ARCHITECTURAL REPORT

Noaaport Broadcast System Processor (Nbsp)

Deep-Dive Analysis from Configuration, Filters, Source Structure, and Protocol Documentation

Version: nbsp-2.3.9r | 1,487 commits | 9 stars | 2 contributors

Languages: Tcl 33.9% | C 33.1% | HTML 16.4% | M4 4.6% | Makefile 3.3% | Shell 3.0%

March 2026

Sources: Official configuration manual (noaaport.net/manual/nbsp/), filter documentation, repository file tree, config files (logrotate.conf, dafilter.conf.in, cleanup scripts), NOAAPort protocol docs, community forums.

Table of Contents

1. Executive Summary

This report analyzes the Nbsp (Noaaport Broadcast System Processor) architecture based exclusively on its official configuration manual, filter documentation, actual configuration and operational files from the repository, NOAAPort protocol specifications, and community technical discussions. No README files were used as sources.

The analysis reveals a system built around a clear separation: a C-language daemon (nbspd) handles real-time UDP multicast reception and frame reassembly, while a Tcl-based filter pipeline provides extensible post-processing. The filter interface protocol is elegantly simple—filters receive a single-line text message on stdin containing sequence metadata and the spool file path. This design allows filters to be written in any language while the core maintains real-time performance.

The system exposes its internal state through eight distinct statistics files (nbspd.status, nbspd.missing, nbspd.rtx, nbspd.qstate, nbspd.server, nbspd.sthreads, nbspd.filter, nbspd.slavestats) rotated daily by logrotate, providing comprehensive operational telemetry. Configuration uses Tcl as a full scripting language, with features.conf acting as the master toggle for subsystem activation and individual .rc/.conf files for per-filter customization.

2. Core Daemon Internals (nbspd)

2.1 Multicast Reception

The nbspd daemon opens UDP multicast sockets on configurable addresses and ports. The default configuration joins four multicast groups (224.0.1.1 through 224.0.1.4 on ports 1201–1204), corresponding to the NOAAPort SBN channels. The multicast groups and ports are configured via the Tcl variables multicastip and multicastport in nbspd.conf. On multi-homed systems, the receiving interface can be specified by name (set ifname "fxp0") or IP address (set ifip "192.168.2.101"), otherwise the kernel routing table selects the interface.

UDP reception is inherently lossy—there is no retransmission mechanism. The daemon must keep up with the incoming data rate without being swapped out. The SBN stream was expanded from 30 Mbps to over 60 Mbps in 2014, with the migration from SES-1 (101°W) to Galaxy 28 (89°W) satellite in late 2017. The broadcast uses DVB-S2 modulation received by hardware such as the Novra S300N, which outputs multicast UDP packets onto the local network.

2.2 Spool Database and File Reassembly

Incoming fragments are tracked by a spool database that manages partial file reassembly. The spooldbslots parameter in nbspd.conf controls whether the daemon manages spool cleanup internally (non-zero value, the default) or delegates it to external scripts. When internally managed, the daemon tracks each file's lifecycle and deletes it from /var/noaaport/nbsp/spool when no longer needed. As a safety measure, the hourly-cleanup script scans the spool directory daily at 23:00 for orphaned files that may have survived due to errors, using find rules like "-mtime +1" for files older than one day.

The spool directory location defaults to /var/noaaport/nbsp/spool but is reconfigurable via set spooldir in the configuration. Complete product files are identified by their WMO headers and AWIPS identifiers, which become part of the file path (e.g., /var/noaaport/nbsp/spool/KMOB/kmob_sdus54-n0rmob.498745).

2.3 Internal Statistics and Telemetry

The daemon maintains eight statistics files under /var/noaaport/nbsp/stats/, each rotated daily with 7-day retention via logrotate:

Statistics File	Purpose
`nbspd.status`	Overall daemon status: uptime, processing rates, error counts, operational state
`nbspd.missing`	Tracking of missing or incomplete product frames—critical for monitoring reception quality
`nbspd.rtx`	Retransmission tracking statistics for frame-level gap analysis
`nbspd.qstate`	Internal queue state: buffer utilization, backpressure indicators, queue depths
`nbspd.server`	Network server module statistics: connected clients, bytes transmitted, protocol errors
`nbspd.sthreads`	Server thread pool status: active threads, idle threads, thread utilization metrics
`nbspd.filter`	Filter pipeline performance: processing times, files dispatched, filter errors

`nbspd.slavestats`	Master-slave replication statistics: slave connection status, sync rates, lag metrics

Additionally, web server logs are maintained under /var/log/nbsp/www/ with 7-day retention, and GemPak decoder logs under /var/log/noaaport/ are compressed daily with 2-day retention.

