# ARCHITECTURAL REPORT

## Noaaport Broadcast System Processor (Nbsp)

Repository: bitbucket.org/noaaport/nbsp
Mirror: github.com/noaaport/nbsp
Current Version: nbsp-2.3.9r

March 2026

# Table of Contents

# 1. Executive Summary

The Noaaport Broadcast System Processor (Nbsp) is a mature, open-source software suite designed to receive, decode, process, and distribute data from NOAA's Satellite Broadcast Network (SBN/NOAAPort). The NOAAPort data stream delivers real-time weather data—including satellite imagery, radar data, model output, text bulletins, and sensor data—via C-band satellite broadcast from the Galaxy 28 satellite.

Nbsp's architecture follows a pipeline model: a high-performance C core handles real-time packet reception and file reassembly, while a comprehensive Tcl-based scripting layer provides extensible post-processing through a modular filter system. The system supports standalone operation and distributed master-slave topologies, with output capabilities spanning HTTP, FTP, NNTP, RSS, LDM queue insertion, and UDP-based product arrival notifications.

The codebase is composed of approximately 33.9% Tcl, 33.1% C, 16.4% HTML, and smaller portions of M4, Makefile, and Shell scripts, reflecting the separation between the performance-critical core and the flexible scripting layer. The project has accumulated over 1,487 commits across two contributors and is actively maintained, with the latest updates (January 2025) adding native GOES-R satellite image processing capabilities.

# 2. System Context and Domain

## 2.1 The NOAAPort Data Stream

NOAAPort is a one-way satellite broadcast system operated by the National Weather Service. Data is uplinked to the Galaxy 28 satellite at 89°W and broadcast over multiple logical channels via DVB-S2 modulation. Each channel carries a multicast UDP stream on a dedicated IP address (224.0.1.1 through 224.0.1.10, plus 224.1.1.1), received by consumer-grade DVB-S2 receivers such as the Novra S300N.

The broadcast carries weather products across several channels: NCEP/NWSTG (text bulletins, model data), GOES (satellite imagery), NCEP/NWSTG2 (additional model output), Non-GOES Imagery/DCP Data, and expanded channels added during the bandwidth expansion from 30 Mbps to over 60 Mbps in 2014. The data arrives as fragmented UDP packets that must be reassembled into complete product files before further processing.

## 2.2 Nbsp's Role in the Ecosystem

Nbsp sits between the raw satellite receiver hardware and downstream consumers of weather data. It serves as the central data ingest and distribution hub, replacing or complementing alternatives such as Unidata's LDM (Local Data Manager). Where LDM focuses on peer-to-peer data relay among institutional users, Nbsp provides a self-contained processing pipeline with built-in web serving, format conversion, and multi-protocol distribution—making it particularly suited for independent operators and smaller institutions.

## 2.3 Supported Platforms

The system is developed primarily on FreeBSD and packages exist for FreeBSD, Almalinux, Debian, and Ubuntu Linux distributions. It has also been run under Cygwin on Windows. Platform-specific configuration files (e.g., nbspd.conf-linux, nbspd.conf-freebsd) handle OS differences in paths and network interface naming.

# 3. High-Level Architecture

## 3.1 Architectural Overview

Nbsp follows a layered pipeline architecture with three primary tiers:

| Tier | Language | Responsibility |
|---|---|---|
| **Reception Core** | C | Multicast UDP reception, packet reassembly, frame decoding, spool management |
| **Filter Pipeline** | Tcl | Post-processing, format conversion, file organization, image generation, distribution |
| **Service Layer** | Tcl / C | Web server, network server (EMWIN/NBS), NNTP gateway, RSS feeds, PAN notifications |

## 3.2 Data Flow Pipeline

The data flows through the system in the following sequence: (1) The DVB-S2 receiver demodulates the satellite signal and outputs multicast UDP packets onto the local network. (2) The nbspd daemon joins the configured multicast groups and receives UDP fragments. (3) The C core reassembles fragments into complete product files, decoding WMO headers and AWIPS identifiers. (4) Complete files are written to the spool directory (/var/noaaport/nbsp/spool). (5) File metadata is dispatched to the filter pipeline. (6) Enabled filters process files in parallel—converting formats, organizing into directory trees, generating images, and distributing to external systems. (7) The built-in web server and optional network servers make processed data available to clients.

## 3.3 Deployment Topologies

Nbsp supports three operational modes:

- **Standalone:** A single instance handles both reception and processing on one machine. This is the default and simplest configuration.
- **Master-Slave:** The master instance receives the broadcast and forwards reassembled files to one or more slave instances via the NBS1 protocol. Slaves handle all post-processing, offloading the reception machine.
- **Chained:** A slave can itself act as a master to additional downstream slaves, creating a hierarchical distribution tree for large-scale deployments.

