플랫폼 설정

날짜 및 시간 설정

시스템에서 NTP(network time protocol)를 구성하거나, 수동으로 날짜 및 시간을 설정하거나, 현재 시스템 시간을 보려면 NTP 페이지을 사용하십시오.

NTP 설정은 Firepower 4100/9300 섀시 및 섀시에 설치된 논리적 디바이스 간에 자동으로 동기화됩니다.

참고 Firepower 4100/9300 섀시에 FTD를 구축할 경우, 스마트 라이센싱의 올바른 작동 및 디바이스 등록 시 올바른 타임스탬프를 보장하려면 Firepower 4100/9300 섀시에서 NTP를 구성해야 합니다. Firepower 4100/9300 섀시 및 FMC에 동일한 NTP 서버를 사용해야 하지만 FMC를 Firepower 4100/9300 섀시의 NTP 서버로 사용할 수는 없습니다.

NTP를 사용하는 경우 **Current Time(**현재 시간**)** 탭에서 전반적인 동기화 상태를 볼 수 있습니다. 또는 **Time Synchronization(**시간 동기화**)** 탭의 **NTP Server(NTP** 서버**)** 테이블에 있는 Server Status(서버 상태) 필드에서 구성된 각 NTP 서버의 동기화 상태를 볼 수 있습니다. 시스템을 특정 NTP 서버와 동기화할 수 없는 경우 Server Status(서버 상태) 옆에 있는 정보 아이콘에 마우스 커서를 대면 자세한 내용을 확인할 수 있습니다.

구성된 날짜 및 시간 보기

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **NTP**를 선택합니다.

단계 **2** **Current Time(**현재 시간**)** 탭을 클릭합니다.

디바이스에 구성된 날짜, 시간 및 시간대가 표시됩니다.

NTP를 사용 중인 경우 **Current Time(**현재 시간**)** 탭에 전체적인 동기화 상태도 표시됩니다. **Time Synchronization(**시간 동기화**)** 탭에서 **NTP Server(NTP** 서버**)** 테이블의 Server Status(서버 상태) 필드를 확인하여 구성된 각 NTP 서버의 동기화 상태를 볼 수 있습니다. 시스템을 특정 NTP 서버와 동기화할 수 없는 경우 Server Status(서버 상태) 옆에 있는 정보 아이콘에 마우스 커서를 대면 자세한 내용을 확인할 수 있습니다.

표준 시간대 설정

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **NTP**를 선택합니다.

단계 **2** **Current Time(**현재 시간**)** 탭을 클릭합니다.

단계 **3** **Time Zone(**표준 시간대**)** 드롭다운 목록에서 섀시에 적절한 표준 시간대를 선택합니다.

NTP를 사용하여 날짜 및 시간 설정

NTP는 네트워크 시스템 간에 정확하게 동기화된 시간을 제공하는 계층적 서버 시스템을 구현하는 데 사용합니다. 정밀한 타임 스탬프가 포함된 CRL 검증과 같이 시간에 민감한 작업에는 이러한 정확성이 필요합니다. 최대 4개까지 NTP 서버를 구성할 수 있습니다.

참고

- FXOS는 NTP 버전 3을 사용합니다.
- 외부 NTP 서버의 stratum 값이 13 이상인 경우 애플리케이션 인스턴스는 FXOS 섀시의 NTP 서버와 동기화할 수 없습니다. NTP 클라이언트가 NTP 서버와 동기화될 때마다 stratum 값이 1씩 증가합니다.

  자체 NTP 서버를 설정한 경우, 서버의 /etc/ntp.conf 파일에서 해당 계층 값을 찾을 수 있습니다. NTP 서버의 stratum 값이 13 이상인 경우 ntp.conf 파일에서 stratum 값을 변경하고 서버를 다시 시작하거나 다른 NTP 서버(예: pool.ntp.org)를 사용할 수 있습니다.

시작하기 전에

NTP 서버의 호스트 이름을 사용하는 경우 DNS 서버를 구성해야 합니다. DNS 서버 구성, 46 페이지를 참조하십시오.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **NTP**를 선택합니다.

Time Synchronization(시간 동기화**)** 탭이 기본적으로 선택되어 있습니다.

단계 **2** **Set Time Source(**시간 소스 설정**)**에서 **Use NTP Server(NTP** 서버 사용**)**.

단계 **3** (선택 사항) NTP 서버로 인증해야 하는 경우 **NTP Server Authentication: Enable(NTP** 서버 인증**:** 활성화**)** 체크 박스을 선택합니다.

인증 키 ID 및 값을 요구하려면 **Yes(**예**)**를 클릭합니다.

NTP 서버 인증에는 SHA1만 지원됩니다.

단계 **4** IP 주소 또는 호스트 이름별로 최대 4개의 NTP 서버를 식별하려면 **Add(**추가**)**를 클릭합니다.

단계 **5** (선택 사항) NTP 서버의 **Authentication Key(**인증 키**)** ID 및 **Authentication Value(**인증 값**)**를 입력합니다.

NTP 서버에서 키 ID 및 값을 가져옵니다. 예를 들어 OpenSSL이 설치된 NTP 서버 버전 4.2.8p8 이상에서 SHA1 키를 생성하려면 **ntp-keygen -M** 명령을 입력한 다음 ntp.keys 파일에서 키 ID 및 값을 확인합니다. message digest를 계산할 때 어떤 키 값을 사용할지를 클라이언트 및 서버에 알려줄 때 키 ID가 사용됩니다.

단계 **6** **Save(**저장**)**를 클릭합니다.

NTP Server(NTP 서버**)** 테이블의 Server Status(서버 상태) 필드를 확인하여 각 서버의 동기화 상태를 볼 수 있습니다. 시스템을 특정 NTP 서버와 동기화할 수 없는 경우 Server Status(서버 상태) 옆에 있는 정보 아이콘에 마우스 커서를 대면 자세한 내용을 확인할 수 있습니다.

참고 시스템 시간을 10분 이상 수정하는 경우, 시스템에서 로그아웃되며 Firepower Chassis Manager에 다시 로그인해야 합니다.

NTP 서버 삭제

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **NTP**를 선택합니다.

단계 **2** **Time Synchronization(**시간 동기화**)** 탭을 클릭합니다.

단계 **3** 제거할 각 NTP 서버에 대해 **NTP Server(NTP** 서버**)** 테이블에서 해당 서버의 **Delete(**삭제**)** 아이콘을 클릭합니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

날짜 및 시간 직접 설정

이 섹션에서는 섀시에 날짜 및 시간을 수동으로 설정하는 방법을 설명합니다. 섀시 날짜 및 시간을 수동으로 설정한 후에는 설치된 논리적 디바이스에 변경 사항이 반영되는 데 다소 시간이 걸릴 수 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **NTP**를 선택합니다.

단계 **2** **Time Synchronization(**시간 동기화**)** 탭을 클릭합니다.

단계 **3** **Set Time Source(**시간 소스 설정**)**에서 **Set Time Manually(**수동으로 시간 설정**)**를 클릭합니다.

단계 **4** **Date(**날짜**)** 드롭다운 목록을 클릭하여 달력을 표시한 다음 달력에서 사용 가능한 컨트롤을 통해 날짜를 설정합니다.

단계 **5** 해당하는 드롭다운 목록을 사용하여 시간을 시, 분 및 AM/PM으로 지정합니다.

팁 **Get System Time(**시스템 시간 가져오기**)**을 클릭하여 Firepower Chassis Manager에 대한 연결에 사용 중인 시스템에 구성되어 있는 날짜 및 시간과 일치하도록 날짜 및 시간을 설정할 수 있습니다.

단계 **6** **Save(**저장**)**를 클릭합니다.

섀시는 날짜 및 시간이 지정된 상태로 구성됩니다.

참고 시스템 시간을 10분 이상 수정하는 경우, 시스템에서 로그아웃되며 Firepower Chassis Manager에 다시 로그인해야 합니다.

SSH 구성

다음 절차에서는 섀시에 대한 SSH 액세스를 활성화하거나 비활성화하는 방법, FXOS 섀시를 SSH 클라이언트로 활성화하는 방법, SSH 서버와 SSH 클라이언트 모두에 대해 암호화, 키 교환 및 메시지 인증을 위해 SSH에서 사용하는 다양한 알고리즘을 구성하는 방법을 설명합니다.

SSH는 기본적으로 활성화되어 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **SSH** > **SSH Server(SSH** 서버**)**를 선택합니다.

단계 **2** 섀시에 대한 SSH 액세스를 활성화하려면 **Enable SSH(SSH** 활성화**)** 체크 박스를 선택합니다. SSH 액세스를 비활성화하려면 **Enable SSH(SSH** 활성화**)** 확인란의 선택을 취소합니다.

단계 **3** 서버의 **Encryption Algorithm(**암호화 알고리즘**)**에 대해 허용되는 각 암호화 알고리즘의 체크 박스를 선택합니다.

참고
- 다음 암호화 알고리즘은 Common Criteria 모드에서 지원되지 않습니다.
  - 3des-cbc
  - chacha20-poly1305@openssh.com
- chacha20-poly1305@openssh.com은 FIPS에서 지원되지 않습니다. FXOS 섀시에서 FIPS 모드가 활성화되어 있으면, chacha20-poly1305@openssh.com를 암호화 알고리즘으로 사용할 수 없습니다.
- 다음 암호화 알고리즘은 기본적으로 활성화되지 않습니다.

```
aes128-cbc
aes192-cbc
aes265-cbc
```

단계 **4** 서버의 **Key Exchange Algorithm(**키 교환 알고리즘**)**에 대해 허용되는 각 DH(Diffie-Hellman) 키 교환의 체크 박스를 선택합니다. DH 키 교환에서는 어느 한쪽에서 단독으로 확인할 수 없는 공유 암호를 제공합니다. 이 키 교환은 서명 및 호스트 키와 연계하여 호스트 인증을 수행합니다. 이 키 교환 방식은 명시적 서버 인증을 수행합니다. DH 키 교환 방법 사용에 대한 자세한 내용은 RFC 4253을 참조하십시오.

참고
- 다음 키 교환 알고리즘은 Common Criteria 모드에서 지원되지 않습니다.
  - diffie-hellman-group14-sha256
  - curve25519-sha256
  - curve25519-sha256@libssh.org
- 다음 키 교환 알고리즘은 FIPS 모드에서 지원되지 않습니다.
  - curve25519-sha256
  - curve25519-sha256@libssh.org

단계 **5** 서버의 **Mac Algorithm(Mac** 알고리즘**)**에 대해 허용되는 각 무결성 알고리즘의 체크 박스를 선택합니다.

단계 **6** 서버의 **Host Key(**호스트 키**)**에 대해 RSA 키 쌍에 대한 모듈러스 크기를 입력합니다.

모듈러스 값(비트 단위)은 1024~2048 범위의 8의 배수입니다. 지정하는 키 모듈러스 크기가 클수록 RSA 키 쌍을 생성하는 데 오래 걸립니다. 권장되는 값은 2048입니다.

단계 **7** 서버의 **Volume Rekey Limit(**볼륨 재생성 제한**)**에 대해 FXOS가 세션에서 연결을 해제하기 전에 연결을 통해 허용되는 트래픽 양(KB 단위)을 설정합니다.

단계 **8** 서버의 **Time Rekey Limit(**시간 키 재생성 제한**)**에 대해 FXOS가 세션에서 연결을 해제하기 전에 SSH 세션이 유휴 상태가 될 수 있는 시간(분 단위)을 설정합니다.

단계 **9** **Save(**저장**)**를 클릭합니다.

단계 **10** FXOS 섀시 SSH 클라이언트를 맞춤화하려면 **SSH Client(SSH** 클라이언트**)** 탭을 클릭합니다.

