LOGZILLA DOCUMENTATION

Syslog pipeline customization

Extend the LogZilla syslog-ng pipeline via config.yaml, conf.d includes, SYSLOG_EXTRA_LOG_RULES, and custom sources for specialized transports

Administration · Generated June 12, 2026 · logzilla.ai/docs/administration/syslog-pipeline-customization

Syslog pipeline customization (advanced)

Administrators who need to customize the syslog-ng pipeline beyond what is available in the UI can use `config.yaml`, the `conf.d/` include directory, dedicated sources, and safe ways to inject extra rules into the main pipeline.

For common changes such as ports, batching, buffering, and debug toggles, use Syslog Settings (https://www.logzilla.ai/docs/administration/syslog-settings).

Configuration locations

- `/etc/logzilla/syslog-ng/config.yaml`
  - Primary YAML used to render the syslog-ng configuration.
- `/etc/logzilla/syslog-ng/conf.d/`
  - Additional `*.conf` files included by the main template.

Important:

- Avoid creating custom top-level `log` statements. Use `SYSLOG_EXTRA_LOG_RULES` or the patterns shown below to keep the main pipeline intact.

When to customize

- Add a dedicated listener or transport not covered by UI.
- Tag a source via `source_type` for specialized parsing.
- Insert targeted filters/rewrites into the main pipeline.
- Create raw TCP/UDP inputs for non-syslog data.

Sources (standard)

Standard sources are generated from settings and usually require no changes:

- `bsd` —- TCP on `SYSLOG_BSD_TCP_PORT` (BSD syslog)
- `bsd_udp` —- UDP on `SYSLOG_BSD_UDP_PORT` (BSD syslog)
- `rfc5424` —- TCP on `SYSLOG_RFC5424_PORT` (RFC 5424 syslog)
- `json` —- TCP on `SYSLOG_JSON_PORT` (newline JSON)
- `tls` —- TCP on `SYSLOG_TLS_PORT` (TLS RFC 5424)
- `raw` —- TCP on `SYSLOG_RAW_PORT` (no parsing)

- `raw_udp` –- UDP on `SYSLOG_RAW_UDP_PORT` (no parsing)

Dedicated sources with `source_type`

To target a subset of events for specific parsing rules:

Set `source_type` on the syslog-ng source in `config.yaml`.
In the relevant Lua rule, set `SOURCE_FILTER = "<tag>"`.
Ensure the tag is included in Parser settings as required.
Only events with the matching `source_type` are processed by rules that declare that tag.

Example: add a TLS source with a dedicated tag

```
sources:
  - name: tls_west
    enabled: true
    type: network
    port: 6514
    transport: tls
    tls_cert_file: /etc/ssl/logzilla/server.crt
    tls_key_file: /etc/ssl/logzilla/server.key
    flags: ["syslog-protocol"]
    program_override: "tls-wf"
    extra_fields:
      site: "west-dc"
    source_type: "west"
```

If rules declare `SOURCE_FILTER = "west"`, ensure the tag is permitted in Parser settings.

Example: add a raw UDP source for unparsed logs

```
sources:
  - name: raw_udp_1516
    enabled: true
    type: network
    port: 1516
    transport: udp
    flags: ["no-parse"]
    program_override: "raw-udp"
    extra_fields:
      log_type: "raw"
    source_type: "devices"
```

Injecting extra log rules (advanced)

Use `SYSLOG_EXTRA_LOG_RULES` in settings or set `extra_log_rules` in `config.yaml` to inject statements into the main `log {}`. Keep injected logic minimal and well-scoped.

```
extra_log_rules: "filter(f_only_host);"
```

When more complex filters are needed, place reusable building blocks in `conf.d/` and reference them from `extra_log_rules`.

Restarting after changes

```
logzilla restart -c syslog
```

Related topics

- Syslog Settings (https://www.logzilla.ai/docs/administration/syslog-settings)
- Syslog troubleshooting (https://www.logzilla.ai/docs/administration/syslog-troubleshooting)
- Syslog basics (https://www.logzilla.ai/docs/administration/syslog-basics)