# 4. Repository Structure

The repository is organized into well-defined directories reflecting the layered architecture:

| Directory | Purpose |
|---|---|
| **src/** | C source code for the core daemon (nbspd), packet processing, spool database management, and network protocols |
| **filters/** | Standard filter implementations (rstfilter, dafilter, gribfilter, metarfilter, etc.) in Tcl, each in its own subdirectory |
| **conf/** | Platform-specific configuration files (nbspd.conf-linux, nbspd.conf-freebsd) and feature toggles (features.conf) |
| **scripts/** | Operational shell scripts for cleanup, maintenance, log rotation, and system monitoring |
| **tclhttpd/** | Embedded web server based on TclHttpd 3.5.1, providing the monitoring UI and file serving |
| **tclgempak/** | Tcl bindings for GemPak weather data processing libraries |
| **tclgrads/** | Tcl bindings for GrADS (Grid Analysis and Display System) integration |
| **tclmetar/** | Tcl library for parsing and processing METAR aviation weather observations |
| **tclupperair/** | Tcl library for upper air sounding data processing |
| **tclldm/** | Tcl interface to Unidata's LDM for product queue insertion (nbsp2ldm) |
| **tclssh/** | Tcl SSH library for secure remote file distribution |
| **build/** | Build system scripts, package generation for multiple platforms (RPM, DEB, TBZ) |
| **tools/** | Utility programs for data inspection, testing, and debugging |
| **utils/** | General-purpose utility functions and helper programs |
| **doc/** | Documentation, man pages, and reference materials |
| **examples/** | Sample configurations, client integration examples (GRLevel3, Digital Atmosphere, IDV) |
| **QUICK_START/** | Getting-started guides for new installations |
| **README**s/** | Platform-specific README files and installation notes |
| **dev-notes/** | Developer documentation including a 6,497-line data dictionary |
| **debug-notes/** | Debugging guides and troubleshooting procedures |

# 5. Core Daemon Architecture (nbspd)

## 5.1 Packet Reception and Reassembly

The nbspd daemon is the central process written in C for maximum performance. It opens UDP multicast sockets on the configured channels (defaulting to all four primary channels: 224.0.1.1 through 224.0.1.4 on ports 1201–1204) and continuously reads incoming data fragments. Since UDP provides no retransmission mechanism, the reception loop must keep up with the incoming data rate without being swapped out by the OS. The daemon uses careful buffer management and non-blocking I/O to maintain real-time throughput.

Fragment reassembly is handled by a spool database that tracks partial files. Each incoming fragment is identified by its sequence number and WMO product header. When all fragments for a product are received, the complete file is written to the spool directory and its metadata is dispatched to the filter pipeline. The spool database can be managed automatically by nbsp (when the spooldbslots parameter is non-zero) or externally via cleanup scripts.

## 5.2 Configuration System

Configuration uses Tcl as its language, allowing full scripting capabilities in configuration files. The primary configuration file is /usr/local/etc/nbsp/nbspd.conf, with supporting files including features.conf for toggling major subsystems and individual filter configuration files (e.g., rstfilter.conf, dafilter.conf). All options have compiled-in defaults, so the system can run without any configuration file present. Key configurable parameters include: multicast group addresses and ports, network interface selection (by name or IP), spool directory location, server type and port, and filter enable/disable flags.

## 5.3 Network Server Module

When the servertype variable is set to a non-zero value, the daemon activates a network server that transmits files to connected clients. Server type 4 enables EMWIN-compatible QBT (Quick Block Transfer “byte blaster”) format, which distributes text bulletins as plain text and imagery as JPG/GIF files. The NBS1 protocol is used for master-slave communication. The listening port defaults to 1000 but is configurable via the serverport variable.

# 6. Filter Pipeline Architecture

The filter system is the primary extension mechanism of Nbsp. When a complete file is saved to the spool directory, the core daemon sends metadata (WMO header, AWIPS ID, file path, product type) to all enabled filters. Filters can be written in any language, though the standard filters are implemented in Tcl using a rich set of shared libraries.

