LOGZILLA DOCUMENTATION

Network Infrastructure Correlation

Detect interface flapping, BGP adjacency loss, and cascading network failures in LogZilla by correlating Cisco and OSPF events with SEC rules

Event Correlation · Generated April 27, 2026 · logzilla.ai/docs/event-correlation/network-infrastructure-correlation

Network Infrastructure Event Correlation

Network infrastructure correlation detects patterns in network device behavior, from simple interface flapping to complex BGP adjacency issues. LogZilla's pre-processing capabilities enable highly efficient SEC rules by extracting network-specific fields before correlation.

Prerequisites: Ensure Event Correlation is enabled and forwarder reloading is available as shown in the Event Correlation Overview (https://www.logzilla.ai/docs/event-correlation/event-correlation-overview).

BGP Adjacency Correlation

Business Value

Monitor BGP neighbor relationships to detect network connectivity issues, calculate outage duration, and identify chronic instability patterns.

LogZilla Forwarder Configuration

```
# /etc/logzilla/forwarder.d/cisco-network-health.yaml
window_size: 0
type: sec
sec_name: cisco-network-health
match:
  - field: cisco_mnemonic
    value:
      - "BGP-5-ADJCHANGE"
      - "BGP-4-MAXPFX"
      - "BGP-3-NOTIFICATION"
      - "OSPF-5-ADJCHG"
rules:
- rewrite:
    message: >-
      [CISCO_NETWORK_HEALTH]
      cisco_mnemonic="${:cisco_mnemonic}"
      host="${:host}"
      srcip="${:SrcIP}"
      dstip="${:DstIP}"
      srcport="${:SrcPort}"
      dstport="${:DstPort}"
      protocol="${:protocol}"
      $MESSAGE
```

SEC Rule: BGP Neighbor Down/Up Correlation

```
# BGP adjacency outage tracker (host + neighbor)

type=Pair

# ---- START (neighbor Down) ----
ptype=RegExp
pattern=\[CISCO_NETWORK_HEALTH\]\s+cisco_mnemonic="BGP-5-ADJCHANGE"\s+host="([^"]+)"\s+srcip="
[^"]*"\s+dstip="[^"]*"\s+srcport="[^"]*"\s+dstport="[^"]*"\s+protocol="[^"]*"\s+.*\bneighbor\s+([0-
9A-Fa-f:.]+)\s+Down\b
context=BGP_ADJ_$1_$2
desc=BGP Neighbor $2 Down on $1
action=eval %router_host ( "$1" ); \
       eval %neighbor_ip ( "$2" ); \
       eval %down_time ( time() ); \
       shellcmd (/usr/bin/host %neighbor_ip > /tmp/bgp-lookup-%neighbor_ip 2>/dev/null); \
       eval %hostname ( readfile("/tmp/bgp-lookup-%neighbor_ip") ); \
       write /var/log/logzilla/sec/network-health.log \
         "BGP_ADJ_DOWN %t router=%router_host neighbor=%neighbor_ip hostname=%hostname"; \
       shellcmd (logger -t SEC-NETWORK-HEALTH \
         "BGP_ADJ_DOWN router=\"%router_host\" neighbor=\"%neighbor_ip\" hostname=\"%hostname\"")

# ---- END (same host + neighbor, Up) ----
ptype2=RegExp
pattern2=\[CISCO_NETWORK_HEALTH\]\s+cisco_mnemonic="BGP-5-ADJCHANGE"\s+host="([^"]+)"\s+srcip="
[^"]*"\s+dstip="[^"]*"\s+srcport="[^"]*"\s+dstport="[^"]*"\s+protocol="[^"]*"\s+.*\bneighbor\s+([0-
9A-Fa-f:.]+)\s+Up\b
context2=BGP_ADJ_$1_$2
desc2=BGP Neighbor $2 Up on $1
action2=eval %up_time ( time() ); \
        eval %outage_duration ( %up_time - %down_time ); \
        write /var/log/logzilla/sec/network-health.log \
          "BGP_ADJ_UP %t router=%router_host neighbor=%neighbor_ip downtime=%outage_duration
hostname=%hostname"; \
        shellcmd (logger -t SEC-NETWORK-HEALTH \
          "BGP_OUTAGE_RESOLVED router=\"%router_host\" neighbor=\"%neighbor_ip\"
downtime=\"%outage_duration\" hostname=\"%hostname\""); \
        delete /tmp/bgp-lookup-%neighbor_ip

# Pairing window (how long to wait for the Up after Down)
window=86400
```