단계 **11** **Strict Host Keycheck(**엄격한 호스트 키 확인**)**에 대해 **enable(**활성화**)**, **disable(**비활성화**)** 또는 **prompt(**프롬프트**)**를 선택하여 SSH 호스트 키 확인을 제어합니다.

- **enable(**활성화**)** - 호스트 키가 FXOS의 알려진 호스트 파일에 없는 경우 연결이 거부됩니다. 시스템/서비스 범위에서 **enter ssh-host** 명령을 사용하여 FXOS CLI에서 호스트를 수동으로 추가해야 합니다.
- **prompt(**프롬프트**)** - 호스트 키가 섀시에 저장되어 있지 않은 경우 호스트 키를 수락하거나 거부하라는 프롬프트가 표시됩니다.
- **disable(**비활성화**)** - (기본값) 이전에 저장한 호스트 키가 없는 경우 섀시가 호스트 키를 자동으로 수락합니다.

단계 **12** 클라이언트의 **Encryption Algorithm(**암호화 알고리즘**)**에 대해 허용되는 각 암호화 알고리즘의 체크 박스를 선택합니다.

참고
- 다음 암호화 알고리즘은 Common Criteria 모드에서 지원되지 않습니다.
  - 3des-cbc
  - chacha20-poly1305@openssh.com

  FXOS 섀시에서 Common Criteria 모드가 활성화되어 있으면 3des-cbc를 암호화 알고리즘으로 사용할 수 없습니다.
- chacha20-poly1305@openssh.com은 FIPS에서 지원되지 않습니다. FXOS 섀시에서 FIPS 모드가 활성화되어 있으면, chacha20-poly1305@openssh.com를 암호화 알고리즘으로 사용할 수 없습니다.
- 다음 암호화 알고리즘은 기본적으로 활성화되지 않습니다.

```
aes128-cbc
aes192-cbc
aes265-cbc
```

단계 **13** 클라이언트의 **Key Exchange Algorithm(**키 교환 알고리즘**)**에 대해 허용되는 각 DH(Diffie-Hellman) 키 교환의 체크 박스를 선택합니다. DH 키 교환에서는 어느 한쪽에서 단독으로 확인할 수 없는 공유 암호를 제공합니다. 이 키 교환은 서명 및 호스트 키와 연계하여 호스트 인증을 수행합니다. 이 키 교환 방식은 명시적 서버 인증을 수행합니다. DH 키 교환 방법 사용에 대한 자세한 내용은 RFC 4253을 참조하십시오.

참고
- 다음 키 교환 알고리즘은 Common Criteria 모드에서 지원되지 않습니다.
  - diffie-hellman-group14-sha256
  - curve25519-sha256
  - curve25519-sha256@libssh.org
- 다음 키 교환 알고리즘은 FIPS 모드에서 지원되지 않습니다.
  - curve25519-sha256
  - curve25519-sha256@libssh.org

단계 **14** 클라이언트의 **Mac Algorithm(Mac** 알고리즘**)**에 대해 허용되는 각 무결성 알고리즘의 체크 박스를 선택합니다.

단계 **15** 클라이언트의 **Volume Rekey Limit(**볼륨 재생성 제한**)**에 대해 FXOS가 세션에서 연결을 해제하기 전에 연결을 통해 허용되는 트래픽 양(KB 단위)을 설정합니다.

단계 **16** 클라이언트의 **Time Rekey Limit(**시간 키 재생성 제한**)**에 대해 FXOS가 세션에서 연결을 해제하기 전에 SSH 세션이 유휴 상태가 될 수 있는 시간(분 단위)을 설정합니다.

단계 **17** **Save(**저장**)**를 클릭합니다.

TLS 구성

TLS(Transport Layer Security) 프로토콜은 통신 중인 두 애플리케이션 간에 프라이버시 및 데이터 무결성을 제공합니다. FXOS CLI를 사용하여 FXOS 섀시가 외부 디바이스와 통신할 때 허용되는 최소 TLS 버전을 구성할 수 있습니다. 최신 TLS 버전은 더 안전한 통신을 제공하며, 이전 TLS 버전에서는 오래된 애플리케이션에 대한 이전 버전과의 호환성이 허용됩니다.

예를 들어 FXOS 섀시에 구성된 최소 TLS 버전이 v1.1인데 클라이언트 브라우저가 v1.0만 실행하도록 구성되어 있으면 클라이언트가 HTTPS를 통해 FXOS Chassis Manager와의 연결을 열 수 없습니다. 따라서 피어 애플리케이션 및 LDAP 서버를 적절하게 구성해야 합니다.

이 절차에서는 FXOS 섀시와 외부 디바이스 간의 통신에 허용되는 최소 TLS 버전을 구성하고 확인하는 방법을 설명합니다.

참고

- FXOS 2.3(1) 릴리스를 기준으로, FXOS 섀시용 기본 최소 TLS 버전은 v1.1입니다.

프로시저

단계 **1** 시스템 모드를 입력합니다.

Firepower-chassis# **scope system**

단계 **2** 시스템에서 사용 가능한 TLS 버전 옵션을 확인합니다.

Firepower-chassis /system #**set services tls-ver**

예제:

```
Firepower-chassis /system #
Firepower-chassis /system # set services tls-ver
    v1_0  v1.0
    v1_1  v1.1
    v1_2  v1.2
```

단계 **3** 최소 TLS 버전을 설정합니다.

Firepower-chassis /system # **set services tls-ver** *version*

예제:

```
Firepower-chassis /system #
Firepower-chassis /system # set services tls-ver v1_2
```

단계 **4** 구성을 커밋합니다.

Firepower-chassis /system #**commit-buffer**

단계 **5** 시스템에 구성된 최소 TLS 버전을 표시합니다.

Firepower-chassis /system #**scope services**

Firepower-chassis /system/services # **show**

예제:

```
Firepower-chassis /system/services # show
Name: ssh
    Admin State: Enabled
    Port: 22
Kex Algo: Diffie Hellman Group1 Sha1,Diffie Hellman Group14 Sha1
Mac Algo: Hmac Sha1,Hmac Sha1 96,Hmac Sha2 512,Hmac Sha2 256
Encrypt Algo: 3des Cbc,Aes256 Cbc,Aes128 Cbc,Aes192 Cbc,Aes256 Ctr,Aes128 Ctr,Ae
s192 Ctr
Auth Algo: Rsa
    Host Key Size: 2048
Volume: None Time: None
Name: telnet
    Admin State: Disabled
    Port: 23
Name: https
    Admin State: Enabled
    Port: 443
    Operational port: 443
    Key Ring: default
    Cipher suite mode: Medium Strength
    Cipher suite: ALL:!ADH:!EXPORT40:!EXPORT56:!LOW:!RC4:!MD5:!IDEA:+HIGH:+MEDIU
M:+EXP:+eNULL
    Https authentication type: Cert Auth
    Crl mode: Relaxed
TLS:
    TLS version: v1.2
```

텔넷 구성

다음 절차에서는 섀시에 대한 Telnet 액세스를 활성화하거나 비활성화하는 방법을 설명합니다. 텔넷은 기본적으로 비활성화되어 있습니다.

참고 텔넷 구성은 현재 CLI를 사용하는 경우에만 사용할 수 있습니다.

프로시저

단계 **1** 시스템 모드를 입력합니다.

Firepower-chassis # **scope system**

단계 **2** 시스템 서비스 모드를 입력합니다.

Firepower-chassis /system # **scope services**

단계 **3** 섀시에 대한 Telnet 액세스를 구성하려면 다음 중 하나를 수행합니다.

- 섀시에 대한 Telnet 액세스를 허용하려면 다음 명령을 입력합니다.

  Firepower-chassis /system/services # **enable telnet-server**

- 섀시에 대한 Telnet 액세스를 거부하려면 다음 명령을 입력합니다.

  Firepower-chassis /system/services # **disable telnet-server**

단계 **4** 시스템 구성에 트랜잭션을 커밋합니다.

Firepower /system/services # **commit-buffer**

예

다음의 예에서는 텔넷을 활성화하고 트랜잭션을 커밋합니다.

```
Firepower-chassis# scope system
Firepower-chassis /system # scope services
Firepower-chassis /services # enable telnet-server
Firepower-chassis /services* # commit-buffer
Firepower-chassis /services #
```

SNMP 구성

SNMP 페이지를 사용하여 섀시에서 단순 네트워크 관리 프로토콜(SNMP)을 구성합니다. 자세한 내용은 다음 항목을 참고하십시오.

SNMP 정보

단순 네트워크 관리 프로토콜(SNMP)은 SNMP 관리자 및 에이전트 간 통신에 메시지 형식을 제공하는 애플리케이션 레이어 프로토콜입니다. SNMP는 네트워크에 있는 디바이스의 모니터링 및 관리에 사용되는 표준화된 프레임워크 및 공통 언어를 제공합니다.

SNMP 프레임워크는 다음 3가지 항목으로 구성됩니다.

- SNMP 관리자 — SNMP를 사용하는 네트워크 디바이스의 활동을 제어하고 모니터링하는 데 쓰이는 시스템.
- SNMP 에이전트 - 섀시에 대한 데이터를 유지 관리하고 필요에 따라 데이터를 SNMP 관리자에게 보고하는 섀시 내의 소프트웨어 구성 요소입니다. 섀시는 MIB 컬렉션 및 에이전트를 포함합니다. SNMP 에이전트를 활성화하고 관리자와 에이전트 간의 관계를 생성하려면 Firepower Chassis Manager 또는 FXOS CLI에서 SNMP를 활성화하고 구성합니다.
- MIB(managed information base) - SNMP 에이전트에 있는 관리되는 개체의 모음.

섀시는 SNMPv1, SNMPv2c 및 SNMPv3를 지원합니다. SNMPv1 및 SNMPv2c는 모두 보안 커뮤니티 기반 양식을 사용합니다. SNMP는 다음에 정의되어 있습니다.

- RFC 3410(http://tools.ietf.org/html/rfc3410)
- RFC 3411(http://tools.ietf.org/html/rfc3411)

- RFC 3412(http://tools.ietf.org/html/rfc3412)
- RFC 3413(http://tools.ietf.org/html/rfc3413)
- RFC 3414(http://tools.ietf.org/html/rfc3414)
- RFC 3415(http://tools.ietf.org/html/rfc3415)
- RFC 3416(http://tools.ietf.org/html/rfc3416)
- RFC 3417(http://tools.ietf.org/html/rfc3417)
- RFC 3418(http://tools.ietf.org/html/rfc3418)
- RFC 3584(http://tools.ietf.org/html/rfc3584)

참고 SNMP 버전 1 및 2c에는 알려진 심각한 보안 문제가 있습니다. 이러한 버전에서 유일한 인증 형식으로 사용되는 커뮤니티 문자열을 포함하여 모든 정보를 암호화 없이 전송합니다.

SNMP 알림

SNMP의 주요 기능은 SNMP 에이전트에서 알림을 생성하는 기능입니다. 이러한 알림에는 SNMP 관리자가 요청을 전송하지 않아도 됩니다. 알림은 잘못된 사용자 인증, 재시작, 연결 종료, 네이버 라우터에 대한 연결 손실 또는 기타 중요한 이벤트를 나타낼 수 있습니다.

섀시는 트랩 또는 알림 중 하나로 SNMP 알림을 생성합니다. 트랩은 SNMP 관리자가 트랩을 수신할 때 승인을 보내지 않고 섀시가 트랩 수신 여부를 확인할 수 없기 때문에 알림보다 신뢰성이 떨어집니다. inform 요청을 수신한 SNMP 관리자는 SNMP 응답 PDU(protocol data unit)로 메시지를 승인합니다. 섀시가 PDU를 수신하지 않으면 알림 요청을 다시 보낼 수 있습니다.

