Air-Gapped Deployment Architectures

A Technical Guide for Deploying SigDrive on SIPRNet, JWICS, and Disconnected Networks

January 11, 2026

Executive Summary

In the contemporary era of near-peer competition, the Department of Defense (DoD) and the Intelligence Community (IC) face a paradoxical challenge: the exponential growth of sensor data, particularly Radio Frequency (RF) and Signals Intelligence (SIGINT) data, versus the rigid constraints of secure, disconnected network environments. Modern warfare is increasingly data-centric, relying on the seamless integration of sensors, effectors, and command nodes as envisioned in the Joint All-Domain Command and Control (JADC2) initiative.[1] However, the most sensitive of these datasets often reside on isolated networks such as SIPRNet (Secret Internet Protocol Router Network), JWICS (Joint Worldwide Intelligence Communications System), or tactical edge networks that lack consistent connectivity to the commercial cloud.

The proliferation of sophisticated electronic warfare (EW) threats, the densification of commercial spectrum use through 5G and 6G technologies, and the rapid expansion of Low Earth Orbit (LEO) satellite constellations have created a "data gravity" challenge that legacy infrastructure is ill-equipped to handle.[1] A single high-bandwidth IQ recording can consume terabytes of storage in minutes, creating massive "binary blobs" that are opaque to traditional search engines. In the absence of specialized management software, engineering teams often revert to physical "sneakernet" logistics, shipping hard drives between test ranges and laboratories.[1] This manual handling results in data loss, version control chaos, and a complete lack of discoverability, where valuable signal recordings are effectively lost in unindexed storage.

This white paper, "Air-Gapped Deployment Architectures," serves as a definitive technical guide for deploying the SigDrive Enterprise RF Data Lake within these constrained environments. Unlike traditional enterprise software that assumes ubiquitous internet access for licensing, updates, and dependency management, SigDrive is architected with an "Air-Gap First" mentality.[1] This document details the architectural patterns, compliance frameworks (NIST 800-53, ICD 503), and operational workflows required to ingest, store, and analyze petabytes of RF data without a single byte crossing the public internet.

The analysis provided herein explores the integration of "Iron Bank" hardened containers, the utilization of "Zarf" for declarative offline software delivery, and the implementation of Cross Domain Solutions (CDS) for moving massive IQ recordings from unclassified collection points to

classified analysis enclaves. By adhering to the Modular Open Systems Approach (MOSA) and leveraging open standards like SigMF, this guide outlines a path to achieving "Data Readiness" for AI/ML pipelines in the most secure environments on Earth.

1. The Strategic Imperative of Disconnected Data Operations

1.1 The "Data Gravity" Problem in Electronic Warfare

The electromagnetic spectrum (EMS) has emerged as a primary domain of maneuver and conflict. Modern sensors, ranging from Software Defined Radios (SDRs) on drones to wideband receivers on satellite ground stations, generate data at rates that overwhelm legacy infrastructure. Radio Frequency data possesses unique volume and velocity characteristics that render standard enterprise IT tools ineffective. A single high-bandwidth IQ recording can consume terabytes of storage in minutes.[1] This creates a phenomenon known as "Data Gravity", as datasets grow, they become increasingly difficult to move, forcing applications and analytics to move to the data rather than vice versa.

In the commercial sector, the solution to data gravity is the hyperscale cloud (AWS, Azure). In the defense sector, however, the most valuable data is often classified or collected in austere environments (e.g., a submarine or a forward operating base) where cloud connectivity is nonexistent or prohibited. This disconnect creates a critical operational bottleneck. Without a centralized, searchable repository, analysts spend the vast majority of their time converting file formats and searching for data, leading to the common complaint: "I know we recorded this signal three years ago, but I can't find it".[1]

SigDrive specifically addresses this "data gravity" by functioning as a specialized Data Lake designed for the lifecycle management of RF data. The system utilizes a modern microservices architecture, incorporating FastAPI, React, PostgreSQL with PostGIS, and S3-compatible object storage, to manage the ingestion and indexing of these massive datasets.[1] By decoupling the storage of heavy binary data from the lightweight metadata index, SigDrive allows users to query petabytes of signal data instantly without the need to retrieve or download the underlying files until necessary.

1.2 Defining the Air-Gapped Environment

An air-gapped network is physically isolated from unsecured networks, including the public internet.[2] While this isolation provides a robust defense against external cyberattacks, protecting critical infrastructure like nuclear facilities and military command systems, it introduces significant friction into the software development and deployment lifecycle (SDLC). The isolation creates an "unbridgeable gap" that prevents remote intrusions but also complicates routine tasks such as patching, dependency management, and license validation.

Characteristics of the Disconnected Environment:

- **No Public Repositories:** Systems cannot run pip install, npm install, or docker pull from public registries. All dependencies must be vendored, scanned, and explicitly allowed into the environment.
- **No Remote Licensing:** Software cannot "phone home" to a license server to validate entitlements or check for updates. Licensing mechanisms must rely on offline cryptographic verification, such as signed JWTs or hardware keys.[3]
- **Manual Updates:** Patches and upgrades must be delivered via physical media (DVDs, specialized hard drives) or secure unidirectional gateways, necessitating a rigorous "Offline Upgrade" process.[1]
- **Strict Compliance:** Every component must meet rigorous security controls defined by the Risk Management Framework (RMF) and Intelligence Community Directives (ICD).[4]

