초보자용 Nexus 치트 시트 문제 해결

목차

소개

이 문서에서는 문제를 진단하고 해결하기 위해 사용할 수 있는 Nexus 제품의 트러블슈팅에 사용할 수 있는 여러 가지 툴에 대해 설명합니다.

개요

어떤 툴을 사용할 수 있는지, 어떤 시나리오에서 어떤 툴을 사용하여 최대의 이득을 얻을 수 있는지 파악하는 것이 중요합니다. 사실, 어떤 도구는 단순히 다른 것에 대해 작동하도록 설계되었기 때문에 실현 가능하지 않을 때도 있다.

이 표에는 Nexus 플랫폼에서 문제를 해결할 수 있는 다양한 툴과 그 기능이 정리되어 있습니다. 자세한 내용과 CLI 예는 Nexus 툴 섹션을 참조하십시오.

툴	기능	활용 사례	장점	단점	지속성	영향받는 비행기	사용된 CLI 명령
에트분석기	CPU를 오가는 트래픽 캡처	트래픽 느림 문제, 지연 및 혼잡	느림, 정체 및 지연 문제에 탁월함	일반적으로 컨트롤 플레인 트래픽, 속도 제한만 표시	해당 없음	컨트롤 플레인. 일부 시나리오에서 데이터 플레인에 사용 가능 (SPAN	#ethanalyzer 인이스 인밴드 #ethanalyzer 인이스 [interface I시 필터 [WORD 예: #ethanalyzer 인이스 Ethernet 6/스플레이 필터 I

						-CPU)	
스팬	여러 패킷 캡처 및 미러링	실패 ping s, 무순서 패킷 등	간헐적인 트래픽 손실에 적합	스니퍼 소프트웨어를 실행하는 외부 장치 필요 TCAM 리소스 필요	SPAN 세션을 구성하고 활성화/비활성화해야 합니다.	제어 + 데이터	#monitor 세션 [# #description [NA #source 인터페 [port ID] #destin 인터페이스 [por #no 종료
오류	Broadcom Nexus 장치에 한해 CPU를 오가는 트래픽 캡처	트래픽 느림 문제, 지연 및 혼잡	느림, 정체 및 지연 문제에 탁월함	Broadcom Nexus 디바이스에만 해당됩니다. 속도 제한 (CloudScale Nexus 9k에는 SPAN-to-CPU가 있음)	해당 없음	컨트롤 플레인. 일부 시나리오에서 데이터 평면에 사용 가능	플랫폼에 따라 다 참조 ELAM 개요 - Cis
엘람	Nexus 스위치를 인그레스(또는 Nexus 7K인 경우 이그레스)하는 단일 패킷을 캡처합니다.	패킷이 Nexus에 도달하는지 확인, 전달 결정 확인, 패킷의 변경 확인, 패킷의 인터페이스/VLAN 확인 등	패킷 플로우 및 포워딩 문제에 매우 적합합니다. 불간섭	하드웨어에 대한 심층적인 이해가 필요합니다. 아키텍처별로 고유한 트리거 메커니즘을 활용합니다. 검사할 트래픽을 알고 있는 경우에만 유용합니다.	해당 없음	제어 + 데이터	# attach module [MODULE NUM # debug platform internal <>
Nexus 9k 패킷 추적기	패킷의 경로 탐지	연결 문제 및 패킷 손실	간헐적/완전한 손실에 유용한 플로우 통계에 대한 카운터를 제공합니다. TCAM 조각이 없는 라인 카드에 적합	ARP 트래픽을 캡처할 수 없습니다. Nexus 9k에서만 작동	해당 없음	데이터 + 제어	# test packet-tra src_IP [SOURC dst_IP [DESTINATION test packet-trace start # test pack tracer stop # tes packet-tracer sh
트레이스라우트	L3 홉과 관련된 패킷 경로 탐지	Ping 실패, 호스트/대상/인터넷에 연결할 수 없음 등	L3 장애를 격리하기 위해 경로에서 다양한 홉을 탐지합니다.	L3 경계가 중단된 위치만 식별(문제 자체는 식별하지 않음)	해당 없음	데이터 + 제어	# traceroute [대 인수는 다음과 같 다. 포트, 포트 번호, 인터페이스, vrf, 인터페이스
핑	네트워크	디바이스 간 연결	연결 테스트	호스트에 연결	해당 없음	데이터	# ping [대상 IP]

	의 두 지점 간 연결 테스트	성 테스트	를 위한 빠르고 간단한 툴	할 수 있는지 여부만 식별합니다		+ 제어	인수는 다음과 같다. 개수, 패킷 크기, 인터페이스, 간격, 티캐스트, 루프백 간 초과
PACL/RACL/VACL	특정 포트 또는 VLAN에서 트래픽 인그레스/이그레스 캡처	호스트 간의 간헐적인 패킷 손실, 패킷이 Nexus에 도착/이탈하는지 확인 등	간헐적인 트래픽 손실에 적합	TCAM 리소스가 필요합니다. 일부 모듈의 경우 수동 TCAM 조각이 필요합니다	영구(적용 대상 running-설정)	데이터 + 제어	# ip access-list [NAME] # ip port access-group [ACL NAME] # ip access-group [ACL NAME] 인수는 다음과 같다. 거부, 조각, 아니용, 설명, 표시, 끝, 종료, pop, 여기서
로그플래시	디바이스 다시 로드에 관계없이 로그 계정, 충돌 파일 및 이벤트와 같은 스위치에 대한 기록 데이터를 전역적으로 저장합니다	갑작스러운 디바이스 다시 로드/종료, 디바이스가 다시 로드될 때마다 로그 플래시 데이터가 분석에 도움이 될 수 있는 몇 가지 정보를 제공합니다	디바이스 다시 로드 시 정보 유지(영구 스토리지)	Nexus 7K의 외부 = 이러한 로그를 수집하려면 수퍼바이저 플랫폼에 설치/통합해야 함 (logflash는 내부 저장 장치의 파티션이므로 3K/9K에는 con이 적용되지 않습니다.)	Reload-Persistent	데이터 + 제어	# dir logflash:
OBFL	장애 및 환경 정보와 같은 특정 모듈에 대한 기록 데이터를 저장합니다.	갑작스러운 디바이스 다시 로드/종료, 디바이스가 다시 로드될 때마다 플래시 데이터를 로깅하면 몇 가지 유용한 정보를 얻을 수 있습니다	디바이스 다시 로드 시 정보 유지(영구 스토리지)	제한된 수의 읽기 및 쓰기 지원	Reload-Persistent	데이터 + 제어	# show logging onboard module 인수는 다음과 같다. boot-uptime, card boot-history, card first-power-on, counter-stats, de version, endtime environment-his error-stats, exception-log, internal, interrup stats, obfl-histor stat-trace, startt status
이벤트 기록	현재 실행 중인 특정 프로세스에	Nexus의 모든 프로세스에는 CDP, STP, OSPF, EIGRP, BGP,	Nexus에서 실행 중인 특정 프로세스 트러블슈팅	디바이스가 다시 로드되면 정보가 손실됩니다(비영구적).	비영구	데이터 + 제어	# show [PROCE internal event-hi [ARGUMENT] 인수는 다음과 같

	대한 정보가 필요한 경우	vPC, LACP 등과 같은 고유한 이벤트 기록이 있습니다					다. 인접성, cli, 이벤 러딩, ha, hello, lsa, msgs, objst redistribution, ri segrt, spf, spf-tr statistics,
디버그	특정 프로세스에 더 세부적인 실시간/라이브 정보가 필요한 경우	CDP, STP, OSPF, IGRP, BGP, vPC, LACP 등 Nexus의 모든 프로세스에서 디버그를 수행할 수 있습니다	Nexus에서 실행되는 특정 프로세스의 문제를 실시간으로 해결하여 더욱 세분화	네트워크 성능에 영향을 미칠 수 있음	비영구	데이터 + 제어	# 디버그 프로세 로세스] 예: # debug ip ospf
골드	하드웨어 구성 요소(예: I/O 및 수퍼바이저 모듈)에 대한 부팅, 런타임 및 온디맨드 진단 제공	USB, Bootflash, OBFL, ASIC 메모리, PCIE, 포트 루프백, NVRAM 등의 테스트 하드웨어	하드웨어의 결함을 감지하고 릴리스 6(2)8 이상에서만 필요한 시정 조치를 취할 수 있음	하드웨어 문제만 감지	비영구	해당 없음	# show diagnos content module show diagnostic description mod [#] test all
EEM	디바이스에서 이벤트를 모니터링하고 필요한 작업을 수행합니다.	인터페이스 종료, 팬 오작동, CPU 사용률 등과 같은 일부 작업/해결/알림이 필요한 디바이스 활동	Python 스크립트 지원	EEM을 구성하려면 네트워크 관리자 권한이 있어야 합니다.	EEM 스크립트 및 트리거가 컨피그레이션에 있음	해당 없음	다양합니다. 참조 Embedded Even Manager 구성

Nexus 툴

다양한 명령 및 해당 구문 또는 옵션에 대한 자세한 설명은 다음을 참조하십시오. Cisco Nexus 9000 Series 스위치 - 명령 참조 - Cisco.