SEC Rule: BGP Flapping Detection

```
# Detect BGP neighbor flapping (multiple up/down cycles)
# 6 state changes within 30 minutes
type=SingleWithThreshold
ptype=RegExp
pattern=\[CISCO_NETWORK_HEALTH\]\s+cisco_mnemonic="BGP-5-ADJCHANGE"\s+host="
```

```
([^"]+)"\s+.*\bneighbor\s+([0-9A-Fa-f:.]+)\s+(Up|Down)
context=BGP_FLAP_$2
desc=BGP neighbor flapping detected
action=eval %hostname ( "$1" ); \
       eval %neighbor ( "$2" ); \
       eval %state ( "$3" ); \
       create BGP_FLAP_%neighbor 1800; \
       write /var/log/logzilla/sec/network-health.log \
         "BGP_FLAP_DETECTED %t neighbor=%neighbor host=%hostname state=%state flap_count=$thresh
window=1800s"; \
       shellcmd (logger -t SEC-NETWORK-HEALTH \
         "BGP_FLAP_DETECTED neighbor=\"%neighbor\" host=\"%hostname\" state=\"%state\"
flap_count=\"$thresh\" window=\"1800s\"")
thresh=5
window=1800
```

Interface Monitoring Correlation

Interface Up/Down Correlation

LogZilla extracts interface names and states from Cisco LINK messages.

Forwarder Configuration

```yaml
# /etc/logzilla/forwarder.d/cisco-interface-health.yaml
window_size: 0
type: sec
sec_name: cisco-interface-health
match:
  - field: cisco_mnemonic
    value:
      - "LINK-3-UPDOWN"
      - "LINK-5-CHANGED"
      - "LINEPROTO-5-UPDOWN"
rules:
- rewrite:
    program: sec-cisco-interface-health
    message: >-
      [CISCO_INTERFACE_HEALTH]
      cisco_mnemonic="${:cisco_mnemonic}"
      host="${:host}"
      $MESSAGE
```

SEC Rule: Interface Flapping Detection

```
# Detect interface flapping (multiple up/down state changes)
# 5 state changes within 10 minutes indicates flapping

type=SingleWithThreshold

ptype=RegExp
pattern=\[CISCO_INTERFACE_HEALTH\]\s+cisco_mnemonic="(?:LINK-3-UPDOWN|LINEPROTO-5-UPDOWN)"\s+host="
([^"]+)"\s+.*\bInterface\s+([^,\s]+).*\bchanged state to (up|down)

context=IFACE_FLAP_$1_$2
desc=Interface flapping detected on $2 at $1
action=eval %hostname ( "$1" ); \
       eval %interface ( "$2" ); \
       eval %state ( "$3" ); \
       create IFACE_FLAP_%hostname_%interface 3600; \
       write /var/log/logzilla/sec/network-health.log \
         "INTERFACE_FLAP_DETECTED %t host=%hostname interface=%interface state=%state
flap_count=$thresh window=600s"; \
       shellcmd (logger -t SEC-INTERFACE-HEALTH \
         "INTERFACE_FLAP_DETECTED host=\"%hostname\" interface=\"%interface\" state=\"%state\"
flap_count=\"$thresh\" window=\"600s\"")

thresh=5
window=600
```

Network Device Health Correlation

Multi-Metric Device Health

Correlate CPU, memory, and interface errors to assess overall device health.