그러나 알림은 안전하지 않은 것으로 간주되어 권장되지 않는 SNMPv2c에서만 사용할 수 있습니다.

참고 SNMP를 사용하는 인터페이스의 ifindex 순서는 FXOS를 재부팅한 후에도 변경되지 않습니다. 그러나 FXOS를 재부팅하면 FXOS 디스크 사용량 OID의 인덱스 번호가 변경됩니다.

SNMP 보안 수준 및 권한

SNMPv1, SNMPv2c 및 SNMPv3는 각각 다른 보안 모델을 나타냅니다. 보안 모델은 선택한 보안 수준과 결합하여 SNMP 메시지를 처리할 때 적용된 보안 메커니즘을 결정합니다.

보안 수준은 SNMP 트랩에 연결된 메시지를 표시하는 데 필요한 권한을 결정합니다. 권한 수준은 메시지가 공개되지 않도록 보호해야 하는지 또는 인증되어야 하는지를 결정합니다. 어떤 보안 모델이 구현되는지에 따라 지원되는 보안 수준이 달라집니다. SNMP 보안 수준은 다음 권한 중 하나 이상을 지원합니다.

- noAuthNoPriv — 인증 또는 암호화 없음
- authNoPriv — 인증은 있지만 암호화 없음
- authPriv — 인증 및 암호화

SNMPv3는 보안 모델 및 보안 수준을 모두 제공합니다. 보안 모델은 사용자 및 사용자 역할을 위해 설정된 인증 전략입니다. 보안 수준은 보안 모델에서 허용된 보안 수준입니다. 보안 모델과 보안 수준을 결합하여 SNMP 패킷을 처리할 때 어떤 보안 메커니즘이 적용되는지 결정합니다.

지원되는 **SNMP** 보안 모델과 수준 결합

다음 표에서는 어떻게 보안 모델과 수준을 결합할 수 있는지에 대해 설명합니다.

표 ***1: SNMP*** 보안 모델과 수준

모델	수준	인증	암호화	결과
v1	noAuthNoPriv	커뮤니티 문자열	없음	인증에 커뮤니티 문자열 일치를 사용합니다.
v2c	noAuthNoPriv	커뮤니티 문자열	없음	인증에 커뮤니티 문자열 일치를 사용합니다.
v3	noAuthNoPriv	사용자 이름	없음	인증에 사용자 이름 일치를 사용합니다. 참고 이를 구성할 수는 있지만, FXOS는 SNMP 버전 3에서 `noAuthNoPriv` 사용을 지원하지 않습니다.
v3	authNoPriv	HMAC-SHA	없음	HMAC SHA(Secure Hash Algorithm) 기반 인증을 제공합니다.
v3	authPriv	HMAC-SHA	DES	HMAC-SHA 알고리즘 기반 인증을 제공합니다. CBC(Cipher Block Chaining) DES(DES-56) 표준 기반의 인증과 함께 DES(Data Encryption Standard) 56비트 암호화도 제공합니다.

SNMPv3 보안 기능

SNMPv3는 네트워크에서 인증 및 암호화 프레임을 결합하여 디바이스에 대한 보안 액세스를 제공합니다. SNMPv3는 구성된 사용자가 수행하는 관리 작업에만 권한을 부여하고 SNMP 메시지를 암호화합니다. SNMPv3 USM(User-Based Security Model)은 SNMP 메시지 수준 보안을 참조하며 다음 서비스를 제공합니다.

- 메시지 통합 — 메시지가 무단으로 변경 또는 손상되지 않았는지, 그리고 데이터 시퀀스가 비악의적인 방식으로 발생할 수 있는 것보다 더 많이 변경되지 않았는지 확인합니다.

- 메시지 출처 인증 — 수신 데이터를 만든 사용자의 클레임된 ID가 확인되도록 보장합니다.
- 메시지 기밀성 및 암호화 — 권한이 없는 개인, 엔티티 또는 프로세스에 정보가 노출 또는 사용되지 않도록 합니다.

SNMP 지원

섀시는 SNMP에 다음을 지원합니다.

MIB 지원

섀시는 MIB에 대해 읽기 전용 액세스를 지원합니다.

사용 가능한 MIB와 이러한 MIB를 획득할 수 있는 위치에 대한 내용은 Cisco FXOS MIB 참조 가이드를 참조하십시오.

SNMPv3 사용자의 인증 프로토콜

섀시는 SNMPv3 사용자에 대해 HMAC-SHA-96(SHA) 인증 프로토콜을 지원합니다.

SNMPv3 사용자를 위한 AES 프라이버시 프로토콜

섀시는 SNMPv3 메시지 암호화를 위한 프라이버시 프로토콜 중 하나로 AES(Advanced Encryption Standard)를 사용하며 RFC 3826을 준수합니다.

프라이버시 비밀번호, 즉 priv 옵션에서는 SNMP 보안 암호화를 위해 DES 또는 128비트 AES 암호화를 선택할 수 있습니다. AES-128 구성을 활성화하고 SNMPv3 사용자에 대한 프라이버시 비밀번호가 있는 경우, Firepower 섀시는 해당 프라이버시 비밀번호를 사용하여 128비트 AES 키를 생성합니다. AES 프라이버시 비밀번호에는 최소 8자 이상을 포함할 수 있습니다. 암호가 일반 텍스트로 지정된 경우, 최대 64자를 지정할 수 있습니다.

SNMP 활성화 및 SNMP 속성 구성

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **SNMP**를 선택합니다.

단계 **2** **SNMP** 영역에서 다음 필드를 완성합니다.

이름	설명
Admin State(관리 상태**)** 체크 박스	SNMP 활성화 또는 비활성화 여부. 시스템에 SNMP 서버와의 통합이 포함된 경우에만 이 서비스를 활성화합니다.
Port(포트**)** 필드	섀시가 SNMP 호스트와 통신할 때 사용하는 포트. 기본 포트를 변경할 수 없습니다.

이름	설명
Community/Username(커뮤니티/사용자 이름**)** 필드	(선택적) SNMP v1 및 v2에서 폴링에 사용되는 커뮤니티 문자열. SNMP 커뮤니티 이름을 지정하면, SNMP 원격 관리자의 폴링 요청에 대해 SNMP 버전 1 및 2c도 자동으로 활성화됩니다. 이 필드는 SNMP v3에는 적용되지 않습니다. SNMP 버전 1 및 2c에는 알려진 심각한 보안 문제가 있습니다. 이러한 버전에서 유일한 인증 형식으로 사용되는 커뮤니티 문자열을 포함하여 모든 정보를 암호화 없이 전송합니다. 영숫자 문자열은 1자~32자로 입력합니다. @, &, ?를 사용하시 마십시오.(물음표)또는 공백을사용하지 마십시오. 기본값은**public**입니다. **Community/Username(**커뮤니티/사용자 이름**)** 필드가 이미 설정된 경우 빈 필드 오른쪽의 텍스트에 **Set: Yes(**설정**:** 예**)**가 표시됩니다. **Community/Username** 필드에 아직 값이 채워지지 않은 경우 빈 필드 오른쪽의 텍스트에 **Set: No(**설정**:** 아니요**)**가 표시됩니다. 참고 CLI 명령 **set snmp community**을 사용하여 기존 커뮤니티 문자열을 삭제할 수 있으므로, SNMP 원격 관리자의 폴링 요청에 대해 SNMP 버전 1 및 2c를 비활성화할 수 있습니다.
System Administrator Name(시스템 관리자 이름**)** 필드	SNMP 구현을 책임지는 담당자입니다. 이메일 주소, 이름, 전화 번호 등 최대 255자의 문자열로 입력합니다.
Location(위치**)** 필드	SNMP 에이전트(서버)가 실행되는 호스트의 위치입니다. 최대 510자의 영숫자 문자열을 입력합니다.

단계 **3** **Save(**저장**)**를 클릭합니다.

다음에 수행할 작업

SNMP 트랩 및 사용자를 생성합니다.

SNMP 트랩 생성

다음 절차에서는 SNMP 트랩을 생성하는 방법을 설명합니다.

참고 최대 8개의 SNMP 트랩을 정의할 수 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **SNMP**를 선택합니다.

단계 **2** **SNMP Traps(SNMP** 트랩**)** 영역에서 **Add(**추가**)**를 클릭합니다.

단계 **3** **Add SNMP Trap(SNMP** 트랩 추가**)** 대화 상자에서 다음 필드를 작성합니다.

이름	설명
Host Name(호스트 이름**)** 필드	섀시가 트랩을 전송해야 하는 SNMP 호스트의 호스트 이름 또는 IP 주소.
Community/Username(커뮤니티/사용자 이름**)** 필드	트랩 대상에 대한 액세스를 허용하는 데 필요한 SNMPv1/v2c 커뮤니티 문자열 또는 SNMPv3 사용자 이름을 입력합니다. 이것은 SNMP 서비스를 위해 구성된 커뮤니티 또는 사용자 이름과 동일해야 합니다. 영숫자 문자열은 1자~32자로 입력합니다. @(at 기호), \(백슬래시), "(큰 따옴표), ? (물음표) 또는 공백을 사용하지 마십시오.
Port(포트**)** 필드	섀시가 트랩을 위해 SNMP 호스트와 통신하는 포트입니다. 1 ~ 65535 범위의 정수를 입력합니다.
Version(버전**)** 필드	트랩에 사용되는 SNMP 버전 및 모델입니다. 다음 중 하나일 수 있습니다. • **V1** • **V2** • **V3** 참고 SNMP 버전 1 및 2c에는 알려진 심각한 보안 문제가 있습니다. 이러한 버전에서 유일한 인증 형식으로 사용되는 커뮤니티 문자열을 포함하여 모든 정보를 암호화 없이 전송합니다.
Type(유형**)** 필드	전송할 트랩 유형을 지정합니다. • 트랩 • 알림(버전이 **V2**인 경우에만 유효)
v3 Privilege(v3 권한**)** 필드	버전을 **V3**로 선택한 경우 트랩과 연결된 권한을 지정합니다. • **Auth** — 인증하지만 암호화 없음 • **Noauth** — 인증 또는 암호화 없음 선택할 수는 있지만 FXOS는 SNMPv3에서 이 보안 레벨을 지원하지 않습니다. • **Priv** — 인증 및 암호화

단계 **4** **OK(**확인**)**를 클릭하여 **Add SNMP Trap(SNMP** 트랩 추가**)** 대화 상자를 닫습니다.

단계 **5** **Save(**저장**)**를 클릭합니다.

SNMP 트랩 삭제

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **SNMP**를 선택합니다.

단계 **2** **SNMP Traps(SNMP** 트랩**)** 영역에서 삭제할 트랩에 해당하는 테이블의 행에 있는 **Delete(**삭제**)** 아이콘을 클릭합니다.

SNMPv3 사용자 생성

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **SNMP**를 선택합니다.

단계 **2** **SNMP Users(SNMP** 사용자**)** 영역에서 **Add(**추가**)**를 클릭합니다.

단계 **3** **Add SNMP User(SNMP** 사용자 추가**)** 대화 상자에서 다음 필드를 작성합니다.

이름	설명
Name(이름**)** 필드	SNMPv3 사용자에게 할당된 사용자 이름입니다. 최대 32자까지 입력할 수 있습니다. 이름은 문자로 시작해야 합니다. 올바른 문자에는 글자, 숫자, _(밑줄)이 포함됩니다. (마침표), @(at 기호) 및 -(하이픈)을 지정할 수 있습니다.
Auth Type(인증 유형**)** 필드	권한 부여 유형: **SHA**.
Use AES-128(AES-128 사용**)** 체크 박스	이 확인란을 선택한 경우, 해당 사용자는 AES-128 암호화를 사용합니다. 참고 SNMPv3는 DES를 지원하지 않습니다. AES-128 상자를 선택하지 않은 상태로 두면 프라이버시 암호화가 수행되지 않으며, 설정된 프라이버시 비밀번호가 적용되지 않습니다.