- **에트분석기**

Ethanalyzer는 패킷 CPU 트래픽을 캡처하도록 설계된 NX-OS 툴입니다. 인그레스(ingress) 또는 이그레스(egress)를 불문하고 CPU에 도달하는 모든 것은 이 툴로 캡처할 수 있습니다. 널리 사용되는 오픈 소스 네트워크 프로토콜 분석기 Wireshark를 기반으로 합니다. 이 도구에 대한 자세한 내용은 Nexus 7000의 Ethanalyzer 문제 해결 가이드 - Cisco를 참조하십시오

일반적으로 Ethanalyzer는 수퍼바이저와 주고받는 모든 트래픽을 캡처합니다. 즉, 인터페이스별 캡처를 지원하지 않습니다. 특정 인터페이스 개선 사항은 최신 코드 포인트에서 일부 플랫폼에 사용할 수 있습니다. 또한 Ethanalyzer는 하드웨어 스위칭이 아닌 CPU 스위칭된 트래픽만 캡처합니다. 예를 들어, 인밴드 인터페이스, 관리 인터페이스 또는 전면 패널 포트(지원되는 경우)에서 트래픽을 캡처할 수 있습니다.

```
Nexus9000_A(config-if-range)# ethanalyzer local interface inband
Capturing on inband
2020-02-18 01:40:55.183177 cc:98:91:fc:55:8b -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/cc:98:91:fc:55:80  Cost = 0  Port = 0x800b
2020-02-18 01:40:55.184031 f8:b7:e2:49:2d:f2 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:40:55.184096 f8:b7:e2:49:2d:f5 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:40:55.184147 f8:b7:e2:49:2d:f4 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:40:55.184190 f8:b7:e2:49:2d:f3 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:40:55.493543 dc:f7:19:1b:f9:85 -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/dc:f7:19:1b:f9:80  Cost = 0  Port = 0x8005
2020-02-18 01:40:56.365722      0.0.0.0 -> 255.255.255.255 DHCP DHCP Discover - Transaction ID
0xc82a6d3
2020-02-18 01:40:56.469094 f8:b7:e2:49:2d:b4 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:40:57.202658 cc:98:91:fc:55:8b -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/cc:98:91:fc:55:80  Cost = 0  Port = 0x800b
2020-02-18 01:40:57.367890      0.0.0.0 -> 255.255.255.255 DHCP DHCP Discover - Transaction ID
0xc82a6d3
10 packets captured

Nexus9000_A(config-if-range)# ethanalyzer local interface mgmt
Capturing on mgmt0
2020-02-18 01:53:07.055100 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:09.061398 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:11.081596 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:13.080874 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:15.087361 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:17.090164 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:19.096518 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:20.391215 00:be:75:5b:d9:00 -> 01:00:0c:cc:cc:cc CDP Device ID:
Nexus9000_A(FDO21512ZES)  Port ID: mgmt0
2020-02-18 01:53:21.119464 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
2020-02-18 01:53:23.126011 cc:98:91:fc:55:94 -> 01:80:c2:00:00:00 STP RST. Root =
32768/46/84:8a:8d:7d:a2:80  Cost = 4  Port = 0x8014
10 packets captured

Nexus9000-A# ethanalyzer local interface front-panel eth1/1
Capturing on 'Eth1-1'
1 2022-07-15 19:46:04.698201919 28:ac:9e:ad:5c:b8  01:80:c2:00:00:00 STP 53 RST. Root =
32768/1/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
2 2022-07-15 19:46:04.698242879 28:ac:9e:ad:5c:b8  01:00:0c:cc:cc:cd STP 64 RST. Root =
32768/1/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
3 2022-07-15 19:46:04.698314467 28:ac:9e:ad:5c:b8  01:00:0c:cc:cc:cd STP 64 RST. Root =
```

```
32768/10/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
4 2022-07-15 19:46:04.698386112 28:ac:9e:ad:5c:b8  01:00:0c:cc:cc:cd STP 64 RST. Root =
32768/20/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
5 2022-07-15 19:46:04.698481274 28:ac:9e:ad:5c:b8  01:00:0c:cc:cc:cd STP 64 RST. Root =
32768/30/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
6 2022-07-15 19:46:04.698555784 28:ac:9e:ad:5c:b8  01:00:0c:cc:cc:cd STP 64 RST. Root =
32768/40/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
7 2022-07-15 19:46:04.698627624 28:ac:9e:ad:5c:b8  01:00:0c:cc:cc:cd STP 64 RST. Root =
32768/50/28:ac:9e:ad:5c:b7 Cost = 0 Port = 0x8001
```

이 출력은 Ethanalyzer로 캡처할 수 있는 메시지 중 일부를 보여줍니다. 기본적으로 Ethanalyzer는 최대 10개의 패킷만 캡처합니다. 그러나 이 명령을 사용하여 CLI에서 패킷을 무기한 캡처하도록 프롬프트를 표시할 수 있습니다. Ctrl+C를 사용하여 캡처 모드를 종료합니다.

```
Nexus9000_A(config-if-range)# ethanalyzer local interface inband limit-captured-frames 0
Capturing on inband
2020-02-18 01:43:30.542588 f8:b7:e2:49:2d:f2 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:30.542626 f8:b7:e2:49:2d:f5 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:30.542873 f8:b7:e2:49:2d:f4 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:30.542892 f8:b7:e2:49:2d:f3 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:31.596841 dc:f7:19:1b:f9:85 -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/dc:f7:19:1b:f9:80  Cost = 0  Port = 0x8005
2020-02-18 01:43:31.661089 f8:b7:e2:49:2d:b2 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:31.661114 f8:b7:e2:49:2d:b3 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:31.661324 f8:b7:e2:49:2d:b5 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:31.776638 cc:98:91:fc:55:8b -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/cc:98:91:fc:55:80  Cost = 0  Port = 0x800b
2020-02-18 01:43:33.143814 f8:b7:e2:49:2d:b4 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:33.596810 dc:f7:19:1b:f9:85 -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/dc:f7:19:1b:f9:80  Cost = 0  Port = 0x8005
2020-02-18 01:43:33.784099 cc:98:91:fc:55:8b -> 01:80:c2:00:00:00 STP RST. Root =
32768/1/cc:98:91:fc:55:80  Cost = 0  Port = 0x800b
2020-02-18 01:43:33.872280 f8:b7:e2:49:2d:f2 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:33.872504 f8:b7:e2:49:2d:f5 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
2020-02-18 01:43:33.872521 f8:b7:e2:49:2d:f4 -> 01:80:c2:00:00:0e LLC U, func=UI; SNAP, OUI
0x00000C (Cisco), PID 0x0134
15 packets captured
```

Ethanalyzer에서 필터를 사용하여 특정 트래픽에 집중할 수도 있습니다. ethanalzyer와 함께 사용할 수 있는 두 가지 유형의 필터는 Capture 필터와 Display 필터로 알려져 있습니다. 캡처 필터는 캡처 필터에 정의된 기준과 일치하는 트래픽만 캡처합니다. 표시 필터는 모든 트래픽을 캡처하지만 표시 필터에 정의된 기준과 일치하는 트래픽만 표시됩니다.

```
Nexus9000_B# ping 10.82.140.106 source 10.82.140.107 vrf management count 2
PING 10.82.140.106 (10.82.140.106) from 10.82.140.107: 56 data bytes
64 bytes from 10.82.140.106: icmp_seq=0 ttl=254 time=0.924 ms
64 bytes from 10.82.140.106: icmp_seq=1 ttl=254 time=0.558 ms

Nexus9000_A(config-if-range)# ethanalyzer local interface mgmt display-filter icmp
Capturing on mgmt0
2020-02-18 01:58:04.403295 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
```

```
2020-02-18 01:58:04.403688 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply
2020-02-18 01:58:04.404122 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 01:58:04.404328 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply

4 packets captured
```

또한 Wireshark에서와 마찬가지로 detail 옵션을 사용하여 패킷을 캡처하고 터미널에서 볼 수 있습니다. 그러면 패킷 분할 결과를 기반으로 전체 헤더 정보를 볼 수 있습니다. 예를 들어, 프레임이 암호화된 경우 암호화된 페이로드를 볼 수 없습니다. 다음 예를 참조하십시오.

```
Nexus9000_A(config-if-range)# ethanalyzer local interface mgmt display-filter icmp detail
Capturing on mgmt0
Frame 2 (98 bytes on wire, 98 bytes captured)
    Arrival Time: Feb 18, 2020 02:02:17.569801000
    [Time delta from previous captured frame: 0.075295000 seconds]
    [Time delta from previous displayed frame: 0.075295000 seconds]
    [Time since reference or first frame: 0.075295000 seconds]
    Frame Number: 2
    Frame Length: 98 bytes
    Capture Length: 98 bytes
    [Frame is marked: False]
    [Protocols in frame: eth:ip:icmp:data]
Ethernet II, Src: 00:be:75:5b:de:00 (00:be:75:5b:de:00), Dst: 00:be:75:5b:d9:00
(00:be:75:5b:d9:00)
    Destination: 00:be:75:5b:d9:00 (00:be:75:5b:d9:00)
        Address: 00:be:75:5b:d9:00 (00:be:75:5b:d9:00)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    Type: IP (0x0800)
>>>>>>>Output Clipped
```

Ethanalyzer를 사용하면 다음을 수행할 수 있습니다.

