0/24"
   display_name = "10.network is private"
  }
```

Similarly, the configuration makes the GKE Cluster accessible through the public internet. Block direct access to the cluster from an external IP address by default. Expose services to public IP addresses on an as-needed basis. The following example blocks any client using an external IP address to connect to the control plane's public endpoint by setting enable_private_endpoint to TRUE. In addition, by setting enable_private_nodes to TRUE, all cluster nodes can only have RFC 1918 private addresses. The example strengthens security by making it difficult for an external host to establish a connection.

```
resource "google_container_cluster" "cluster-example" {

...
 private_cluster_config {
  enable_private_endpoint = true
  enable_private_nodes = true
 }
...
}
```

For the resource "google_container_node_pool" "dss_pool", the Terraform configuration also sets up GKE nodes using a configurable reference (google_machine_type.tf) that may not run Container-Optimized OS. By default, GKE nodes run with Container-Optimized OS (COS). COS is an operating system image optimized to run GKE nodes on Google Compute Engine instances. Opting out of the default forgoes the benefits of enhanced security and efficiency. Consider modifying the google_machine_type.tf file to explicitly define image_type as the Google-managed COS image to ensure the operating system stays up-to-date with the best secure practices. For example:

```
resource "google_container_node_pool" "node_pool_demo" {
```

```
 ...
 node_config {
  image_type = "COS_CONTAINERD"
  ...
 }
```

cmds\core-service\main.go

The DSS server initialized on line 332 uses an insecure protocol instead of a secure protocol. Communication over HTTP is unauthenticated and unencrypted, increasing the risk of compromise. Unless there is rationale otherwise, consider using TLS / HTTPS for communication.

```
        logger.Info("Starting DSS HTTP server")
        return httpServer.ListenAndServe()
}
```

interfaces\aux_\aux_.yaml

In aux_.yaml, the OpenAPI specification needs to include the securitySchemes definition. The securitySchemes definition specifies the security mechanisms used globally or by specific API operations. The securitySchemes definition is typically specified under the reusable components object and is referenced globally or by particular operations to dictate security requirements for interaction. Specify an appropriate securitySchemes definition: https://learn.openapis.org/specification/security.html. For example,

```
openapi: 3.0.3
  info:
    title: My API
    version: 1.0.0
  components:
   schemas:
     GeneralError:
       type: object
       properties:
        ...
   securitySchemes:
     api_key:
        type: apiKey
        name: api_key
        in: header
     petstore_auth:
```

```
    type: oauth2
    flows:
  ...
```

deploy\infrastructure\dependencies\terraform-aws-kubernetes\cluster.tf

The resource "aws_eks_cluster" "kubernetes_cluster" and resource "aws_eks_node_group" "eks_node_group" configurations may have overly broad access permissions; endpoint_public_access is set to TRUE, and source_security_groups_ids unset. Loose access configurations can expose systems and broaden an organization's attack surface. Services open to interaction with the public are typically subjected to near-continuous scanning and probing by malicious entities. Open services are especially problematic when a zero-day exploit for an exposed service is discovered and published, such as Heartbleed. Attackers can actively pursue and search for unpatched and exposed systems to exploit. The cluster configuration should have network rules that restrict access to services and ensure the services are available to as limited an audience as possible. For example, ports for service management activities should not be publicly accessible. Ensure these ports are only available to a restricted pool of systems that require authorized access. The following improves the configuration in Example 1. It specifies an aws_eks_cluster resource with endpoint_public_access no longer set to the insecure value TRUE.

