DESIGN AND DEMONSTRATION OF CONFIGURATION MANAGEMENT DATABASE (CMDB) FOR IT SECURITY MANAGEMENT

David Abiodun

1058 Words

CMDB Design

1. Security Management Requirements

1.1: Asset Management tracks IT hardware and software.

1.2: Vulnerability Tracking monitors IT asset vulnerabilities through scans, prioritisation, asset linking, remediation monitoring, and records of data sources.

1.3: Compliance Monitoring tracks asset compliance, relates requirements to assets, audits assures regulatory compliance, and sends remediation notifications.

1.4: Incident Response identifies, manages, and resolves security incidents by detecting and classifying them, tracking impacted assets, documenting containment steps, investigating root causes, and maintaining detailed records.

Data Model for the CMDB

2.1: Key Configuration Items (Cis): Assets, Vulnerabilities, Compliance Requirements, Incidents, Users, Services (In attached spreadsheet)

2.2: Attributes for Each Configuration Item (CI):

Assets

Asset ID	Asset Type	Owner	Location	Criticality	Status	Configuration

(The attached spreadsheet contains the full tables for all tables below)

Vulnerabilities

Vulnerability ID	Affected Asset	Discovery Date	Mitigation Status	Vulnerability Source

Compliance Requirements

Compliance ID	Regulation / Standard	Asset ID (Affected)	Compliance Status	Last Audit Date

Incidents

Incident ID	Incident Type	Severity Level	Asset ID (Affected)	Incident Description	Root Cause / Resolution Notes	Date of Incident	Mitigation Actions