2.4 Server Module and Protocol Support

The server module is activated by setting the servertype variable to a non-zero value. Type 4 enables EMWIN-compatible QBT (Quick Block Transfer "byte blaster") distribution on configurable port (default 1000). The NBS1 server protocol (enabled via set feature(nbs1server) 1 in features.conf) supports master-slave file distribution. The server thread pool status is tracked separately in nbspd.sthreads, indicating a multi-threaded server implementation.

2.5 Configuration Architecture

Configuration files use Tcl as a full scripting language, enabling conditional logic, variable interpolation, and procedural configuration. The hierarchy is: (1) nbspd.conf as the main daemon configuration with all parameters having compiled-in defaults; (2) features.conf as the master subsystem toggle using feature flags (e.g., set feature(rstfilter) 1); (3) per-filter .conf files for parameter overrides; (4) per-filter .rc files for rule definitions and matching logic; (5) a site/ subdirectory for local overrides that survive package upgrades.

The -C command-line switch dumps all compiled-in default values, while -c allows specifying an alternate configuration file (intended for debugging, not production use). The man page (man nbspd) documents additional command-line options.

3. Filter Pipeline Architecture

3.1 Filter Interface Protocol

The filter interface is the core extensibility mechanism. When a complete file is saved to the spool directory, the daemon sends a single-line text message to each enabled filter's standard input. The message format is:

```
88164933 4 1 17 /var/noaaport/nbsp/spool/KMOB/kmob_sdus54-
n0rmob.498745
```

The five space-delimited fields are: (1) SBN sequence number, (2) product type, (3) category, (4) code—as defined by the NOAAPort system—and (5) the full filesystem path to the saved file. Filters read these lines from stdin in a loop and process the referenced file as needed. This protocol is language-agnostic: the documentation includes example implementations in Perl and Tcl, and explicitly states filters can be written in C or any other language.

3.2 Filter Activation and Lifecycle

Filters are enabled through two mechanisms. The standard filters are toggled via feature flags in features.conf (e.g., set feature(gribfilter) 1). User-written filters can be specified in the filterlist variable in nbspd.conf, or placed in the /var/noaaport/nbsp/dev directory for automatic discovery. When the daemon receives a SIGHUP signal, it reloads all filters from the dev directory without requiring a restart, enabling hot-deployment of new filters.

3.3 Standard Filter Configuration Pattern

Each standard filter follows a consistent three-file pattern: an .rc file (rule configuration defining matching logic and processing actions), an optional .conf file (parameter overrides such as output directories, enable/disable sub-features, and integration settings), and a shared nbsp.conf dependency (common definitions). The .rc files use Tcl's scripting capabilities for pattern matching against WMO headers, AWIPS IDs, product types, and geographical zones.

4. Standard Filters — Detailed Analysis

4.1 rstfilter (Redistribution/Storage)

The primary post-processing filter. Processes satellite image files (converting to PNG/JPG) and text files (saving as plain text), organizing them into a browsable directory tree under /var/noaaport/data/nbsp with txt/, sat/, and rad/ subdirectories. Enabled via set feature(rstfilter) 1 in features.conf, requiring nbsp.conf and rstfilter.rc in the configuration directory.

Radar GIF generation is disabled by default and requires the gpmap_gif program from GemPak. It is enabled by setting rstfilterrad_enable to 1 in rstfilter.conf. The radmap support files are assumed to be in /home/gempak/gempak/radmap/ (configurable). The rstfilter is automatically invoked by the emwin filter, so enabling both simultaneously would duplicate processing. As of v2.3.9r, the rstfilter integrates with nbspgoesr for native GOES-R satellite image production without GemPak.

4.2 dafilter (Digital Atmosphere)

Saves files in Digital Atmosphere (DA) compatible format under /var/noaaport/data/digatmos/ (configurable in dafilter.conf). The configuration file (dafilter.conf.in, 211 lines) reveals detailed options including: NNTP group prefix configuration (dafilter(nntp_groupprefix)), hourly file format strings (dafilter(hourlyfilefmt) "%Y%m%d%H"), and a satellite processing fallback mode—when the rstfilter is disabled, the dafilter can take over satellite data file processing using regex matching against directory names (e.g., "sat/gini/tigp01"). Requires Samba export of the data directory for Windows DA clients.