- 다양한 대상 파일 시스템의 지정된 파일 이름에 출력(PCAP 파일)을 기록합니다. bootflash, logflash, USB 등 그런 다음 저장된 파일을 디바이스 외부로 전송하고 필요에 따라 Wireshark에서 볼 수 있습니다.
- bootflash에서 파일을 읽고 터미널에 표시합니다. CPU 인터페이스에서 직접 읽을 때와 마찬가지로 detail 키워드를 사용하면 전체 패킷 정보를 표시할 수 있습니다.

다양한 인터페이스 소스 및 출력 옵션에 대해서는 이 예제를 참조하십시오.

```
Nexus9000_A# ethanalyzer local interface mgmt capture-filter "host 10.82.140.107" write
bootflash:TEST.PCAP
Capturing on mgmt0
10
Nexus9000_A# dir bootflash:
       4096    Feb 11 02:59:04 2020  .rpmstore/
       4096    Feb 12 02:57:36 2020  .swtam/
       2783    Feb 17 21:59:49 2020  09b0b204-a292-4f77-b479-1ca1c4359d6f.config
       1738    Feb 17 21:53:50 2020  20200217_215345_poap_4168_init.log
       7169    Mar 01 04:41:55 2019  686114680.bin
       4411    Nov 15 15:07:17 2018  EBC-SC02-M2_303_running_config.txt
   13562165    Oct 26 06:15:35 2019  GBGBLD4SL01DRE0001-CZ07-
        590    Jan 10 14:21:08 2019  MDS20190110082155835.lic
       1164    Feb 18 02:18:15 2020  TEST.PCAP
>>>>>>>Output Clipped

Nexus9000_A# copy bootflash: ftp:
Enter source filename: TEST.PCAP
Enter vrf (If no input, current vrf 'default' is considered): management
Enter hostname for the ftp server: 10.122.153.158
```

```
Enter username: calo
Password:
***** Transfer of file Completed Successfully *****
Copy complete, now saving to disk (please wait)...
Copy complete.

Nexus9000_A# ethanalyzer local read bootflash:TEST.PCAP
2020-02-18 02:18:03.140167 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 02:18:03.140563 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply
2020-02-18 02:18:15.663901 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 02:18:15.664303 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply
2020-02-18 02:18:15.664763 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 02:18:15.664975 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply
2020-02-18 02:18:15.665338 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 02:18:15.665536 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply
2020-02-18 02:18:15.665864 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 02:18:15.666066 10.82.140.106 -> 10.82.140.107 ICMP Echo (ping) reply

RTP-SUG-BGW-1# ethanalyzer local interface front-panel eth1-1 write bootflash:e1-1.pcap
Capturing on 'Eth1-1'
10

RTP-SUG-BGW-1# ethanalyzer local read bootflash:e1-1.pcap detail
Frame 1: 53 bytes on wire (424 bits), 53 bytes captured (424 bits) on interface Eth1-1, id 0
    Interface id: 0 (Eth1-1)
        Interface name: Eth1-1
    Encapsulation type: Ethernet (1)
    Arrival Time: Jul 15, 2022 19:59:50.696219656 UTC
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1657915190.696219656 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 53 bytes (424 bits)
    Capture Length: 53 bytes (424 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:llc:stp]
```

- **스팬**

SPAN은 SwitchPort Analyzer의 약자로, 인터페이스에서 모든 트래픽을 캡처하고 해당 트래픽을 대상 포트로 미러링하는 데 사용됩니다. 목적지 포트는 일반적으로 Wireshark를 실행하는 PC와 같은 네트워크 분석기 툴에 연결되며, 이를 통해 해당 포트를 통과하는 트래픽을 분석할 수 있습니다. 단일 포트 또는 여러 포트 및 VLAN의 트래픽에 대해 SPAN할 수 있습니다.

SPAN 세션에는 소스 포트 및 대상 포트가 포함됩니다. 소스 포트는 이더넷 포트(하위 인터페이스 없음), 포트 채널, 수퍼바이저 인밴드 인터페이스일 수 있으며 동시에 목적지 포트가 될 수 없습니다. 또한 9300 및 9500 플랫폼과 같은 일부 디바이스에서는 FEX(Fabric Extender) 포트도 지원됩니다. 대상 포트는 이더넷 포트(액세스 또는 트렁크), 포트 채널(액세스 또는 트렁크)일 수 있으며 9300 업링크 포트와 같은 일부 장치의 경우 지원되지만 FEX 포트는 대상이 지원되지 않습니다.

여러 SPAN 세션을 인그레스/이그레스/둘 다로 구성할 수 있습니다. 개별 디바이스에서 지원할 수 있는 총 SPAN 세션 수에는 제한이 있습니다. 예를 들어, Nexus 9000은 최대 32개의 세션을 지원할 수 있지만 Nexus 7000은 16개만 지원할 수 있습니다. CLI에서 확인하거나 사용하는 제품에 대한 SPAN 컨피그레이션 가이드를 참조하십시오.

각 NX-OS 릴리스와 제품 유형에 대해 지원되는 인터페이스 유형과 기능은 서로 다릅니다. 사용하는 제품 및 버전에 대한 최신 구성 지침 및 제한 사항을 참조하십시오. 다음은 각각 Nexus 9000 및

Nexus 7000에 대한 링크입니다.

Cisco Nexus 9000 Series NX-OS System Management 컨피그레이션 가이드, 릴리스 9.3(x) - SPAN 구성 [Cisco Nexus 9000 Series 스위치] - Cisco

Cisco Nexus 7000 Series NX-OS 시스템 관리 컨피그레이션 가이드 - SPAN 구성 [Cisco Nexus 7000 Series 스위치] - Cisco

다양한 유형의 SPAN 세션이 있습니다. 몇 가지 일반적인 유형은 다음과 같습니다.

- Local SPAN(로컬 SPAN): 소스 호스트와 대상 호스트가 모두 스위치에 로컬인 SPAN 세션의 유형입니다. 다시 말해, SPAN 세션을 설정하는 데 필요한 모든 컨피그레이션이 단일 스위치에 적용되며, 이는 소스 및 목적지 호스트 포트가 있는 스위치와 동일합니다.
- RSPAN(Remote SPAN): 소스 및 대상 호스트가 스위치에 로컬이 아닌 SPAN 세션의 유형입니다. 즉, 하나의 스위치에서 소스 RSPAN 세션을 구성하고 대상 스위치에서 대상 RSPAN을 구성한 다음 RSPAN VLAN과의 연결을 확장합니다.

참고: RSPAN은 Nexus에서 지원되지 않음

- 확장된 원격 SPAN(ERSPAN): 스위치는 복사된 프레임을 GRE(Generic Routing Encapsulation) 터널 헤더로 캡슐화하고 패킷을 구성된 대상으로 라우팅합니다. 캡슐화 및 캡슐화 해제 스위치(서로 다른 두 디바이스)에서 소스 및 대상 세션을 구성합니다. 이를 통해 레이어 3 네트워크를 통해 트래픽을 스팬할 수 있습니다.
- SPAN-to-CPU: 대상 포트가 수퍼바이저 또는 CPU인 특수한 유형의 SPAN 세션에 지정된 이름입니다. 로컬 SPAN 세션의 형태이며 표준 SPAN 세션을 활용할 수 없는 경우에 사용할 수 있습니다. 몇 가지 일반적인 이유는 다음과 같습니다. 사용 가능하거나 적합한 SPAN 대상 포트가 없거나, 사이트에 액세스할 수 없거나 관리되지 않는 사이트, SPAN 대상 포트에 연결할 수 있는 장치가 없습니다. 자세한 내용은 Nexus 9000 Cloud Scale ASIC NX-OS SPAN-to-CPU Procedure - Cisco 링크를 참조하십시오. SPAN-to CPU는 CoPP(Control Plane Policing)에 의해 속도가 제한되므로 sniffing 폴리서를 초과하는 하나 이상의 소스 인터페이스가 SPAN-CPU 세션을 드롭할 수 있습니다. 이 경우 데이터가 유선 데이터의 100% 반영되지 않으므로 CPU에 대한 SPAN이 높은 데이터 속도 및/또는 간헐적 손실이 있는 문제 해결 시나리오에 적합하지 않습니다. SPAN to CPU 세션을 구성하고 관리를 활성화한 후에는 Ethanalyzer를 실행하여 CPU로 전송된 트래픽을 확인하여 그에 따라 분석을 수행해야 합니다.