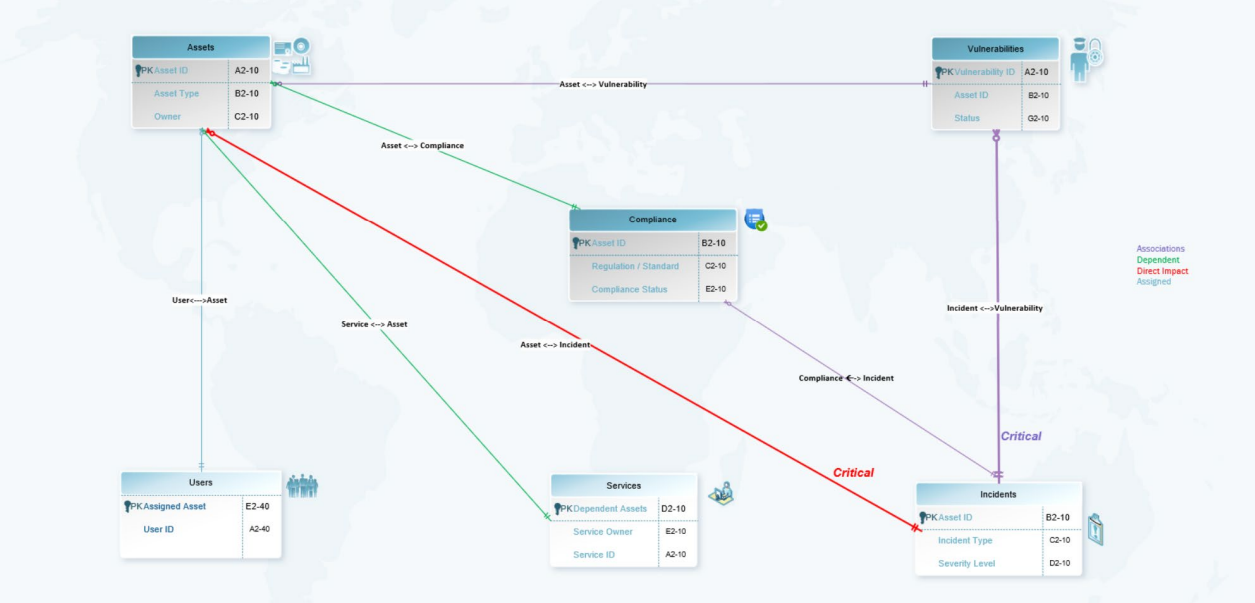
Users

User ID	Department / Role	Access Level	Affected Assets	Username

Services

Service ID	Dependencies	Criticality Level	Associated Incidents

2.3: Relationships Between Cis



- **Asset ⟷ Vulnerability:**
- **Asset ⟷ Compliance**
- **Asset ⟷ Incident**
- **Service ⟷ Asset**
- **Incident ⟷ Vulnerability**
- **User ⟷ Asset**
- **Compliance ⟷ Incident**

3. CMDB Design Document

3.1: Architecture: The centralised CMDB stores all CI data with RBAC, which is read-only for end users, read-write for asset managers and IT support, and full access for database managers. Data integration from several sources requires import routines, APIs, or ETL for automatic modifications.

3.2: Data Schema

Assets store data about all assets in the organisation.

Field Name	Data Type	Description
Asset ID	Integer (PK)	Unique identifier
Asset Type	Varchar	Asset type
Owner	Varchar	Owner
Location	Varchar	Physical/virtual
Status	Varchar	Status
Last Updated	Date	Last Update

Relationships

Vulnerabilities: One-to-Many

Compliance: Many-to-One

Incidents: Many-to-One

Users: Many-to-One

Services: Many-to-Many

Vulnerabilities track vulnerabilities associated with assets.

Field Name	Data Type	Description
Vulnerability ID	Integer (PK)	Unique identifier
Asset ID	Integer (FK)	ID of the affected asset
Severity	Varchar	Severity level
Status	Varchar	Status
Description	Text	Brief description
Date Identified	Date	Date detected

Relationships

Assets: Many-to-One

Incidents: Many-to-Many

Compliance tracks standards and policies for assets.

Field Name	Data Type	Description
Compliance ID	Integer (PK)	Unique identifier
Asset ID	Integer (FK)	ID of the asset
Regulation/Standard	Varchar	Regulatory standard
Compliance Status	Varchar	Status
Last Audit Date	Date	Last audit date

Relationships:

Assets: Many-to-One

Incidents: Many-to-Many

Incidents track incidents related to assets, vulnerabilities, and compliance.

Incidents Table 4:

Field Name	Data Type	Description
Incident ID	Integer (PK)	Unique identifier
Asset ID	Integer (FK)	Asset ID
Incident Type	Varchar	Incident Type
Severity Level	Varchar	Severity level
Description	Text	Description
Date Reported	Date	Incident date

Relationships

Assets: Many-to-One.

Vulnerabilities: Many-to-Many.

Compliance: Many-to-Many.

Users store data about users.

Field Name	Data Type	Description
User ID	Integer (PK)	Unique identifier
Assigned Asset	Integer (FK)	Asset ID
Access Level	Varchar	Access rights
Department	Varchar	Users Department

Relationships:

Assets: Many-to-One.

Incidents: Many-to-Many.

Services represent services provided by assets.

Field Name	Data Type	Description
Service ID	Integer (PK)	Unique identifier
Service Name	Varchar	Service Name
Service Owner	Varchar	Service Owner
Dependent Assets	Integer (FK)	Assets ID for affected assets
Status	Varchar	Operational status

Relationships:

Assets: Many-to-Many.

Incidents: Many-to-Many.

3.3: Data Validation Rules maintain CMDB accuracy across tables.

Assets Table

- Asset ID: Unique, non-null, and auto-incremented.
- Asset Type: Predefined list.
- Owner: User ID in the user's table.
- Location: Predefined locations.
- Status: Active, Inactive, Retired.
- Last Updated: Valid date and automatically set to the current update date.

Vulnerabilities Table

- Vulnerability ID: Unique and non-null.
- Asset ID: Asset ID in the assets table.
- Severity: Low, Medium, High, Critical.
- Status: Open, In Progress, Resolved, Closed.
- Date Identified: Valid date.