이름	설명
Password(비밀번호**)** 필드	이 사용자의 비밀번호입니다. FXOS에서는 다음 요건을 충족하지 않는 모든 비밀번호를 거부합니다. • 8자 이상, 80자 이하여야 합니다. • 문자, 숫자 및 다음 문자만 포함해야 합니다. ~`!@#%^&*()_-+{}[]\\:;"'<,>./ • 다음 기호는 포함할 수 없습니다. 예: $(달러 기호), ? (물음표) 또는 = (등호). • 각기 다른 문자를 5자 이상 포함해야 합니다. • 연속적으로 증가하거나 감소하는 문자나 숫자를 너무 많이 포함하면 안 됩니다. 예를 들어 "12345" 문자열에는 이러한 문자가 4개 포함되고 "ZYXW" 문자열에는 3개 포함됩니다. 증가/감소 문자의 총수가 특정 한도를 초과하는 경우(대개 해당 문자가 4~6개 이상 포함되는 경우) 단순성 검사에 실패하게 됩니다. 참고 연속적으로 증가하거나 감소하는 문자 사이에 증가하거나 감소하지 않는 문자가 사용되는 경우에는 증가/감소 문자 수가 재설정되지 않습니다. 예를 들어 abcd&!21의 경우 비밀번호 검사에 실패하지만 abcd&!25의 경우에는 비밀번호 검사에 통과합니다.
Confirm Password(비밀번호 확인**)** 필드	확인을 위해 다시 한 번 입력하는 비밀번호입니다.

이름	설명
Privacy Password(프라이버시 비밀번호**)** 필드	이 사용자의 프라이버시 비밀번호입니다. FXOS에서는 다음 요건을 충족하지 않는 모든 비밀번호를 거부합니다. • 8자 이상, 80자 이하여야 합니다. • 문자, 숫자 및 다음 문자만 포함해야 합니다. ~`!@#%^&*()_-+{}[]\\:;"'<,>./ • 다음 기호는 포함할 수 없습니다. 예: $(달러 기호), ? (물음표) 또는 = (등호). • 각기 다른 문자를 5자 이상 포함해야 합니다. • 연속적으로 증가하거나 감소하는 문자나 숫자를 너무 많이 포함하면 안 됩니다. 예를 들어 "12345" 문자열에는 이러한 문자가 4개 포함되고 "ZYXW" 문자열에는 3개 포함됩니다. 증가/감소 문자의 총수가 특정 한도를 초과하는 경우(대개 해당 문자가 4~6개 이상 포함되는 경우) 단순성 검사에 실패하게 됩니다. 참고 연속적으로 증가하거나 감소하는 문자 사이에 증가하거나 감소하지 않는 문자가 사용되는 경우에는 증가/감소 문자 수가 재설정되지 않습니다. 예를 들어 abcd&!21의 경우 비밀번호 검사에 실패하지만 abcd&!25의 경우에는 비밀번호 검사에 통과합니다.
Confirm Privacy Password(프라이버시 비밀번호 확인**)** 필드	확인을 위해 다시 한 번 입력하는 프라이버시 비밀번호입니다.

단계 **4** **OK(**확인**)**를 클릭하여 **Add SNMP User(SNMP** 사용자 추가**)** 대화 상자를 닫습니다.

단계 **5** **Save(**저장**)**를 클릭합니다.

SNMPv3 사용자 삭제

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **SNMP**를 선택합니다.

단계 **2** **SNMP Users(SNMP** 사용자**)** 영역에서 삭제할 사용자에 해당하는 테이블의 행에 있는 **Delete(**삭제**)** 아이콘을 클릭합니다.

HTTPS 구성

이 섹션에서는 Firepower 4100/9300 섀시에서 HTTPS를 구성하는 방법을 설명합니다.

참고 Firepower Chassis Manager 또는 FXOS CLI를 사용하여 HTTPS 포트를 변경할 수 있습니다. 다른 모든 HTTPS 구성 작업에는 FXOS CLI만 사용해야 합니다.

인증서, 키 링, 트러스트 포인트

HTTPS에서는 PKI(Public Key Infrastructure)의 구성 요소를 사용하여 두 디바이스, 이를테면 클라이언트 브라우저와 Firepower 4100/9300 섀시 간의 보안 통신을 설정합니다.

암호화 키 및 키 링

각 PKI 디바이스는 비대칭 RSA(Rivest-Shamir-Adleman) 암호화 키의 쌍을 보유합니다. 개인 키와 공개 키로 구성된 이 쌍은 내부 키 링에 저장됩니다. 두 키 중 하나로 암호화한 메시지는 나머지 키로 해독할 수 있습니다. 암호화된 메시지를 보낼 때 발신자는 수신자의 공개 키로 메시지를 암호화하며 수신자는 자신의 개인 키로 그 메시지를 해독합니다. 또한 발신자는 자체 개인 키로 알려진 메시지를 암호화('서명'이라고도 함)하여 공개 키의 소유권을 증명할 수도 있습니다. 수신자가 해당 공개 키를 사용하여 성공적으로 메시지를 해독할 수 있다면 발신자가 개인 키를 소유하고 있음이 입증됩니다. 암호화 키의 길이는 다양하지만, 일반적으로 512바이트 ~ 2048바이트입니다. 일반적으로는 길이가 더 긴 키가 짧은 키보다 안전합니다. FXOS에서는 초기 2048비트 키 쌍으로 기본 키 링을 제공하며 사용자가 추가 키 링을 생성할 수 있습니다.

클러스터 이름이 바뀌거나 인증서가 만료될 경우 기본 키 링 인증서를 수동으로 재생성해야 합니다.

인증서

안전한 통신을 위해 일차적으로 두 디바이스가 디지털 인증서를 교환합니다. 인증서는 디바이스 공개 키 및 디바이스 ID에 대한 서명된 정보를 포함하는 파일입니다. 디바이스에서 단순히 암호화된 통신을 지원하기 위해서는 자신의 키 쌍 및 자체 서명된 인증서를 생성할 수 있습니다. 원격 사용자가 자체 서명 인증서가 있는 디바이스에 연결할 경우 이 사용자가 디바이스의 ID를 용이하게 확인할 방법이 없으므로 사용자의 브라우저는 초기에 인증 경고를 표시합니다. 기본적으로 FXOS에는 기본 키 링의 공개 키를 포함하는 자체 서명 인증서가 내장되어 있습니다.

신뢰 지점

FXOS에 대한 더 강력한 인증을 제공하기 위해 신뢰할 수 있는 소스 또는 트러스트 포인트로부터 디바이스의 ID를 확인하는 서드파티 인증서를 얻어 설치할 수 있습니다. 서드파티 인증서는 해당 신뢰 지점에서 서명하는데, 이는 루트 CA(certificate authority), 중간 CA 또는 루트 CA로 연결되는 신뢰 체인의 일부인 Trust anchor가 될 수 있습니다. 새 인증서를 얻으려면 FXOS를 통해 인증서 요청을 생성하고 트러스트 포인트에 해당 요청을 제출해야 합니다.

중요 인증서는 Base64 인코딩 X.509(CER) 형식이어야 합니다.

키 링 생성

FXOS는 기본 키 링을 포함하여 최대 8개의 키 링을 지원합니다.

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 키 링의 이름을 생성합니다.

Firepower-chassis # **create keyring** *keyring-name*

단계 **3** SSL 키 길이(비트)를 설정합니다.

Firepower-chassis # **set modulus** {**mod1024** | **mod1536** | **mod2048** | **mod512**}

단계 **4** 트랜잭션을 커밋합니다.

Firepower-chassis # **commit-buffer**

예

다음 예에서는 키 크기 1024비트의 키 링을 생성합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # create keyring kr220
Firepower-chassis /security/keyring* # set modulus mod1024
Firepower-chassis /security/keyring* # commit-buffer
Firepower-chassis /security/keyring #
```

다음에 수행할 작업

이 키 링에 대한 인증서 요청을 생성합니다.

기본 키 링 재생성

클러스터 이름이 바뀌거나 인증서가 만료될 경우 기본 키 링 인증서를 수동으로 재생성해야 합니다.

참고 기본 키 링은 FXOS의 FCM에서만 사용됩니다.

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 기본 키 링에 대한 키 링 보안 모드로 들어갑니다.

Firepower-chassis /security # **scope keyring default**

단계 **3** 기본 키 링 재생성:

Firepower-chassis /security/keyring # **set regenerate yes**

단계 **4** 트랜잭션을 커밋합니다.

Firepower-chassis # **commit-buffer**

예

다음 예에서는 기본 키 링을 재생성합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # scope keyring default
Firepower-chassis /security/keyring* # set regenerate yes
Firepower-chassis /security/keyring* # commit-buffer
Firepower-chassis /security/keyring #
```

키 링에 대한 인증서 요청 생성

기본 옵션으로 키 링에 대한 인증서 요청 생성

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 키 링에 대한 구성 모드로 들어갑니다.

Firepower-chassis /security # **scope keyring** *keyring-name*

단계 **3** 지정된 IPv4 또는 IPv6 주소 또는 fabric interconnect의 이름을 사용하여 인증서 요청을 만듭니다. 인증서 요청에 대한 비밀번호를 입력하라는 프롬프트가 표시됩니다.

Firepower-chassis /security/keyring # **create certreq** {**ip** [*ipv4-addr* | *ipv6-v6*] | **subject-name** *name*}

단계 **4** 트랜잭션을 커밋합니다.

Firepower-chassis /security/keyring/certreq # **commit-buffer**

단계 **5** Trust anchor 또는 인증 증명으로 복사하여 전송할 수 있는 인증서 요청을 표시합니다.

Firepower-chassis /security/keyring # **show certreq**

예

다음 예는 기본 옵션으로 키 링에 대한 IPv4 주소와 함께 인증서 요청을 만들고 표시합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # scope keyring kr220
Firepower-chassis /security/keyring # create certreq ip 192.168.200.123 subject-name sjc04
Certificate request password:
Confirm certificate request password:
Firepower-chassis /security/keyring* # commit-buffer
Firepower-chassis /security/keyring # show certreq
Certificate request subject name: sjc04
Certificate request ip address: 192.168.200.123
Certificate request e-mail name:
Certificate request country name:
State, province or county (full name):
Locality (eg, city):
Organization name (eg, company):
Organization Unit name (eg, section):
Request:
-----BEGIN CERTIFICATE REQUEST-----
MIIBfTCB5wIBADARMQ8wDQYDVQQDEwZzYW1jMDQwgZ8wDQYJKoZIhvcNAQEBBQAD
gY0AMIGJAoGBALpKn1t8qMZO4UGqILKFXQQc2c8b/vW2rnRF8OPhKbhghLA1YZ1F
JqcYEG5Yl1+vgohLBTd45s0GC8m4RTLJWHo4SwccAUXQ5Zngf45YtX1WsylwUWV4
0re/zgTk/WCd56RfOBvWR2Dtztu2pGA14sd761zLxt29K7R8mzj6CAUVAgMBAAGg
LTArBgkqhkiG9w0BCQ4xHjAcMBoGA1UdEQEB/wQQMA6CBnNhbWMwNIcECsEiXjAN
BgkqhkiG9w0BAQQFAAOBgQCsxN0qUHYGFoQw56RwQueLTNPnrndqUwuZHUO03Teg
nhsyu4satpyiPqVV9viKZ+spvc6x5PWIcTWgHhH8BimOb/0OKuG8kwfIGGsEDlAv
TTYvUP+BZ9OFiPbRIA718S+V8ndXr1HejiQGxlDNqoN+odCXPc5kjoXD0lZTL09H
BA==
-----END CERTIFICATE REQUEST-----

