LOGZILLA DOCUMENTATION

Application Ports on LogZilla Cloud

Enable vendor parsing apps on LogZilla Cloud by tagging events with a source type at the relay, the cloud equivalent of the on-prem Application Ports setting

Getting Started with LogZilla Cloud · Generated June 17, 2026 · logzilla.ai/docs/logzilla-cloud/application-ports-on-cloud

Several vendor apps (Arista EOS, Palo Alto, Cisco Meraki, and others) need to know which app should parse an event *before* the message is parsed, because their log format is otherwise ambiguous or non-standard. On a self-hosted server this is handled by the **Application Ports** setting: each vendor gets a dedicated syslog port, and events received on that port are automatically tagged so the matching app rule runs.

On LogZilla Cloud there is no Application Ports page, because devices do not send to the cloud directly - they send to a relay (https://www.logzilla.ai/docs/logzilla-cloud/cloud-onboarding) that forwards events over HTTPS to `/incoming`. This page explains how to reproduce the Application Ports behavior on the relay.

How Application Ports Work

The dedicated port is only a convenience. What it actually does is stamp an extra field, `_source_type`, on every event it receives. Each vendor app's rule is gated on that field and ignores any event that is not tagged for it. For example, the Arista EOS rule processes an event only when `extra_fields._source_type` equals `arista_eos`.

On the cloud platform the same tag is applied at the relay. Nothing else about the app changes - once the tag is present, the cloud-side rule parses the event exactly as it would on-prem.

Tag Reference

Set `_source_type` to the value below for the vendor being sent. These are the same values the on-prem Application Ports stamp.

Vendor / App	Application Ports field (on-prem)	`_source_type` value	Relay message format
Arista EOS	Syslog Arista Eos Port	`arista_eos`	Standard
Check Point	Syslog Checkpoint Port	`checkpoint`	Raw (no-parse)
IBM DataPower	Syslog Datapower Port	`datapower`	Raw (no-parse)
Dell N-Series	Syslog Dell N Series Port	`dell_n_series`	Standard
FireEye	Syslog Fireeye Port	`fireeye`	Standard
Infoblox NIOS	Syslog Infoblox Port	`infoblox`	Raw (no-parse)
Cisco Meraki	Syslog Meraki Port	`meraki`	Standard

Vendor / App	Application Ports field (on-prem)	`_source_type` value	Relay message format
Palo Alto	Syslog Paloalto Port	`paloalto`	Standard
Palo Alto Prisma SD-WAN ION	Syslog Paloalto Sdwan Ion Port	`paloalto_sdwan_ion`	Standard
Netgate pfSense	Syslog Pfsense Port	`pfsense`	RFC 5424
Symantec Endpoint Protection	Syslog Symantec Port	`symantec_epm`	Raw (no-parse)
Ubiquiti UniFi	Syslog Unifi Port	`unifi`	Standard
VMware vSphere	Syslog Vmware Port	`vmware`	RFC 5424

The **Relay message format** column matters when building the relay source - see Vendors That Need the Raw Message below. Vendors marked *Standard* need no special handling.

Tag Events at the Relay (syslog-ng)

This builds on the syslog-ng relay from Onboarding LogZilla Cloud (https://www.logzilla.ai/docs/logzilla-cloud/cloud-onboarding). The tag is added with one `--pair` line under an `extra_fields.` prefix, the same way user tags are added under `user_tags.`:

```
        --pair extra_fields._source_type="arista_eos"
```

Scope the Tag to the Right Devices

A `--pair` line applies to **every** event sent through that destination, so it must not be added to the main catch-all destination, which would tag all traffic. Instead, give the vendor's devices a dedicated listener port on the relay and their own destination, so only those events are tagged. This mirrors how a dedicated Application Port keeps the vendor's traffic separate on-prem.

```
# Devices for this app send syslog to a dedicated port on the relay
source s_arista {
    network(ip("0.0.0.0") port(5514) transport("udp"));
    network(ip("0.0.0.0") port(5514) transport("tcp"));
};

destination d_logzilla_arista {
    http(
```

```
        url("https://YOUR-HOST.logzilla.cloud/incoming")
        method("POST")
        user-agent("syslog-ng User Agent")
        headers(
            "Content-Type: application/json",
            "Authorization: token YOUR_GENERATED_TOKEN"
        )
        body-prefix("{\"events\": [\n")
        delimiter(",\n")
        body('$(format-json
            --pair priority=int($PRI)
            --pair host="$HOST"
            --pair program="$PROGRAM"
            --pair message="$MESSAGE"
            --pair extra_fields._source_type="arista_eos"
        )')
        body-suffix("\n]}")
        batch-lines(10000)
        batch-bytes(10485760)
        batch-timeout(500)
    );
};

log { source(s_arista); destination(d_logzilla_arista); };
```