SigDrive addresses these challenges by bundling all dependencies into a self-contained "Mission Pack," utilizing a microservices architecture that can be deployed via offline orchestration tools like Kubernetes or Docker Compose, ensuring that the system is fully functional from the moment of installation.[1] This "Air-Gap Ready" designation is a fundamental architectural choice, not merely a deployment option, creating a significant competitive moat against cloud-native competitors that rely on continuous internet connectivity.[1]

1.3 Alignment with DoD Doctrine

Three primary DoD initiatives create a massive tailwind for SigDrive's disconnected architecture, positioning it as a compliance enabler rather than a discretionary purchase [1]:

1. **JADC2 (Joint All-Domain Command and Control):** The Pentagon's vision to connect "sensors to shooters" across all branches requires a common data architecture. JADC2 relies on the ability to move data seamlessly between domains. SigDrive's use of open standards (SigMF) facilitates the "data fabric" required for joint operations.[1]
2. **MOSA (Modular Open Systems Approach):** Mandated by Congress to prevent vendor lock-in, MOSA requires systems to use modular interfaces and open standards. SigDrive's support for generic hardware metadata and "Pluggable Architecture" for extractors aligns perfectly with this mandate, in contrast to competitors who rely on proprietary file formats.[1]
3. **AI/ML Integration (Project Linchpin/Maven):** "Data readiness" is widely recognized as the primary bottleneck in military AI adoption. SigDrive's ability to normalize disparate headers into a canonical schema effectively solves the "data wrangling" problem, positioning it as an ideal "Feature Store" for trusted AI pipelines.[1]

2. Regulatory Compliance and Security Frameworks

Deploying software on SIPRNet or JWICS requires more than just functional code; it requires an Authority to Operate (ATO). The ATO is a formal declaration by a Designated Authorizing Official (DAO) that the system's security posture is acceptable, explicitly accepting the risk to agency operations.[1] For SigDrive, achieving an ATO involves strict adherence to federal standards and a comprehensive strategy for risk mitigation.

2.1 ICD 503 and the Risk Management Framework (RMF)

For Intelligence Community networks (JWICS), the guiding standard is Intelligence Community Directive (ICD) 503, "Intelligence Community Information Technology Systems Security Risk Management".[4] This directive aligns with the NIST Risk Management Framework (RMF) but often imposes stricter controls regarding continuous monitoring, supply chain risk management, and the protection of sources and methods.

Key RMF Steps for SigDrive Deployment:

1. **Categorize System:** Determine the impact level (Low, Moderate, High) of the data. RF data containing specific emitter signatures is often categorized as High Impact for Confidentiality due to the potential to reveal sensitive collection capabilities or locations.
2. **Select Controls:** Implement security controls from NIST SP 800-53 (Revision 5). SigDrive specifically addresses controls in the Audit and Accountability (AU), Access Control (AC), and System and Communications Protection (SC) families.[7]
3. **Implement Controls:** Configuration of the software to meet Security Technical Implementation Guides (STIGs). For example, the PostgreSQL database backing SigDrive must be configured according to the DISA STIG for PostgreSQL, and the underlying Linux OS must be hardened.[8]
4. **Assess Controls:** Validation by a Security Control Assessor (SCA). This often involves automated scanning (ACAS) and manual review of the System Security Plan (SSP).
5. **Authorize:** The Authorizing Official (AO) reviews the Security Assessment Report (SAR) and signs the ATO.
6. **Monitor:** Continuous monitoring of logs and configuration drift to ensure the system remains compliant over time.[10]

2.2 NIST SP 800-53 Control Mapping

SigDrive's architecture is designed to map directly to critical NIST controls required for air-gapped systems.[1]

Control Family	ID	Control Name	SigDrive Implementation Strategy
Audit & Accountability	AU-2	Event Logging	The Audit_logs table and Audit Service capture every metadata change, file access, and login attempt with immutable timestamps. Logs track the "who, what, where, and when" of every interaction.[1]
Access Control	AC-3	Access Enforcement	A robust Role-Based Access Control (RBAC) engine restricts visibility of RF recordings based on user clearance and "Need-to-Know" groups.[1]
Identification & Auth	IA-2	Identification & Authentication	Support for PKI/CAC/PIV authentication via mTLS and integration with disconnected Identity Providers (Keycloak), supporting multi-factor authentication (MFA) requirements.[1]
System Protection	SC-18	Mobile Code	Restriction of executed code; the frontend is a compiled React application served from a local Nginx container, with no external CDN references or unauthorized script execution.[13]
Config Management	CM-6	Configuration Settings	"STIG Mode" enables strict password policies, session timeouts, and disables unneeded API endpoints automatically to match DISA benchmarks.[1]
System Integrity	SI-4	Information System Monitoring	Integration with host-based intrusion detection and centralized logging mechanisms to detect unauthorized data exfiltration or modification.[1]

Table 1: NIST Control Implementation Matrix

2.3 The "Iron Bank" and Container Hardening

To expedite the ATO process on DoD networks, software should be delivered as hardened containers. The DoD's "Iron Bank" (Repo One) is a centralized repository of digitally signed, binary-hardened container images that have been scanned for vulnerabilities.[15] By utilizing Iron Bank base images, SigDrive inherits a trusted security baseline, significantly reducing the burden of vulnerability management.

