LOGZILLA DOCUMENTATION

Event Enrichment

LogZilla App Store application: Event Enrichment

LogZilla App Store · Generated April 27, 2026 · logzilla.ai/docs/logzilla-appstore/event-enrichment

Overview

The Event Enrichment app adds contextual metadata to log events by looking up information from external data sources. This process enhances events with additional user tags, making them more searchable and useful for dashboards, alerting, and operational analysis.

App Function

Event enrichment works by:

- **Matching Events**: Uses configurable criteria to identify events for enrichment
- **Looking Up Data**: Searches metadata files for matching information
- **Adding Context**: Applies found data as user tags to events
- **Optional Message Enhancement**: Can append enrichment data to log messages

Enriched data becomes available as user tags for filtering, dashboard creation, and automated workflows.

Vendor Documentation

This is a LogZilla utility app. No external vendor documentation applies.

Incoming Log Format

The Event Enrichment app processes all log formats. It operates on events after they have been parsed by other rules, applying metadata based on configurable lookup criteria.

Parsed Metadata Fields

The Event Enrichment app applies dynamic user tags from external metadata files. The specific tags depend on the configuration and metadata files created by the administrator.

Log Examples

The Event Enrichment app works with any log format. See the configuration examples below for usage patterns.

Prerequisites

IMPORTANT: Event enrichment requires manual configuration before it will function. Administrators must create both configuration rules and metadata files.

Configuration

Installation

Install the Event Enrichment app from the LogZilla App Store
Configure lookup rules and metadata files
Reload rules to activate enrichment

Configuration Files

Event enrichment uses configuration files in `/etc/logzilla/apps/event_enrichment/config/`:

- **config.yaml**: Defines enrichment tasks and lookup rules
- **metaData.yaml**: Sample metadata file (create additional files as needed)

Note: Root permissions are required to edit configuration files.

Basic Configuration

config.yaml

```
---
- name: Simple Host Lookup
  metadata_file: metaData
  lookup_field: host
  apply_to_message: false
```

Configuration Options

Option	Required	Type	Description
`name`	No	string	Descriptive name for the enrichment task
`metadata_file`	Yes	string	Name of metadata file (without .yaml extension)
`lookup_field`	Yes	string	Event field to use for lookup (e.g., `host`, `message`, user tag)

Option	Required	Type	Description
`filter`	No	array	LogZilla filter to limit which events are enriched
`lookup_re`	No	string	Regular expression to extract lookup key from field
`use_cidr_match`	No	boolean	Enable CIDR network matching for IP addresses
`apply_to_message`	No	boolean	Append enrichment data to event message text

Metadata Files

Create metadata files containing enrichment data organized by lookup keys. Each key corresponds to a value that will be matched from incoming events.

metaData.yaml

```
---
"server1.example.com":
  device_role: web server
  location: datacenter-east
  contact: webteam@example.com
"192.168.1.1":
  device_role: router
  location: branch-office
  contact: netops@example.com
"firewall-01":
  device_role: security
  location: datacenter-west
  contact: security@example.com
```

When an event arrives with `host` matching one of these keys, the corresponding key-value pairs are added as user tags to the event.

CIDR Network Matching

Enable CIDR matching to enrich events based on IP network ranges:

CIDR Configuration

```
---
- name: Network Zone Enrichment
  metadata_file: networks
  lookup_field: host
  use_cidr_match: true
```

networks.yaml

```
---
"10.0.0.0/8":
  network_type: internal
  security_zone: trusted
"192.168.1.0/24":
  location: branch_office
  vlan: 100
"172.16.0.0/12":
  network_type: dmz
  security_zone: restricted
```

Note: More specific networks override broader matches. An IP like `192.168.1.50` would match both `10.0.0.0/8` and `192.168.1.0/24`, with the more specific `/24` network taking precedence.

Activate Configuration

Reload rules to activate enrichment:

```
sudo logzilla rules reload
```

Advanced Configuration

Event Filtering

Use filters to restrict enrichment to specific events:

```
---
- name: Firewall Event Enrichment
  metadata_file: firewalls
  lookup_field: host
  filter:
  - field: program
    op: eq
    value: "%ASA"
```

Filters use LogZilla's standard filter syntax. See Data Transformation - Rewrite Rules for complete filter documentation.

Regular Expression Extraction

Use `lookup_re` to extract specific values from event fields:

```
---
- name: IP Address Enrichment
  metadata_file: ipInfo
  lookup_field: message
  lookup_re: "src_ip=(\\d+\\.\\d+\\.\\d+\\.\\d+)"
```

This extracts IP addresses from messages like `"Connection from src_ip=192.168.1.100"` and uses the IP address as the lookup key.

Built-in Regex Patterns

LogZilla provides built-in patterns for common extractions:

```
lookup_re: "src_ip=(%IP_REGEX%)"
```

The `%IP_REGEX%` token matches IPv4 addresses.