다음은 Nexus 9000 스위치에서 간단한 로컬 SPAN 세션을 구성하는 방법의 예입니다.

```
Nexus9000_A(config-monitor)# monitor session ?

*** No matching command found in current mode, matching in (config) mode ***
  <1-32>
  all     All sessions

Nexus9000_A(config)# monitor session 10
Nexus9000_A(config-monitor)#?
  description  Session description (max 32 characters)
  destination  Destination configuration
  filter       Filter configuration
  mtu          Set the MTU size for SPAN packets
  no           Negate a command or set its defaults
  show         Show running system information
  shut         Shut a monitor session
  source       Source configuration
  end          Go to exec mode
```

```
  exit         Exit from command interpreter
  pop          Pop mode from stack or restore from name
  push         Push current mode to stack or save it under name
  where        Shows the cli context you are in

Nexus9000_A(config-monitor)# description Monitor_Port_e1/1
Nexus9000_A(config-monitor)# source interface ethernet 1/1
Nexus9000_A(config-monitor)# destination interface ethernet 1/10
Nexus9000_A(config-monitor)# no shut
```

다음 예에서는 실행된 CPU 세션에 대한 SPAN의 컨피그레이션을 보여 주고, Ethanalyzer를 사용하여 트래픽을 캡처합니다.

```
N9000-A#show run monitor

monitor session 1
source interface Ethernet1/7 rx
destination interface sup-eth0 << this is what sends the traffic to CPU
no shut

RTP-SUG-BGW-1# ethanalyzer local interface inband mirror limit-c 0
Capturing on 'ps-inb'
2020-02-18 02:18:03.140167 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
2020-02-18 02:18:15.663901 10.82.140.107 -> 10.82.140.106 ICMP Echo (ping) request
```

- **거울**

Dmirror는 Broadcom 기반 Nexus 플랫폼을 위한 SPAN-CPU 세션 유형입니다. 이 개념은 SPAN-to-CPU와 동일하며 속도가 50pps(초당 패킷)로 제한됩니다. 이 기능은 bcm-셸 CLI를 사용하여 내부 데이터 경로를 디버깅하도록 구현되었습니다. 관련 제한 때문에 사용자가 Sup에 대한 SPAN 세션을 구성할 수 있도록 허용하는 NX-OS CLI는 없으며, 이는 제어 트래픽에 영향을 미치고 CoPP 클래스를 사용할 수 있습니다.

- **엘람**

ELAM은 Embedded Logic Analyzer Module을 의미합니다. ASIC를 살펴보고 **단일** 패킷에 대해 어떤 포워딩 결정을 내렸는지 확인할 수 있는 기능을 제공합니다. 따라서 ELAM을 사용하면 패킷이 포워딩 엔진에 도착하는지 여부와 어떤 포트/VLAN 정보를 확인할 수 있습니다. 또한 L2 - L4 패킷 구조 및 패킷에 변경이 있었는지 여부를 확인할 수 있습니다.

ELAM은 아키텍처에 따라 다르며, 패킷을 캡처하는 절차는 내부 아키텍처에 따라 플랫폼마다 다르다는 점을 이해하는 것이 중요합니다. 툴을 올바르게 적용하려면 하드웨어의 ASIC 매핑을 알고 있어야 합니다. Nexus 7000의 경우, 단일 패킷에 대해 두 개의 캡처가 취해지는데, 하나는 결정이 **데이터 버스(DBUS)로** 이루어지기 전이고, 다른 하나는 결정이 **결과 버스(RBUS)로** 이루어진 후입니다. DBUS 정보를 볼 때 패킷이 수신된 위치 및 레이어 2~4 정보를 볼 수 있습니다. RBUS의 결과로 패킷이 어디로 전달되었는지, 그리고 프레임이 변경되었는지 확인할 수 있습니다. DBUS 및 RBUS에 대한 트리거를 설정하고 준비가 되었는지 확인한 다음 패킷을 실시간으로 캡처해야 합니다. 다양한 라인 카드의 절차는 다음과 같습니다.

다양한 ELAM 절차에 대한 자세한 내용은 다음 표의 링크를 참조하십시오.

ELAM 개요	ELAM 개요 - Cisco
Nexus 7K F1 Module	Nexus 7000 F1 모듈 ELAM 절차 - Cisco
Nexus 7K F2 Module	Nexus 7000 F2 모듈 ELAM 절차 - Cisco
Nexus 7K F3 Module	F3- ELAM 예

Nexus 7K M Module	Nexus 7000 M-Series Module ELAM 절차 - Cisco
Nexus 7K M1/M2 및 F2 모듈	M1/M2 및 F2 및 Ethanalyzer용 Nexus 7K ELAM
Nexus 7K M3 Module	Nexus 7000 M3 모듈 ELAM 절차 - Cisco

ELAM for Nexus 7000 - M1/M2(Eureka 플랫폼)

- **show** module 명령을 사용하여 모듈 번호를 **확인합니다**.
- 모듈 x를 **연결하여 모듈**에 연결합니다. 여기서 x는 모듈 번호입니다.
- **show hardware internal dev-port-map** 명령을 사용하여 내부 ASIC 매핑을 확인하고 L2LKP 및 L3LKP를 확인합니다.

```
Nexus7000(config)#show module
Mod  Ports  Module-Type                        Model              Status
---  -----  ---------------------------------- ------------------ ----------
1    0      Supervisor Module-2                N7K-SUP2E          active *
2    0      Supervisor Module-2                N7K-SUP2E          ha-standby
3    48     1/10 Gbps Ethernet Module          N7K-F248XP-25E     ok
4    24     10 Gbps Ethernet Module            N7K-M224XP-23L     ok


Nexus7000(config)# attach module 4
Attaching to module 4 ...
To exit type 'exit', to abort type '$.'
Last login: Fri Feb 14 18:10:21 UTC 2020 from 127.1.1.1 on pts/0

module-4# show hardware internal dev-port-map
--------------------------------------------------------------
CARD_TYPE:       24 port 10G
>Front Panel ports:24
--------------------------------------------------------------
 Device name             Dev role              Abbr num_inst:
--------------------------------------------------------------
> Skytrain               DEV_QUEUEING           QUEUE  4
> Valkyrie               DEV_REWRITE            RWR_0  4
> Eureka                 DEV_LAYER_2_LOOKUP     L2LKP  2
> Lamira                 DEV_LAYER_3_LOOKUP     L3LKP  2
> Garuda                 DEV_ETHERNET_MAC       MAC_0  2
> EDC                    DEV_PHY                PHYS   6
> Sacramento Xbar ASIC   DEV_SWITCH_FABRIC      SWICHF 1
+-----------------------------------------------------------------------+
+----------------+++FRONT PANEL PORT TO ASIC INSTANCE MAP+++------------+
+-----------------------------------------------------------------------+
FP port |  PHYS | SECUR | MAC_0 | RWR_0 | L2LKP | L3LKP | QUEUE |SWICHF
   1       0       0       0       0,1     0       0       0,1     0
   2       0       0       0       0,1     0       0       0,1     0
   3       0       0       0       0,1     0       0       0,1     0
   4       0       0       0       0,1     0       0       0,1     0
   5       1       0       0       0,1     0       0       0,1     0
   6       1       0       0       0,1     0       0       0,1     0
   7       1       0       0       0,1     0       0       0,1     0
   8       1       0       0       0,1     0       0       0,1     0
   9       2       0       0       0,1     0       0       0,1     0
   10      2       0       0       0,1     0       0       0,1     0
   11      2       0       0       0,1     0       0       0,1     0
   12      2       0       0       0,1     0       0       0,1     0
   13      3       1       1       2,3     1       1       2,3     0
```

```
   14      3       1       1       2,3     1       1       2,3     0
   15      3       1       1       2,3     1       1       2,3     0
   16      3       1       1       2,3     1       1       2,3     0
   17      4       1       1       2,3     1       1       2,3     0
   18      4       1       1       2,3     1       1       2,3     0
   19      4       1       1       2,3     1       1       2,3     0
   20      4       1       1       2,3     1       1       2,3     0
   21      5       1       1       2,3     1       1       2,3     0
   22      5       1       1       2,3     1       1       2,3     0
   23      5       1       1       2,3     1       1       2,3     0
   24      5       1       1       2,3     1       1       2,3     0
+-------------------------------------------------------------------+
+-------------------------------------------------------------------+
```