Firepower-chassis /security/keyring #
```

다음에 수행할 작업

- BEGIN 및 END 줄을 포함한 인증서 요청의 텍스트를 복사하여 파일에 저장합니다. 키 링에 대한 인증서를 얻을 수 있도록, 인증서 요청이 포함된 파일을 Trust anchor 또는 인증 증명으로 전송합니다.
- 신뢰 지점을 생성하고 Trust anchor로부터 받은 신뢰 인증서에 대한 인증서 체인을 설정합니다.

고급 옵션으로 키 링에 대한 인증서 요청 생성

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 키 링에 대한 구성 모드로 들어갑니다.

Firepower-chassis /security # **scope keyring** *keyring-name*

단계 **3** 인증서 요청을 생성합니다.

Firepower-chassis /security/keyring # **create certreq**

단계 **4** 회사가 소재한 국가의 국가 코드를 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set country** *country name*

단계 **5** 요청과 연결된 DNS(Domain Name Server) 주소를 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set dns** *DNS Name*

단계 **6** 인증서 요청과 연결된 이메일 주소를 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set e-mail** *E-mail name*

단계 **7** Firepower 4100/9300 섀시의 IP 주소를 지정합니다.

Firepower-chassis /security/keyring/certreq*# **set ip** {*certificate request ip-address*|*certificate request ip6-address*}

단계 **8** 인증서를 요청하는 회사의 본사가 위치한 시/읍/면을 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set locality** *locality name (eg, city)*

단계 **9** 인증서를 요청하는 조직을 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set org-name** *organization name*

단계 **10** 조직 단위를 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set org-unit-name** *organizational unit name*

단계 **11** 인증서 요청에 대한 비밀번호를 지정합니다(선택 사항).

Firepower-chassis /security/keyring/certreq* # **set password** *certificate request password*

단계 **12** 인증서를 요청하는 회사의 본사가 위치한 시/도를 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set state** *state, province or county*

단계 **13** Firepower 4100/9300 섀시의 FQDN(Fully Qualified Domain Name)을 지정합니다.

Firepower-chassis /security/keyring/certreq* # **set subject-name** *certificate request name*

단계 **14** 트랜잭션을 커밋합니다.

Firepower-chassis /security/keyring/certreq # **commit-buffer**

단계 **15** Trust anchor 또는 인증 증명으로 복사하여 전송할 수 있는 인증서 요청을 표시합니다.

Firepower-chassis /security/keyring # **show certreq**

예

참고 2.7 이전 릴리스의 경우 FQDN 없이 "set dns" 또는 "set subject-name"을 사용하여 버퍼를 커밋하지 않는 것이 좋습니다. FQDN이 아닌 DNS 또는 주체 이름으로 인증 요구 사항을 생성하려고 하면 오류가 발생합니다.

다음 예는 고급 옵션으로 키 링에 대한 IPv4 주소와 함께 인증서 요청을 만들고 표시합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # scope keyring kr220
Firepower-chassis /security/keyring # create certreq
Firepower-chassis /security/keyring/certreq* # set "ip 192.168.200.123"
Firepower-chassis /security/keyring/certreq* #  set subject-name "sjc04"
Firepower-chassis /security/keyring/certreq* # set country "US"
Firepower-chassis /security/keyring/certreq* # set dns "bg1-samc-15A"
Firepower-chassis /security/keyring/certreq* # set email "test@cisco.com"
Firepower-chassis /security/keyring/certreq* # set locality "new york city"
Firepower-chassis /security/keyring/certreq* # set org-name "Cisco Systems"
Firepower-chassis /security/keyring/certreq* # set org-unit-name "Testing"
Firepower-chassis /security/keyring/certreq* # set state "new york"
Firepower-chassis /security/keyring/certreq* # commit-buffer
Firepower-chassis /security/keyring/certreq # show certreq
Certificate request subject name: sjc04
Certificate request ip address: 192.168.200.123
Certificate request e-mail name: test@cisco.com
Certificate request country name: US
State, province or county (full name): New York
Locality name (eg, city): new york city
Organization name (eg, company): Cisco
Organization Unit name (eg, section): Testing
Request:
-----BEGIN CERTIFICATE REQUEST-----
MIIBfTCB5wIBADARMQ8wDQYDVQQDEwZzYW1jMDQwgZ8wDQYJKoZIhvcNAQEBBQAD
gY0AMIGJAoGBALpKn1t8qMZO4UGqILKFXQQc2c8b/vW2rnRF8OPhKbhghLA1YZ1F
JqcYEG5Yl1+vgohLBTd45s0GC8m4RTLJWHo4SwccAUXQ5Zngf45YtX1WsylwUWV4
0re/zgTk/WCd56RfOBvWR2Dtztu2pGA14sd761zLxt29K7R8mzj6CAUVAgMBAAGg
LTArBgkqhkiG9w0BCQ4xHjAcMBoGA1UdEQEB/wQQMA6CBnNhbWMwNIcECsEiXjAN
BgkqhkiG9w0BAQQFAAOBgQCsxN0qUHYGFoQw56RwQueLTNPnrndqUwuZHUO03Teg
nhsyu4satpyiPqVV9viKZ+spvc6x5PWIcTWgHhH8BimOb/0OKuG8kwfIGGsEDlAv
TTYvUP+BZ9OFiPbRIA718S+V8ndXr1HejiQGxlDNqoN+odCXPc5kjoXD0lZTL09H
BA==
-----END CERTIFICATE REQUEST-----

Firepower-chassis /security/keyring/certreq #
```

다음에 수행할 작업

- BEGIN 및 END 줄을 포함한 인증서 요청의 텍스트를 복사하여 파일에 저장합니다. 키 링에 대한 인증서를 얻을 수 있도록, 인증서 요청이 포함된 파일을 Trust anchor 또는 인증 증명으로 전송합니다.
- 신뢰 지점을 생성하고 Trust anchor로부터 받은 신뢰 인증서에 대한 인증서 체인을 설정합니다.

트러스트 포인트 생성

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 신뢰 지점을 생성합니다.

Firepower-chassis /security # **create trustpoint** *name*

단계 **3** 이 신뢰 지점에 대한 인증서 정보를 지정합니다.

Firepower-chassis /security/trustpoint # **set certchain** [*certchain*]

명령에 인증서 정보를 지정하지 않은 경우, 루트 CA(Certificate Authority)에 인증 경로를 정의하는 Trust Point 목록 또는 인증서를 입력하라는 프롬프트가 표시됩니다. 해당 정보를 입력한 후 다음 행에 **ENDOFBUF**를 입력하여 완료합니다.

중요 인증서는 Base64 인코딩 X.509(CER) 형식이어야 합니다.

단계 **4** 트랜잭션을 커밋합니다.

Firepower-chassis /security/trustpoint # **commit-buffer**

예

다음 예에서는 신뢰 지점을 만들고 신뢰 지점에 대한 인증서를 제공합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # create trustpoint tPoint10
Firepower-chassis /security/trustpoint* # set certchain
Enter lines one at a time. Enter ENDOFBUF to finish. Press ^C to abort.
Trustpoint Certificate Chain:
> -----BEGIN CERTIFICATE-----
> MIIDMDCCApmgAwIBAgIBADANBgkqhkiG9w0BAQQFADB0MQswCQYDVQQGEwJVUzEL
> BxMMU2FuIEpvc2UsIENBMRUwEwYDVQQKEwxFeGFtcGxlIEluYy4xEzARBgNVBAsT
> ClRlc3QgR3JvdXAxGTAXBgNVBAMTEHRlc3QuZXhhbXBsZS5jb20xHzAdBgkqhkiG
> 9w0BCQEWEHVzZXJAZXhhbXBsZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJ
> AoGBAMZw4nTepNIDhVzb0j7Z2Je4xAG56zmSHRMQeOGHemdh66u2/XAoLx7YCcYU
> ZgAMivyCsKgb/6CjQtsofvtrmC/eAehuK3/SINv7wd6Vv2pBt6ZpXgD4VBNKOND1
> GMbkPayVlQjbG4MD2dx2+H8EH3LMtdZrgKvPxPTE+bF5wZVNAgMBAAGgJTAjBgkq
> hkiG9w0BCQcxFhMUQSBjaGFsbGVuZ2UgcGFzc3dvcmQwDQYJKoZIhvcNAQEFBQAD
> gYEAG61CaJoJaVMhzCl903O6Mg51zq1zXcz75+VFj2I6rH9asckCld3mkOVx5gJU
> Ptt5CVQpNgNLdvbDPSsXretysOhqHmp9+CLv8FDuy1CDYfuaLtvlWvfhevskV0j6
> jtcEMyZ+f7+3yh421ido3nO4MIGeBgNVHSMEgZYwgZOAFLlNjtcEMyZ+f7+3yh42
> 1ido3nO4oXikdjB0MQswCQYDVQQGEwJVUzELMAkGA1UECBMCQ0ExFDASBgNVBAcT
> C1NhbnRhIENsYXJhMRswGQYDVQQKExJOdW92YSBTeXN0ZW1zIEluYy4xFDASBgNV
> BAsTC0VuZ2luZWVyaW5nMQ8wDQYDVQQDEwZ0ZXN0Q0GCAQAwDAYDVR0TBAUwAwEB
> /zANBgkqhkiG9w0BAQQFAAOBgQAhWaRwXNR6B4g6Lsnr+fptHv+WVhB5fKqGQqXc
> wR4pYiO4z42/j9Ijenh75tCKMhW51az8copP1EBmOcyuhf5C6vasrenn1ddkkYt4
> PR0vxGc40whuiozBolesmsmjBbedUCwQgdFDWhDIZJwK5+N3x/kfa2EHU6id1avt
```

```
> 4YL5Jg==
> -----END CERTIFICATE-----
> ENDOFBUF
Firepower-chassis /security/trustpoint* # commit-buffer
Firepower-chassis /security/trustpoint #
```

다음에 수행할 작업

Trust anchor 또는 인증 증명에서 키 링 인증서를 받아 키 링으로 가져옵니다.

키 링으로 인증서 가져오기

시작하기 전에

- 키 링 인증서에 대한 인증서 체인을 포함하는 신뢰 지점을 구성합니다.
- Trust anchor 또는 인증 증명에서 키 링 인증서를 가져옵니다.

참고 HTTPS에 이미 구성된 키 링에서 인증서를 변경하는 경우 새 인증서를 적용하려면 HTTPS를 다시 시작해야 합니다. 자세한 내용은 HTTPS 재시작, 29 페이지를 참조하십시오.

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 인증서를 수신할 키 링에 대한 구성 모드로 들어갑니다.

Firepower-chassis /security # **scope keyring** *keyring-name*

단계 **3** 키 링 인증서를 수신한 Trust anchor 또는 인증 증명에 대한 신뢰 지점을 지정합니다.

Firepower-chassis /security/keyring # **set trustpoint** *name*

단계 **4** 키 링 인증서를 입력 및 업로드할 대화 상자를 엽니다.

Firepower-chassis /security/keyring # **set cert**

프롬프트에 Trust anchor 또는 인증 증명으로부터 받은 인증서의 텍스트를 붙여넣습니다. 인증서의 바로 다음 줄에 **ENDOFBUF**를 입력하여 인증서 입력을 완료합니다.

중요 인증서는 Base64 인코딩 X.509(CER) 형식이어야 합니다.

단계 **5** 트랜잭션을 커밋합니다.

Firepower-chassis /security/keyring # **commit-buffer**

예

다음 예에서는 신뢰 지점을 지정하고 인증서를 키 링으로 가져옵니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # scope keyring kr220
Firepower-chassis /security/keyring # set trustpoint tPoint10
Firepower-chassis /security/keyring* # set cert
Enter lines one at a time. Enter ENDOFBUF to finish. Press ^C to abort.
Keyring certificate:
> -----BEGIN CERTIFICATE-----
> MIIB/zCCAWgCAQAwgZkxCzAJBgNVBAYTAlVTMQswCQYDVQQIEwJDQTEVMBMGA1UE
> BxMMU2FuIEpvc2UsIENBMRUwEwYDVQQKEwxFeGFtcGxlIEluYy4xEzARBgNVBAsT
> ClRlc3QgR3JvdXAxGTAXBgNVBAMTEHRlc3QuZXhhbXBsZS5jb20xHzAdBgkqhkiG
> 9w0BCQEWEHVzZXJAZXhhbXBsZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJ
> AoGBAMZw4nTepNIDhVzb0j7Z2Je4xAG56zmSHRMQeOGHemdh66u2/XAoLx7YCcYU
> ZgAMivyCsKgb/6CjQtsofvtrmC/eAehuK3/SINv7wd6Vv2pBt6ZpXgD4VBNKONDl
> GMbkPayVlQjbG4MD2dx2+H8EH3LMtdZrgKvPxPTE+bF5wZVNAgMBAAGgJTAjBgkq
> hkiG9w0BCQcxFhMUQSBjaGFsbGVuZ2UgcGFzc3dvcmQwDQYJKoZIhvcNAQEFBQAD
> gYEAG61CaJoJaVMhzCl903O6Mg51zq1zXcz75+VFj2I6rH9asckCld3mkOVx5gJU
> Ptt5CVQpNgNLdvbDPSsXretysOhqHmp9+CLv8FDuy1CDYfuaLtvlWvfhevskV0j6
> mK3Ku+YiORnv6DhxrOoqau8r/hyI/L43l7IPN1HhOi3oha4=
> -----END CERTIFICATE-----
> ENDOFBUF
Firepower-chassis /security/keyring* # commit-buffer
Firepower-chassis /security/keyring #
```

