SRS - Software Requirements and Specifications

Project Name: Automated Misconfiguration & Threat Detection in Public Cloud Storage

Project Developers: Jamal Tannous
Asya Persan
Dan Ydov

1. Overview

The system is designed to automatically detect misconfigurations and security threats in public cloud storage services such as AWS S3, Azure Blob Storage, and Google Cloud Storage. The system will perform periodic scans, identify high severity security issues, and provide real-time remediation recommendations.
In addition, the system would identify configuration weaknesses,
anomalies in access patterns, and provide valid indicators of attack suspicion or potential data leakage.
The main KPI: On our synthetic benchmark dataset, the system should correctly flag at least 90% of injected misconfigurations (recall), with a false-positive rate under 10%.

2. Problem Description and Motivation

Problem Description
Many organisations store sensitive information in public cloud services, and configuration errors frequently occur, which can expose the data to the public or unauthorised parties.
Data breaches resulting from misconfigurations
are among the most common and dangerous security threats in the cloud era.

Motivation

- Prevent sensitive data leaks
- Comply with regulatory and security standards (GDPR, HIPAA, ISO 27001)
- Early detection of security vulnerabilities before attackers exploit them
- Reduce costs associated with security incidents
- Improve organisational security culture

3. Project Goals

- Automatically identify common cloud storage misconfigurations, including public access, permission mismatches, missing encryption, and disabled logging
- Apply unsupervised machine-learning techniques to synthetic access logs to flag abnormal behaviour or potential misuse
- Present results in a concise, human-readable form, including severity levels (Low, Medium, High) and remediation suggestions
- Use a modular architecture that allows future integration with additional cloud providers beyond the initial target
- Offer a simple CLI or basic web interface that allows users to trigger scans and view results without unnecessary complexity
- Ability for REST API and similar secure integrations

4. Approach

The system combines cloud API integration, rule-based checks, and lightweight ML to detect misconfigurations and suspicious activity.

- **Cloud API Integration** - Retrieve bucket configurations (permissions, encryption,
- logging, policies) through the provider’s secure API
- **Rule-Based Checks** - Evaluate configurations against known best practices to detect public access, weak permissions, missing encryption, and similar risks
- **Unsupervised ML Detection** - Analyse synthetic access logs using a simple unsupervised model to flag abnormal or suspicious behaviour
- **Severity Classification & Reporting** - Assign severity levels to findings and present results in a clear, human-readable report with remediation suggestions
- **Modular Architecture** - Separate modules for data retrieval, analysis, ML, and reporting to support maintainability and future cloud provider extensions
- **Simple & Minimal User Interface** - Provide a simple CLI or small web interface to trigger scans and review results. The UI would be easy to use and with clear fonts on a clean background chosen by the customer

5. Stakeholders and Scenarios

Stakeholders

- **Security Engineer** - Primary user. Runs scans, reviews misconfigurations and anomalies, and prioritises remediation
- **DevOps / Cloud Engineer** - Uses the tool during deployment or configuration changes to ensure cloud storage settings follow best practices
- **Compliance Officer / Project Manager** - Reviews reports and severity classifications for compliance needs

Scenarios

- **Pre-Deployment Check** - DevOps engineers create or modify a cloud storage bucket and run the tool before deployment.
  The system flags misconfigurations such as public access or missing encryption and guides fixes.
- **Periodic Security Scan** - Security engineers schedule or manually run weekly scans across all cloud storage buckets.
  The tool detects misconfigurations and ML-based anomalies, helping them prioritise issues.
- **Compliance Review** - Compliance officers request a summary of all high-severity issues.
  The system exports a structured report sorted by severity level.

6. Functional Requirements

Configuration Retrieval

- **FR1:** The system shall connect securely to the selected cloud provider.
- **FR2:** The system shall retrieve each bucket's configuration (permissions, encryption, logging, policies).
- **FR3:** The system shall keep configuration data only in memory and shall not store sensitive information on disk.

Rule-Based Misconfiguration Detection

- **FR4:** The system shall check bucket configurations against a predefined set of best-practice rules.
- **FR5:** The system shall detect buckets or objects that are publicly accessible.
- **FR6:** The system shall detect when encryption is missing or incorrectly configured.
- **FR7:** The system shall detect bucket or IAM policies that grant excessive or unnecessary permissions.
- **FR8:** The system shall assign each issue a severity level (Low, Medium, High).