- 먼저 패킷을 L2에 캡처하여 전달 결정이 올바른지 확인합니다. 이렇게 하려면 L2LKP 매핑 열을 살펴보고 포트에 해당하는 ASIC 인스턴스 번호를 식별합니다.
- 그런 다음 elam asic eureka instance **x** 명령을 사용하여 이 인스턴스에서 **ELAM을 실행합니다** .여기서 x는 ASIC 인스턴스 번호이며 DBUS 및 RBUS에 대한 트리거를 구성합니다. 명령 상태와 함께 트리거 상태를 **확인하고** 트리거가 구성되었는지 확인합니다.

```
module-4(eureka-elam)# trigger dbus dbi ingress ipv4 if source-ipv4-address 192.0.2.2
destination-ipv4-address 192.0.2.4 rbi-corelate
module-4(eureka-elam)# trigger rbus rbi pb1 ip if cap2 1

module-4(eureka-elam)# status

Slot: 4, Instance: 1
EU-DBUS: Configured
trigger dbus dbi ingress ipv4 if source-ipv4-address 192.168.10.1
EU-RBUS: Configured
trigger rbus rbi pb1 ip if cap2 1
```

- 명령 **start**로 트리거를 활성화하고 명령 상태의 트리거 상태를 확인하여 **트리거**가 작동하고 있는지 확인합니다.

```
module-4(eureka-elam)# start
module-4(eureka-elam)# status

Slot: 4, Instance: 1 EU-DBUS: Armed <<<<<<<<<<
trigger dbus dbi ingress ipv4 if source-ipv4-address 192.168.10.1
EU-RBUS: Armed <<<<<<<<<<
trigger rbus rbi pb1 ip if cap2 1
```

- 상태가 트리거가 무장된 것으로 표시되면 캡처할 준비가 된 것입니다. 이 때 트리거가 실제로 트리거되었는지 확인하려면 트래픽을 전송하고 상태를 다시 확인해야 합니다.

```
module-4(eureka-elam)# status

Slot: 4, Instance: 1
EU-DBUS: Triggered    <<<<<<<<<<trigger dbus dbi ingress ipv4 if source-ipv4-address
192.168.10.1 EU-RBUS: Triggered <<<<<<<<<<
trigger rbus rbi pb1 ip if cap2 1
```

- 트리거되면 rbus 및 dbus 모두에 대한 패킷 시퀀스 번호를 확인하여 둘 다 동일한 패킷을 캡처했는지 확인합니다. 이 작업은 show dbus 명령을 사용하여 수행할 수 **있습니다 | 시퀀스 ; RBUS 표시 | 시퀀스**. 시퀀스 번호가 일치하면 dbus 및 rbus의 내용을 볼 수 있습니다. 그렇지 않은 경우 동일한 패킷을 캡처할 수 있을 때까지 캡처를 다시 실행합니다.

참고: 더 정확하게 전달하려면 항상 ELAM을 여러 번 실행하여 전달 문제를 확인합니다.

- show dbus 및 show rbus 명령을 사용하여 rbus 및 dbus의 **내용**을 볼 **수 있습니다**. 캡처에서 중요한 것은 시퀀스 번호와 소스/대상 인덱스입니다. Dbus는 패킷을 수신한 포트를 알려주는 소스 인덱스를 보여줍니다. Rbus는 패킷이 전달되는 포트의 목적지 인덱스를 보여줍니다. 또한 소스 및 목적지 IP/MAC 주소와 VLAN 정보도 확인할 수 있습니다.
- 소스 및 대상 인덱스(LTL 인덱스라고도 함)를 사용하면 **show system internal pixm info ltl #** 명령을 사용하여 연결된 전면 패널 포트를 확인할 수 있습니다.

ELAM for Nexus 7000 - M1/M2(Lamira 플랫폼)

절차는 Lamira 플랫폼에서도 동일하지만 몇 가지 차이점이 있습니다.

- 키워드 Lamira elam asic lamira 인스턴스 **x로 ELAM을 실행합니다**.
- ELAM을 트리거하는 명령은 다음과 같습니다.

```
module-4(lamira-elam)#trigger dbus ipv4 if source-ipv4-address 192.0.2.2 destination-ipv4-
address 192.0.2.4
module-4(lamira-elam)# trigger rbus
```

- status 명령을 사용하여 상태를 **확인하고 트래픽**을 보내기 전에 Armed인지, 캡처한 후에 트리거되는지 확인합니다.
- 그런 다음 Eureka에 표시된 것과 유사한 방식으로 dbus 및 show bus의 출력을 해석할 수 있습니다.

ELAM for Nexus 7000 - F2/F2E(Clipper 플랫폼)

다시, 절차는 비슷하며, 트리거만 다릅니다. 몇 가지 차이점은 다음과 같습니다.

- Clipper elam asic clipper 인스턴스 **x 키워드로 ELAM을 실행하고** Layer 2 또는 Layer 3 모드를 지정합니다.

```
module-4# elam asic  clipper instance 1
module-4(clipper-elam)#
```

- ELAM을 트리거하는 명령은 다음과 같습니다.

```
module-4(clipper-l2-elam)# trigger dbus ipv4 ingress if source-ipv4-address 192.0.2.3
destination-ipv4-address 192.0.2.2
module-4(clipper-l2-elam)# trigger rbus ingress if trig
```

- status 명령을 사용하여 상태를 **확인하고 트래픽**을 보내기 전에 Armed인지, 캡처한 후에 트리거되는지 확인합니다.
- 그런 다음 Eureka에 표시된 것과 유사한 방식으로 dbus 및 show bus의 출력을 해석할 수 있습니다.

ELAM for Nexus 7000 - F3(Flanker Platform)

다시 말하지만, 절차는 비슷하며, 트리거만 다릅니다. 몇 가지 차이점은 다음과 같습니다.

- Flanker elam asic flanker 인스턴스 **x 키워드로 ELAM을 실행하고** Layer 2 또는 Layer 3 모드를 지정합니다.

```
module-4# elam asic flanker instance 1
module-4(flanker-elam)#
```

- ELAM을 트리거하는 명령은 다음과 같습니다.

```
module-9(fln-l2-elam)# trigger dbus ipv4 if destination-ipv4-address 10.1.1.2
module-9(fln-l2-elam)# trigger rbus ingress if trig
```

- **status** 명령으로 상태를 확인하고 트래픽을 보내기 전에 Armed인지, 캡처한 후에 트리거되는지 확인합니다.
- 그런 다음 Eureka에 표시된 것과 유사한 방식으로 dbus 및 rbus의 출력을 해석할 수 있습니다.

ELAM for Nexus 9000(Tahoe 플랫폼)

Nexus 9000에서는 절차가 Nexus 7000과 조금 다릅니다. Nexus 9000의 경우 Nexus 9000 Cloud Scale ASIC(Tahoe) NX-OS ELAM - Cisco 링크를 참조하십시오.

- 먼저 show hardware internal tah interface **#** 명령을 사용하여 **인터페이스 매핑을 확인합니다**. 이 출력에서 가장 중요한 정보는 **ASIC #, 슬라이스 # 및 소스 ID(srcid) #입니다.**
- 또한 **show system internal ethpm info interface # 명령을** 사용하여 이 정보를 다시 확인할 수도 있습니다 **나는 src**. 앞서 나열한 것 외에 여기서 중요한 것은 dpid와 dmod 값이다.
- **show** module 명령을 사용하여 모듈 번호를 **확인합니다**.
- 모듈 x를 **연결하여 모듈**에 연결합니다. 여기서 x는 모듈 번호입니다.
- module**-1# debug platform internal tah** elam asic **# 명령을 사용하여 모듈에서 ELAM을 실행합니다.**
- 캡처할 트래픽 유형(L2, L3, GRE 또는 VXLAN과 같은 캡슐화된 트래픽 등)에 따라 내부 또는 외부 트리거를 구성합니다.

```
Nexus9000(config)# attach module 1
module-1# debug platform internal tah elam asic 0
module-1(TAH-elam)# trigger init asic # slice # lu-a2d 1 in-select 6 out-select 0 use-src-id #
module-1(TAH-elam-insel6)# reset
module-1(TAH-elam-insel6)# set outer ipv4 dst_ip 192.0.2.1 src_ip 192.0.2.2
```

- 트리거가 설정되면 **start** 명령을 사용하여 ELAM을 시작하고, 트래픽을 전송하고, 명령 보고서를 사용하여 출력을 **확인합니다**. 보고서의 출력에는 발신 및 수신 인터페이스가 vlan ID, 소스 및 목적지 IP/MAC 주소와 함께 표시됩니다.

```
SUGARBOWL ELAM REPORT SUMMARY
slot - 1, asic - 1, slice - 1
==============================

Incoming Interface: Eth1/49
Src Idx : 0xd, Src BD : 10
Outgoing Interface Info: dmod 1, dpid 14
```

```
Dst Idx : 0x602, Dst BD : 10

Packet Type: IPv4
Dst MAC address: CC:46:D6:6E:28:DB
Src MAC address: 00:FE:C8:0E:27:15
.1q Tag0 VLAN: 10,  cos = 0x0
Dst IPv4 address: 192.0.2.1
Src IPv4 address: 192.0.2.2

Ver     =  4, DSCP    =    0, Don't Fragment = 0 Proto   =  1, TTL     =   64, More Fragments =
0 Hdr len = 20, Pkt len =   84, Checksum       = 0x667f
```

ELAM for Nexus 9000(NorthStar 플랫폼)

NorthStar 플랫폼의 절차는 Tahoe 플랫폼과 동일하며, 유일한 차이점은 ELAM 모드를 입력할 때 **키워드**가 tah 대신 **ns**가 사용된다는 것입니다.

```
module-1#debug platform internal ns elam asic 0
```

- **N9K 패킷 추적기**

Nexus 9000 패킷 추적기 툴은 패킷의 경로를 추적하는 데 사용할 수 있으며 플로우 통계용 카운터가 내장되어 있어 간헐적/완전한 트래픽 손실 시나리오에 유용한 툴입니다. TCAM 리소스가 제한되어 있거나 다른 도구를 실행할 수 없는 경우에는 매우 유용합니다. 또한 이 툴은 ARP 트래픽을 캡처할 수 없으며 Wireshark와 같은 패킷 콘텐츠의 세부 사항을 표시하지 않습니다.

