changelog.md for @push.rocks/smartmta

2026-03-02 - 5.3.1 - fix(mail)

add periodic cleanup timers and proper shutdown handling for bounce manager and delivery queue; avoid mutating maps during iteration and prune stale rate-limiter stats to prevent memory growth

- BounceManager: add cleanupInterval to periodically remove bounce records older than 7 days and log removals; add stop() to clear the interval and prevent leaks
- UnifiedDeliveryQueue: introduce cleanupTimer started in startProcessing() and cleared in stopProcessing(); cleanupOldItems now collects IDs first to avoid mutating the Map while iterating and logs cleaned items; shutdown now relies on stopProcessing to clear timers
- UnifiedRateLimiter: prune stale stats.byIp and stats.byPattern entries for IPs/patterns that no longer have active counters or blocks to reduce memory usage and keep stats accurate
- Auto-cleanup tasks log errors rather than throwing to avoid crashing processing loops

2026-02-26 - 5.3.0 - feat(mailer-bin)

use mimalloc as the global allocator for mailer-bin

- Add mimalloc dependency to workspace Cargo.toml
- Enable workspace mimalloc in rust/crates/mailer-bin/Cargo.toml
- Register mimalloc as the #[global_allocator] in mailer-bin/src/main.rs
- Update Cargo.lock with new mimalloc and libmimalloc-sys entries

2026-02-26 - 5.2.6 - fix(postinstall)

remove legacy postinstall binary installer and packaging entry

- Deleted scripts/install-binary.js (legacy postinstall script that downloaded platform-specific binaries).
- Removed reference to scripts/install-binary.js from package.json "files" array so the installer is no longer included in published packages.
- This prevents automatic binary downloads during npm install and reduces package size; recommend a patch version bump.

2026-02-26 - 5.2.5 - fix(package)

remove CLI bin wrapper and exclude bin/ from published files

- Removed "bin" entry from package.json (mailer wrapper)
- Removed "bin/" from files array to prevent including CLI wrapper in published package

2026-02-26 - 5.2.4 - fix(repo)

no changes detected — no version bump required

- git diff contains no changes
- package.json version is 5.2.3
- no files modified — no release required

2026-02-26 - 5.2.3 - fix(delivery)

prevent throttle reset timer from firing after stop and avoid scheduling duplicate timers

- add throttleResetTimer property to track scheduled throttle-reset timeout
- clear throttleResetTimer when stopping to prevent it firing after shutdown
- clear existing throttleResetTimer before scheduling a new one and null it when fired to avoid duplicate timers and potential leaks

2026-02-12 - 5.2.2 - fix(deps)

bump dependencies: @push.rocks/smartrust to ^1.2.1, lru-cache to ^11.2.6

- Bumped @push.rocks/smartrust from ^1.2.0 to ^1.2.1
- Bumped lru-cache from ^11.2.5 to ^11.2.6

2026-02-11 - 5.2.1 - fix(rust-bridge)

map Node.js platform/arch to tsrust-style suffix and add platform-specific and dev localPaths for RustBridge

- Add getPlatformSuffix() to map process.platform/process.arch to tsrust-style suffixes (e.g. linux_amd64)
- Include dist_rust/mailer-bin_{suffix} when available to prefer cross-compiled binaries
- Consolidate localPaths and add local dev build paths (rust/target/release and rust/target/debug)
- Pass the computed localPaths array into plugins.smartrust.RustBridge (searchSystemPath remains disabled)

2026-02-11 - 5.2.0 - feat(packaging)

add package exports entry, include ts/dist_ts in package files, and add TS barrel index re-exports

- package.json: add "exports" mapping "." -> "./dist_ts/index.js" to provide a module entry point
- package.json: add "ts/**/*" and "dist_ts/**/*" to "files" so TypeScript sources and built output are published
- ts/index.ts: new barrel that re-exports './00_commitinfo_data.js', './mail/index.js', and './security/index.js'

2026-02-11 - 5.1.3 - fix(docs)

clarify sendEmail default behavior and document automatic MX discovery and delivery modes

- Updated README to describe automatic MX record discovery and grouping behavior when using sendEmail() (MTA mode)

- Added a Delivery Modes section and API signature for sendEmail(mode) describing mta, forward, and process options
- Expanded examples to show multi-recipient delivery, explicit mode usage, and retained low-level sendOutboundEmail example

2026-02-11 - 5.1.2 - fix(readme)

adjust ASCII architecture diagram alignment in README

- Whitespace and alignment tweaks to the ASCII architecture diagram in readme.md
- No code or behavior changes; documentation-only edit

2026-02-11 - 5.1.1 - fix(release)

no changes

- No files changed in this commit.
- Current package version remains 5.1.0 (from package.json).