ML-Based Anomaly Detection

- **FR9:** The system shall load simple synthetic access logs containing fields such as user, time, action, and source.
- **FR10:** The system shall use a lightweight unsupervised ML model to identify unusual or suspicious activity.
- **FR11:** The system shall flag suspicious log entries and assign an anomaly score.

Reporting and User Interaction

- **FR12:** The system shall present all results in a clear and readable report.
- **FR13:** The system shall group findings by severity and include short suggestions for fixing each issue.
- **FR14:** The system shall allow users to export results in JSON or text format.
- **FR15:** The system shall provide a simple CLI or small web interface to run scans and view results.

Scan Execution

- **FR16:** The system shall allow users to run scans on demand.
- **FR17:** The system shall support automatic scheduled scans.

7. Non-Functional Requirements

Performance Requirements

- **NFR1:** The system shall retrieve configuration data within a reasonable time (e.g., under 5 seconds for up to 50 buckets).
- **NFR2:** The anomaly detection model shall process log entries quickly enough to support smooth, interactive use.

Security Requirements

- **NFR3:** All communication with cloud providers shall use secure protocols (HTTPS/TLS).
- **NFR4:** The system shall not store sensitive credentials or configuration data on disk.
- **NFR5:** Access to scans and results shall be restricted to authorised users.

Usability Requirements

- **NFR6:** The interface (CLI or simple web UI) shall be easy to use without advanced training.
- **NFR7:** Reports shall be formatted clearly and be easy to interpret.

Reliability Requirements

- **NFR8:** The system shall handle missing or invalid configuration data gracefully.
- **NFR9:** The system shall retry cloud API calls when temporary failures occur.

Portability Requirements:

- **NFR10:** The system shall run on major operating systems, including Windows, macOS, and Linux.

8. Use Cases

UC1: Onboarding Cloud Account

Actor: Security Admin
Description: Connect a new cloud account to the system
Pre-conditions: User authenticated to the system.

Main Flow:

1. User selects **“Add Cloud Account”**.
2. User selects cloud provider (AWS/Azure/GCP).
3. User enters read-only credentials.
4. System validates credential format.
5. System attempts connection using secure API.
6. System retrieves basic configuration preview **in memory only**.
7. System displays a short summary of discovered storage resources.

Alternative Flows:

- 4a. Authentication failed - system displays an error and requests new credentials.
- 5a. Permission insufficient - system shows missing permissions.

Post-conditions:

- Cloud account connected and automatic scans scheduled.
- System ready for on-demand or scheduled scans.

UC2: Automated Security Scan

Actor: System (scheduled)/Security Admin (Manual)
Description: Periodic automated security scan
Pre-conditions: At least one cloud account configured
Main Flow:

1. System (or user) initiates a scan.
2. System retrieves storage configurations.
3. System performs rule-based checks:
    - Public access.
    - Encryption.
    - IAM/policy permissions.
    - Logging status.
4. System detects and classifies findings by **Low/Medium/High**.
5. System optionally loads synthetic access logs.
6. System runs ML anomaly detection.
7. System aggregates all findings into a unified results list.

8. System stores **only processed findings** in DB.
9. System generates a human-readable results report.

Postconditions:

- Findings stored and ready for viewing.

UC3: Investigate Security Finding

Actor: Security Analyst
Description: Examine a specific security finding
Pre-conditions: Findings exist in the system.

Main Flow:

1. User opens the findings view (CLI or simple UI).
2. User filters findings by:
    - Severity
    - Provider
    - Type (Misconfiguration / Anomaly)
3. System displays selected finding with:
    - Description.
    - Assigned severity.
    - Short remediation suggestion.
    - Associated metadata (resource, policy, etc.).
4. User optionally applies fix in cloud console (outside system scope).
5. User may re-run scans to verify fix.

Post-conditions: None.

UC4: Export Results Report

Actor: Compliance Officer
Description: Export a periodic report summarizing scan findings.
Pre-conditions: Scans performed during the period
Main Flow:

1. User selects **“Export Report”**.
2. User chooses format: JSON or text.
3. System compiles the findings grouped by severity.
4. System generates the export file.
5. User downloads the file.

Post-conditions: Report file generated and available for download.

UC5: Configure Scan Scheduling

Actor: Security Admin
Description: Configure recurring automated scan schedules.
Pre-conditions: At least one cloud account onboarded.

