VIOE - Vulnerability Intelligence & Orchestration Engine

System Architecture Overview (Non-Code)

Document Version: 1.0 **Classification:** Internal / Vendor / Executive **Last Updated:** January 2026

Table of Contents

1. Document Purpose

This document provides a high-level overview of VIOE's system architecture for non-technical stakeholders, executives, and vendor partners. It explains how the system is structured and how components interact without requiring programming knowledge.

Intended Audience:

- Product Owners
- CTO / Technical Leadership

- Vendor Partners
- Enterprise Clients (Technical Evaluation)
- Auditors (Technical Assessment)

---

2. Architecture Summary

2.1 Architecture Style

VIOE is built as a **modern web application** using a client-server architecture with the following characteristics:

- **Single Page Application (SPA):** The user interface loads once and updates dynamically
- **API-Driven:** All data operations occur through secure application programming interfaces
- **Cloud-Native:** Designed to run on cloud infrastructure
- **AI-Augmented:** Machine learning capabilities embedded in core workflows

2.2 High-Level Architecture Diagram

```
[Diagram Placeholder: High-Level System Architecture]

Description: Three-tier architecture showing:
- Presentation Layer (User Interface)
- Application Layer (Business Logic & AI)
- Data Layer (Database & Storage)

With external integrations connecting to:
- Vulnerability Scanners
- Jira
- Slack
- Directory Services
```

2.3 Key Architectural Principles

Principle	Description
Separation of Concerns	Each layer handles specific responsibilities
API-First	All functionality accessible via APIs
Security by Design	Security embedded in all architectural decisions
Scalability	Architecture supports growth in users and data
Extensibility	New integrations can be added without core changes

---

3. Frontend Architecture

3.1 Overview

The frontend is the user-facing portion of VIOE—everything users see and interact with in their web browser.

3.2 Technology Approach

VIOE uses a **React-based** single page application, which means:

- **Fast User Experience:** Pages update instantly without full reloads
- **Responsive Design:** Works on desktop, tablet, and mobile devices
- **Modern Interface:** Contemporary look and feel with smooth interactions
- **Offline Resilience:** Continues working during brief network interruptions

3.3 User Interface Components

```
[Diagram Placeholder: UI Component Hierarchy]

Description: Tree structure showing:
- App Shell (Navigation, Header, Footer)
  - Dashboard Components (KPI Cards, Charts)
  - Vulnerability Components (List, Detail, Filters)
  - Task Components (Cards, Status, Actions)
  - Settings Components (Tabs, Forms, Toggles)
  - Reporting Components (Compliance, Analytics)
```

3.4 Design System

The interface is built using:

Element	Description
Component Library	Consistent, reusable interface elements
Design Tokens	Standardized colors, spacing, typography
Accessibility	WCAG-compliant for inclusive access
Dark Theme	Optimized for security operations environments

3.5 Key Frontend Capabilities

Capability	Benefit
Real-time Updates	See changes without refreshing
Interactive Charts	Drill into data visually
Smart Filtering	Find items quickly
Responsive Tables	Handle large data sets
Toast Notifications	Immediate feedback on actions

---

4. Backend Architecture

4.1 Overview

The backend handles all business logic, data processing, and integrations. It sits between the user interface and the database, enforcing rules and coordinating activities.

4.2 Platform Foundation

VIOE's backend is built on the **Base44 Platform**, which provides:

Service	Function
Authentication	User identity verification
Authorization	Access control enforcement
Data Storage	Secure database operations
File Storage	Document and file management
AI Services	Machine learning capabilities
Integration Hub	External system connectivity

4.3 Service Architecture

```
[Diagram Placeholder: Backend Services Architecture]

Description: Service blocks showing:
- Authentication Service
- Vulnerability Service
- Remediation Service
- Asset Service
- Incident Service
- Compliance Service
- AI Orchestration Service
- Integration Gateway
```

4.4 API Structure

All functionality is exposed through RESTful APIs organized by domain:

API Domain	Purpose
/vulnerabilities	Vulnerability CRUD operations
/tasks	Remediation task management
/assets	Asset inventory operations
/teams	Team management
/incidents	Incident response
/compliance	Compliance reporting
/settings	Configuration management
/import	Data import operations

4.5 Business Logic Layer

Key business processes handled by the backend:

Process	Description
Vulnerability Triage	AI-powered analysis and assignment
Status Management	Enforce valid state transitions
SLA Calculation	Track and alert on deadlines
Suppression Evaluation	Apply filtering rules
Compliance Mapping	Framework control alignment

---

5. Database Architecture

5.1 Overview

VIOE uses a cloud-managed database to store all application data securely and reliably.