SigDrive Hardening Pipeline:

1. **Base Image Selection:** All SigDrive microservices (Ingest, API, Database) are built on top of the Iron Bank ubi8 (Universal Base Image 8) or distroless images rather than standard Alpine or Debian images. This ensures the underlying OS layer is compliant with DoD standards.[16]
2. **Dependency Scanning:** During the build process (which occurs on a connected network prior to transfer), all Python (pip) and Node.js (npm) dependencies are pinned to specific hashes and scanned against CVE databases using tools like Twistlock or Anchore.[16]
3. **Justification:** Any findings (vulnerabilities) that cannot be remediated must be formally justified with a Plan of Action and Milestones (POAM) to the Authorizing Official.[18] This transparency is critical for risk acceptance.
4. **Air-Gap Transfer:** The final hardened images are saved as .tar archives and transferred to the classified network, ensuring that the software running on SIPRNet is bit-for-bit identical to the scanned version.[19]

3. High-Level System Architecture

SigDrive operates as a distributed system comprised of several microservices. In an air-gapped environment, these services must be orchestrated locally without reliance on external APIs. The architecture prioritizes modularity, scalability, and fault tolerance, leveraging a containerized approach for consistent deployment across diverse hardware environments.

3.1 Core Components

The system is composed of the following key services, each containerized and managed via orchestration tools:

- **API Gateway (Nginx/Traefik):** Handles TLS termination (FIPS 140-2 compliant) and routes traffic to backend services. In air-gapped modes, it serves the static frontend assets (React) directly, removing dependencies on public CDNs.[20] It acts as the single entry point for all client requests, enforcing strict traffic rules.
- **Backend API (FastAPI/Python):** The core logic engine. It manages metadata extraction (SigMF), search queries, and user authentication. It is built using Iron Bank hardened Python images [21] and implements the business logic for data management and analysis.

- **Metadata Store (PostgreSQL + PostGIS):** Relational database storing user profiles, RBAC policies, and geospatial indexes of RF recordings. It is hardened according to the Crunchy Data STIG [8] and serves as the source of truth for all structured data.
- **Object Storage (MinIO):** A high-performance, S3-compatible object store optimized for large binary files (IQ recordings). It replaces AWS S3 in the disconnected environment [22], providing scalable and durable storage for the raw signal data.
- **Message Broker (Redis/RabbitMQ):** Manages asynchronous tasks such as file ingestion, transcoding, and spectrogram generation. It decouples the ingestion process from the user interface, ensuring responsiveness even during heavy load.
- **User Interface (React):** A modern, single-page application (SPA) providing visualization, search, and administrative capabilities. It is built with offline-first principles, bundling all necessary assets.

3.2 The "Air-Gap Ready" Design Philosophy

Traditional software often fails in air-gapped environments due to hidden dependencies, a font loader trying to reach Google Fonts, a library checking a revocation list online, or an analytics script pinging a remote server. SigDrive architecture mitigates this via a rigorous "Air-Gap First" design philosophy.[1]

Architectural Adaptations for Disconnected Operations:

- **Vendor Bundling:** All frontend assets, including fonts (Roboto, Material Icons) and mapping libraries (OpenLayers/Leaflet), are bundled directly into the Nginx container.[23] The build process explicitly verifies that no external URLs are referenced in the source code.
- **Local Licensing:** Licensing is handled via cryptographically signed JWTs (JSON Web Tokens) or hardware dongles, eliminating the need for a license server connection.[3] The system verifies the license signature against a locally stored public key.
- **Offline Maps:** The geospatial interface connects to a local tile server (e.g., Martin or TileServer-GL) hosting NGA-standard vector tiles (MBTiles/GeoPackage), rather than calling out to Google Maps or OpenStreetMap.[25]
- **Self-Contained Documentation:** All user manuals and API documentation are embedded within the application, accessible without internet access.

4. Infrastructure & Platform: Kubernetes and Zarf

While Docker Compose is suitable for small lab deployments [1], enterprise deployment on SIPRNet typically utilizes Kubernetes. However, installing and managing Kubernetes without the internet is notoriously difficult. To solve this, SigDrive leverages **Zarf** and the **Platform One "Big Bang"** ecosystem to simplify the delivery and operations of the platform.

4.1 Zarf: The Air-Gap Package Manager

Zarf is an open-source tool born out of the DoD's need to deploy software to "tactical edge" and disconnected environments.[27] It acts as a declarative package manager that bundles the "entire internet" needed for an application into a single compressed tarball. It solves the "chicken and egg" problem of needing a registry to pull images to start a registry.

The Zarf Workflow for SigDrive:

1. **Package Definition (zarf.yaml):** A manifest file defines the SigDrive application. It lists all Docker images, Helm charts, and raw files (e.g., database schemas, STIG config files) required.[29] It essentially describes the desired state of the cluster.
2. **Package Creation (Connected):** On a secure, internet-connected build machine, the zarf package create command is executed. Zarf pulls all images from the registry, downloads Helm charts, and compiles them into a single .zst compressed archive.[30]
    - *SBOM Generation:* Zarf automatically generates a Software Bill of Materials (SBOM) for the package, a critical requirement for DoD supply chain security.[28]
3. **Sneakernet Transfer:** The Zarf package (e.g., zarf-package-sigdrive-v1.0.tar.zst) is burned to a DVD or loaded onto a scanned hard drive and physically carried to the air-gapped facility.
4. **Package Deployment (Disconnected):** On the air-gapped server, the administrator runs zarf package deploy. Zarf spins up a temporary internal registry ("seed registry"), pushes the images, and installs the Helm charts into the cluster, all without any external network traffic.[29] It automatically rewrites the Helm chart image references to point to this new internal registry.