패킷 추적기를 구성하려면 다음 명령을 사용합니다.

```
N9K-9508#test packet-tracer src_ip



              <==== provide your src and dst ip
N9K-9508# test packet-tracer start                                    <==== Start packet tracer
N9K-9508# test packet-tracer stop                                     <==== Stop packet tracer
N9K-9508# test packet-tracer show                                     <==== Check for packet
matches
```

자세한 내용은 Nexus 9000 링크를 참조하십시오. 패킷 추적기 툴 설명 - Cisco

- **트레이스라우트(Traceroute)와 핑스(Ping)**

이 명령은 연결 문제를 빠르게 식별할 수 있는 가장 유용한 두 가지 명령입니다.

Ping은 ICMP(Internet Control Message Protocol)를 사용하여 특정 대상에 ICMP 에코 메시지를 보내고 해당 대상에서 ICMP 에코 응답이 올 때까지 기다립니다. 호스트 간의 경로가 문제 없이 제대로 작동하면 회신이 다시 오고 ping이 성공한 것을 확인할 수 있습니다. ping 명령은 기본적으로 5x ICMP 에코 메시지(양방향으로 동일한 크기)를 전송하며, 모든 것이 정상적으로 작동하는 경우 5x ICMP 에코 응답을 확인할 수 있습니다. ARP(Address Resolution Protocol) 요청 중에 스위치에서 MAC 주소를 학습할 때 초기 에코 요청이 실패하는 경우가 있습니다. 이후에 즉시 ping을 다시 실행하면 초기 ping 손실이 없습니다. 또한 다음 키워드를 사용하여 ping 수, 패킷 크기, 소스, 소스 인터페이스 및 시간 초과 간격을 설정할 수도 있습니다.

```
F241.04.25-N9K-C93180-1# ping 10.82.139.39 vrf management
PING 10.82.139.39 (10.82.139.39): 56 data bytes
36 bytes from 10.82.139.38: Destination Host Unreachable
Request 0 timed out
64 bytes from 10.82.139.39: icmp_seq=1 ttl=254 time=23.714 ms
```

```
64 bytes from 10.82.139.39: icmp_seq=2 ttl=254 time=0.622 ms
64 bytes from 10.82.139.39: icmp_seq=3 ttl=254 time=0.55 ms
64 bytes from 10.82.139.39: icmp_seq=4 ttl=254 time=0.598 ms
```

```
F241.04.25-N9K-C93180-1# ping 10.82.139.39 ?
  <CR>
  count             Number of pings to send
  df-bit            Enable do not fragment bit in IP header
  interval          Wait interval seconds between sending each packet
  packet-size       Packet size to send
  source            Source IP address to use
  source-interface  Select source interface
  timeout           Specify timeout interval
  vrf               Display per-VRF information
```

Traceroute는 패킷이 목적지에 도달하기 전에 취하는 다양한 홉을 식별하는 데 사용됩니다. 장애가 발생하는 L3 경계를 식별하는 데 도움이 되므로 매우 중요한 툴입니다. 포트, 소스 및 소스 인터페이스를 다음 키워드와 함께 사용할 수도 있습니다.

```
F241.04.25-N9K-C93180-1# traceroute 10.82.139.39 ?
  <CR>
  port              Set destination port
  source            Set source address in IP header
  source-interface  Select source interface
  vrf               Display per-VRF information
```

```
Nexus_1(config)# traceroute 192.0.2.1
traceroute to 192.0.2.1 (192.0.2.1), 30 hops max, 40 byte packets
 1 198.51.100.3 (198.51.100.3)  1.017 ms  0.655 ms  0.648 ms
 2 203.0.113.2 (203.0.113.2)  0.826 ms  0.898 ms  0.82 ms
 3 192.0.2.1 (192.0.2.1)  0.962 ms  0.765 ms  0.776 ms
```

- PACL/RACL/VACL

ACL은 Access Control List를 의미합니다. 관련 정의된 기준에 따라 트래픽을 필터링할 수 있는 중요한 툴입니다. ACL이 일치 기준에 대한 항목으로 채워지면 인바운드 또는 아웃바운드 트래픽을 캡처하는 데 적용할 수 있습니다. ACL의 중요한 측면은 플로우 통계에 대한 카운터를 제공하는 기능입니다. PACL/RACL/VACL이라는 용어는 ACL을 강력한 트러블슈팅 툴로 사용할 수 있도록 하는 이러한 ACL의 다양한 구현을 나타냅니다. 특히 간헐적인 트래픽 손실을 방지합니다. 이 조건은 여기에서 간략하게 설명합니다.

- PACL은 Port Access Control List를 의미합니다. L2 switchport/interface에 액세스 목록을 적용할 때 이 액세스 목록을 PACL이라고 합니다.
- RACL은 Router Access Control List를 의미합니다. L3 라우팅 포트/인터페이스에 액세스 목록을 적용할 때 해당 액세스 목록을 RACL이라고 합니다.
- VACL은 VLAN Access Control List를 의미합니다. VACL을 구성하여 VLAN으로 들어오거나 VLAN에서 라우팅되거나 VLAN 내에 브리지된 모든 패킷에 적용할 수 있습니다. VACL은 보안 패킷 필터를 위한 것이며 트래픽을 특정 물리적 인터페이스로 리디렉션하기 위한 것입니다. VACL은 방향(인그레스 또는 이그레스)에 의해 정의되지 않습니다.

이 표에서는 ACL 버전을 비교합니다.

ACL 유형	PACL	라클	바클
기능	L2 인터페이스에서 수신된 트래픽을 필터링합니다.	L3 인터페이스에서 수신된 트래픽 필터링	vLAN 트래픽 필터링
적용 날짜	- L2 인터페이스/포트 - L2 포트 채널 인터페이스.	- VLAN 인터페이스 - 물리적 L3 인터페이스.	활성화되면 ACL은 해당 VLAN의 모든 포트에 적용

	- 트렁크 포트에 적용되는 경우 ACL은 해당 트렁크 포트에서 허용되는 모든 VLAN의 트래픽을 필터링합니다.	- L3 하위 인터페이스. - L3 포트 채널 인터페이스. - 관리 인터페이스.	다(트렁크 포트 포함).
적용된 방향	인바운드 전용.	인바운드 또는 아웃바운드	-

다음은 액세스 목록을 구성하는 방법의 예입니다. 자세한 내용은 Cisco Nexus 9000 Series NX-OS Security Configuration Guide, Release 9.3(x) - Configuring IP ACLs [Cisco Nexus 9000 Series Switches] - Cisco 링크를 참조하십시오.

```
Nexus93180(config)# ip access-list



Nexus93180(config-acl)# ?
  <1-4294967295>  Sequence number
  deny            Specify packets to reject
  fragments       Optimize fragments rule installation
  no              Negate a command or set its defaults
  permit          Specify packets to forward
  remark          Access list entry comment
  show            Show running system information
  statistics      Enable per-entry statistics for the ACL
  end             Go to exec mode
  exit            Exit from command interpreter
  pop             Pop mode from stack or restore from name
  push            Push current mode to stack or save it under name
  where           Shows the cli context you are in

Nexus93180(config)# int e1/1

Nexus93180(config-if)# ip port access-group



         >>>>>> When you configure ACL like this, it is PACL.
in  Inbound packets
Nexus93180(config-if)# ip access-group



             >>>>>> When you configure ACL like this, it is RACL.
  in   Inbound packets
  out  Outbound packets
```