5.2 Data Entities

The database stores the following primary data types:

Entity	Description	Volume
Vulnerabilities	Security findings	High
Remediation Tasks	Fix tracking	High
Assets	IT inventory	Medium
Teams	Organization structure	Low
Incidents	Security events	Medium
Compliance Reports	Framework assessments	Low
Threat Alerts	Threat intelligence	Medium
Suppression Rules	Filtering configuration	Low
Ownership Logs	Assignment audit trail	High
Configuration	System settings	Low

5.3 Data Relationships

```
[Diagram Placeholder: Entity Relationship Overview]

Description: Simplified ER diagram showing:
- Vulnerability → Team (assigned to)
- Vulnerability → Asset (affects)
- Vulnerability → Remediation Task (has many)
- Vulnerability → Ownership Log (has many)
- Team → Remediation Task (responsible for)
- Incident → Asset (affects)
```

5.4 Data Integrity

Mechanism	Purpose
Referential Integrity	Relationships enforced
Validation Rules	Data quality assurance
Audit Trails	Change history preserved
Soft Deletes	Data recovery capability

5.5 Data Retention

Data Type	Retention
Active records	Indefinite
Resolved vulnerabilities	2 years
Audit logs	3 years
Compliance evidence	5 years

---

6. AI & Intelligence Layer

6.1 Overview

VIOE incorporates artificial intelligence throughout the platform to automate analysis, prediction, and decision support.

6.2 AI Capabilities

```
[Diagram Placeholder: AI Capabilities Map]

Description: Mind map showing AI functions:
- Ownership Assignment
  - Git analysis
  - CODEOWNERS parsing
  - Directory mapping
- Threat Intelligence
  - Pattern detection
  - Vulnerability prediction
  - Emerging threat identification
- Compliance
  - Control mapping
  - Gap analysis
  - Policy recommendations
- Incident Response
  - Threat assessment
  - Blast radius calculation
  - Playbook generation
```

6.3 AI Services Used

Service	Purpose
Large Language Model (LLM)	Natural language analysis, report generation
Pattern Recognition	Trend analysis, anomaly detection
Classification	Severity assessment, category assignment
Prediction	Vulnerability forecasting

6.4 AI Integration Points

Function	AI Role
triageVulnerability	Analyze and assign ownership
bulkTriageVulnerabilities	Batch ownership assignment
estimateRemediationEffort	Effort prediction
proactiveThreatHunting	Threat pattern detection
generateThreatModel	STRIDE analysis
generateComplianceReport	Framework mapping
analyzeCodebase	Security pattern assessment
triageIncident	Threat level assessment

6.5 AI Confidence Model

All AI outputs include confidence scoring:

Level	Confidence	Interpretation
High	90-100%	Trusted output
Medium	70-89%	Review recommended
Low	Below 70%	Manual verification required

---

7. Integration Architecture

7.1 Overview

VIOE connects with external systems to import data, synchronize workflows, and send notifications.

7.2 Integration Map

```
[Diagram Placeholder: Integration Architecture]

Description: Hub-and-spoke diagram showing VIOE at center with connections to:

Inbound (Data Sources):
- Snyk
- SonarQube
- Checkmarx
- Qualys
- Tenable
- Rapid7

Bidirectional (Workflow):
- Jira

Outbound (Notifications):
- Slack
- Email

Identity (Authentication):
- Okta
- Active Directory
```

7.3 Integration Types

Type	Direction	Purpose
Scanner Integration	Inbound	Import vulnerability data
Jira Integration	Bidirectional	Task synchronization
Slack Integration	Outbound	Team notifications
Email Integration	Outbound	Alerts and digests
Directory Integration	Inbound	User/team mapping

7.4 Integration Security

Control	Description
API Keys	Secure credential storage
OAuth	Standard authentication protocol
HTTPS	Encrypted communications
Rate Limiting	Protect against overload
Audit Logging	Track all integration activity

7.5 Supported Scanners

Scanner	Integration Method	Data Types
Snyk	API / File	Application vulnerabilities
SonarQube	API / File	Code quality issues
Checkmarx	API / File	SAST findings
Qualys	API / File	Infrastructure vulnerabilities
Tenable	API / File	Network vulnerabilities
Rapid7	API / File	Security assessments

---

8. Data Flow Overview

8.1 Vulnerability Import Flow

```
[Diagram Placeholder: Vulnerability Import Data Flow]

Description: Sequential flow:
1. Scanner produces findings
2. File uploaded or API sync triggered
3. Data extracted and normalized
4. AI analyzes for ownership
5. Vulnerabilities created with assignments
6. Dashboard updated
7. Notifications sent (if configured)
```