4.2 Platform One "Big Bang" Integration

For larger installations (e.g., a major JWICS data center), SigDrive can be deployed as an "addon" to Platform One's **Big Bang**. Big Bang is a standardized "Infrastructure as Code" (IaC) baseline that deploys a hardened Kubernetes cluster with pre-integrated security tools (Istio, monitoring, logging).[32]

Integration Benefits:

- **Inherited Security:** SigDrive fits into the Big Bang architecture as a "Customer Package." This ensures that it inherits the security posture of the underlying platform, including FIPS-validated encryption modules provided by the platform's service mesh.[12]
- **Standardized Tooling:** By leveraging Big Bang, SigDrive utilizes standard monitoring (Prometheus/Grafana) and logging (Elastic/Fluentd) stacks, simplifying operations for DoD administrators who are already familiar with these tools.
- **Continuous ATO:** Integration with Big Bang facilitates the concept of a Continuous ATO (cATO), where the platform's continuous monitoring capabilities allow for faster updates and feature releases.[34]

5. The Storage Layer: Solving the 5TB Problem

The defining characteristic of SigDrive is its ability to handle massive files. A single 5TB IQ recording cannot be uploaded via standard HTTP POST methods, especially on networks with high latency or aggressive timeout settings like SIPRNet. The storage layer must be robust, scalable, and capable of handling resumable transfers.

5.1 MinIO: The S3 of the Air-Gap

SigDrive utilizes **MinIO**, a high-performance, distributed object storage server that is API-compatible with Amazon S3.[22] MinIO is designed for private cloud infrastructure and excels in handling large binary objects. It allows SigDrive to offer an "S3-like" experience on-premises.

Architectural Configuration:

- **Erasure Coding:** In a distributed setup (e.g., 4 servers with 12 drives each), MinIO uses erasure coding to split objects into data and parity blocks. This allows the system to tolerate the loss of multiple drives or entire nodes without data loss, critical for tactical environments where hardware replacement logistics are slow.[35] For example, with an erasure code parity of 4, the system can lose up to 4 drives and still serve data.
- **Bitrot Protection:** MinIO silently detects and heals corrupted data using hashing algorithms, ensuring that the high-fidelity signal data recorded years ago remains bit-perfect when retrieved for analysis.[22] This is essential for archival integrity in defense applications.
- **Distributed Mode:** MinIO can be deployed in a distributed mode, aggregating storage from multiple nodes into a single namespace. This allows for horizontal scaling of capacity and performance as the volume of RF data grows.[36]

5.2 Multipart Uploads and Resumability

To robustly handle 5TB files, SigDrive implements the **TUS protocol** or S3 Multipart Uploads logic on the client side.[37] Standard file uploads are prone to failure over unstable connections; resumability is non-negotiable.

The Resumable Upload Workflow:

1. **Chunking:** The browser-based uploader splits the 5TB file into manageable chunks (e.g., 100MB).
2. **Parallel Uploads:** Chunks are uploaded in parallel to the MinIO backend to maximize network throughput.[39]
3. **State Tracking:** If the network connection drops (common in tactical scenarios), the client retains the state of the transfer. When connectivity is restored, the upload resumes from the last successful chunk rather than restarting the entire 5TB transfer.[1]
4. **Assembly:** Once all parts are received, MinIO concatenates them into a single object, verifying the checksum of the final artifact.
5. **TUS Integration:** The TUS protocol provides a standardized way to handle resumption, ensuring compatibility and reliability across different browser environments.[37]

5.3 FIPS 140-2 Compliance for Storage

For classified data, encryption at rest is mandatory. MinIO is configured to use **FIPS 140-2 validated cryptographic modules**.[40]

- **Server-Side Encryption (SSE):** All data written to disks is encrypted using AES-256-GCM.
- **Key Management:** MinIO integrates with an external Key Management Service (KMS) such as HashiCorp Vault (often available in Big Bang deployments) via the KES (Key Encryption Service) interface. In strictly isolated setups, MinIO can manage keys locally, provided the physical security of the server meets requirements.[40]
- **TLS Configuration:** All data in transit between MinIO nodes and clients is encrypted via TLS 1.2+, using FIPS-approved cipher suites to prevent interception.[42]

6. Database Management: PostgreSQL in the Secure Enclave

The metadata layer, containing frequency, time, location, and classification tags, is stored in PostgreSQL. This relational database serves as the backbone for the system's search and indexing capabilities.

6.1 High Availability (HA) with Patroni

In a disconnected environment, administrators cannot rely on cloud-managed database services (like AWS RDS) to handle failover and replication. SigDrive employs **Patroni**, a template for PostgreSQL High Availability.[43]

Patroni Architecture:

- **Cluster Management:** Patroni manages a cluster of PostgreSQL nodes (Primary and Replicas). It uses a distributed consensus store (like **etcd**, often bundled with Kubernetes) to elect a leader and maintain cluster state.
- **Automated Failover:** If the primary node fails, Patroni automatically promotes a replica to leader. This ensures the Data Lake remains accessible even during hardware failures, without requiring manual intervention from a DBA who might not be on-site.[44]
- **Self-Healing:** Patroni monitors the health of the database instances and can automatically restart failed nodes or reconfigure replication to restore redundancy.[43]

6.2 STIG Compliance (Crunchy Data)

To maintain the ATO, the database must comply with the DISA STIGs. SigDrive utilizes the **Crunchy Data PostgreSQL** distribution, which is pre-configured to meet these requirements.[8]

Key STIG Configurations:

- **Audit Logging:** pgaudit is enabled to log specific classes of statements (READ, WRITE, DDL) required by DoD policy. These logs are essential for forensic analysis and accountability.[46]
- **Connection Security:** Enforced TLS 1.2 with FIPS-compliant ciphers for all client connections.[11] The system rejects any non-encrypted connection attempts.
- **Session Termination:** Idle sessions are automatically terminated after a configurable period (e.g., 15 minutes) to prevent unauthorized access at unattended terminals.[47]
- **Access Controls:** Strict file permissions on the database data directory and configuration files ensure that only the database user can access sensitive information.[48]

7. Cross Domain Solutions (CDS) & Data Ingestion

A critical operational challenge is moving data from the point of collection (often unclassified or "Secret") to the point of analysis (often "Top Secret" or JWICS). This requires a Cross Domain Solution (CDS) to ensure the secure transfer of information across security boundaries.