4.3 panfilter (Product Arrival Notification)

Implements an event-driven notification system using UDP transmissions. When a file matches a rule in panfilter.rc, the filter sends a UDP datagram to configured client addresses containing: the WMO header, AWIPS ID (if present), and the file name. Client machines use this metadata to construct a path or URL to retrieve the file via NFS, HTTP, or other exported access. Example client implementations are provided as

nbspudpread.pl (Perl) and nbspudpread.tcl (Tcl) in the documentation directory /usr/local/share/doc.

4.4 metarfilter

Parses METAR aviation weather observations and saves per-station and per-collective summary files under the metar/metarweather subdirectory. Files are formatted for direct import by MetarWeather (a free Windows utility from nirsoft.net). Opening a collective file (e.g., Louisiana/Louisiana.txt) in MetarWeather displays the latest reports from all stations in that collective; opening a station file shows its historical reports.

4.5 gribfilter

Processes GRIB (Gridded Binary) meteorological model data files, collecting them under /var/noaaport/data/grib. By default, only file collection is enabled; GrADS integration is disabled. When GrADS is installed, enabling gradsctlenable and gradsplotsenable in gribfilter.conf activates automatic generation of GrADS control files and plot images. The GRIB file naming convention is documented in gribnames.README, and GrADS integration details are in grads-grib.README.

4.6 trackfilter (Hurricane Tracking)

Collects NHC (National Hurricane Center) bulletin data and plots hurricane tracks and forecasts. Uses GrADS for plot generation (disableable via set trackfilter(grads_enable) 0 in trackfilter.conf). When GrADS is disabled, raw data is still collected under /var/noaaport/data/track/ for use by external tools. Both data and plots are accessible through the built-in web interface.

4.7 msgfilter (Notification/Alerting)

A sophisticated alerting system that sends email and SMS notifications based on product codes and geographical zones. Configuration requires: enabling in features.conf, setting SMTP originator and server addresses in site/msgfilter.conf, and defining subscriber lists.

The subscriber specification language supports regex matching against product-zone combinations. For example, the pattern "(svs|tor)-txz14[5-9],svr-vaz014" routes all Severe Weather Statements and Tornado

warnings for Texas zones 145–149, plus all Severe Thunderstorm warnings for Virginia zone 014, to a specified email address. Delivery flags (F=full file, S=summary) control message content. The filter speaks SMTP directly to a configurable server (local or remote), offloading delivery to avoid blocking the pipeline.

Subscriber lists can be defined inline in the .rc file or imported from flat files using the msgfilter_add_flat_file function, enabling external management of the subscriber database.

4.8 nbspfilter (General-Purpose Framework)

A Tcl-based general-purpose filter framework that performs preliminary file identification work and then sources a user-provided .rc script for custom processing. The framework pre-parses the filter input to identify SBN sequence numbers, product types, categories, and codes, making these available as Tcl variables (seq, type, cat, code). This eliminates the need for user scripts to implement their own parsing of the stdin protocol. The nbspfilter.rc-sample file documents the framework API with working examples.

5. Runtime File System Layout

The operational file system layout, derived from configuration files and cleanup scripts:

Path	Purpose and Retention
/var/noaaport/nbsp/spool/	Active spool directory. Managed by daemon (spooldbslots) or hourly cleanup. Organized by station ID (e.g., KMOB/).
/var/noaaport/nbsp/stats/	Eight daemon statistics files, rotated daily with 7-day retention.
/var/noaaport/data/nbsp/	Processed data (rstfilter output): txt/, sat/, rad/ subdirectories for web-ready products.
/var/noaaport/data/digatmos/	Digital Atmosphere format output (dafilter). Exported via Samba.
/var/noaaport/data/gempak/	GemPak-compatible directory tree: images/, model/, nexrad/, nwx/ subdirectories.
/var/noaaport/data/grib/	GRIB model data files (gribfilter output). Optionally with GrADS control files and plots.

/var/noaaport/data/track/	Hurricane tracking data and plots (trackfilter output).
/var/noaaport/data/inv/gis/sat/	GIS satellite inventory files. Cleaned every 6 hours for files older than 6 hours.
/var/noaaport/nbsp/dev/	Hot-deployable user filter directory. Scanned on SIGHUP.
/var/log/nbsp/www/	Web server log files with 7-day retention.
/var/log/noaaport/	GemPak decoder logs. Compressed daily, 2-day retention.
/usr/local/etc/nbsp/	Configuration directory: nbspd.conf, features.conf, filter .rc/.conf files, site/ overrides.
/usr/local/libexec/nbsp/	Filter executables and scripts (e.g., gpfilter).
/usr/local/share/doc/	Documentation, man pages, example client scripts (nbspudpread.pl, nbspudpread.tcl).