Main Flow:

1. User selects **“Scheduling Settings”**.
2. User sets recurrence (e.g., daily/weekly).
3. System validates schedule format.
4. System activates schedule.

Alternative Flows:

- Invalid time format – system shows correction prompt.

Post-conditions: Future scans triggered automatically.

9. System Flows

Flow 1: End-to-End Scanning Process

1. User or scheduler triggers a scan.
2. System retrieves configuration data from cloud provider.
3. System keeps raw data in memory only.
4. Rule Engine evaluates misconfigurations.
5. ML Engine processes synthetic logs.
6. System assigns severity levels (Low/Medium/High).
7. System stores processed findings in database.
8. System produces human-readable structured output.

Flow 2: Findings Viewing & Filtering

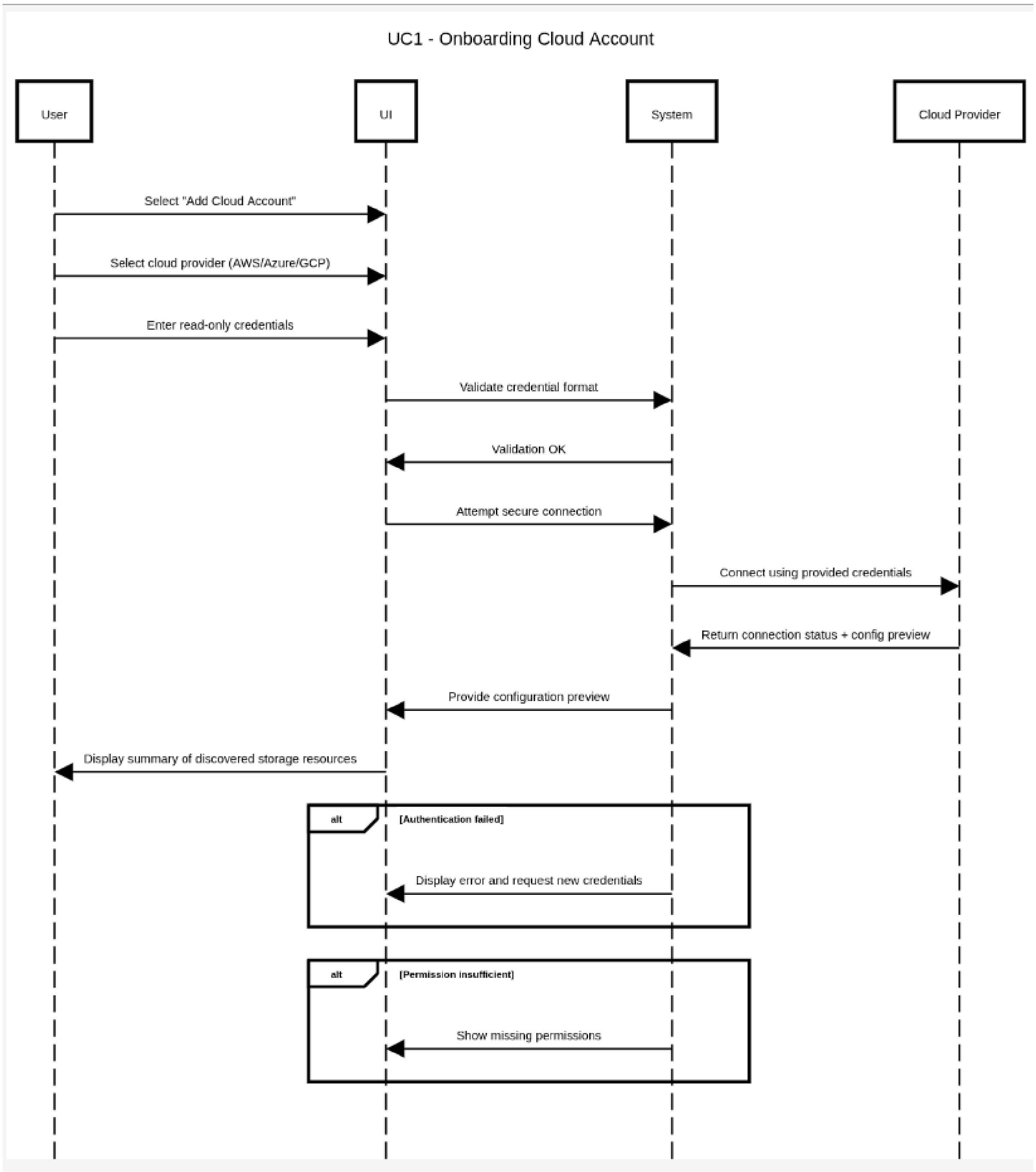
1. User opens findings view.
2. System loads findings from DB.
3. User applies filters (severity, provider, type).
4. System returns filtered results.
5. User inspects finding - system shows summary + remediation suggestion.