2026-02-11 - 5.1.0 - feat(mailer-smtp)

add SCRAM-SHA-256 auth, Ed25519 DKIM, opportunistic TLS, SNI cert selection, pipelining and delivery/bridge improvements

- Add server-side SCRAM-SHA-256 implementation in Rust (scram.rs) and wire up SCRAM credential request/response between Rust and TypeScript bridge (ScramCredentialRequest / scramCredentialResult).
- Support SCRAM-SHA-256 auth mechanism in SMTP command parsing and advertise AUTH PLAIN LOGIN SCRAM-SHA-256 capability.
- Add opportunistic TLS mode for MTA-to-MTA delivery: configurable tls_opportunistic flag, an OpportunisticVerifier that skips cert verification per RFC 7435, and plumbing into connect/upgrade TLS paths.
- Add pipelined envelope support for MAIL FROM + multiple RCPT TO (send_pipelined_envelope) and use pipelining when server advertises PIPELINING to improve outbound performance.

- Add Ed25519 DKIM signing support and auto-dispatch: sign_dkim_ed25519, sign_dkim_auto, dkim_dns_record_value_typed, and TS changes to detect key type and call the auto signing API.
- Expose additional per-domain TLS certs (additionalTlsCerts) and implement SNI-based certificate resolver on the server to select certs by hostname; parsing helpers and fallback default cert handling included.
- Install ring crypto provider early in mailer-bin main for rustls operations and add related rust dependencies (sha2, hmac, pbkdf2) and workspace entries.
- TypeScript delivery and server bridge changes: group recipients by domain, MX resolution fallback to A record, MTA delivery loop over MX hosts, DKIM options propagation, TLS opportunistic option passed to outbound client, SCRAM credential computation in TS using PBKDF2/HMAC/SHA256 and sending results back to Rust.
- Add new tests and utilities: IPv6 DNSBL support and tests, SCRAM unit tests, DKIM Ed25519 tests, node-level MTA delivery integration test, and various test updates.
- Public API additions on the Rust <-> TS bridge: signDkim accepts keyType, new scram credential result command, onScramCredentialRequest/onScramCredentialResult helpers and sendScramCredentialResult.
- Various refactors and safety/feature improvements across mailer-core/smtp/security: envelope handling, stream buffering detection, and error handling for auth flows.

2026-02-11 - 5.0.0 - BREAKING CHANGE(mail)

remove DMARC and DKIM verifier implementations and MTA error classes; introduce DkimManager and EmailActionExecutor; simplify SPF verifier and update routing exports and tests

- Removed ts/mail/security/classes.dmarcverifier.ts and ts/mail/security/classes.dkimverifier.ts — DMARC and DKIM verifier implementations deleted
- Removed ts/errors/index.ts — MTA-specific error classes removed
- Added ts/mail/routing/classes.dkim.manager.ts — new DKIM key management and rotation logic
- Added ts/mail/routing/classes.email.action.executor.ts — centralized email action execution (forward/process/deliver/reject)
- Updated ts/mail/security/classes.spfverifier.ts to retain SPF parsing but removed verify/verifyAndApply logic delegating to Rust bridge
- Updated ts/mail/routing/index.ts to export new routing classes and adjusted import paths (e.g. delivery queue import updated)
- Tests trimmed: DMARC tests and rate limiter tests removed; SPF parsing test retained and simplified

- This set of changes alters public exports and removes previously available verifier APIs — major version bump recommended

2026-02-11 - 4.1.1 - fix(readme)

clarify architecture and IPC, document outbound flow and testing, and update module and crate descriptions in README