6. Deployment Topologies

6.1 Master-Slave Configuration

The master requires only set feature(nbs1server) 1 in features.conf and optionally disabling all filters (set filterlist "") to dedicate the machine to reception. The slave configuration requires set feedmode 2 and set mastername "mastername.domainname" in nbspd.conf. A slave can act as master to further slaves, creating hierarchical distribution trees. Slave performance is tracked in the nbspd.slavestats file on the master.

6.2 Multi-Platform Build System

The build directory contains platform-specific packaging for FreeBSD (build/bsdng/), Debian/Ubuntu (.deb), Almalinux/CentOS (.rpm), using M4 macros (4.6% of codebase) and Makefile templates (Makefile.in, Makefile.inc.in) with configure.sh and configure.inc for platform detection. The configure system generates platform-appropriate paths, compiler flags, and package metadata. Platform-specific config files (nbspd.conf-linux at 42 lines, nbspd.conf-freebsd at 43 lines) handle OS differences in paths and network interface naming.

6.3 Operational Automation

The scripts/ directory provides operational automation including hourly-cleanup scripts (platform-specific variants for FreeBSD and Linux) that manage: spool directory orphan removal, GIS inventory pruning (every 6 hours), web log rotation (7-day retention), and conditional radar data cleanup. The system auto-starts at boot via init scripts installed by the package manager. The 1,487-commit history across two contributors reflects sustained maintenance since at least 2011.

7. Tcl Library Ecosystem

The repository contains six specialized Tcl libraries, each in its own top-level directory, providing domain-specific functionality for weather data processing:

Library	Function and Integration
tclgempak	Tcl bindings for the GemPak weather data processing suite. Provides scripted access to GemPak decoders and image generation. Used by gpfilter for data format conversion.
tclgrads	Tcl bindings for GrADS (Grid Analysis and Display System). Contains a test suite (src/lib/test/) with output validation files. Powers the gribfilter plot generation and trackfilter hurricane track visualization.
tclmetar	Tcl library for parsing METAR aviation weather observation format. Extracts structured data (wind, visibility, ceiling, temperature, etc.) from raw text bulletins. Used by the metarfilter.
tclupperair	Tcl library for processing upper air sounding data (RAOB/radiosonde observations). Handles the specialized encoding of vertical atmospheric profiles.
tclldm	Tcl interface to Unidata's LDM (Local Data Manager). Bridges nbsp to the LDM product queue for integration with institutional data distribution networks. Powers the ldmfilter via the nbsp2ldm program.

tclssh	Tcl SSH library for secure remote file distribution. Enables encrypted file transfer to remote servers as a distribution mechanism beyond HTTP/FTP/NNTP.

8. Web Server and GIS Processing

8.1 Embedded Web Server

The tclhttpd/ directory contains a full embedded web server based on TclHttpd 3.5.1, including its complete source tree (src/tclhttpd3.5.1/htdocs/). The server listens on port 8015 by default and serves both the monitoring interface (real-time statistics from the eight nbspd.* files) and processed weather data products. The web interface includes a Statistics menu with device monitoring integration via Npstats, displaying Novra S300 receiver signal levels and quality metrics.

8.2 Native GIS Processing

The nbspgislib companion library (C programs and library) converts NOAAPort radar (NIDS) and satellite (GINI) files to GIS-compatible formats including shapefiles. It supports specific radar products including Storm Relative Velocity (e.g., N1S) and Storm Precipitation Accumulation (e.g., N1P). The nbspgislibmap extension adds map overlay and projection rendering. Together, the nbspgoesr and nbspgoesrinfo programs process GOES-R data files without requiring external applications. Binary packages are available for FreeBSD 10.1, Ubuntu 14.04, and CentOS 6/7.

9. External Integration Protocols

Protocol	Filter/Module	Implementation Detail
QBT/EMWIN	Server (type 4)	Quick Block Transfer "byte blaster" format. Distributes text as plain text, satellite as JPG, radar as GIF. Port 1000 default.
NBS1	Server	Proprietary master-slave protocol for complete file replication between nbsp instances.
HTTP	tclhttpd	TclHttpd 3.5.1 on port 8015. Serves processed products and monitoring UI.
NNTP	nntpfilter	Gateway to INN news server. Raw files via nntpfilter, processed products via rstnntpfilter.
RSS	rssfilter	Template-based RSS feed generation for text products, accessible via web server.
LDM	ldmfilter	Product insertion via nbsp2ldm bridge into Unidata LDM queue for pqact processing.
UDP PAN	panfilter	Product Arrival Notification. UDP datagrams with WMO header + AWIPS ID + filename.
SMTP	msgfilter	Direct SMTP speaking for email/SMS alerting. Supports remote SMTP servers.