## 6.1 Standard Filters

| Filter | Description |
|---|---|
| **rstfilter** | The primary post-processing filter. Converts satellite images to PNG/JPG, saves text files as plain text, and optionally generates radar GIF images. Organizes output into a browsable directory tree under /var/noaaport/data/nbsp with txt/, sat/, and rad/ subdirectories. Supports loop sequence generation for animated imagery. |
| **dafilter** | Saves files in a format compatible with Digital Atmosphere (DA) for Windows-based weather analysis. Output goes to /var/noaaport/data/digatmos in a naming convention DA can import directly. Also supports NNTP distribution of processed files. |
| **gpfilter** | The GemPak compatibility filter. Processes files into the directory structure and naming format expected by GemPak viewers and decoders. Requires the GemPak software package for full radar and annotated satellite image generation. |
| **gribfilter** | Processes GRIB (Gridded Binary) data files, which contain numerical weather prediction model output. Organizes GRIB files and makes them accessible for downstream analysis tools. |
| **metarfilter** | Parses and processes METAR (aviation routine weather reports) observations, extracting structured data from the raw text bulletins for programmatic access. |
| **panfilter** | Implements the Product Arrival Notification (PAN) system. Sends UDP messages to configured client machines when specific products arrive, enabling event-driven architectures. |
| **ldmfilter** | Inserts received products into a Unidata LDM product queue using the nbsp2ldm bridge program, enabling integration with LDM-based processing pipelines. |
| **nntpfilter** | Distributes raw data files via NNTP (news protocol) through an INN news server gateway, allowing news reader clients to browse weather products. |
| **rssfilter** | Generates RSS feeds for text products, accessible through the built-in web server. Template-based for customizable feed formats. |
| **trackfilter** | Tracks and catalogs received products for monitoring and statistical purposes. |
| **msgfilter** | Handles message-type products and special notifications. |
| **nbspfilter** | Core utility filter providing shared functionality used by other filters. |

## 6.2 User-Defined Filters

Beyond the standard filters, Nbsp provides a well-defined interface for custom user filters. These can be written in any language and receive the same product metadata as standard

filters. User filters are installed in the configuration directory and enabled through the filterlist variable. The system documents both the installation procedure and the API for writing custom filters, including the metadata fields available and the execution model.

## 6.3 Filter Configuration Pattern

Each filter follows a consistent configuration pattern: an .rc file (rule configuration) that defines matching rules and processing actions, an optional .conf file for parameter overrides, and supporting Tcl library files. Filters are enabled either through the features.conf master toggle or by including them in the filterlist. This pattern makes it straightforward to enable, configure, and extend any filter independently.

# 7. GIS and Satellite Image Processing

A significant recent architectural addition is the native GIS processing capability, eliminating the previous dependency on external applications like GemPak for image generation.

## 7.1 Nbsp GIS Libraries

Two companion libraries handle GIS transformations. The NbspGisLib (nbspgislib) is a C library and set of programs for converting NOAAPort radar (NIDS) and satellite (GINI) files to GIS-compatible formats including shapefiles. The NbspGisLibMap (nbspgislibmap) extends this with map overlay and rendering capabilities. Together, these libraries power the nbspgoesr and nbspgoesrinfo programs that produce images directly from GOES-R data files without any external dependencies.

## 7.2 GOES-R Processing Pipeline

As of version 2.3.9r (January 2025), Nbsp can process GOES-R satellite data files automatically through the rstfilter and dafilter. The nbspgoesr program decodes GOES-R data products, while nbspgoesrinfo extracts metadata. The processing is integrated into the existing filter pipeline, so enabling satellite image generation requires no special configuration beyond the standard filter setup. This represents a major architectural improvement, as previous versions required external GemPak decoders for full image processing.

# 8. Web Server and Monitoring Interface

Nbsp includes an embedded web server based on TclHttpd 3.5.1, listening on port 8015 by default. The web interface serves a dual purpose: monitoring the system's internal state and serving processed weather data products to end users.

The monitoring interface provides:

- Real-time statistics about received files, processing rates, and error counts
- Browsable directory trees of processed text bulletins, satellite images, and radar products
- RSS feed access for text product subscriptions
- Device monitoring integration via Npstats for tracking signal quality across multiple receivers
- Internal state reporting on spool utilization, filter performance, and connection status

# 9. Technology Stack Summary

| Component | Technology Details |
|---|---|
| **Core Language** | C (33.1% of codebase) — used for the nbspd daemon, packet processing, spool management, and performance-critical GIS libraries |
| **Scripting Language** | Tcl (33.9% of codebase) — all filters, configuration files, libraries (tclmetar, tclgempak, tclgrads, tclupperair, tclldm, tclssh), and web server |
| **Web Interface** | HTML (16.4%) with TclHttpd 3.5.1 embedded server |
| **Build System** | M4 macros (4.6%) with Makefile templates (3.3%) and configure.sh; platform-specific package generation (RPM, DEB, TBZ) |
| **Automation** | Shell scripts (3.0%) for system operations, cleanup, log rotation, and monitoring |
| **Configuration** | Tcl-based configuration files with compiled-in defaults; features.conf for subsystem toggles |
| **Data Formats** | WMO bulletins, GINI satellite, NIDS radar, GRIB model data, GOES-R, METAR, upper air soundings |
| **Output Formats** | PNG, JPG, GIF imagery; plain text; shapefiles; GemPak format; Digital Atmosphere format; GRIB |
| **Protocols** | UDP multicast (reception), NBS1 (master-slave), QBT/EMWIN (byte blaster), HTTP, FTP, NNTP, RSS, LDM, UDP PAN |
| **Platforms** | FreeBSD (primary development), Almalinux, Debian, Ubuntu; Cygwin (limited) |