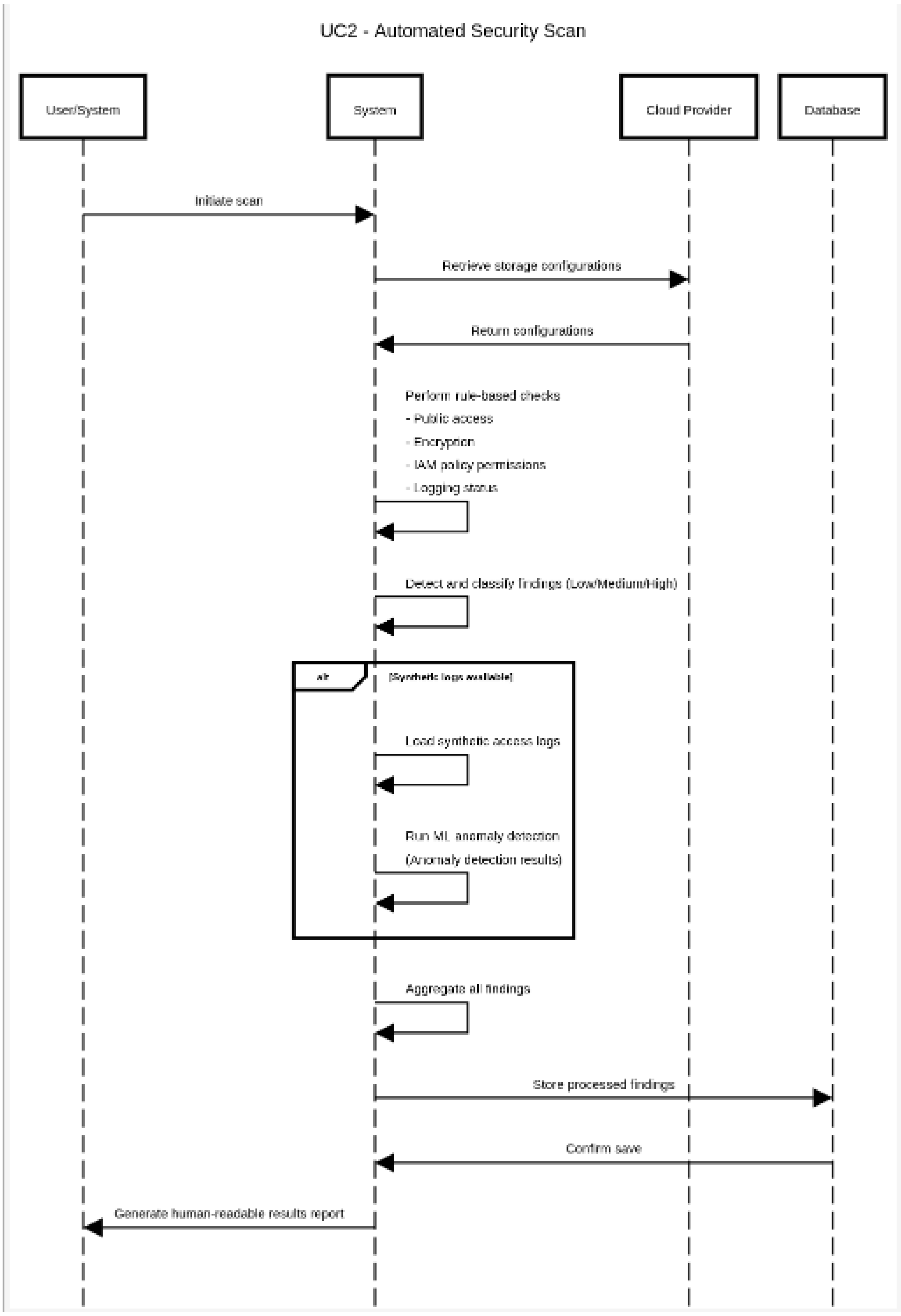
Flow 3: Report Export

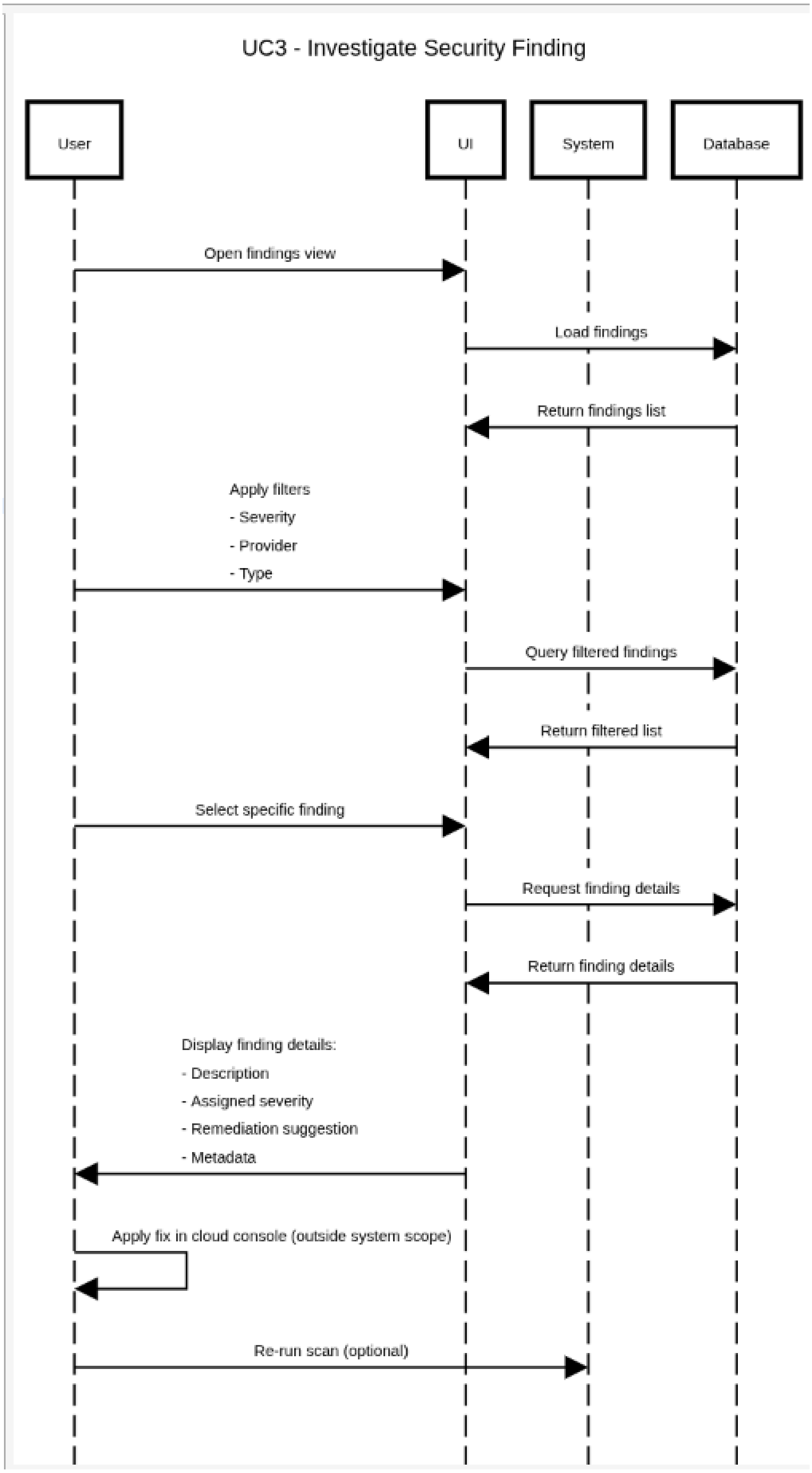
1. User selects report export option.
2. User chooses JSON or text.
3. System formats findings by severity.
4. System generates file.
5. User downloads.