Message Text Enhancement

Set `apply_to_message: true` to append enrichment data to event messages:

```
---
- name: User Lookup with Message Enhancement
  metadata_file: users
  lookup_field: message
  lookup_re: "user=([\\w.]+)"
  apply_to_message: true
```

Original Message:

```
Login successful for user=jdoe from 192.168.1.100
```

Enhanced Message:

```
Login successful for user=jdoe from 192.168.1.100 department="Engineering" phone="555-0123"
```

Multiple Enrichment Tasks

Define multiple enrichment tasks in a single configuration:

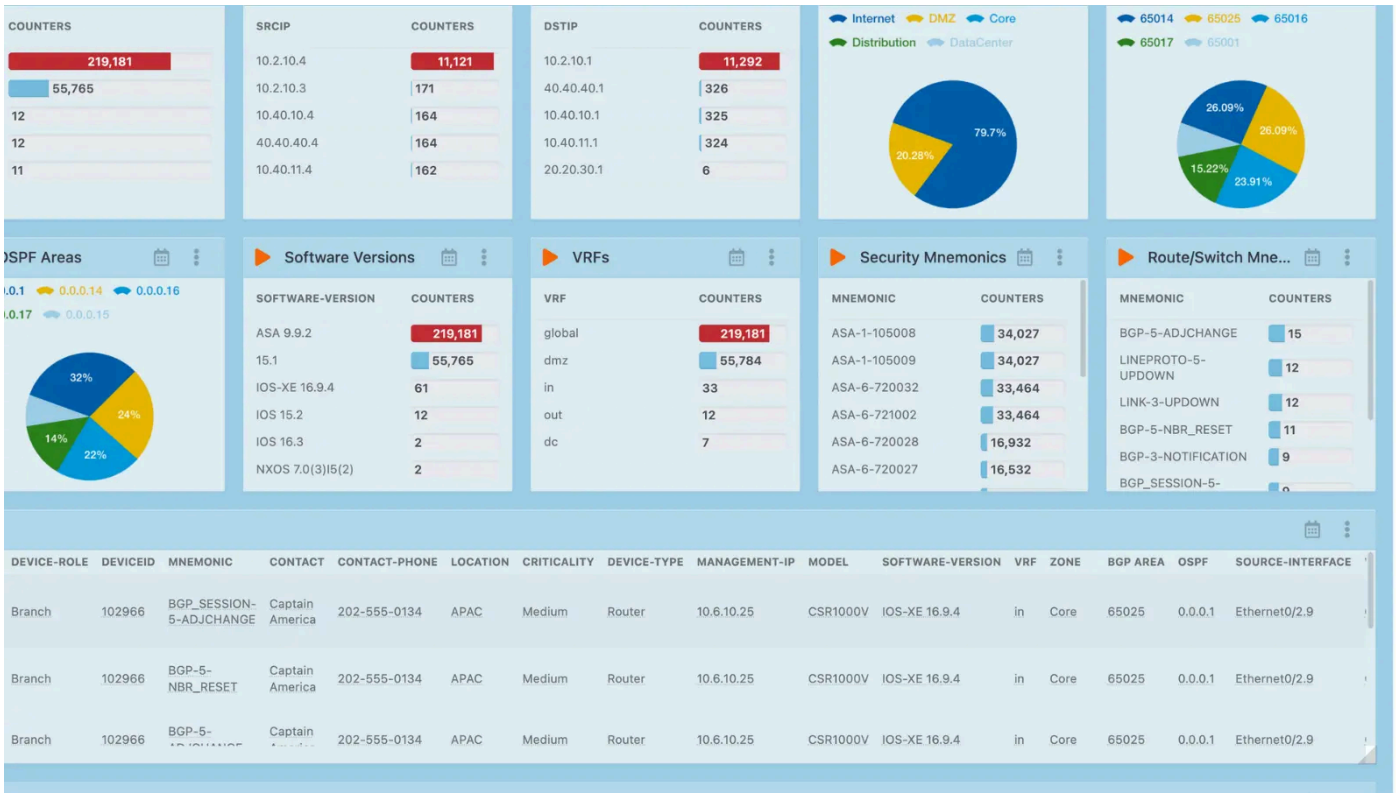
```
---
- name: Host Information
  metadata_file: hosts
```

```
    lookup_field: host
  - name: Application Data
    metadata_file: applications
    lookup_field: program
  - name: User Information
    metadata_file: users
    lookup_field: message
    lookup_re: "user=([\\w.]+)"
    filter:
    - field: message
      op: "=*"
      value: "login"
```

Each task runs independently, allowing multiple enrichments per event.

Real-World Example

The dashboard below shows the power of event enrichment in action:



In this example, all columns except "Mnemonics" are populated through event enrichment, providing operational context including:

- **Device Information**: Role, device ID, model, software version
- **Contact Details**: Responsible person and phone number
- **Network Context**: Location, criticality, management IP, VRF, zones
- **Infrastructure Details**: BGP areas, OSPF settings, source interfaces

This enrichment transforms basic network device logs into actionable operational intelligence, enabling faster troubleshooting, better monitoring, and more effective incident response.

Benefits

Event enrichment provides:

- **Enhanced Context**: Transform basic logs into actionable intelligence
- **Faster Resolution**: Immediate access to device and contact information
- **Better Dashboards**: Rich metadata enables detailed visualizations
- **Improved Alerting**: Context-aware alerts based on device criticality
- **Operational Efficiency**: Centralized asset information reduces manual lookups