Then point the vendor's devices at the relay on the dedicated port (`5514` in this example), and replace `arista_eos` with the value from the Tag Reference for the relevant vendor.

Replace Placeholders

- `YOUR-HOST` → instance hostname.
- `YOUR_GENERATED_TOKEN` → ingest-only token from the provisioning email.
- `5514` → any unused port on the relay; it does not need to match any on-prem value.
- `arista_eos` → the `_source_type` value for the relevant vendor.

Validate and Restart

```
syslog-ng --syntax-only
sudo systemctl restart syslog-ng
```

Vendors That Need the Raw Message

Most apps work with the standard relay configuration above. A few vendors send a log format that the app rule must parse in full, so the relay has to forward the original message unchanged rather than syslog-ng's parsed version:

- **Raw (no-parse)** - Check Point, IBM DataPower, Infoblox NIOS, and Symantec Endpoint Protection do not include a standard program field; the rule parses the entire raw line itself. Configure the dedicated relay source with `flags(no-parse)` so `$MESSAGE` carries the complete original line:

```
source s_checkpoint {
    network(ip("0.0.0.0") port(5601) transport("udp") flags(no-parse));
    network(ip("0.0.0.0") port(5601) transport("tcp") flags(no-parse));
};
```

- **RFC 5424** - Netgate pfSense and VMware vSphere send RFC 5424 syslog. Configure the dedicated relay source with `flags(syslog-protocol)` so the message is parsed correctly:

```
source s_pfsense {
    network(ip("0.0.0.0") port(5602) transport("udp") flags(syslog-protocol));
    network(ip("0.0.0.0") port(5602) transport("tcp") flags(syslog-protocol));
};
```

In both cases the destination and the `extra_fields._source_type` tag are the same as in the standard example - only the `flags(...)` on the source change.

Tag Events at the Relay (rsyslog)

If the relay uses rsyslog with `omhttp` (see Onboarding LogZilla Cloud (https://www.logzilla.ai/docs/logzilla-cloud/cloud-onboarding), Option B), add an `extra_fields` object to the JSON template for the vendor's destination:

```
template(name="lz_event_arista" type="list") {
  constant(value="{\"events\":[{")
  constant(value="\"priority\":") property(name="pri")
  constant(value=",\"host\":\"") property(name="hostname") constant(value="\"")
  constant(value=",\"program\":\"") property(name="programname") constant(value="\"")
  constant(value=",\"message\":\"") property(name="msg") constant(value="\"")
  constant(value=",\"extra_fields\":{\"_source_type\":\"arista_eos\"}}")
  constant(value="}]} ")
}
```

Reference this template from the `omhttp` action that handles the vendor's devices, and route only those devices to it (for example, with a `ruleset`/`if $fromhost-ip` filter) so the tag is not applied to unrelated traffic.

Verify

After the relay restarts and the devices are sending:

In the LogZilla UI, search for events from the vendor's devices.
Confirm the **Program** column shows the value set by the app (for example, `Arista EOS`) rather than `Unknown`, and that the app's tags (Vendor, Event Class, and so on) are populated.
If events still show `Program: Unknown` with no tags, the `_source_type` tag is most likely missing or misspelled, or - for the vendors above - the relay source is not preserving the raw or RFC 5424 message.

Related

- Onboarding LogZilla Cloud (https://www.logzilla.ai/docs/logzilla-cloud/cloud-onboarding)
- Syslog Settings (https://www.logzilla.ai/docs/administration/syslog-settings)
- Receiving Syslog Events (https://www.logzilla.ai/docs/receiving-data/receiving-syslog-events)