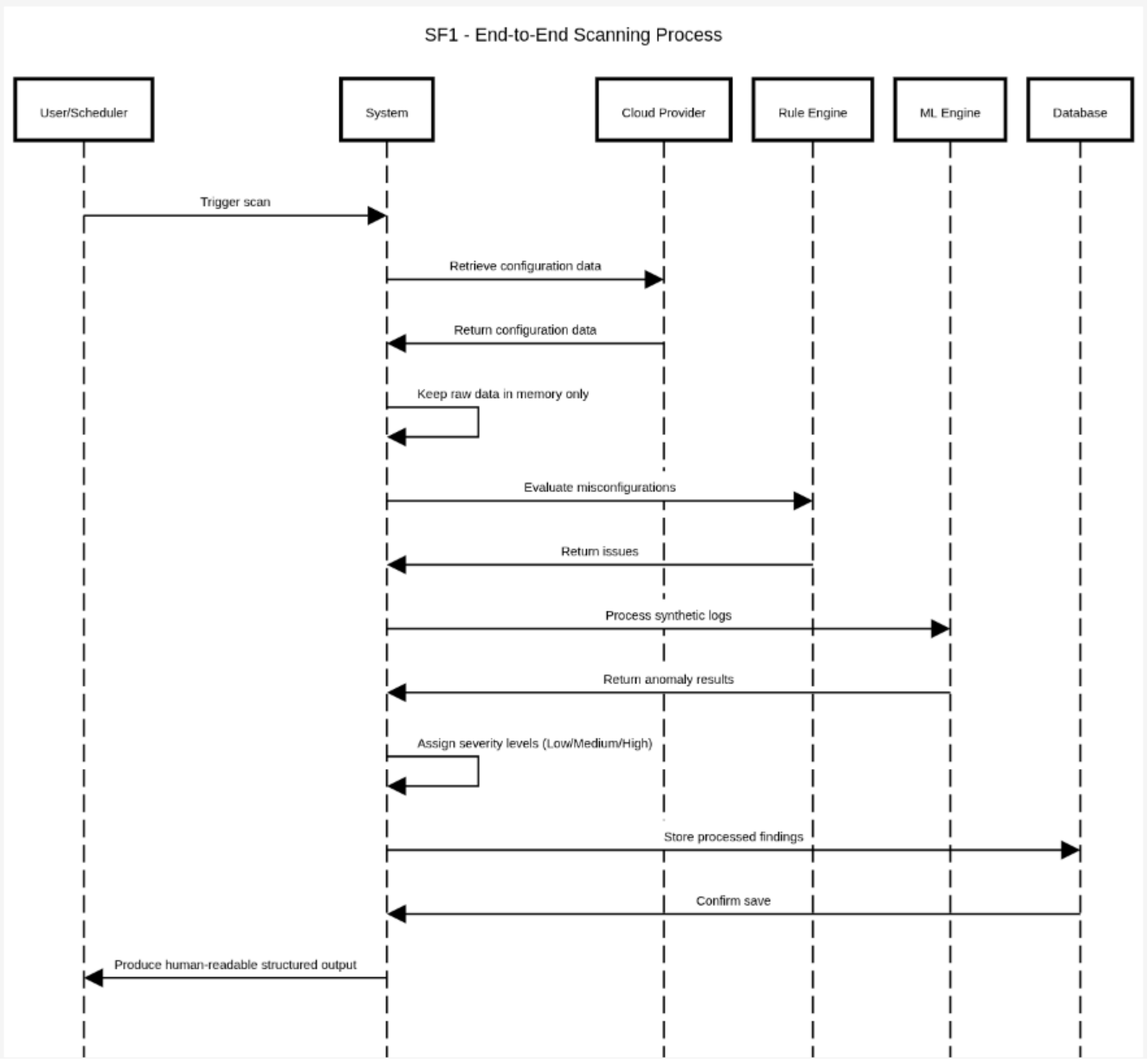
Summary

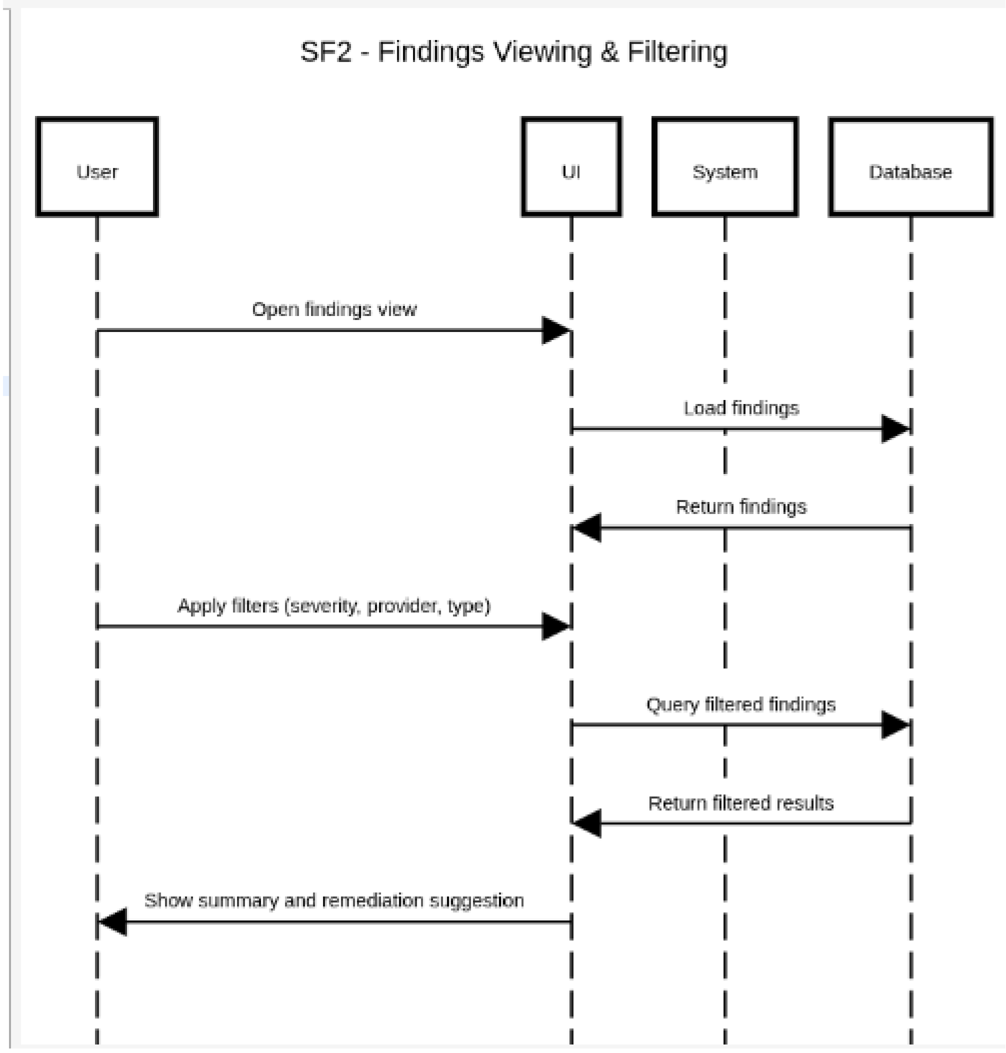
This SRS document defines the complete requirements for an automated misconfiguration and threat detection system for public cloud storage services. The system will provide organisations with a powerful automated tool to maintain cloud asset security while complying with regulatory standards and continuously improving organisational security posture.