```
resource "aws_eks_cluster" "example" {
  vpc_config {
    endpoint_public_access = false
  }
...
}
```

deploy/infrastructure/dependencies/terraform-aws-kubernetes/network_vpc.tf

Similar to the example above, map_public_ip_on_launch set to the value TRUE. Unless there is rationale to the contrary, consider limiting access by default and configuring appropriate connections later.

deploy/operations/Docker

deploy/infrastructure/utils/Docker

interfaces/openapi-to-go-server/example/Docker

interfaces/openapi-to-go-server/Docker

test/repo_hygiene/Docker

Docker

These Docker files do not specify a USER, so it defaults to running with a root user. When a Dockerfile does not specify a USER, Docker containers run with superuser privileges by default. These superuser privileges are propagated to the code running inside the container, which is usually more permission than necessary. Running the Docker container with super user privileges broadens the attack surface, which might enable attackers to perform more severe forms of exploitation. Running your containers as a non-root user is good practice when possible.

Dockerfile:8,9,26

deploy/operations/Dockerfile:5,9,22

Some of the Docker files have build dependencies using a non-specific version, which can leave the build system vulnerable to malicious binaries or cause the system to experience unexpected behavior. Dockerfiles can specify an unbound range of versions for dependencies and base images. If an attacker can add malicious versions of dependencies to a repository or trick the build system into downloading dependencies from a repository under the attacker's control, if docker is configured without specific versions of dependencies, then docker will silently download and run the compromised dependency. Additionally, the latest tag automatically indicates the version level of an image that doesn't use a digest or unique tag to provide a version. Docker automatically assigns the newest tag as mechanism to point to the most recent image manifest file. Because tags are mutable, an attacker can replace an image or layer using a latest (or weak tags such as imagename-lst, imagename-last, myimage).

Consider using version pinning or simple pinning. Version pinning explicitly specifies the version of images, libraries, and support packages on which an application or system depends. For example,

```
RUN zypper install <package_name>=<version> \
RUN gem install <package_name> --version <version>
RUN gem install <package_name> -v <version>
RUN apk add <package_name>=<version>
RUN apt-get update && apt-get install -y  \
  <package_name>=<version>  \
  <package_name>=<version> \
  <package_name>=<version> \
  && rm -rf /var/lib/apt/lists/*
```

Where <package_name> is the name of the dependency to install and <version> is the exact version or release the application should use.

cmds/db-manager/main.go

```
// Identify files defining version migration steps
        files, err := os.ReadDir(*path)
        if err != nil {
```

A malicious user may be able to leverage the file system path argument to ReadDir() at main.go line 181, used for migrating db migration, to cause harm (i.e. classic path manipulation vulnerability). The approximate path through the code in question is:

```
main.go:33 - String(return)
main.go:33 - Assignment to github.com.interuss.dss.cmds.db-manager::path
main.go:46 - Read github.com.interuss.dss.cmds.db-manager::path
main.go:46 - enumerateMigrationSteps(0)
main.go:181 – ReadDir()
```

Consider testing malicious inputs to the migration path and implementing a sanitization method or checking for the path. For example, create a list of characters that are permitted to appear in the resource name and accept input composed exclusively of characters in the approved set. There are likely some characters that are undesirable for said input.

deploy/infrastructure/dependencies/terraform-aws-kubernetes/cluster.tf

Note that cluster.tf, a configuration creates a resource without encryption at rest enabled, which could expose data to unauthorized access and potential theft. Consider enforcing encryption at rest if applicable. For example,

deploy/infrastructure/dependencies/terraform-aws-kubernetes/cluster.tf

Note that cluster.tf, a configuration creates a resource without encryption at rest enabled, which could expose data to unauthorized access and potential theft. Consider enforcing encryption at rest if applicable. For example,

```
esource "aws_eks_cluster" "example" {
  encryption_config {
    provider {
      key_arn = "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  }
...
}
```

interfaces/aux_/aux_.yaml

In aux_.yaml, the OpenAPI specification does not define responses for security-relevant status codes for an operation that uses an authentication scheme. E.g.

aux_.yaml:1 - ConfigMap
aux_.yaml:70 - ConfigPair
aux_.yaml:26 – ConfigMap

A best practice is configuring APIs to use custom error responses instead of relying on the underlying framework's default responses. Default responses might give detailed information about the error that occurred and inadvertently leak sensitive information, which attackers could use for future attacks. In this case, the operation inherits a global security definition but does not define the 401 or 403 status code responses.

Make sure the specification provides 401 and 403 status code response definitions for security-sensitive operations. RFC 2616 for the HTTP protocol requires a 401 response when client authentication is missing or cannot be performed using the provided credentials. The specification also requires a 403 response if the client cannot access the requested resource. For example,

```
openapi: 3.0.3
info:
 ...
security:
 - basicAuth: []
paths:
 /greet:
  get:
   responses:
    '200':
     ...
    '401':
     ...
    '403':
     ...
```

build/dev/extract_json_field.py

Code ignores an exception which can cause the program to overlook unexpected states and conditions. At a minimum, log the fact that the exception was thrown so that it will be possible to come back later and make sense of the resulting program behavior.

try:

```
  for line in sys.stdin:
    s += line + "\n"
except KeyboardInterrupt:
    pass
```