- Changed IPC description to JSON-over-stdin/stdout (clarifies communication format between Rust and TypeScript)
- Added Rust SMTP client entry and documented outbound mail data flow (TypeScript -> Rust signing/delivery -> result back)
- Expanded testing instructions with commands for building Rust binary and running unit/E2E tests
- Updated architecture diagram labels and Rust crate/module descriptions (mailer-smtp now includes client; test counts noted)
- Documentation-only changes; no source code behavior modified

2026-02-11 - 4.1.0 - feat(e2e-tests)

add Node.js end-to-end tests covering server lifecycle, inbound SMTP handling, outbound delivery and routing actions

- Adds four end-to-end test files: test.e2e.server-lifecycle.node.ts, test.e2e.inbound-smtp.node.ts, test.e2e.outbound-delivery.node.ts, test.e2e.routing-actions.node.ts
- Tests exercise UnifiedEmailServer start/stop, SMTP handshake and transactions, outbound delivery via a mock SMTP server, routing actions (process, deliver, reject, forward), concurrency, and RSET handling mid-session
- Introduces a minimal mock SMTP server to avoid IPC deadlock with the Rust SMTP client during outbound delivery tests
- Tests will skip when the Rust bridge or server cannot start (binary build required)

2026-02-11 - 4.0.0 - BREAKING CHANGE(smtp-client)

Replace the legacy TypeScript SMTP client with a new Rust-based SMTP client and IPC bridge for outbound delivery

- Introduce a Rust SMTP client crate with connection handling, TLS, protocol engine, and connection pooling (new modules: connection, pool, protocol, error, config).
- Add IPC handlers and management commands in the Rust binary: sendEmail, sendRawEmail, verifySmtpConnection, closeSmtpPool, getSmtpPoolStatus and integrate a SmtpClientManager into the runtime.
- Update TypeScript bridge (RustSecurityBridge) with new types and methods (ISmtpSendOptions, ISmtpSendResult, verifySmtpConnection, sendOutboundEmail, sendRawEmail, getSmtpPoolStatus, closeSmtpPool) and rework UnifiedEmailServer to use the Rust bridge for outbound delivery and DKIM signing.
- Remove the previous TypeScript SMTP client implementation and associated tests/utilities (many ts/mail/delivery/smtpclient modules and tests deleted) in favor of the Rust implementation.
- Bump dependencies and cargo config: @push.rocks/smartrust to ^1.2.0 in package.json and add/require crates (uuid, base64, webpki-roots) in Rust Cargo files.

2026-02-10 - 3.0.0 - BREAKING CHANGE(security)

implement resilience and lifecycle management for RustSecurityBridge (auto-restart, health checks, state machine and eventing); remove legacy TS SMTP test helper and DNSManager; remove deliverability IP-warmup/sender-reputation integrations and related types; drop unused dependencies

- RustSecurityBridge now extends EventEmitter and includes a BridgeState state machine, IBridgeResilienceConfig with DEFAULT_RESILIENCE_CONFIG, auto-restart with exponential backoff, periodic health checks, restart/restore logic, and descriptive ensureRunning() guards on command methods.
- Added static methods: resetInstance() (test-friendly) and configure(...) to tweak resilience settings at runtime.
- Added stateChange events and logging for lifecycle transitions; new tests added for resilience: test/test.rustsecuritybridge.resilience.node.ts.
- Removed the TypeScript SMTP test helper (test/helpers/server.loader.ts), the DNSManager (ts/mail/routing/classes.dnsmanager.ts), and many deliverability-related interfaces/implementations (IP warmup manager and sender reputation monitor) from unified email server.
- Removed public types ISmtpServerOptions and ISmtpTransactionResult from ts/mail/delivery/interfaces.ts, which is a breaking API change for consumers relying on those types.

- Removed unused dependencies from package.json: ip and mailauth.