7.1 Data Diodes and Unidirectional Gateways

The most secure method for bulk data transfer is a **Data Diode**, a hardware device that physically enforces one-way data flow.[49]

Data Diode Mechanics:

- **Low-to-High Transfer:** RF data collected on a drone (Unclassified) is plugged into a "Blue" (Low) server. The data diode optically transmits the data to a "Red" (High) server. The physics of the device prevent any light (and thus data) from traveling in the reverse direction, making it physically impossible for a high-side attacker to exfiltrate data to the low side.[2]
- **Protocol Adaptation:** Standard TCP/IP requires a two-way handshake (SYN/ACK), which diodes break. SigDrive utilizes specialized proxy software (e.g., **Owl Cyber Defense** or **Forcepoint High Speed Guard**) that wraps the file transfer in a unidirectional UDP stream (often using proprietary Forward Error Correction) to jump the gap.[51]
- **High Throughput:** These solutions are designed for speed, supporting transfer rates of 10Gbps or more, essential for moving terabyte-scale RF datasets.[53]

7.2 The "Drop Box" Ingestion Pattern

SigDrive implements a "Folder Watcher" pattern to integrate with CDS workflows, automating the ingestion of data once it arrives on the high side.[54]

Ingestion Workflow:

1. **Ingest Landing Zone:** The CDS deposits sanitized files into a specific directory on the SigDrive server (e.g., /mnt/ingest/cds_import).
2. **Watcher Service:** A Python daemon uses the watchdog library to monitor this directory.[55] It listens for file system events such as FileCreated or FileClosed.
3. **Stability Check:** To prevent reading a file while it is still being written, the service verifies the file size is stable before initiating processing.
4. **Automated Indexing:** When a new file (e.g., a 2TB .iq file and its .sigmf-meta sidecar) is ready, the service triggers the ingestion pipeline. It parses the metadata, indexes it in PostGIS, moves the binary to MinIO, and notifies analysts of the new asset.[1]
5. **Error Handling:** Corrupt or malformed files are moved to a quarantine directory, and an alert is generated for administrative review.

7.3 File Sanitization (Raise the Bar)

Under the NSA's "Raise the Bar" initiative, complex file formats must be sanitized before crossing domains. While raw binary IQ data is often considered "safe" due to its lack of executable structure, the metadata (JSON/XML) must be validated.[56]

Validation and Sanitization:

- **SigMF Validation:** SigDrive's ingest parser performs strict schema validation against the SigMF standard.[57] It checks for required fields, data types, and value ranges.
- **Input Sanitization:** Any text fields in the metadata are sanitized to remove potential script injection or control characters.

- **Rejection:** Any file failing validation is rejected at the ingest point, preventing potential exploits from entering the system. This ensures that the high-side environment remains protected from malformed or malicious data structures.[58]

8. Geospatial Intelligence: Offline Mapping

Analysts need to visualize *where* a signal was recorded to correlate it with other intelligence. In the cloud, this is done via Google Maps or Bing API calls. On SIPRNet, the map must be served locally.

8.1 NGA "Map of the World" and Vector Tiles

The National Geospatial-Intelligence Agency (NGA) provides foundation data for DoD systems. SigDrive supports the ingestion of NGA-standard map data, typically in **GeoPackage** or **MBTiles** formats.[59]

Vector Tiles vs. Raster:

- **Efficiency:** Unlike raster tiles (images), vector tiles (PBF format) contain the raw geometry and attributes of map features. They are significantly smaller than raster tilesets, reducing the storage footprint of the base map from terabytes to gigabytes, a crucial saving in resource-constrained environments.[61]
- **Styling:** Vector tiles allow for client-side styling. The map appearance (colors, labels, languages) can be changed dynamically without redownloading data.
- **Offline Standards:** MBTiles and GeoPackage are standard formats for storing tiled map data in a single file (SQLite database), making them easy to transport and deploy.[62]

8.2 Hosting the Tile Server

SigDrive includes a lightweight, containerized tile server, such as **Martin** (written in Rust) or **TileServer-GL**, to serve these maps.[25]

Tile Server Architecture:

- **Data Source:** The tile server reads directly from a local .mbtiles file (e.g., world_basemap.mbtiles) located on the server's filesystem.
- **Internal API:** It serves the vector tiles via an internal HTTP API (e.g., GET /v1/tiles/{z}/{x}/{y}.pbf).
- **Frontend Integration:** The React frontend uses **OpenLayers** or **MapLibre GL JS**.[64] These libraries are configured to point to the local tile server URL instead of a public endpoint.
- **Local Styling:** Map styles (defining colors, road widths, labels) are loaded from a local JSON file, ensuring the visual rendering complies with military standards (e.g., MIL-STD-2525 symbology) without external dependencies.[66]
- **Performance:** Martin is optimized for speed and low resource usage, making it ideal for tactical servers with limited CPU and RAM.[26]

9. Identity & Access Management (IAM)

Security on classified networks relies on strong identity verification, typically using Common Access Cards (CAC) or Personal Identity Verification (PIV) cards. The system must support these hardware tokens while operating completely offline.