Steps:

1. **Source:** Vulnerability scanner generates findings
2. **Ingestion:** Data imported via file upload or API
3. **Normalization:** Data standardized to common format
4. **Enrichment:** AI adds ownership and confidence
5. **Storage:** Records created in database
6. **Presentation:** Displayed in user interface
7. **Notification:** Alerts sent for critical items

8.2 Remediation Flow

```
[Diagram Placeholder: Remediation Data Flow]

Description: Sequential flow:
1. Vulnerability identified
2. Task created
3. (Optional) Jira issue created
4. Task worked by team
5. Status updated in VIOE
6. Jira synced bidirectionally
7. Vulnerability marked resolved
```

8.3 Compliance Reporting Flow

```
[Diagram Placeholder: Compliance Report Data Flow]

Description: Sequential flow:
1. User requests compliance report
2. System queries vulnerability data
3. AI maps to framework controls
4. Gap analysis performed
5. Evidence collected
6. Report generated
7. Results displayed
```

8.4 Incident Response Flow

```
[Diagram Placeholder: Incident Response Data Flow]

Description: Sequential flow:
1. Threat detected
2. Incident record created
3. AI assessment generated
4. Affected assets identified
5. Containment actions tracked
6. Timeline documented
7. Resolution recorded
8. Report generated
```

---

9. Security Architecture

9.1 Security Layers

```
[Diagram Placeholder: Security Architecture Layers]

Description: Concentric circles showing:
- Outer: Network Security (HTTPS, Firewalls)
- Middle: Application Security (Auth, RBAC)
- Inner: Data Security (Encryption, Access Control)
- Core: Audit & Monitoring
```

9.2 Authentication

Mechanism	Description
User Authentication	Username/password with MFA option
Session Management	Secure token-based sessions
SSO Support	Integration with enterprise identity
API Authentication	Token-based API access

9.3 Authorization

Control	Description
Role-Based Access	Permissions by role assignment
Team Scoping	Data visibility by team
Feature Flags	Enable/disable capabilities
Admin Restrictions	Elevated privilege controls

9.4 Data Protection

Mechanism	Scope
Encryption in Transit	All network communications
Encryption at Rest	Database and file storage
Access Logging	All data access recorded
Data Masking	Sensitive field protection

9.5 Compliance Controls

Framework	Relevant Controls
SOC 2	CC6.1-CC6.8 (Logical Access)
ISO 27001	A.9, A.10, A.12, A.13, A.14
GDPR	Article 25, 32
PCI DSS	Requirements 3, 4, 7, 8

10. Deployment Architecture

10.1 Deployment Model

VIOE supports multiple deployment configurations:

Model	Description	Use Case
Cloud-Hosted (SaaS)	Fully managed by vendor	Most customers
Private Cloud	Customer's cloud account	Enterprise isolation
Hybrid	Split deployment	Specific compliance needs

10.2 Cloud Architecture

```
[Diagram Placeholder: Cloud Deployment Architecture]

Description: Cloud infrastructure showing:
- Load Balancer (traffic distribution)
- Application Servers (horizontal scaling)
- Database Cluster (managed, replicated)
- File Storage (object storage)
- CDN (static asset delivery)
- AI Services (managed ML)
```

10.3 Environment Structure

Environment	Purpose	Access
Production	Live customer use	Controlled
Staging	Pre-release testing	Limited
Development	Feature development	Development team

10.4 High Availability

Component	Availability Approach
Application	Multiple instances, load balanced
Database	Managed service with replication
Storage	Distributed with redundancy
CDN	Globally distributed

---

11. Scalability Considerations

11.1 Scaling Dimensions

Dimension	Approach
Users	Horizontal application scaling
Data Volume	Database scaling, archival
Integrations	Queue-based processing
AI Processing	Managed service scaling

11.2 Performance Characteristics

Metric	Target
Page Load Time	< 2 seconds
API Response Time	< 500ms
Import Processing	~1000 records/minute
Report Generation	< 30 seconds

11.3 Capacity Planning

Factor	Consideration
Vulnerability Volume	Database sizing
Active Users	Application instances
Integration Frequency	API capacity
Report Generation	AI service capacity

11.4 Growth Support

The architecture supports growth through:

- Horizontal scaling of application tier
- Managed database scaling
- Queue-based integration processing
- Cloud-native elasticity

Document Control

Version	Date	Author	Changes
1.0	January 2026	Documentation Team	Initial release

This document provides architectural overview for planning and evaluation purposes.

VIOE - Vulnerability Intelligence & Orchestration Engine *System Architecture Overview*