ESA CSOC/ESOC — Rsyslog Load-Balancer on RHEL 9 (TCP/TLS Round-Robin)

Author: Gaetan De Dobbeleer (Logpoint)
Date: 12 Aug 2025
Scope: Implement a robust rsyslog relay on RHEL 9 that load-balances outgoing syslog traffic to multiple Logpoint collectors/backends over **TCP/TLS** using the **native `omfwd` target pool** feature. Ports are **fixed by Logpoint**: **UDP 514**, **TCP 514 (clear)**, **TCP 6514 (TLS)**.

---

1) Executive summary

- **OS**: RHEL 9 (standard build; no containers).
- **Rsyslog**: **v8.2502+ (recommended)** from the Adiscon RHEL 9 repo.
  - Rationale: native **load-balancing in `omfwd`** requires **≥ 8.2408**; v8.2502 additionally fixes a TLS handshake blocking issue and brings reliability/perf improvements.
- **Transport & fixed ports (Logpoint)**: **UDP 514**, **TCP 514 (clear)**, **TCP 6514 (TLS)**.
- **Ingress on relay**: listen on UDP 514, TCP 514, and TCP 6514 (TLS).
- **Egress from relay**: **preferred** over **TCP 6514 (TLS)** with round-robin LB; optional cleartext egress over **TCP 514** provided (disabled by default).
- **Buffering & failover**: disk-assisted queues with automatic retry; if one target is down the pool rotates; if **all** targets are down, events are **buffered on disk** and automatically replayed when a backend returns.
- **Observability**: `impstats` every 60s, JSON/CEE format; actionable counters for queue depth, submits, failures, and suspension state.
- **Security**: SELinux **Enforcing**; firewalld opened only for 514/udp, 514/tcp, 6514/tcp; TLS defaults set (anonymous auth by default, can switch to mTLS later).

> **Outcome**: transparent round-robin distribution of events across multiple Logpoint backends with automatic removal/retry of down targets, safe buffering when none are reachable, and clean recovery with no manual intervention.

---

2) Why this rsyslog version? (Design decision)

- The **native load-balancing** (array syntax `target=["ip1","ip2",...]` in `omfwd`) landed in **8.2408.0**.
- RHEL 9 stock repos carry **8.2310.x**, which **does not** include that feature.
- Installing from the **Adiscon RHEL 9 repo** provides **current v8** quickly and safely.
- **Recommendation**: **8.2502+** (released Feb 2025) — includes a **TLS handshake fix** preventing rare stalls in `omfwd` under load.
- Acceptable “scheduled-stable” alternative if change control is strict: **8.2412.0** (still includes LB).

---

3) Prerequisites

- RHEL 9 server (x86_64), root access.

- Network reachability to Logpoint backends.
- **Fixed Logpoint ports: UDP 514**, **TCP 514 (clear)**, **TCP 6514 (TLS)**.
- TLS files (if TLS enabled):
  - `/etc/rsyslog.d/tls/ca.crt`
  - `/etc/rsyslog.d/tls/server.crt`
  - `/etc/rsyslog.d/tls/server.key` (0600)

Backends: `<BACKEND_1>`, `<BACKEND_2>`

3bis) Customization — Replace placeholders with ESA production targets

Before going live, **replace the placeholders** with the actual ESA Logpoint collector IPs or hostnames:

- `<BACKEND_1>` — first Logpoint backend (collector)
- `<BACKEND_2>` — second Logpoint backend (collector)

Quick example with `sed` (replace with your real values):

```
sudo sed -i 's/<BACKEND_1>/192.0.2.10/' /etc/rsyslog.d/10-esa-lb.conf
sudo sed -i 's/<BACKEND_2>/192.0.2.11/' /etc/rsyslog.d/10-esa-lb.conf
sudo systemctl restart rsyslog
```

Connectivity test (optional):

```
# TLS 6514
timeout 3 bash -c 'cat < /dev/null > /dev/tcp/<BACKEND_1>/6514' && echo "OK" ||
echo "FAIL"
timeout 3 bash -c 'cat < /dev/null > /dev/tcp/<BACKEND_2>/6514' && echo "OK" ||
echo "FAIL"
# Clear TCP 514 if you intend to enable it later
timeout 3 bash -c 'cat < /dev/null > /dev/tcp/<BACKEND_1>/514' && echo "OK" ||
echo "FAIL"
timeout 3 bash -c 'cat < /dev/null > /dev/tcp/<BACKEND_2>/514' && echo "OK" ||
echo "FAIL"
```