- **로그플래시**

LogFlash는 Nexus 플랫폼에서 외부 컴팩트 플래시, USB 장치 또는 수퍼바이저에 내장된 디스크로 사용할 수 있는 영구 스토리지의 유형입니다. 스위치에서 제거된 경우, 시스템은 LogFlash가 없음을 사용자에게 주기적으로 알립니다. Logflash는 수퍼바이저에 설치되며 어카운팅 로그, syslog 메시지, 디버그 및 EEM(Embedded Event Manager) 출력과 같은 기록 데이터를 보관합니다. EEM에 대해서는 이 문서의 뒷부분에서 설명합니다. 다음 명령을 사용하여 LogFlash의 내용을 확인할 수 있습니다.

```
Nexus93180(config)# dir logflash:
          0    Nov 14 04:13:21 2019  .gmr6_plus
```

```
      20480    Feb 18 13:35:07 2020  ISSU_debug_logs/
         24    Feb 20 20:43:24 2019  arp.pcap
         24    Feb 20 20:36:52 2019  capture_SYB010L2289.pcap
       4096    Feb 18 17:24:53 2020  command/
       4096    Sep 11 01:39:04 2018  controller/
       4096    Aug 15 03:28:05 2019  core/
       4096    Feb 02 05:21:47 2018  debug/
    1323008    Feb 18 19:20:46 2020  debug_logs/
       4096    Feb 17 06:35:36 2020  evt_log_snapshot/
       4096    Feb 02 05:21:47 2018  generic/
       1024    Oct 30 17:27:49 2019  icamsql_1_1.db
      32768    Jan 17 11:53:23 2020  icamsql_1_1.db-shm
     129984    Jan 17 11:53:23 2020  icamsql_1_1.db-wal
       4096    Feb 14 13:44:00 2020  log/
      16384    Feb 02 05:21:44 2018  lost+found/
       4096    Aug 09 20:38:22 2019  old_upgrade/
       4096    Feb 18 13:40:36 2020  vdc_1/

Usage for logflash://sup-local
 1103396864 bytes used
 7217504256 bytes free
 8320901120 bytes total
```

사용자가 디바이스를 다시 로드했거나 이벤트로 인해 갑자기 스스로 디바이스를 다시 로드한 경우 로그 정보가 모두 손실됩니다. 이러한 시나리오에서 LogFlash는 문제의 가능한 원인을 파악할 수 있도록 검토할 수 있는 기록 데이터를 제공할 수 있습니다. 물론, 이 사건이 다시 발생할 경우 무엇을 찾아야 하는지에 대한 힌트를 제공하는 근본 원인을 파악하기 위해 추가적인 실사가 필요합니다 .

디바이스에 logflash를 설치하는 방법에 대한 자세한 내용은 Nexus 7000 로깅 기능 - Cisco 링크를 참조하십시오.

- **OBFL**

OBFL은 OnBoard Failure Logging을 의미합니다. Nexus ToR(Top of Rack) 및 모듈형 스위치에서 모두 사용할 수 있는 영구 스토리지 유형입니다. LogFlash와 마찬가지로, 디바이스가 다시 로드되면 정보가 보존됩니다. OBFL은 장애 및 환경 데이터와 같은 정보를 저장합니다. 정보는 플랫폼과 모듈마다 다르지만, 다음은 Nexus 93108 플랫폼의 모듈 1의 샘플 출력입니다(즉, 하나의 모듈만 있는 고정 섀시).

```
Nexus93180(config)# show logging onboard module 1 ?
*** No matching command found in current mode, matching in (exec) mode ***
  <CR>
  >                     Redirect it to a file
  >>                    Redirect it to a file in append mode
  boot-uptime           Boot-uptime
  card-boot-history     Show card boot history
  card-first-power-on   Show card first power on information
  counter-stats         Show OBFL counter statistics
  device-version        Device-version
  endtime               Show OBFL logs till end time mm/dd/yy-HH:MM:SS
  environmental-history Environmental-history
  error-stats           Show OBFL error statistics
  exception-log         Exception-log
  internal              Show Logging Onboard Internal
  interrupt-stats       Interrupt-stats
  obfl-history          Obfl-history
  stack-trace           Stack-trace
  starttime             Show OBFL logs from start time mm/dd/yy-HH:MM:SS
  status                Status
  |                     Pipe command output to filter
```

```
Nexus93180(config)# show logging onboard module 1 status
----------------------------
OBFL Status
----------------------------
    Switch OBFL Log:                                      Enabled
    Module:  1 OBFL Log:                                  Enabled
    card-boot-history                                     Enabled
    card-first-power-on                                   Enabled
    cpu-hog                                               Enabled
    environmental-history                                 Enabled
    error-stats                                           Enabled
    exception-log                                         Enabled
    interrupt-stats                                       Enabled
    mem-leak                                              Enabled
    miscellaneous-error                                   Enabled
    obfl-log (boot-uptime/device-version/obfl-history)  Enabled
    register-log                                          Enabled
    system-health                                         Enabled
    temp Error                                            Enabled
    stack-trace                                           Enabled
```

이 정보는 사용자가 의도적으로 또는 다시 로드를 트리거한 이벤트로 인해 디바이스가 다시 로드되는 경우에 유용합니다. 이 경우 OBFL 정보를 통해 라인 카드의 관점에서 무엇이 잘못되었는지 파악할 수 있습니다. 명령 show **logging onboard**를 시작하는 것이 좋습니다. 필요한 모든 것을 얻으려면 모듈 컨텍스트 내에서 캡처해야 합니다. show logging onboard **module x 또는 attach mod** x를 **사용해야 합니다. show logging onboard.**

- **이벤트 기록**

이벤트 기록은 Nexus에서 실행되는 프로세스에 대해 발생하는 다양한 이벤트에 대한 정보를 제공할 수 있는 강력한 툴 중 하나입니다. 즉, Nexus 플랫폼에서 실행되는 모든 프로세스에는 백그라운드에서 실행되는 이벤트 기록이 있으며 해당 프로세스의 다양한 이벤트에 대한 정보를 저장합니다(끊임없이 실행되는 디버그로 생각함). 이러한 이벤트 기록은 비지속적이며, 디바이스 다시 로드 시 저장된 모든 정보가 손실됩니다. 특정 프로세스에 문제가 있음을 확인하고 해당 프로세스에 대한 트러블슈팅을 원할 때 매우 유용합니다. 예를 들어 OSPF 라우팅 프로토콜이 제대로 작동하지 않을 경우 OSPF와 관련된 이벤트 기록을 사용하여 OSPF 프로세스가 실패한 위치를 식별할 수 있습니다. CDP/STP, UDLD, LACP/OSPF, EIGRP/BGP 등 Nexus 플랫폼의 거의 모든 프로세스와 관련된 이벤트 기록을 찾을 수 있습니다.

이는 일반적으로 참조 예제를 사용하여 프로세스의 이벤트 기록을 확인하는 방법입니다. 모든 프로세스에는 여러 옵션이 있으므로 사용하시겠습니까**?** - 프로세스에서 사용할 수 있는 다양한 옵션을 확인합니다.

```
Nexus93180(config)# show
```

```
Nexus93180# show ip ospf event-history ?
  adjacency       Adjacency formation logs
  cli             Cli logs
  event           Internal event logs
  flooding        LSA flooding logs
  ha              HA and GR logs
  hello           Hello related logs
  ldp             LDP related logs
  lsa             LSA generation and databse logs
```

```
  msgs            IPC logs
  objstore        DME OBJSTORE related logs
  redistribution  Redistribution logs
  rib             RIB related logs
  segrt           Segment Routing logs
  spf             SPF calculation logs
  spf-trigger     SPF TRIGGER related logs
  statistics      Show the state and size of the buffers
  te              MPLS TE related logs

Nexus93180# show spanning-tree internal event-history ?
  all      Show all event historys
  deleted  Show event history of deleted trees and ports
  errors   Show error logs of STP
  msgs     Show various message logs of STP
  tree     Show spanning tree instance info
  vpc      Show virtual Port-channel event logs
```

- **디버그**

디버그는 NX-OS 내의 강력한 도구로서 실시간 문제 해결 이벤트를 실행하고 이를 파일에 기록하거나 CLI에 표시할 수 있습니다. 디버그 출력은 CPU 성능에 영향을 미치므로 파일에 기록하는 것이 좋습니다. CLI에서 디버그를 직접 실행하기 전에 주의하십시오.

디버그는 일반적으로 단일 프로세스로 문제를 식별한 경우에만 실행되며, 네트워크의 실제 트래픽에서 이 프로세스가 실시간으로 어떻게 동작하는지 확인하고자 합니다. 정의된 사용자 계정 권한을 기반으로 디버그 기능을 활성화해야 합니다.

이벤트 기록과 마찬가지로 CDP/STP, UDLD, LACP/OSPF, EIGRP/BGP 등 Nexus 디바이스의 모든 프로세스에 대해 디버그를 실행할 수 있습니다.