2026-02-10 - 2.4.0 - feat(docs)

document Rust-side in-process security pipeline and update README to reflect SMTP server behavior and crate/test counts

- Clarifies that the Rust SMTP server accepts the full SMTP protocol and runs the security pipeline in-process (DKIM/SPF/DMARC verification, content scanning, IP reputation/DNSBL) to avoid IPC round-trips
- Notes that Rust now emits an emailReceived IPC event with pre-computed security results attached for TypeScript to use in routing/delivery decisions
- Updates mailer-smtp crate description to include the in-process security pipeline and increments its test count from 72 to 77
- Adjusts TypeScript directory comments to reflect removal/relocation of the legacy TS SMTP server and the smtpclient path

2026-02-10 - 2.3.2 - fix(tests)

remove large SMTP client test suites and update SmartFile API usage

- Deleted ~80 test files under test/suite/ (multiple smtpclient command, connection, edge-cases, email-composition, error-handling and performance test suites)
- Updated SmartFile usage in test/test.smartmail.ts: replaced plugins.smartfile.SmartFile.fromString(...) with plugins.smartfile.SmartFileFactory.nodeFs().fromString(...)
- Removes a large set of tests to reduce test surface / simplify test runtime

2026-02-10 - 2.3.1 - fix(npmextra)

update .gitignore and npmextra.json to add ignore patterns, registries, and module metadata

- .gitignore: expanded ignore list for artifacts, installs, caches, builds, AI dirs, and rust target path
- npmextra.json: added npmjs registry alongside internal Verdaccio registry for @git.zone/cli release settings
- npmextra.json: added projectType and module metadata (githost, gitscope, gitrepo, description, npmPackagename, license) for @git.zone/cli and added empty @ship.zone/szci entry

2026-02-10 - 2.3.0 - feat(mailer-smtp)

add in-process security pipeline for SMTP delivery (DKIM/SPF/DMARC, content scanning, IP reputation)

- Integrate mailer_security verification (DKIM/SPF/DMARC) and IP reputation checks into the Rust SMTP server; run concurrently and wrapped with a 30s timeout.
- Add MIME parsing using mailparse and an extract_mime_parts helper to extract subject, text/html bodies and attachment filenames for content scanning.
- Wire MessageAuthenticator and TokioResolver into server and connection startup; pass them into the delivery pipeline and connection handlers.
- Run content scanning (mailer_security::content_scanner), combine results (dkim/spf/dmarc, contentScan, ipReputation) into a JSON object and attach as security_results on EmailReceived events.
- Update Rust crates (Cargo.toml/Cargo.lock) to include mailparse and resolver usage and add serde::Deserialize where required; add unit tests for MIME extraction.
- Remove the TypeScript SMTP server implementation and many TS tests; replace test helper (server.loader.ts) with a stub that points tests to use the Rust SMTP server and provide small utilities (getAvailablePort/isPortFree).

2026-02-10 - 2.2.1 - fix(readme)

Clarify Rust-powered architecture and mandatory Rust bridge; expand README with Rust workspace details and project structure updates

- Emphasizes that the SMTP server is Rust-powered (high-performance) and not a nodemailer-based TS server.
- Documents that the Rust binary (mailer-bin) is required — if unavailable UnifiedEmailServer.start() will throw an error.
- Adds installation/build note: run `pnpm build` to compile the Rust binary.
- Adds a new Rust Acceleration Layer section listing workspace crates and responsibilities (mailer-core, mailer-security, mailer-smtp, mailer-bin, mailer-napi).
- Updates project structure: marks legacy TS SMTP server as fallback/legacy, adds dist_rust output, and clarifies which operations run in Rust vs TypeScript.

2026-02-10 - 2.2.0 - feat(mailer-smtp)

implement in-process SMTP server and management IPC integration

- Add full SMTP protocol engine crate (mailer-smtp) with modules: command, config, connection, data, response, session, state, validation, rate_limiter and server
- Introduce SmtpServerConfig, DataAccumulator (DATA phase handling, dot-unstuffing, size limits) and SmtpResponse builder with EHLO capability construction
- Add in-process RateLimiter using DashMap and runtime-configurable RateLimitConfig
- Add TCP/TLS server start/stop API (start_server) with TlsAcceptor building from PEM and SmtpServerHandle for shutdown and status
- Integrate callback registry and oneshot-based correlation callbacks in mailer-bin management mode for email processing/auth results and JSON IPC parsing for SmtpServerConfig
- TypeScript bridge and routing updates: new IPC commands/types (startSmtpServer, stopSmtpServer, emailProcessingResult, authResult, configureRateLimits) and event handlers (emailReceived, authRequest)
- Update Cargo manifests and lockfile to add dependencies (dashmap, regex, rustls, rustls-pemfile, rustls-pki-types, uuid, serde_json, base64, etc.)
- Add comprehensive unit tests for new modules (config, data, response, session, state, rate_limiter, validation)