# 10. External Integrations and Client Support

Nbsp's architecture is designed for extensive interoperability with downstream weather analysis software:

- **GemPak:** The gpfilter organizes data into GemPak's expected directory structure and file format. With the gpak package, GemPak decoders and viewers work directly against the Nbsp data directory.
- **Digital Atmosphere:** The dafilter saves files in DA-compatible format under /var/noaaport/data/digatmos. Combined with Samba, Windows DA clients can load products directly.
- **GRLevel3:** Configuration files and server support for GRLevel3 radar visualization software, with the dafilter maintaining the required list files.
- **Unidata LDM:** The ldmfilter and nbsp2ldm bridge allow product insertion into LDM queues for further distribution through pqact-based processing.
- **IDV and MetarWeather:** Integration documented for Unidata's Integrated Data Viewer and the MetarWeather client application.
- **WeatherScope:** Support for the WeatherScope visualization platform via the ONDAS (Open Nbsp Data Access Specification) protocol.
- **INN News Server:** The nntpfilter and rstnntpfilter enable NNTP distribution through an INN gateway, with the nbsp-news package providing a ready-made configuration.

# 11. Key Architectural Characteristics

## 11.1 Strengths

- **Clean Separation of Concerns:** The C core handles only real-time reception while Tcl handles all extensible processing, resulting in a system that is both performant and adaptable.
- **Modular Filter Design:** Each filter is independently configurable, enable-able, and replaceable. New output formats require only a new filter, not core changes.
- **Self-Contained Deployment:** The embedded web server, built-in protocol support, and native image processing eliminate most external dependencies.
- **Scalable Distribution:** The master-slave topology with chaining support enables hierarchical data distribution across large installations.
- **Comprehensive Protocol Support:** Distributing via HTTP, NNTP, RSS, LDM, EMWIN, and PAN from a single installation covers virtually all weather data consumer requirements.

## 11.2 Considerations and Trade-offs

- **Niche Technology Stack:** Heavy reliance on Tcl may limit the contributor pool, as Tcl is less commonly used in modern software development compared to Python or Go.
- **Small Team:** With only two contributors across the project's history, there is inherent bus-factor risk for a system used in weather data infrastructure.
- **Build Complexity:** The M4 macro/configure.sh build system and multi-platform packaging require significant expertise to modify and maintain.
- **Legacy Protocol Support:** Supporting older protocols like QBT/EMWIN byte blaster adds complexity, though it serves a real user base that still relies on these formats.

# 12. Companion Projects and Ecosystem

| Project | Description |
|---|---|
| **Npemwin** | A standalone EMWIN server that can operate independently of the full Nbsp installation, providing lighter-weight EMWIN data distribution |
| **Npstats** | A global monitoring system for NOAAPort signal quality, receiver performance, and Nbsp processing status across multiple installations |
| **Novramon** | A text-console monitoring tool for the Novra S300 DVB-S2 receiver, with device reset and data logging capabilities |
| **NbspGisLib** | Standalone C library for NIDS radar and GINI satellite to shapefile conversion; also used as a linked library within Nbsp |
| **NbspGisLibMap** | Map rendering library extending NbspGisLib with overlay and projection capabilities for image generation |
| **gpak** | Packaged GemPak distribution designed for easy installation alongside Nbsp on supported platforms |

# 13. Conclusion

Nbsp represents a well-engineered, domain-specific software system that has evolved over more than 15 years to address the complete lifecycle of NOAAPort weather data—from raw satellite reception through processing, format conversion, and multi-protocol distribution. Its architecture successfully balances the need for real-time performance in the C core with the flexibility and extensibility of the Tcl scripting layer.

The modular filter pipeline is the key architectural insight, allowing the system to serve diverse downstream consumers (GemPak, Digital Atmosphere, GRLevel3, LDM, web browsers, news readers) from a single data ingest point. The recent addition of native GOES-R processing through the GIS libraries marks a significant step toward reducing external dependencies while expanding capabilities.

For organizations and individuals operating NOAAPort receiving stations, Nbsp provides a comprehensive, self-contained solution that continues to be actively maintained and adapted to evolving NOAA data formats and satellite infrastructure.