---

4) Installation on RHEL 9 (no Docker)

1. **Enable Adiscon repo & install** (v8-stable or daily-stable):

   ```
   cd /etc/yum.repos.d/
   sudo curl -O https://rpms.adiscon.com/v8-stable/rsyslog-rhel.repo
   # (option) daily-stable for faster fixes
   sudo curl -O https://rpms.adiscon.com/v8-stable-daily/rsyslog-daily-
   rhel.repo
   ```

```
# ensure gpgcheck is enabled in the .repo file (security hygiene)
sudo sed -i 's/^gpgcheck=.*/gpgcheck=1/' /etc/yum.repos.d/rsyslog-*.repo

sudo dnf clean all && sudo dnf makecache
sudo dnf install -y rsyslog rsyslog-gnutls
rsyslogd -v   # verify: aim for 8.2502+ (or ≥ 8.2412 at minimum)
```

2. **Open firewalld** (fixed ports):

```
sudo firewall-cmd --permanent --add-port=514/tcp
sudo firewall-cmd --permanent --add-port=514/udp
sudo firewall-cmd --permanent --add-port=6514/tcp
sudo firewall-cmd --reload
```

3. **SELinux** — keep **Enforcing**. If a non-default port is ever used, map it:

```
# example for TCP 30514 if used one day:
sudo semanage port -a -t syslogd_port_t -p tcp 30514
```

---

5) Configuration layout

Files

- `/etc/rsyslog.conf` — core, inputs, TLS defaults, `impstats`.
- `/etc/rsyslog.d/10-esa-lb.conf` — outbound to Logpoint via LB + buffering + failover.
- `/etc/rsyslog.d/tls/` — TLS material.

5.1 `/etc/rsyslog.conf` (excerpt)

```
# Work directory for persistent queues (disk-assisted)
global(workDirectory="/var/spool/rsyslog")

# Inputs
module(load="imudp")
module(load="imtcp")
module(load="impstats" interval="60" format="cee")  # stats/health

# TLS defaults (used by listener on 6514 and egress if enabled)
global(
  defaultNetstreamDriver="gtls"
  defaultNetstreamDriverCAFile="/etc/rsyslog.d/tls/ca.crt"
  defaultNetstreamDriverCertFile="/etc/rsyslog.d/tls/server.crt"
  defaultNetstreamDriverKeyFile="/etc/rsyslog.d/tls/server.key"
)

# Inputs from sources (fixed ports)
```

```
input(type="imudp" port="514" ruleset="to_logpoint")
input(type="imtcp" port="514" ruleset="to_logpoint")
input(type="imtcp" port="6514" StreamDriver.name="gtls" StreamDriver.mode="1"
StreamDriver.authmode="anon" ruleset="to_logpoint")

# Include drop-ins
$IncludeConfig /etc/rsyslog.d/*.conf
```

> Notes:
>
> - `omfwd` is builtin (no `module(load="omfwd")`).
> - Template used later: `RSYSLOG_SyslogProtocol23Format` (RFC5424-like).

5.2 `/etc/rsyslog.d/10-esa-lb.conf` — **LB + buffering + failover**

```
# Keep impstats out of the forward stream (optional)
if ($syslogtag == 'impstats:') then {
  action(type="omfile" file="/var/log/rsyslog_stats.json")
  stop
}

# Main forwarding ruleset to Logpoint
ruleset(name="to_logpoint") {
  # --- Preferred: TLS 6514 with native round-robin load-balancing
  action(
    name="lp_tls_rr"
    type="omfwd" protocol="tcp"
    StreamDriver="gtls" StreamDriverMode="1" StreamDriverAuthMode="anon"
    target=["<BACKEND_1>","<BACKEND_2>"]   # round-robin pool
    port="6514"
    template="RSYSLOG_SyslogProtocol23Format"

    # --- Buffering (disk-assisted action queue)
    queue.type="LinkedList"                          # async, supports DA-queue
    queue.filename="q_logpoint_tls"                 # enables spooling under
workDirectory
    queue.maxdiskspace="10g"                         # cap for on-disk backlog
    queue.size="50000"                               # in-memory elements
    queue.highwatermark="40000"
    queue.lowwatermark="10000"
    queue.dequeuebatchsize="1024"
    queue.workerthreads="2"
    queue.saveonshutdown="on"                        # persist on planned reboot

    # --- Availability / retry policy
    action.resumeRetryCount="-1"                     # retry forever
    action.resumeInterval="30"                       # backoff between attempts (s)
  )

  # --- Optional (disabled): clear TCP 514 with round-robin load-balancing
  # action(
  #   name="lp_tcp_rr" type="omfwd" protocol="tcp"
  #   target=["<BACKEND_1>","<BACKEND_2>"] port="514"
  #   template="RSYSLOG_SyslogProtocol23Format"
  #   queue.type="LinkedList" queue.filename="q_logpoint_tcp"
  #   queue.maxdiskspace="10g" queue.size="50000"
  #   queue.highwatermark="40000" queue.lowwatermark="10000"
  #   queue.dequeuebatchsize="1024" queue.workerthreads="2"
  #   queue.saveonshutdown="on"
  #   action.resumeRetryCount="-1" action.resumeInterval="30"
  # )

  # --- Local fallback (executes only if previous action is suspended)
  action(
    name="local_fallback" type="omfile"
    file="/var/log/esa_fallback-buffer.log"
    execOnlyWhenPreviousIsSuspended="on"
  )
}
```