2026-02-10 - 2.1.0 - feat(security)

migrate content scanning and bounce detection to Rust security bridge; add scanContent IPC command and Rust content scanner with tests; update TS RustSecurityBridge and callers, and adjust CI package references

- Add Rust content scanner implementation (rust/crates/mailer-security/src/content_scanner.rs) with pattern-based detection and unit tests (~515 lines)
- Expose new IPC command 'scanContent' in mailer-bin and marshal results via JSON for the RustSecurityBridge
- Update TypeScript RustSecurityBridge with scanContent typing and method, and replace local JS detection logic (bounce/content) to call Rust bridge
- Update tests to start/stop the RustSecurityBridge and rely on Rust-based detection (test updates in test.bouncemanager.ts and test.contentscanner.ts)
- Update CI workflow messages and package references from @serve.zone/mailer to @push.rocks/smartmta

- Add regex dependency to rust mailer-security workspace (Cargo.toml / Cargo.lock updated)

2026-02-10 - 2.0.1 - fix(docs/readme)

update README: clarify APIs, document RustSecurityBridge, update examples and architecture diagram

- Documented RustSecurityBridge: startup/shutdown, automatic delegation, compound verifyEmail API, and individual operations
- Clarified verification APIs: SpfVerifier.verify() and DmarcVerifier.verify() examples now take an Email object as the first argument
- Updated example method names/usages: scanEmail, createEmail, evaluateRoutes, checkMessageLimit, isEmailSuppressed, DKIMCreator rotation and output formatting
- Reformatted architecture diagram and added Rust Security Bridge and expanded Rust Acceleration details
- Rate limiter example updated: renamed/standardized config keys (maxMessagesPerMinute, domains) and added additional limits (maxRecipientsPerMessage, maxConnectionsPerIP, etc.)
- DNS management documentation reorganized: UnifiedEmailServer now handles DNS record setup automatically; DNSManager usage clarified for standalone checks
- Minor wording/formatting tweaks throughout README (arrow styles, headings, test counts)

2026-02-10 - 2.0.0 - BREAKING CHANGE(smartmta)

Rebrand package to @push.rocks/smartmta, add consolidated email security verification and IPC handler

- Package renamed from @serve.zone/mailer to @push.rocks/smartmta (package.json, commitinfo, README and homepage/bugs/repository URLs updated) — breaking for consumers who import by package name.
- Added new compound email security API verify_email_security that runs DKIM, SPF and DMARC in a single call (rust/crates/mailer-security/src/verify.rs) and exposed it from the mailer-security crate.

- Added IPC handler "verifyEmail" in mailer-bin to call the new verify_email_security function from the Rust side.
- Refactored DKIM and SPF code to convert mail-auth outputs to serializable results (dkim_outputs_to_results and SpfResult::from_output) and wired them into the combined verifier.
- Updated TypeScript plugin exports and dependencies: added @push.rocks/smartrust and exported smartrust in ts/plugins.ts.
- Large README overhaul to reflect rebranding, install instructions, architecture and legal/company info.