9.1 Disconnected Identity Provider: Keycloak

SigDrive deploys **Keycloak** as a private, air-gapped Identity Provider (IdP).[12] Keycloak provides a comprehensive IAM solution that can function without external connectivity.

Authentication Workflow:

- **Federation:** Keycloak is configured to federate with the local Active Directory (AD) or LDAP server if available on the SIPRNet enclave.[1] This allows users to use their existing domain credentials.
- **PKI/Smart Card Auth:** Keycloak is configured to accept X.509 certificates presented by the user's browser (mTLS). It parses the Subject Alternative Name (SAN) from the CAC to extract the user's EDIPI (DoD ID number) and map it to a SigDrive user account.[1]
- **OCSP/CRL Handling:** In a connected world, the IdP checks the Certificate Revocation List (CRL) online to see if a certificate is valid. In an air-gap, the CRL must be manually updated (sneaker-net) or cached on the network. SigDrive's configuration allows for "soft-fail" or strictly local CRL checking to prevent lockout when the external CRL server is unreachable.[12]

9.2 Role-Based Access Control (RBAC)

SigDrive implements fine-grained RBAC to enforce "Least Privilege" and "Need-to-Know" principles.

RBAC Structure:

- **Global Roles:** Pre-defined roles such as Admin, Analyst, and Read-Only determine the high-level permissions of a user.[1]
- **Data Scoping:** Users can be restricted to specific "Mission Collections." For example, a Navy analyst may only have access to "Task Force 70" recordings, while an Army analyst sees "Project Linchpin" data. This enforcement happens at the API layer, filtering database queries based on the user's JWT claims.[1]
- **Attribute-Based Control:** Access can also be granted based on user attributes (e.g., clearance level, citizenship) extracted from their certificate or AD profile.[1]

9.3 User Lifecycle Management

Managing user accounts in a disconnected system requires robust tooling. SigDrive includes a **User Lifecycle Manager** service.[1]

Lifecycle Automation:

- **Provisioning:** The system can automatically create accounts based on AD group membership or SCIM pushes from a local HRIS.
- **Deprovisioning:** To meet security SLAs, accounts are deactivated within 15 minutes of a termination event (detected via AD update or manual trigger).[1]
- **Asset Transfer:** Upon deactivation, the system prompts administrators to transfer ownership of the user's recordings and collections to a designated successor to prevent data loss.[1]

10. Operational Scenarios and Workflows

Understanding how SigDrive operates in distinct environments illustrates the flexibility of the architecture.

10.1 Scenario A: The Tactical Edge (Submarine/FOB)

Environment: Highly constrained, intermittent power, zero connectivity.

- **Hardware:** Single-node robust server (e.g., Dell XR11 or AWS Snowball Edge).
- **Deployment:** zarf package deploy directly from a USB drive.
- **Data Flow:** Local sensors (SDRs) write directly to the MinIO ingress bucket via 10GbE LAN.
- **Operations:** Analysts work locally on the LAN. The map tile server runs locally on the node.
- **Sync:** When the unit returns to port/base, a "Data Mule" drive is exported from MinIO containing the mission data, which is then physically transported to the national archive.

10.2 Scenario B: The Enterprise Data Center (JWICS)

Environment: Stable power, high-speed internal network, massive storage SAN.

- **Hardware:** Kubernetes Cluster (OpenShift/RKE2) with robust SAN storage.
- **Deployment:** Platform One Big Bang Helm release.
- **Data Flow:** Automated ingestion from satellite downlinks via high-speed Cross Domain Solutions (Data Diodes).
- **Operations:** Multiple analysts connect via VDI (Virtual Desktop Infrastructure) workstations. They perform geospatial searches and launch AI/ML training jobs on the accumulated data lake.
- **Scale:** The system scales horizontally, adding more MinIO and PostgreSQL nodes to handle petabytes of data.

11. Conclusion: Achieving Data Dominance

Deploying modern data infrastructure in air-gapped environments is a formidable engineering challenge, requiring a departure from convenient cloud-native assumptions. It demands a rigorous focus on dependency management, supply chain security, and compliant architecture. The "Data Gravity" of modern RF operations necessitates that the analytics move to the data, even if that data resides in a bunker or on a ship.

SigDrive's approach, leveraging the Zarf packaging ecosystem, Iron Bank hardened containers, and a flexible microservices architecture, provides a viable path for the DoD to modernize its RF operations. By solving the "Data Gravity" problem through optimized local storage (MinIO) and ensuring "Data Readiness" via SigMF standardization, SigDrive transforms isolated islands of data into a connected, queryable, and actionable enterprise asset.

This architecture not only meets the current requirements of ICD 503 and NIST 800-53 but prepares the defense enterprise for the future of JADC2, where data must move securely and seamlessly from the tactical edge to the strategic core, regardless of the network environment. By enabling advanced analytics and AI in disconnected environments, SigDrive ensures that the U.S. and its allies maintain electromagnetic dominance in an increasingly contested spectrum.