다음에 수행할 작업

HTTPS 서비스를 키 링으로 구성합니다.

HTTPS 구성

⚠

주의 HTTPS에서 사용하는 포트 및 키 링 변경을 포함하여 HTTPS 구성을 완료한 후 트랜잭션을 저장하거나 커밋하자마자 모든 현재 HTTP 및 HTTPS 세션이 종료됩니다.

프로시저

단계 **1** 시스템 모드를 입력합니다.

Firepower-chassis# **scope system**

단계 **2** 시스템 서비스 모드를 입력합니다.

Firepower-chassis /system #**scope services**

단계 **3** HTTPS 서비스를 활성화합니다.

Firepower-chassis /system/services # **enable https**

단계 **4** (선택 사항) HTTPS 연결에 사용할 포트를 지정합니다.

Firepower-chassis /system/services # **set https port** *port-num*

단계 **5** (선택 사항) HTTPS에 대해 생성한 키 링의 이름을 지정합니다.

Firepower-chassis /system/services # **set https keyring** *keyring-name*

단계 **6** (선택 사항) 도메인에서 사용하는 Cipher Suite 보안 레벨을 지정합니다.

Firepower-chassis /system/services # **set https cipher-suite-mode** *cipher-suite-mode*

*cipher-suite-mode*는 다음 키워드 중 하나일 수 있습니다.

- **high-strength**
- **medium-strength**
- **low-strength**
- **custom**- 사용자 정의 Cipher Suite 사양 문자열을 지정할 수 있습니다.

단계 **7** (선택 사항) **cipher-suite-mode**가 **custom**으로 설정된 경우 도메인에 대한 Cipher Suite 보안의 커스텀 레벨을 지정합니다.

Firepower-chassis /system/services # **set https cipher-suite** *cipher-suite-spec-string*

*cipher-suite-spec-string*은 최대 256자이며 OpenSSL Cipher Suite 사양을 준수해야 합니다. 공백 또는 특수 문자를 사용할 수 없습니다. 단, !(느낌표), +(덧셈 기호), -(하이픈), :(콜론)은 사용할 수 있습니다. 자세한 내용은 http://httpd.apache.org/docs/2.0/mod/mod_ssl.html#sslciphersuite를 참조하십시오.

예를 들어 FXOS에서 기본값으로 사용하는 중간 강도 사양 문자열은 다음과 같습니다.
ALL:!ADH:!EXPORT56:!LOW:RC4+RSA:+HIGH:+MEDIUM:+EXP:+eNULL

참고 **cipher-suite-mode**가 **custom** 이외의 값으로 설정되어 있으면 이 옵션은 무시됩니다.

단계 **8** (선택 사항) 인증서 해지 목록 확인을 활성화 또는 비활성화합니다.

set revoke-policy *{ relaxed | strict }*

단계 **9** 시스템 구성에 트랜잭션을 커밋합니다.

Firepower-chassis /system/services # **commit-buffer**

예

다음 예에서는 HTTPS를 활성화하고, 포터 번호를 443으로 설정하고, 키 링 이름을 kring7984로 설정하고, Cipher Suite 보안 레벨을 high로 설정하고, 트랜잭션을 커밋합니다.

```
Firepower-chassis# scope system
Firepower-chassis /system # scope services
Firepower-chassis /system/services # enable https
Firepower-chassis /system/services* # set https port 443
Warning: When committed, this closes all the web sessions.
Firepower-chassis /system/services* # set https keyring kring7984
Firepower-chassis /system/services* # set https cipher-suite-mode high
```

```
Firepower-chassis /system/services* # commit-buffer
Firepower-chassis /system/services #
```

HTTPS 포트 변경

HTTPS 서비스는 기본적으로 포트 443에서 활성화되어 있습니다. HTTPS를 비활성화할 수는 없지만, HTTPS 연결에 사용할 포트를 변경할 수 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **HTTPS**를 선택합니다.

단계 **2** HTTPS 연결에 사용할 포트를 **Port(**포트**)** 필드에 입력합니다. 1~65535 사이의 정수를 입력합니다. 이 서비스는 기본적으로 포트 443에서 활성화됩니다.

단계 **3** **Save(**저장**)**를 클릭합니다.

섀시는 HTTPS 포트가 지정된 상태로 구성됩니다.

HTTPS 포트를 변경한 후에는 현재의 모든 HTTPS 세션이 종료됩니다. 사용자는 다음과 같이 새 포트를 사용하여 Firepower Chassis Manager에 다시 로그인해야 합니다.

https://*<chassis_mgmt_ip_address>*:*<chassis_mgmt_port>*

이때 <*chassis_mgmt_ip_address*>는 사용자가 초기 구성을 설정하는 동안 입력한 섀시의 IP 주소 또는 호스트 이름이며 <*chassis_mgmt_port*>는 방금 구성한 HTTPS 포트입니다.

HTTPS 재시작

HTTPS에 이미 구성된 키 링에서 인증서를 변경하는 경우, 새 인증서를 적용하려면 HTTPS를 다시 시작해야 합니다. 업데이트된 키 링으로 HTTPS를 재설정하려면 다음 절차를 사용합니다.

프로시저

단계 **1** 시스템 모드를 입력합니다.

Firepower-chassis# **scope system**

단계 **2** 시스템 서비스 모드를 입력합니다.

Firepower-chassis /system #**scope services**

단계 **3** HTTPS 키 링을 기본값으로 다시 설정합니다.

Firepower-chassis /system/services # **set https keyring default**

단계 **4** 시스템 구성에 트랜잭션을 커밋합니다.

Firepower-chassis /system/services # **commit-buffer**

단계 **5** 5초 동안 기다립니다.

단계 **6** 생성한 키 링으로 HTTPS를 설정합니다.

Firepower-chassis /system/services # **set https keyring** *keyring-name*

단계 **7** 시스템 구성에 트랜잭션을 커밋합니다.

Firepower-chassis /system/services # **commit-buffer**

키 링 삭제

프로시저

단계 **1** 보안 모드를 입력합니다.

Firepower-chassis # **scope security**

단계 **2** 명명된 키 링을 삭제합니다.

Firepower-chassis /security # **delete keyring** *name*

단계 **3** 트랜잭션을 커밋합니다.

Firepower-chassis /security # **commit-buffer**

예

다음 예에서는 사용자 계정을 삭제합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # delete keyring key10
Firepower-chassis /security* # commit-buffer
Firepower-chassis /security #
```

트러스트 포인트 삭제

시작하기 전에

신뢰 지점이 키 링에서 사용하지 않음을 확인합니다.

프로시저

단계 **1** 보안 모드로 들어갑니다.

Firepower-chassis# **scope security**

단계 **2** 명명된 신뢰 지점을 삭제합니다.

Firepower-chassis /security # **delete trustpoint** *name*

단계 **3** 트랜잭션을 커밋합니다.

Firepower-chassis /security # **commit-buffer**

예

다음 예에서는 신뢰 지점을 삭제합니다.

```
Firepower-chassis# scope security
Firepower-chassis /security # delete trustpoint tPoint10
Firepower-chassis /security* # commit-buffer
Firepower-chassis /security #
```

HTTPS 비활성화

프로시저

단계 **1** 시스템 모드를 입력합니다.

Firepower-chassis# **scope system**

단계 **2** 시스템 서비스 모드를 입력합니다.

Firepower-chassis /system # **scope services**

단계 **3** HTTPS 서비스를 비활성화합니다.

Firepower-chassis /system/services # **disable https**

단계 **4** 시스템 구성에 트랜잭션을 커밋합니다.

Firepower-chassis /system/services # **commit-buffer**

예

다음 예에서는 HTTPS를 비활성화하고 트랜잭션을 커밋합니다.