2026-02-10 - 1.3.1 - fix(deps)

add workspace dependency entries for multiple crates across mailer-bin, mailer-core, and mailer-security

- rust/crates/mailer-bin/Cargo.toml: add clap.workspace = true
- rust/crates/mailer-core/Cargo.toml: add regex.workspace = true, base64.workspace = true, uuid.workspace = true
- rust/crates/mailer-security/Cargo.toml: add serde_json.workspace = true, tokio.workspace = true, hickory-resolver.workspace = true, ipnet.workspace = true, rustls-pki-types.workspace = true, psl.workspace = true
- Purpose: align and enable workspace-managed dependencies for the affected crates

2026-02-10 - 1.3.0 - feat(mail/delivery)

add error-count based blocking to rate limiter; improve test SMTP server port selection; add tsbuild scripts and devDependency; remove stale backup file

- Add TokenBucket error tracking (errors, firstErrorTime) and initialize fields for global and per-key buckets
- Introduce RateLimiter.recordError(key, window, threshold) to track errors and decide blocking when threshold exceeded
- Update test SMTP server loader to dynamically find a free port using smartnetwork and add a recordError stub to the mock server
- Add build and check scripts to package.json and add @git.zone/tsbuild to devDependencies
- Remove a large backup file (classes.emailsendjob.ts.backup) from the repo; minor whitespace and logging cleanups in SMTP server code

2025-10-24 - 1.2.1 - fix(mail/delivery)

Centralize runtime/plugin imports and switch modules to use plugins exports; unify EventEmitter usage; update Deno dependencies and small path/server refactors

- Centralized Node and third-party imports in ts/plugins.ts and re-exported commonly used utilities (net, tls, dns, fs, smartfile, smartdns, smartmail, mailauth, uuid, ip, LRUCache, etc).
- Replaced direct EventEmitter / Node built-in imports with plugins.EventEmitter across delivery, smtpclient, routing and the unified email server to standardize runtime integration.
- Updated deno.json dependency map: added @push.rocks/smartfile, @push.rocks/smartdns, @tsclass/tsclass and ip; reordered lru-cache entry.
- Stopped exporting ./dns/index.ts from ts/index.ts (DNS is available via mail/routing) to avoid duplicate exports.
- Added keysDir alias and dnsRecordsDir in ts/paths.ts and small path-related fixes.
- Added a placeholder SmtpServer and other minor delivery/smtpserver refactors and sanitizations.
- Added a local .claude/settings.local.json for development permissions (local-only configuration).

2025-10-24 - 1.2.0 - feat(plugins)

Add smartmail, mailauth and uuid to Deno dependencies and export them from plugins; include local dev permissions file

- Add @push.rocks/smartmail, mailauth and uuid to deno.json dependencies so email parsing, DKIM and UUID utilities are available.
- Export smartmail, mailauth and uuid from ts/plugins.ts for centralized access across the codebase.
- Add .claude/settings.local.json to define local development permissions (tooling/dev environment configuration).

2025-10-24 - 1.1.0 - feat(ci)

Add CI, release and npm-publish automation; introduce release template and local settings

- Add CI workflow (.gitea/workflows/ci.yml) to run TypeScript checks, linting, formatting and platform compilation tests
- Add release workflow (.gitea/workflows/release.yml) to compile binaries for multiple platforms, generate checksums, create/update Gitea releases and upload assets
- Add npm publish workflow (.gitea/workflows/npm-publish.yml) to build package from a tag and publish precompiled binaries to npm (tag-triggered)
- Add a Gitea release template (.gitea/release-template.md) describing platforms, checksums and installation options
- Add local tool permission settings (.claude/settings.local.json) used by local tooling
- No API or runtime source changes — this commit is focused on CI/CD, release automation and packaging

2025-10-24 - 1.0.1 - fix(dev)

Add local development settings file to grant tooling permissions

- Add a local development settings file to configure permissions used by repository tooling (file-system reads and a small set of shell/CLI operations).
- This is a developer-only configuration — it doesn't change runtime code or published artifacts.
- No functional changes to the mailer library; bumps patch version only to record the repository config change.

1.0.0 (2025-10-24)

Features

- Initial release of @serve.zone/mailer
- SMTP server and client implementation
- HTTP REST API (Mailgun-compatible)
- Automatic DNS management via Cloudflare
- Systemd daemon service
- CLI interface for all operations
- DKIM signing and SPF validation
- Email routing and delivery queue
- Rate limiting and bounce management

Revision #2
Created 2026-03-28 13:10:58 UTC by foss.global Team
Updated 2026-03-29 16:54:03 UTC by foss.global Team