Device Health Forwarder Configuration

```
# /etc/logzilla/forwarder.d/cisco-device-health.yaml
window_size: 0
type: sec
sec_name: cisco-device-health
match:
  - field: cisco_mnemonic
    value:
      - "SYS-1-CPURISINGTHRESHOLD"
      - "SYS-4-FREEMEMLOW"
      - "SYS-4-THRESHOLD_TK"
      - "LINK-3-UPDOWN"
      - "LINEPROTO-5-UPDOWN"
```

```
rules:
  - rewrite:
      message: >-
        [CISCO_DEVICE_HEALTH]
        cisco_mnemonic="${:cisco_mnemonic}"
        host="${:host}"
      $MESSAGE
```

SEC Rule: Device Health Correlation

```
# ============================================
# Cisco Device Health (CPU≥80%, Mem low, Iface down)
# ============================================

type=EventGroup3
context=HOST_$1
window=300

# -------- CPU HIGH (>=80%) --------
ptype=RegExp
pattern=\[CISCO_DEVICE_HEALTH\]\s+cisco_mnemonic="SYS-1-CPURISINGTHRESHOLD"\s+host="
([^"]+)"\b.*\b(8\d|9\d|100)%\b
thresh=3

# -------- MEMORY LOW --------
ptype2=RegExp
pattern2=\[CISCO_DEVICE_HEALTH\]\s+cisco_mnemonic="(?:SYS-4-FREEMEMLOW|SYS-4-THRESHOLD_TK)"\s+host="
([^"]+)"\b.*
thresh2=2

# -------- INTERFACE DOWN --------
ptype3=RegExp
pattern3=\[CISCO_DEVICE_HEALTH\]\s+cisco_mnemonic="(?:LINK-3-UPDOWN|LINEPROTO-5-UPDOWN)"\s+host="
([^"]+)"\b.*?\bInterface\s+([^,]+),\s+changed state to down\b
thresh3=1

desc=Device health degradation on host=$1 (CPU>=80% x3, Mem low x2, IfDown x1 within 5m)
action=write /var/log/logzilla/sec/network-health.log \
         "DEVICE_HEALTH_ALERT %t host=$1 symptoms=cpu_ge80x3,mem_lowx2,ifdownx1"; \
       shellcmd (logger -t SEC-CRITICAL -p local0.crit \
         "DEVICE_HEALTH_ALERT host=\"$1\" correlation=\"cpu>=80x3,mem_lowx2,ifdownx1\"")
```

OSPF Neighbor Correlation

OSPF Adjacency Monitoring

Track OSPF neighbor state changes and detect routing instability.

SEC Rule: OSPF Neighbor Correlation

```
# OSPF Neighbor State Change Tracker
# Tracks OSPF neighbor down/up events and detects instability

type=Pair

# ---- START (neighbor Down) ----
ptype=RegExp
pattern=\[CISCO_NETWORK_HEALTH\]\s+cisco_mnemonic="OSPF-5-ADJCHG"\s+host="([^"]+)"\s+srcip="
([^"]+)"\s+dstip="[^"]*"\s+srcport="[^"]*"\s+dstport="[^"]*"\s+protocol="[^"]*"\s+.*\bto DOWN\b
context=OSPF_DOWN_$1_$2
desc=OSPF neighbor $2 down on $1
action=eval %router_host ( "$1" ); \
       eval %neighbor_id ( "$2" ); \
       eval %down_time ( time() ); \
       create OSPF_DOWN_%router_host_%neighbor_id 7200; \
       write /var/log/logzilla/sec/network-health.log \
         "OSPF_NEIGHBOR_DOWN %t router=%router_host neighbor=%neighbor_id"; \
       shellcmd (logger -t SEC-NETWORK-HEALTH -p local0.notice \
         "OSPF_NEIGHBOR_DOWN router=\"%router_host\" neighbor=\"%neighbor_id\"")

# ---- END (same neighbor to Full) ----
ptype2=RegExp
pattern2=\[CISCO_NETWORK_HEALTH\]\s+cisco_mnemonic="OSPF-5-ADJCHG"\s+host="([^"]+)"\s+srcip="
([^"]+)"\s+dstip="[^"]*"\s+srcport="[^"]*"\s+dstport="[^"]*"\s+protocol="[^"]*"\s+.*\bto FULL\b
context2=OSPF_DOWN_$1_$2
desc2=OSPF neighbor $2 restored on $1
action2=eval %up_time ( time() ); \
        eval %outage_duration ( %up_time - %down_time ); \
        write /var/log/logzilla/sec/network-health.log \
          "OSPF_NEIGHBOR_RESTORED %t router=%router_host neighbor=%neighbor_id
downtime=%outage_duration"; \
        shellcmd (logger -t SEC-NETWORK-HEALTH -p local0.info \
          "OSPF_NEIGHBOR_RESTORED router=\"%router_host\" neighbor=\"%neighbor_id\"
downtime=\"%outage_duration\""); \
        delete OSPF_DOWN_%router_host_%neighbor_id

# Pairing window (2 hours = 7200 seconds)
window=7200
```