List of Abbreviations

- **ATO:** Authority to Operate
- **CDS:** Cross Domain Solution
- **FIPS:** Federal Information Processing Standards
- **ICD:** Intelligence Community Directive
- **JADC2:** Joint All-Domain Command and Control
- **MOSA:** Modular Open Systems Approach
- **NGA:** National Geospatial-Intelligence Agency
- **RMF:** Risk Management Framework
- **SigMF:** Signal Metadata Format
- **SIPRNet:** Secret Internet Protocol Router Network
- **STIG:** Security Technical Implementation Guide
- **JWICS:** Joint Worldwide Intelligence Communications System
- **SDR:** Software Defined Radio
- **RBAC:** Role-Based Access Control
- **PKI:** Public Key Infrastructure
- **CRL:** Certificate Revocation List

References

1. Internal documentation, available on request
2. Air Gap Security: A Complete Guide, https://www.micromindercs.com/blog/air-gap-security
3. keygen-sh/air-gapped-activation-example - GitHub, https://github.com/keygen-sh/air-gapped-activation-example
4. DoDM 5205.07, "Special Access Program Security Manual," January 17, 2025 - Executive Services Directorate, https://www.esd.whs.mil/Portals/54/Documents/DD/issuances/dodm/520507m1.PDF?ver=o_3_m4lDAtLKPOLQHatcYA%3D%3D
5. Protect National Interests With Secure Government Data Exchange - Kiteworks, https://www.kiteworks.com/solutions/government/
6. Technical Specifications for Construction and Management of Sensitive Compartmented Information Facilities - DNI.gov, https://www.dni.gov/files/Governance/IC-Tech-Specs-for-Const-and-Mgmt-of-SCIFs-v15.pdf
7. Security and Privacy Controls for Information Systems and Organizations - NIST Technical Series Publications, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf
8. Secure Technical Implementation Guidance - STIG - for Postgres - Crunchy Data, https://www.crunchydata.com/solutions/postgres-stig
9. Security Technical Implementation Guides (STIGs) - Cyber Exchange, https://www.cyber.mil/stigs
10. NIST Special Publication (SP) 800-53 Revision 4 Title: Security and Privacy Controls for Federal Information, https://csrc.nist.gov/files/pubs/sp/800/53/r4/final/docs/sp800-53-rev4-ipd.pdf
11. The Automation Controller servers must use encrypted communication for all channels given the high impact of those services to an organization's infrastructure. - STIG Viewer, https://www.stigviewer.com/stigs/red_hat_ansible_automation_controller_web_server/2024-08-27/finding/V-256941
12. DevSecOps Enterprise Container Hardening Guide - DoD Cyber Exchange, https://dl.dod.cyber.mil/wp-content/uploads/devsecops/pdf/Final_DevSecOps_Enterprise_Container_Hardening_Guide_1.2.pdf
13. SC-18: Mobile Code - CSF Tools, https://csf.tools/reference/nist-sp-800-53/r4/sc/sc-18/
14. Training Agenda 2025 - Alamo Chapter of the Armed Forces Communications and Electronics Association, https://www.alamoafcea.org/mpage/training2025agenda
15. Iron Bank - Platform One, https://p1.dso.mil/ironbank
16. README.md - Iron Bank Containers / dccscr, https://repo1.dso.mil/dsop/dccscr/-/blob/b88a0be6ea3ef84c87709e8e74cf9e9fda90fea4/README.md
17. Iron Bank Containers / dccscr · GitLab - Repo One, https://repo1.dso.mil/dsop/dccscr/-/tree/python-hardening-guide

18. Securing Kubernetes Clusters in Federal Government Environments: A Technical Guide - AlphaBravo Engineering Blog, https://blog.alphabravo.io/securing-k8s-federal-gov-tech-guide/
19. Advanced Battle Management System: Needs, Progress, Challenges, and Opportunities Facing the Department of the Air Force, https://www.fie.undef.edu.ar/ceptm/wp-content/uploads/2022/06/ADVANCED-BATTLE-MANAGEMENT-SYSTEM-US-AIRFORCE.pdf
20. Guidelines for Deploying React - Max Rozen, https://maxrozen.com/guidelines-for-deploying-react
21. Security - FastAPI - Tiangolo, https://fastapi.tiangolo.com/tutorial/security/
22. Airgapped MinIO Deployments, https://blog.min.io/airgapped-minio-deployments/
23. How to host material icons offline? - Stack Overflow, https://stackoverflow.com/questions/37270835/how-to-host-material-icons-offline
24. Use Openlayers in Offline mode for land-survey applications! | by Krishna G. Lodha, https://medium.com/random-gis-talks/use-openlayers-offline-for-survey-4ded3998d9cc
25. Serve maps with TileServer GL - OpenMapTiles, https://openmaptiles.org/docs/host/tileserver-gl/
26. maplibre/martin: Blazing fast and lightweight PostGIS, MBtiles and PMtiles tile server, tile generation, and mbtiles tooling. - GitHub, https://github.com/maplibre/martin
27. Zarf: airplane mode for your application delivery, https://zarf.dev/
28. zarf-dev/zarf: The Airgap Native Packager Manager for Kubernetes - GitHub, https://github.com/zarf-dev/zarf
29. Deploy a Package | Zarf, https://docs.zarf.dev/ref/deploy/
30. Deploy and Update Zarf Packages in an Air Gap | by Brandi McCall | Medium, https://medium.com/@bm54cloud/deploy-and-update-zarf-packages-in-an-air-gap-b2e3ec43abf7
31. The 'init' Package | Zarf, https://docs.zarf.dev/ref/init-package/
32. airgap_quickstart.md · main · Big Bang / Customers / air-gap-deployment - Repo One, https://repo1.dso.mil/big-bang/apps/sandbox/air-gap-deployment/-/blob/main/airgap_quickstart.md?ref_type=heads
33. Headquarters U.S. Air Force How did the Department of Defense move to Kubernetes and Istio?, https://csrc.nist.gov/CSRC/media/Presentations/dod-enterprise-devsecops-initiative/images-media/DoD%20Enterprise%20DevSecOps%20Initiative%20%20v2.5.pdf
34. DevSecOps Continuous Authorization Implementation Guide - DoD CIO, https://dodcio.defense.gov/Portals/0/Documents/Library/DoDCIO-ContinuousAuthorizationImplementationGuide.pdf
35. CI/CD Deploy with MinIO distributed cluster on Kubernetes, https://blog.min.io/ci-cd-distributed-cluster-kubernetes/
36. How to Install MinIO in Distributed Mode on AWS EC2, https://blog.min.io/install-minio-distributed-mode-aws-ec2/