> **Behavior**
>
> - If one backend is **down**, `omfwd` **skips it** and sends to the next one in the pool.
> - If **all** backends are down, the action becomes **suspended** and events are **buffered** (memory → disk) until a target returns.
> - On recovery, rsyslog **automatically resumes** and **replays** the backlog in order.
> - The **local fallback** only activates while the TLS action is suspended, ensuring you still retain local evidence.

---

6) Verification & smoke tests

```
# Syntax check
sudo rsyslogd -N1

# Enable & start
sudo systemctl enable --now rsyslog
sudo systemctl status rsyslog --no-pager

# Send a test event
logger -t ESA_SMOKE "rsyslog LB end-to-end OK"
```

Operational checks

- Round-robin: temporarily stop one backend and verify events continue to the remaining target; restore and observe alternation.
- Buffering: stop **both** backends, generate traffic, confirm growth of spool files under `/var/spool/rsyslog/` and of the impstats queue metrics; restart backends and confirm automatic drain.

---

7) Monitoring / operations with `impstats`

We enabled `impstats` (JSON/CEE) every 60s to `/var/log/rsyslog_stats.json`. Typical usage:

- **Quick view (latest 20 lines):**

  ```
  tail -n 20 /var/log/rsyslog_stats.json
  ```

- **Follow in real time:**

  ```
  tail -f /var/log/rsyslog_stats.json
  ```

- **Parse JSON (strip the @cee: prefix) and summarize the action/queue health** — requires `jq`:

  ```
  sed 's/^@cee: //' /var/log/rsyslog_stats.json |   jq -r
  'select(.name=="action" and .actionName=="lp_tls_rr") |
         "\(.timegenerated) submitted=\(.submitted) failed=\(.failed)
  suspended=\(.suspended) queuesize=\(.queuesize)"'
  ```

What to watch

- `submitted` steadily increasing; `failed` near zero in normal conditions.
- `suspended=true` indicates the action is paused (e.g., all targets unavailable).
- `queuesize` and on-disk spool count/size (`du -sh /var/spool/rsyslog`) should **stabilize** then **drain** after recovery.
- For the fallback file, check growth of `/var/log/esa_fallback-buffer.log` during suspension.

Housekeeping

- Cap disk usage via `queue.maxdiskspace` (here `10g`). Increase if necessary for longer outages.
- Review `/var/log/messages` (or journald) for rsyslog notices about suspension/resume events.

---

8) Rollback / change control

- To **pause LB** quickly, comment the `target=[...]` action and point to a single backend.
- To **revert packages**: disable Adiscon repo and `dnf downgrade` to the previous rsyslog build.
- Configuration is self-contained under `/etc/rsyslog*`; keep a backup before changes.

---

9) Appendix — Quick FAQ

Q: Why not UDP for load-balancing?
A: The target pool is applied for TCP/TLS; UDP sends to the first target only. Use TCP/TLS for LB and reliability.

Q: How does rsyslog detect target availability?
A: Via the TCP/TLS connection lifecycle and write results. On connect or send failure, the current target is skipped; if all fail, the action is suspended and the queue buffers data until recovery.

Q: Can we require client certs (mTLS)?
A: Yes. Switch `StreamDriverAuthMode` from `anon` to `x509/name` and manage trust anchors accordingly.

Q: What about TLS ciphers?
A: Default OpenSSL policy on RHEL 9 is sane; strict ciphers can be pinned via `gtls` driver options if a policy mandates it.