```
Firepower-chassis# scope system
Firepower-chassis /system # scope services
Firepower-chassis /system/services # disable https
Firepower-chassis /system/services* # commit-buffer
Firepower-chassis /system/services #
```

AAA 구성

이 섹션에서는 인증, 권한 부여 및 어카운팅에 관해 설명합니다. 자세한 내용은 다음 항목을 참고하십시오.

AAA 정보

AAA(인증, 권한 부여 및 계정 관리)는 네트워크 리소스에 대한 액세스 제어를 위한 서비스의 집합으로, 정책을 구현하고, 사용량을 평가하고 서비스에 대한 청구에 필요한 정보를 제공합니다. 인증은 사용자를 식별합니다. 권한 부여는 사용자가 액세스할 수 있는 리소스와 서비스를 결정하는 정책을 구현합니다. 어카운팅은 청구 및 분석을 위해 사용되는 시간과 데이터를 추적합니다. 이 과정은 효과적인 네트워크 관리 및 보안을 위해 중요한 부분으로 간주됩니다.

인증

인증은 액세스를 부여하기 전에 사용자가 유효한 사용자 이름과 유효한 암호를 입력하게 하여 각 사용자를 식별하는 방법을 제공합니다. AAA 서버는 사용자의 인증 크리덴셜을 데이터베이스에 저장된 다른 사용자의 크리덴셜과 비교합니다. 크리덴셜이 일치하면 사용자는 네트워크에 액세스할 수 있습니다. 크리덴셜이 일치하지 않으면, 인증에 실패하고 네트워크 액세스가 거부됩니다.

다음 세션을 포함하는 섀시에 대한 관리 연결을 인증하도록 Firepower 4100/9300 섀시를 구성할 수 있습니다.

- HTTPS
- SSH
- 시리얼 콘솔

권한 부여

권한 부여는 정책을 구현하는 프로세스로, 사용자의 액세스가 허용된 활동, 리소스 또는 서비스 유형을 결정합니다. 권한이 부여되면, 사용자는 다양한 액세스 또는 활동 유형에 대한 권한을 가질 수 있습니다.

어카운팅

어카운팅은 사용자가 액세스 중 소비하는 리소스를 측정합니다. 여기에는 시스템 사용 시간, 사용자가 세션 중 보내거나 받는 데이터의 양 등이 포함됩니다. 어카운팅은 세션 통계 및 사용량 정보 기록을 통해 이루어지며 이는 권한 부여 제어, 청구, 경향 분석, 리소스 활용도 및 용량 계획 활동에 사용됩니다.

인증, 권한 부여 및 어카운팅 간 상호 작용

인증을 단독으로 사용하거나 권한 부여 및 어카운팅과 함께 사용할 수 있습니다. 권한 부여에서는 항상 먼저 사용자의 인증 여부를 확인해야 합니다. 어카운팅을 단독으로 사용하거나 인증 및 권한 부여와 함께 사용할 수 있습니다.

지원되는 인증 유형

FXOS는 다음 유형의 사용자 인증을 지원합니다.

- 원격 - 다음 네트워크 AAA 서비스가 지원됩니다.
  - LDAP
  - RADIUS
  - TACACS+
- 로컬 - 섀시는 사용자가 사용자 프로파일을 채울 수 있는 로컬 데이터베이스를 유지합니다. AAA 서버 대신 로컬 데이터베이스를 사용하여 사용자 인증, 권한 부여 및 어카운팅을 제공할 수 있습니다.

사용자 역할

FXOS는 사용자 역할 할당 형식으로 로컬 및 원격 권한 부여를 지원합니다. 할당할 수 있는 역할은 다음과 같습니다.

- **Admin** - 전체 시스템에 대한 완전한 읽기 및 쓰기 액세스. 기본 관리자 계정이 기본적으로 이 역할에 할당되며 변경할 수 없습니다.
- **AAA Administrator** - 사용자, 역할 및 AAA 구성에 대한 읽기-쓰기 액세스. 나머지 시스템에 대한 읽기 액세스
- **Operations** - NTP 구성, 스마트 라이센싱을 위한 Smart Call Home 구성, 시스템 로그(syslog 서버 및 장애 포함)에 대한 읽기 및 쓰기 액세스. 나머지 시스템에 대한 읽기 액세스
- **Read-Only** - 시스템 상태를 수정할 수 있는 권한이 없는 시스템 구성에 대한 읽기 전용 액세스

로컬 사용자 및 역할 할당에 대한 자세한 내용은 사용자 관리 섹션을 참조하십시오.

AAA 설정

이 단계에서는 Firepower 4100/9300 어플라이언스에서 AAA(Authentication, Authorization and Accounting)를 설정하기 위한 기본 개요를 제공합니다.

1. 원하는 사용자 인증 유형을 구성합니다.
   - 로컬 - 사용자 정의 및 로컬 인증이 사용자 관리의 일부입니다.
   - 원격 - 원격 AAA 서버 액세스 구성은 플랫폼 설정의 일부입니다. 구체적으로는 다음과 같습니다.
     - LDAP 제공자 구성, 34 페이지
     - RADIUS 제공자 구성, 38 페이지
     - TACACS+ 제공자 구성, 40 페이지

참고 원격 AAA 서버를 사용하려는 경우, 섀시에서 원격 AAA 서버 액세스를 구성하기 전에 원격 서버에서 AAA 서비스를 활성화하고 구성해야 합니다.

2. 기본 인증 방법을 지정합니다. 이 역시 사용자 관리의 일부입니다.

참고 Default Authentication(기본 인증) 및 Console Authentication(콘솔 인증)이 모두 동일한 원격 인증 프로토콜(RADIUS, TACACS+ 또는 LDAP)을 사용하도록 설정된 경우, 이러한 사용자 설정을 업데이트해야 해당 서버 구성의 특정 측면(예: 해당 서버 삭제 또는 할당 순서 변경)을 변경할 수 있습니다.

LDAP 제공자 구성

LDAP 제공자 속성 구성

이 작업에서 구성하는 속성은 이 유형의 모든 제공자 연결에 대한 기본 설정입니다. 개별 제공자에 이러한 속성 중 하나에 대한 설정이 포함된 경우 FXOS에서는 해당 설정을 사용하고 기본 설정을 무시합니다.

Active Directory를 LDAP 서버로 사용하는 경우에는 Active Directory 서버에서 사용자 어카운트를 생성하여 FXOS와 바인딩합니다. 이 계정은 만료되지 않는 비밀번호를 가져야 합니다.

프로시저

단계 1 **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 2 **LDAP** 탭을 클릭합니다.

단계 3 **Properties(**속성**)** 영역에서 다음 필드를 완성합니다.

이름	설명
Timeout(시간 초과**)** 필드	시간이 초과되기 전에 시스템이 TACACS+ 데이터베이스에 연결을 시도하는 데 필요한 시간(초). 1~60초의 정수를 입력합니다. 기본값은 30초입니다. 이 속성은 필수 항목입니다.
Attribute(속성**)** 필드	사용자 역할 및 로케일에 대해 값을 저장하는 LDAP 속성. 이 속성은 항상 이름값 쌍입니다. 시스템은 사용자 레코드를 쿼리하여 이 속성 이름과 일치하는 값을 찾습니다. LDAP 제공자에 대한 속성을 구성할 때는 `shell:roles="admin,aaa"` 속성 값이 필요합니다.

이름	설명
Base DN(기본 **DN)** 필드	원격 사용자가 로그인하고 시스템에서 사용자 이름에 기초한 사용자 DN을 얻으려고 시도할 때 서버에서 검색을 시작해야 하는 LDAP 계층 구조에서 특정한 고유 이름입니다. 기본 DN 길이는 최대 255자에서 *cn=$userid* 길이를 뺀 문자 수로 설정될 수 있습니다. 이때 $userid는 LDAP 인증을 사용하여 섀시에 액세스를 시도하는 원격 사용자를 식별합니다. 이 속성은 LDAP 제공자에 필요합니다. 이 탭에서 기본 DN을 지정하지 않으면 정의하는 각 LDAP 제공자에 하나를 지정해야 합니다.
Filter(필터**)** 필드	LDAP 서버에 사용할 필터 속성을 입력합니다(예: *cn=$userid* 또는 *sAMAccountName=$userid*). LDAP 검색은 정의된 필터와 일치하는 사용자 이름으로 제한됩니다. 필터는 *$userid*를 포함해야 합니다. 이 속성은 필수 항목입니다. 이 탭에서 필터를 지정하지 않으면 정의하는 각 LDAP 제공자에 하나를 지정해야 합니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

다음에 수행할 작업

LDAP 제공자를 생성합니다.

LDAP 제공자 생성

다음 단계에 따라 LDAP 공급자, 즉 이 어플라이언스에 LDAP 기반 AAA 서비스를 제공하는 특정 원격 서버를 정의하고 구성합니다.

참고 FXOS에서는 최대 16개의 LDAP 제공자를 지원합니다.

시작하기 전에

Active Directory를 LDAP 서버로 사용하는 경우에는 Active Directory 서버에서 사용자 어카운트를 생성하여 FXOS와 바인딩합니다. 이 계정은 만료되지 않는 비밀번호를 가져야 합니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 **2** **LDAP** 탭을 클릭합니다.

단계 **3** 추가할 각 LDAP 제공자에 대해 다음을 수행합니다.

a) **LDAP Providers(LDAP** 제공자**)** 영역에서 **Add(**추가**)**를 클릭합니다.

b) **Add LDAP Provider(LDAP** 제공자 추가**)** 대화 상자에서 다음 필드를 작성합니다.

이름	설명
호스트 이름**/FQDN(**또는 **IP** 주소**)** 필드	LDAP 서버의 호스트 이름 또는 IP 주소. SSL이 활성화된 경우 이 필드는 LDAP 데이터베이스 보안 인증서의 CN(Common Name)과 정확히 일치해야 합니다.
Order(순서**)** 필드	FXOS에서 사용자를 인증하기 위해 이 제공자를 사용하는 순서입니다. Firepower Chassis Manager 또는 FXOS CLI에 정의되어 있는 다른 제공자에 기반하여 FXOS이 다음으로 사용 가능한 순서를 할당하게 하려면 1~16 사이의 정수를 입력하거나 **`lowest-available`** 또는 **`0`**(숫자 0)을 입력합니다.
Bind DN(바인드 **DN)** 필드	기본 DN에 속하는 모든 객체에 대한 읽기 및 검색 권한이 있는 LDAP 데이터베이스 어카운트의 고유 이름(DN)입니다. 지원되는 최대 문자열 길이는 ASCII 255자입니다.
Base DN(기본 **DN)** 필드	원격 사용자가 로그인하고 시스템에서 사용자 이름에 기초한 사용자 DN을 얻으려고 시도할 때 서버에서 검색을 시작해야 하는 LDAP 계층 구조에서 특정한 고유 이름입니다. 기본 DN 길이는 최대 255자에서 CN=$userid 길이를 뺀 문자 수로 설정될 수 있습니다. 이때 $userid는 LDAP 인증을 사용하여 Firepower Chassis Manager 또는 FXOS CLI에 액세스를 시도하는 원격 사용자를 식별합니다. 이 값은 기본 DN의 기본값이 **LDAP** 탭에 설정되지 않은 경우 필요합니다.
Port(포트**)** 필드	Firepower Chassis Manager 또는 FXOS CLI에서 LDAP 데이터베이스와 통신할 때 사용하는 포트입니다. 표준 포트 번호는 389입니다.
Enable SSL(SSL 활성화**)** 체크박스	이 확인란을 선택한 경우, LDAP 데이터베이스와의 통신에 암호화가 필요합니다. 이 확인란이 선택되지 않은 경우, 인증 정보는 암호화되지 않은 텍스트로 전송됩니다. LDAP는 STARTTLS를 사용합니다. 이는 포트 389를 사용하여 암호화된 통신을 허용합니다. 참고 STARTTLS 작업을 수행하려면 FXOS 인증서 체인에 LDAP 제공자의 CA 인증서를 설치해야 합니다.

이름	설명
Filter(필터**)** 필드	LDAP 서버에 사용할 필터 속성을 입력합니다(예: cn=$userid 또는 sAMAccountName=$userid). LDAP 검색은 정의된 필터와 일치하는 사용자 이름으로 제한됩니다. 필터는 $userid를 포함해야 합니다. 이 값은 기본 필터가 **LDAP** 탭에 설정되지 않은 경우 필요합니다.
Attribute(속성**)** 필드	사용자 역할 및 로케일에 대해 값을 저장하는 LDAP 속성. 이 속성은 항상 이름값 쌍입니다. 시스템은 사용자 레코드를 쿼리하여 이 속성 이름과 일치하는 값을 찾습니다. 이 값은 기본 속성이 **LDAP** 탭에 설정되지 않은 경우 필요합니다.
Key(키**)** 필드	**Bind DN(**바인드 **DN)** 필드에 지정된 LDAP 데이터베이스 어카운트의 비밀번호입니다. 공백, §(섹션 기호), ?(물음표) 또는 =(등호)를 제외한 모든 표준 ASCII 문자를 입력할 수 있습니다.
Confirm Key(키 확인**)** 필드	확인을 위해 다시 입력하는 LDAP 데이터베이스 비밀번호.
Timeout(시간 초과**)** 필드	시간이 초과되기 전에 시스템이 TACACS+ 데이터베이스에 연결을 시도하는 데 필요한 시간(초). 1~60초의 정수를 입력하거나 0(숫자 0)을 입력하여 **LDAP** 탭에 지정된 전역 시간제한 값을 사용합니다. 기본값은 30초입니다.
Vendor(벤더**)** 필드	이 선택사항으로 LDAP 제공자 또는 서버 상세 정보를 제공하는 벤더를 식별합니다. • LDAP 제공자가 Microsoft Active Directory인 경우, **MS AD**를 선택합니다. • LDAP 제공자가 Microsoft Active Directory가 아닌 경우, **Open LDAP(LDAP** 열기**)**를 선택합니다. 기본값은 **Open LDAP(LDAP** 열기**)**입니다.