Network Security Correlation

Firewall Connection Correlation

Monitor firewall connection patterns for security analysis.

Firewall Monitoring Configuration

Required Apps: `cisco__asa` app (for `SrcIP` and `DstIP` user tags), `sonicwall` app (for `fw_action` user tag)

```
# /etc/logzilla/forwarder.d/cisco-firewall.yaml
window_size: 0
type: sec
sec_name: cisco-firewall
match:
  - field: cisco_mnemonic
    value:
      - "ASA-2-106001"
rules:
- rewrite:
    message: >-
      [CISCO_FIREWALL]
      cisco_mnemonic="${:cisco_mnemonic}"
      host="${:host}"
      srcip="${:SrcIP}"
      dstip="${:DstIP}"
      srcport="${:SrcPort}"
      dstport="${:DstPort}"
      protocol="${:protocol}"
      $MESSAGE
```

SEC Rule: Suspicious Connection Patterns

```
# Port Scan Detection (Cisco ASA)
# Detects when source IP attempts connections to 20+ different destination ports
# within 5 minutes - indicates port scanning activity

type=SingleWithThreshold

ptype=RegExp
pattern=\[CISCO_FIREWALL\]\s+cisco_mnemonic="ASA-2-106001"\s+host="([^"]+)"\s+srcip="
([^"]+)"\s+dstip="([^"]+)"\s+srcport="[^"]*"\s+dstport="([^"]+)"\s+protocol="[^"]*"\s+

context=PORT_SCAN_$2_$4
desc=Port scan detected from $2
action=eval %firewall_host ( "$1" ); \
       eval %src_ip ( "$2" ); \
       eval %dst_ip ( "$3" ); \
       eval %dst_port ( "$4" ); \
       create PORT_SCAN_DETECTED_%src_ip 300 \
         (report PORT_SCAN_DETECTED_%src_ip \
           /bin/bash -c 'logger -t SEC-SECURITY -p local0.alert "PORT_SCAN_DETECTED
src_ip=\"%src_ip\" unique_ports=\"$thresh\" window=\"300s\" - BLOCKING"; \
             iptables -I INPUT -s %src_ip -j DROP 2>/dev/null || true; \
             echo "%src_ip" >> /var/log/logzilla/sec/blocked-ips.log'); \
```

```
        add PORT_SCAN_DETECTED_%src_ip %dst_port

thresh=20
window=300
```

Related Topics

- Event Correlation Rule Types (https://www.logzilla.ai/docs/event-correlation/event-correlation-rule-types)
- Advanced Event Correlation Walkthrough (https://www.logzilla.ai/docs/event-correlation/advanced-event-correlation-walkthrough)
- SOAR Security Orchestration (https://www.logzilla.ai/docs/event-correlation/soar-security-orchestration)