Compliance Table

- Compliance ID: Unique and non-null.
- Asset ID: Asset ID in the assets table
- Compliance Status: Compliant, Non-Compliant, Pending.
- Last Audit Date: Valid date.

Incidents Table

- Incident ID: Unique and non-null.
- Asset ID: Asset ID in the assets table.
- Incident Type: Security Breach, Service Outage, Data Loss.
- Severity Level: Critical, High, Medium, Low.
- Date Reported: Valid date.

Users Table

- User ID: Unique.
- Assigned Asset: Asset ID in the assets table.
- Access Level: Admin, User, Viewer.
- Role: Job title.

Services Table

- Service ID: Unique and non-null.
- Service Owner: User ID in the users table.
- Dependent Assets: Asset IDs in the assets table
- Status: Online, Offline, Maintenance.

3.4: User Interface Specifications

User Interface Overview

Dashboard widgets show Total Assets by category, current Open Incidents, Compliance Status by asset, and Top Vulnerabilities by severity. Details can be accessed through each widget.

Asset View displays attributes, connected vulnerabilities, compliance requirements, incidents, and actions, including updating details, adding vulnerabilities, and viewing reports.

Incident Management centralises incident data, showing an incident list, detailed views, filtering options, timeline progression, and actions to create incidents, update status, and link vulnerabilities.

Compliance Monitoring provides asset compliance summaries, asset inventories, compliance criteria, and an audit log. Check compliance reports and update asset criteria.

Search and Reporting offers powerful filtering, global table search, and bespoke reports. Search Bar, Report Builder for new reports, Predefined Reports for convenient access, and Multiple Export Formats are vital features. Saving report configurations and scheduling report generation is possible.

CMDB Demonstration Phase

Scenario 1: Adding New Security Assets to the CMDB

Identify assets, enter data into the Assets table, validate, configure permissions, monitor, verify, and document.

Scenario 2: Updating Configuration Data

Access CMDB configuration data, adjust information, apply validation, update dependencies, record changes, evaluate and approve updates, interact with stakeholders, and document the process.

Scenario 3: Conducting Vulnerability Assessments

Select assets, gather vulnerability data, enter results into the CMDB, link assets, prioritise by severity, validate data, generate reports, assign remediation activities, and schedule follow-up assessments.

Scenario 4: Generating Compliance Reports:

Define compliance reporting needs, identify relevant assets and standards, collect CMDB data, apply filters, build and validate the report, deliver to stakeholders, store for audits, and schedule regular report generation.

Conclusion:

The CMDB's centralised design supports efficient configuration, compliance, and incident response management, driven by the need for unified visibility and control in complex IT environments (Farayola et al., 2023). Key design choices, such as role-based access and automated data integration, ensure accuracy and accessibility, though challenges like complex relationship mapping and data validation persist (Mohamed et al., 2008). Future enhancements, including AI-driven predictive analytics, will further strengthen CMDB functionality and resilience (Chi, 2024).

Bibliography

Farayola, O. A., Hassan, A. O., Adaramodu, O. R., Fakeyede, O. G., & Oladeinde, M. (2023) Configuration Management in the Modern Era: Best Practices, Innovations, and Challenges. *Computer Science & IT Research Journal*, 4(2), 140-157. DOI: 10.51594/csitrj.v4i2.613 [Accessed 01/11/2024].

Chi, Y. (2024) *Mastering Digital Complexity: The Role of Configuration Management Database (CMDB) in Modern Infrastructure Management*. International Journal of Management, IT, and Engineering, 14(3). http://www.ijmra.us [Accessed 08/11/2024].

Mohamed, M. S., Ribiere, V. M., O'Sullivan, K. J., & Mohamed, M. A. (2008) The re-structuring of ITIL implementation using knowledge management framework. *VINE: The Journal of Information and Knowledge Management Systems*, 38(3), 315-333. www.emeraldinsight.com/0305-5728.htm [Accessed 07/11/2024].