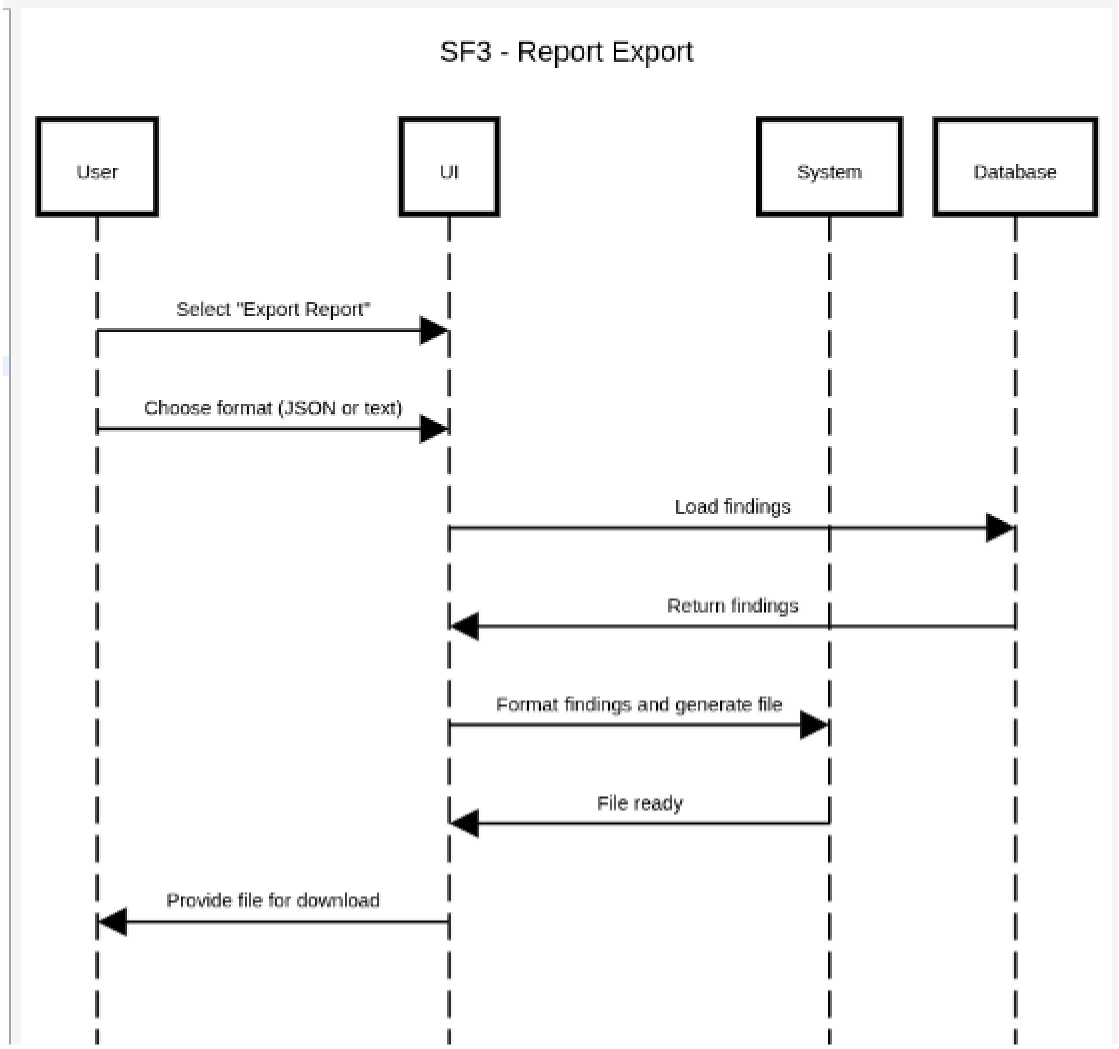

UC1 - Onboarding Cloud Account
User
UI
System
Cloud Provider
Select "Add Cloud Account"
Select cloud provider (AWS/Azure/GCP)
Enter read-only credentials
Validate credential format
Validation OK
Attempt secure connection
Connect using provided credentials
Return connection status + config preview
Provide configuration preview
Display summary of discovered storage resources
alt
[Authentication failed]
Display error and request new credentials
alt
[Permission insufficient]
Show missing permissions


UC2 - Automated Security Scan
User/System
System
Cloud Provider
Database
Initiate scan
Retrieve storage configurations
Return configurations
Perform rule-based checks
- Public access
- Encryption
- IAM policy permissions
- Logging status
Detect and classify findings (Low/Medium/High)
alt
[Synthetic logs available]
Load synthetic access logs
Run ML anomaly detection
(Anomaly detection results)
Aggregate all findings
Store processed findings
Confirm save
Generate human-readable results report


UC3 - Investigate Security Finding
User
UI
System
Database
Open findings view
Load findings
Return findings list
Apply filters
- Severity
- Provider
- Type
Query filtered findings
Return filtered list
Select specific finding
Request finding details
Return finding details
Display finding details:
- Description
- Assigned severity
- Remediation suggestion
- Metadata
Apply fix in cloud console (outside system scope)
Re-run scan (optional)


SF1 - End-to-End Scanning Process
User/Scheduler
System
Cloud Provider
Rule Engine
ML Engine
Database
Trigger scan
Retrieve configuration data
Return configuration data
Keep raw data in memory only
Evaluate misconfigurations
Return issues
Process synthetic logs
Return anomaly results
Assign severity levels (Low/Medium/High)
Store processed findings
Confirm save
Produce human-readable structured output


SF2 - Findings Viewing & Filtering
User
UI
System
Database
Open findings view
Load findings
Return findings
Apply filters (severity, provider, type)
Query filtered findings
Return filtered results
Show summary and remediation suggestion


SF3 - Report Export
User
UI
System
Database
Select "Export Report"
Choose format (JSON or text)
Load findings
Return findings
Format findings and generate file
File ready
Provide file for download