SMB/CIFS	dafilter	Data directory exported via Samba for Windows Digital Atmosphere client access.
NFS	Various	Spool and data directories exportable via NFS for remote processing architectures.
SSH/SCP	tclssh	Secure remote file distribution via Tcl SSH library.
Multicast UDP	Core daemon	Reception on 224.0.1.1–10 + 224.1.1.1, ports 1201+. DVB-S2 from Galaxy 28 at 89°W.

10. Client Software Integration

The examples/ directory and filter configurations document integration with specific downstream applications:

- **GRLevel3:** The examples/scripts/ondas/ directory contains grlevel3.cfg with product-to-directory mappings for 18+ radar products (N0R, N0S, N0V, N0Z, N1P, N1R, N1S, etc.) across 150+ NEXRAD sites. The dafilter maintains the list file required by GRLevel3.
- **Digital Atmosphere:** The dafilter creates /var/noaaport/data/digatmos/nexrad/ with NIDS product subdirectories (nids/sss/n0r, etc.) exported via Samba.
- **GemPak:** The gpfilter populates /var/noaaport/data/gempak/ with images/, model/, nexrad/, nwx/ subdirectories. The gpak convenience package handles GemPak installation. GEMDATA environment variable must point to the data directory.
- **MetarWeather:** The metarfilter creates station-organized METAR files directly importable by the MetarWeather Windows application.
- **WeatherScope/IDV:** Supported via the ONDAS (Open Nbsp Data Access Specification) protocol documented in the examples/ directory.
- **Unidata LDM:** The nbsp2ldm bridge program (source in tclldm/) inserts products into LDM queues, enabling distribution via properly configured pqact.conf.

11. Architectural Characteristics

11.1 Design Strengths

- **Elegant Filter Protocol:** The stdin-based, single-line text protocol for filters is simple, language-agnostic, and requires no shared libraries or IPC complexity. Any program that reads stdin can be a filter.
- **Hot Deployment:** SIGHUP-triggered filter reloading from the dev/ directory enables production filter changes without daemon restart.
- **Comprehensive Telemetry:** Eight separate statistics files provide granular operational visibility into every daemon subsystem, from reception quality (nbspd.missing) to server thread utilization (nbspd.sthreads).
- **Tcl Configuration Language:** Using a full scripting language for configuration enables conditional logic, variable computation, and procedural setup that would be impossible with INI or YAML formats.
- **Self-Contained GIS Processing:** The nbspgislib/nbspgislibmap libraries eliminate the previous external dependency on GemPak for image generation.
- **Multi-Protocol Distribution:** 12 distinct distribution protocols from a single installation cover virtually every consumer scenario.

11.2 Trade-offs and Risks

- **Tcl Ecosystem Dependency:** 33.9% of the codebase in Tcl limits the contributor pool. Modern alternatives (Python, Go) would broaden accessibility but would require rewriting the entire filter and library ecosystem.
- **Two-Person Team:** With only two contributors across 1,487 commits, institutional knowledge concentration is a significant risk for infrastructure software.
- **M4/configure Build System:** The custom build system (4.6% M4, 3.3% Makefile) requires specialized knowledge to maintain compared to modern alternatives like CMake or Meson.

- **Legacy Protocol Maintenance:** Supporting QBT/EMWIN byte blaster, NNTP, and other legacy protocols adds complexity, though each serves a real user community.

12. Conclusion

Analyzed through its configuration manual, filter documentation, operational scripts, and statistics infrastructure—rather than its README descriptions—Nbsp reveals itself as a deeply considered piece of engineering. The filter interface protocol (a single-line stdin message) is a masterclass in simplicity: it imposes no language dependencies, no IPC complexity, and no shared-memory coordination, yet it enables an ecosystem of 8+ standard filters and unlimited user extensions.

The eight-file telemetry system (nbspd.status through nbspd.slavestats) demonstrates operational maturity: every subsystem—reception, queuing, server threads, filters, and slave replication—is independently observable. The hourly-cleanup scripts with their time-based file retention policies show a system designed for unattended long-term operation.

The configuration architecture—Tcl as a configuration language, features.conf as a feature-flag registry, site/ overrides for upgrade-safe customization, and SIGHUP hot-reload for filter deployment—reflects years of operational experience distilled into a mature configuration management pattern. This is infrastructure software built by someone who runs it in production.