37. tus/tusd - the open protocol for resumable file uploads - GitHub, https://github.com/tus/tusd
38. Uploading and copying objects using multipart upload in Amazon S3 - AWS Documentation, https://docs.aws.amazon.com/AmazonS3/latest/userguide/mpuoverview.html
39. Performance design patterns for Amazon S3 - Amazon Simple Storage Service, https://docs.aws.amazon.com/AmazonS3/latest/userguide/optimizing-performance-design-patterns.html
40. MinIO AIStor Object Store Key Management Server, https://www.min.io/product/aistor/key-management-server
41. Enable FIPS Mode | AIStor Object Store Documentation - MinIO, https://docs.min.io/enterprise/aistor-object-store/installation/kubernetes/fips-mode/
42. MinIO Network Encryption - It's All Good, https://blog.min.io/tls-good/
43. PostgreSQL High Availability in Action | by Mehman Jafarov | Nov, 2025 - Medium, https://medium.com/@mehmanjafarov1905/postgresql-high-availability-in-action-49aadf181549
44. Achieving PostgreSQL High Availability: Strategies, Tools, and Best Practices - pgEdge, https://www.pgedge.com/blog/postgresql-high-availability-strategies-tools-best-practice
45. Checklist Crunchy Data Postgres 16 STIG - NCP, https://ncp.nist.gov/checklist/1246
46. Crunchy Data Postgres 16 Security Technical Implementation Guide - STIG Viewer, https://www.stigviewer.com/stigs/crunchy_data_postgres_16
47. PostgreSQL must automatically terminate a user session after organization-defined conditions or trigger events requiring session disconnect. - STIG Viewer, https://stigviewer.com/stigs/crunchy_data_postgresql/2024-08-27/finding/V-233613
48. Crunchy Data Postgres 16 Security Technical Implementation Guide Version: 1 Release, https://www.crunchydata.com/files/stig/PGSQL_16_STIG_V1R1.pdf
49. FAQ For Cross Domain Solutions - General Dynamics Mission Systems, https://gdmissionsystems.com/cross-domain-solutions/faqs
50. Fundamentals of Cross Domain Solutions - Australian Cyber Security Centre, https://www.cyber.gov.au/sites/default/files/2025-03/Fundamentals%20of%20Cross%20Domain%20Solutions%20%28October%202021%29.pdf
51. Cross Domain Solutions - Owl Cyber Defense, https://owlcyberdefense.com/wp-content/uploads/2019/05/owlcyberdefense-brochure_cross-domain-solutions.pdf
52. High Speed Guard - Military Expos, https://www.militaryexpos.com/wp-content/uploads/2021/10/datasheet_forcepoint_high_speed_guard_en.pdf
53. Cross Domain Solutions: Data diode solution™ - BAE Systems, https://www.baesystems.com/en/product/data-diode-solution
54. netboxlabs/diode-sdk-python - GitHub, https://github.com/netboxlabs/diode-sdk-python
55. Master Watchdog: Real-Time File & Folder Monitoring in Python - YouTube, https://www.youtube.com/watch?v=T4xLPnR7W6s

56. National Cross Domain Strategy & Management Office, https://www.nsa.gov/Cybersecurity/Partnership/National-Cross-Domain-Strategy-Management-Office/
57. sigmf/SigMF: The Signal Metadata Format Specification - GitHub, https://github.com/sigmf/SigMF
58. Security principles for cross domain solutions - NCSC.GOV.UK, https://www.ncsc.gov.uk/collection/cross-domain-solutions
59. NGA Products & Services - National Geospatial-Intelligence Agency, https://www.nga.mil/resources/Products_&_Services.html
60. Output formats: GeoPackage, MBTiles, folder | Guides | Map tiling hosting | Data processing, https://docs.maptiler.com/guides/map-tiling-hosting/data-processing/folder-vs-mbtiles-vs-geopackage/
61. OSM offline tiles -- mbtiles in openlayers - Geographic Information Systems Stack Exchange, https://gis.stackexchange.com/questions/202119/osm-offline-tiles-mbtiles-in-openlayers
62. OGC GeoPackage, https://www.geopackage.org/
63. Mbtiles vs. geopackage for a simple offline vector tile server : r/gis - Reddit, https://www.reddit.com/r/gis/comments/1ppbi6w/mbtiles_vs_geopackage_for_a_simple_offline_vector/
64. OpenLayers Examples, https://openlayers.org/en/latest/examples/
65. Offline hosting using openlayers - General talk - OpenStreetMap Community Forum, https://community.openstreetmap.org/t/offline-hosting-using-openlayers/93353
66. OGC Testbed-14: Symbology Engineering Report, https://docs.ogc.org/per/18-029.pdf
67. Unicorn Delivery Service, https://uds.defenseunicorns.com/