일반적으로 프로세스에 대해 디버그를 실행하는 방법입니다. 모든 프로세스에는 여러 옵션이 있으므로 사용하시겠습니까**?** - 프로세스에서 사용할 수 있는 다양한 옵션을 확인합니다.

```
Nexus93180# debug
```

```
Nexus93180# debug spanning-tree ?
  all      Configure all debug flags of stp
  bpdu_rx  Configure debugging of stp bpdu rx
  bpdu_tx  Configure debugging of stp bpdu tx
  error    Configure debugging of stp error
  event    Configure debugging of Events
  ha       Configure debugging of stp HA
  mcs      Configure debugging of stp MCS
  mstp     Configure debugging of MSTP
  pss      Configure debugging of PSS
  rstp     Configure debugging of RSTP
  sps      Configure debugging of Set Port state batching
  timer    Configure debugging of stp Timer events
  trace    Configure debugging of stp trace
  warning  Configure debugging of stp warning

Nexus93180# debug ip ospf ?
  adjacency               Adjacency events
  all                     All OSPF debugging
```

```
  database                 OSPF LSDB changes
  database-timers          OSPF LSDB timers
  events                   OSPF related events
  flooding                 LSA flooding
  graceful-restart         OSPF graceful restart related debugs
  ha                       OSPF HA related events
  hello                    Hello packets and DR elections
  lsa-generation           Local OSPF LSA generation
  lsa-throttling           Local OSPF LSA throttling
  mpls                     OSPF MPLS
  objectstore              Objectstore Events
  packets                  OSPF packets
  policy                   OSPF RPM policy debug information
  redist                   OSPF redistribution
  retransmission           OSPF retransmission events
  rib                      Sending routes to the URIB
  segrt                    Segment Routing Events
  snmp                     SNMP traps and request-response related events
  spf                      SPF calculations
  spf-trigger              Show SPF triggers
```

- **골드**

GOLD는 Generic OnLine Diagnostics의 약어입니다. 이름에서 알 수 있듯이 이러한 테스트는 일반적으로 시스템 상태 점검으로 사용되며 문제의 하드웨어를 점검하거나 확인하는 데 사용됩니다. 다양한 온라인 테스트가 수행되며 사용 중인 플랫폼을 기반으로 일부 테스트는 중단을 초래하지만 일부는 중단을 일으키지 않습니다. 이러한 온라인 테스트는 다음과 같이 분류할 수 있습니다.

- **부팅 진단:** 이러한 테스트는 디바이스가 부팅될 때 실행되는 테스트입니다. 또한 모든 ASIC에 대한 데이터 및 컨트롤 플레인 간의 연결을 포함하는 모듈과 수퍼바이저 간의 연결도 확인합니다. ManagementPortLoopback 및 EOBCLoopback과 같은 테스트는 중단되지만 OBFL 및 USB에 대한 테스트는 중단되지 않습니다.
- **런타임 또는 상태 모니터링 진단:** 이러한 테스트는 디바이스의 상태에 대한 정보를 제공합니다. 이러한 테스트는 무중단 방식으로 진행되며 하드웨어 안정성을 위해 백그라운드에서 실행됩니다. 필요에 따라 또는 문제 해결을 위해 이러한 테스트를 활성화/비활성화할 수 있습니다.
- **온디맨드 진단:** 문제를 현지화하기 위해 언급된 모든 테스트를 온디맨드 방식으로 다시 실행할 수 있습니다.

다음 명령을 사용하여 스위치에 사용할 수 있는 다양한 유형의 온라인 테스트를 확인할 수 있습니다.

```
Nexus93180(config)# show diagnostic content module all
Diagnostics test suite attributes:
B/C/* - Bypass bootup level test / Complete bootup level test / NA
P/*   - Per port test / NA
M/S/* - Only applicable to active / standby unit / NA
D/N/* - Disruptive test / Non-disruptive test / NA
H/O/* - Always enabled monitoring test / Conditionally enabled test / NA
F/*   - Fixed monitoring interval test / NA
X/*   - Not a health monitoring test / NA
E/*   - Sup to line card test / NA
L/*   - Exclusively run this test / NA
T/*   - Not an ondemand test / NA
A/I/* - Monitoring is active / Monitoring is inactive / NA

Module 1: 48x10/25G + 6x40/100G Ethernet Module (Active)

                                                    Testing Interval
ID     Name                                 Attributes      (hh:mm:ss)
____   ___________________________________ ____________   _________________
```

```
 1)    USB--------------------------->     C**N**X**T*     -NA-
 2)    NVRAM------------------------->     ***N******A     00:05:00
 3)    RealTimeClock----------------->     ***N******A     00:05:00
 4)    PrimaryBootROM---------------->     ***N******A     00:30:00
 5)    SecondaryBootROM-------------->     ***N******A     00:30:00
 6)    BootFlash--------------------->     ***N******A     00:30:00
 7)    SystemMgmtBus----------------->     **MN******A     00:00:30
 8)    OBFL-------------------------->     C**N**X**T*     -NA-
 9)    ACT2-------------------------->     ***N******A     00:30:00
10)    Console----------------------->     ***N******A     00:00:30
11)    FpgaRegTest------------------->     ***N******A     00:00:30
12)    Mce--------------------------->     ***N******A     01:00:00
13)    AsicMemory-------------------->     C**D**X**T*     -NA-
14)    Pcie-------------------------->     C**N**X**T*     -NA-
15)    PortLoopback------------------>     *P*N**XE***     -NA-
16)    L2ACLRedirect----------------->     *P*N***E**A     00:01:00
17)    BootupPortLoopback------------>     CP*N**XE*T*     -NA-
```

언급된 17개의 각 테스트에서 수행하는 작업을 표시하려면 다음 명령을 사용할 수 있습니다.

```
Nexus93180(config)#show diagnostic description module 1 test all
USB :
        A bootup test that checks the USB controller initialization
        on the module.

NVRAM :
        A health monitoring test, enabled by default that checks the
        sanity of the NVRAM device on the module.

RealTimeClock :
        A health monitoring test, enabled by default that verifies
        the real time clock on the module.

PrimaryBootROM :
        A health monitoring  test that verifies the primary BootROM
        on the module.

SecondaryBootROM :
        A health monitoring  test that verifies the secondary BootROM
        on the module.

BootFlash :
        A Health monitoring test, enabled by default, that verifies
        access to the internal compactflash devices.

SystemMgmtBus :
        A Health monitoring test, enabled by default, that verifies
        the standby System Bus.

OBFL :
        A bootup test that checks the onboard flash used for failure
        logging (OBFL) device initialization on the module.

ACT2 :
        A Health monitoring test, enabled by default, that verifies
        access to the ACT2 device.

Console :
        A health monitoring test,enabled by default that checks health
        of console device.

FpgaRegTest :
```

```
        A health monitoring test,enabled by default that checks read/write
        access to FPGA scratch registers on the module.

Mce :
        A Health monitoring test, enabled by default, that check for
        machine errors on sup.

AsicMemory :
        A bootup test that checks the asic memory.

Pcie :
        A bootup test that tests pcie bus of the module

PortLoopback :
        A health monitoring test that tests the packet path from
        the Supervisor card to the physical port in ADMIN DOWN state
        on Linecards.

L2ACLRedirect :
        A health monitoring test, enabled by default, that does a
        non disruptive loopback for TAHOE asics to check the ACL Sup
        redirect with the CPU port.

BootupPortLoopback :
        A Bootup test that tests the packet path from the Supervisor
        card to all of the physical ports at boot time.
```

- **EEM**

EEM은 Embedded Event Manager를 의미합니다. 특정 이벤트가 발생할 경우 특정 작업을 수행하도록 디바이스를 프로그래밍할 수 있는 강력한 도구입니다. 디바이스에서 다양한 이벤트를 모니터링한 다음 문제 해결에 필요한 조치를 취하고 복구할 수 있습니다. EEM은 세 가지 주요 구성 요소로 구성되며, 각 구성 요소는 여기에서 간략하게 설명합니다.

- **이벤트 설명:** 이는 Nexus에서 특정 작업(예: 해결 방법, SNMP 서버에 알림, CLI 로그 표시)을 수행하도록 모니터링하고 원하는 이벤트입니다.
- **작업 설명:** 이는 이벤트가 트리거되면 EEM에서 수행하는 단계입니다. 이러한 작업은 인터페이스를 비활성화하거나 일부 show 명령을 실행하고 출력을 ftp 서버의 파일에 복사하거나, 이메일을 보내는 등의 작업을 수행하는 것입니다.
- **정책:** 기본적으로 CLI 또는 bash 스크립트를 통해 수퍼바이저에 구성할 수 있는 하나 이상의 작업 명령문과 결합된 이벤트입니다. python 스크립트를 사용하여 EEM을 호출할 수도 있습니다. 정책이 수퍼바이저에 정의되면 해당 모듈에 정책을 푸시합니다.

EEM에 대한 자세한 내용은 Cisco Nexus 9000 Series NX-OS System Management Configuration Guide, Release 9.2(x) - Configuring the Embedded Event Manager [Cisco Nexus 9000 Series Switches] - Cisco 링크를 참조하십시오.

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.