c) **OK(**확인**)**를 클릭하여 **Add LDAP Provider(LDAP** 제공자 추가**)** 대화 상자를 닫습니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

단계 **5** (선택 사항) 인증서 해지 목록 확인을 활성화합니다.

Firepower-chassis /security/ldap/server # **set revoke-policy** *{strict | relaxed}*

참고 이 구성은 SSL 연결이 활성화된 경우에만 적용됩니다.

LDAP 제공자 삭제

프로시저

단계 1 **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 2 **LDAP** 탭을 클릭합니다.

단계 3 **LDAP Providers(LDAP** 제공자**)** 영역에서 삭제할 LDAP 제공자에 해당하는 테이블의 행에 있는 **Delete(**삭제**)** 아이콘을 클릭합니다.

RADIUS 제공자 구성

RADIUS 제공자의 속성 구성

이 작업에서 구성하는 속성은 이 유형의 모든 제공자 연결에 대한 기본 설정입니다. 개별 제공자에 이러한 속성 중 하나에 대한 설정이 포함된 경우, FXOS에서는 해당 설정을 사용하고 기본 설정을 무시합니다.

프로시저

단계 1 **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 2 **RADIUS** 탭을 클릭합니다.

단계 3 **Properties(**속성**)** 영역에서 다음 필드를 완성합니다.

이름	설명
Timeout(시간 초과**)** 필드	시간이 초과되기 전에 시스템이 RADIUS 데이터베이스에 연결을 시도하는 데 필요한 시간(초)입니다. 1~60의 정수를 입력합니다. 기본값은 180초입니다. 이 속성은 필수입니다.
Retries(재시도 횟수**)** 필드	요청에 실패한 것으로 간주하기 전에 연결을 재시도할 횟수입니다.

단계 4 **Save(**저장**)**를 클릭합니다.

다음에 수행할 작업

RADIUS 제공자를 생성합니다.

RADIUS 제공자 생성

RADIUS 제공자, 즉 이 어플라이언스에 대해 RADIUS 기반 AAA 서비스를 제공하는 특정 원격 서버를 정의하고 구성하려면 다음 단계를 수행하십시오.

참고 FXOS에서는 최대 16개의 RADIUS 제공자를 지원합니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 **2** **RADIUS** 탭을 클릭합니다.

단계 **3** 추가할 각 RADIUS 제공자에 대해 다음을 수행합니다.

a) **RADIUS Providers(RADIUS** 제공자**)** 영역에서 **Add(**추가**)**를 클릭합니다.

b) **Add RADIUS Provider(RADIUS** 제공자 추가**)** 대화 상자에서 다음 필드를 작성합니다.

이름	설명
호스트 이름/**FQDN(**또는 **IP** 주소**)** 필드	RADIUS 서버의 호스트 이름 또는 IP 주소
Order(순서**)** 필드	FXOS에서 사용자를 인증하기 위해 이 제공자를 사용하는 순서입니다. Firepower Chassis Manager 또는 FXOS CLI에 정의되어 있는 다른 제공자에 기반하여 FXOS이 다음으로 사용 가능한 순서를 할당하게 하려면 1~16 사이의 정수를 입력하거나 **lowest-available** 또는 **0**(숫자 0)을 입력합니다.
Key(키**)** 필드	데이터베이스에 대한 SSL 암호화 키입니다. 공백, §(섹션 기호), ? (물음표) 또는 =(등호)를 제외한 모든 표준 ASCII 문자를 입력할 수 있습니다.
Confirm Key(키 확인**)** 필드	확인을 위해 다시 한 번 입력하는 SSL 암호화 키
Authorization Port(권한 부여 포트**)** 필드	Firepower Chassis Manager 또는 FXOS CLI에서 RADIUS 데이터베이스와 통신할 때 사용하는 포트입니다. 유효한 범위는 1~65535입니다. 표준 포트 번호는 1700입니다.
Timeout(시간 초과**)** 필드	시간이 초과되기 전에 시스템이 RADIUS 데이터베이스에 연결을 시도하는 데 필요한 시간(초) 길이 1~60초의 정수를 입력하거나 0(숫자 0)을 입력하여 **RADIUS** 탭에 지정된 전역 시간제한 값을 사용합니다. 기본값은 5일입니다.

이름	설명
Retries(재시도 횟수**)** 필드	요청에 실패한 것으로 간주하기 전에 연결을 재시도할 횟수입니다. 필요 시 0~5의 정수를 입력합니다. 값을 지정하지 않은 경우, Firepower Chassis Manager에서는 **RADIUS** 탭에 지정된 값을 사용합니다.

c) **OK(**확인**)**를 클릭하여 **Add RADIUS Provider(RADIUS** 제공자 추가**)** 대화 상자를 닫습니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

RADIUS 제공자 삭제

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 **2** **RADIUS** 탭을 클릭합니다.

단계 **3** **RADIUS Providers(RADIUS** 제공자**)** 영역에서 삭제할 RADIUS 제공자에 해당하는 테이블의 행에 있는 **Delete(**삭제**)** 아이콘을 클릭합니다.

TACACS+ 제공자 구성

TACACS+ 제공자 속성 구성

이 작업에서 구성하는 속성은 이 유형의 모든 제공자 연결에 대한 기본 설정입니다. 개별 제공자에 이러한 속성 중 하나에 대한 설정이 포함된 경우, FXOS에서는 해당 설정을 사용하고 기본 설정을 무시합니다.

참고 FXOS 섀시는 TACACS+(Terminal Access Controller Access-Control System Plus) 프로토콜에 대한 명령 어카운팅을 지원하지 않습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 **2** **TACACS** 탭을 클릭합니다.

단계 **3** **Properties(**속성**)** 영역에서 다음 필드를 완성합니다.

이름	설명
Timeout(시간 초과**)** 필드	시간이 초과되기 전에 시스템이 TACACS+ 데이터베이스에 연결을 시도하는 데 필요한 시간(초). 1~60의 정수를 입력합니다. 기본값은 180초입니다. 이 속성은 필수입니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

다음에 수행할 작업

TACACS+ 제공자를 만듭니다.

TACACS+ 제공자 생성

다음 단계를 따라 TACACS+ 제공자, 즉 이 어플라이언스에 대해 TACACS 기반 AAA 서비스를 제공하는 특정 원격 서버를 정의하고 구성합니다.

참고 FXOS에서는 최대 16개의 TACACS+ 제공자를 지원합니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 **2** **TACACS** 탭을 클릭합니다.

단계 **3** 추가할 각 TACACS+ 제공자에 대해 다음을 수행합니다.

a) **TACACS Providers(TACACS** 제공자**)** 영역에서 **Add(**추가**)**를 클릭합니다.

b) **Add TACACS Provider(TACACS** 제공자 추가**)** 대화 상자에서 다음 필드를 작성합니다.

이름	설명
호스트 이름**/FQDN(**또는 **IP** 주소**)** 필드	TACACS+ 서버의 호스트 이름 또는 IP 주소
Order(순서**)** 필드	FXOS에서 사용자를 인증하기 위해 이 제공자를 사용하는 순서입니다. Firepower Chassis Manager 또는 FXOS CLI에 정의되어 있는 다른 제공자에 기반하여 FXOS이 다음으로 사용 가능한 순서를 할당하게 하려면 1~16 사이의 정수를 입력하거나 **lowest-available** 또는 **0**(숫자 0)을 입력합니다.

이름	설명
Key(키**)** 필드	데이터베이스에 대한 SSL 암호화 키입니다. 공백, §(섹션 기호), ? (물음표) 또는 =(등호)를 제외한 모든 표준 ASCII 문자를 입력할 수 있습니다.
Confirm Key(키 확인**)** 필드	확인을 위해 다시 한 번 입력하는 SSL 암호화 키
Port(포트**)** 필드	Firepower Chassis Manager 또는 FXOS CLI이 TACACS+ 서버와 통신할 때 사용하는 포트. 1~65535의 정수를 입력합니다. 기본 포트는 49입니다.
Timeout(시간 초과**)** 필드	시간이 초과되기 전에 시스템이 TACACS+ 데이터베이스에 연결을 시도하는 데 필요한 시간(초). 1~60초의 정수를 입력하거나 0(숫자 0)을 입력하여 **TACACS+** 탭에 지정된 전역 시간제한 값을 사용합니다. 기본값은 5일입니다.

c) **OK(**확인**)**를 클릭하여 **Add TACACS Provider(TACACS** 제공자 추가**)** 대화 상자를 닫습니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

TACACS+ 제공자 삭제

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **AAA**를 선택합니다.

단계 **2** **TACACS** 탭을 클릭합니다.

단계 **3** **TACACS Providers(TACACS** 제공자**)** 영역에서 삭제할 TACACS+ 제공자에 해당하는 테이블의 행에 있는 **Delete(**삭제**)** 아이콘을 클릭합니다.

Syslog 구성

시스템 로깅은 디바이스의 메시지를 syslog 데몬을 실행 중인 서버로 수집하는 방식입니다. 중앙 syslog 서버에 로깅하면 로그와 경고를 종합하는 데 도움이 됩니다. syslog 서비스는 메시지를 수신하고 파일로 저장하거나 간단한 구성 파일에 따라 인쇄합니다. 이 로깅 양식을 통해 로그를 안전하게 장기 보관할 수 있습니다. 로그는 일상적인 문제 해결과 인시던트 처리에 모두 유용합니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **Syslog**를 선택합니다.

단계 **2** 로컬 대상을 구성합니다.

a) **Local Destinations(**로컬 대상**)** 탭을 클릭합니다.

b) **Local Destinations(**로컬 대상**)** 탭에서 다음 필드를 입력합니다.

이름	설명
Console(콘솔**)** 섹션	
Admin State(관리 상태**)** 필드	섀시가 콘솔에 syslog 메시지를 표시하는지 여부. 콘솔에 syslog 메시지를 표시하고 로그에 추가하려는 경우 **Enable(**활성화**)** 확인란을 선택합니다. **Enable(**활성화**)** 확인란이 선택되지 않은 경우, syslog 메시지는 로그에 추가되지만 콘솔에 표시되지 않습니다.
Level(레벨**)** 필드	**Console - Admin State(**콘솔 - 관리 상태**)**의 **Enable(**활성화**)** 확인란을 선택한 경우, 콘솔에 표시할 가장 낮은 메시지 수준을 선택합니다. 섀시가 콘솔에 해당 레벨 이상의 메시지를 표시합니다. 다음 중 하나일 수 있습니다. • **Emergencies** • **Alerts** • **Critical**
Monitor(모니터**)** 섹션	
Admin State(관리 상태**)** 필드	섀시가 모니터에 syslog 메시지를 표시하는지 여부. 모니터에 syslog 메시지를 표시하고 로그에 추가하려는 경우 **Enable(**활성화**)** 확인란을 선택합니다. **Enable(**활성화**)** 확인란이 선택되지 않은 경우, syslog 메시지는 로그에 추가되지만 모니터에 표시되지 않습니다.

이름	설명
Level(수준**)** 드롭다운 목록	**Monitor - Admin State(**모니터 - 관리 상태**)**의 **Enable(**활성화**)** 확인란을 선택한 경우, 모니터에 표시할 가장 낮은 메시지 수준을 선택합니다. Firepower 섀시는 모니터에 해당 수준 이상의 메시지를 표시합니다. 다음 중 하나일 수 있습니다. • **Emergencies** • **Alerts** • **Critical** • **Errors** • **Warnings** • **Notifications** • **Information(**정보**)** • **Debugging**

c) **Save(**저장**)**를 클릭합니다.

단계 **3** 원격 대상을 구성합니다.

a) **Remote Destination(**원격 대상**)** 탭을 클릭합니다.

b) **Remote Destination(**원격 대상**)** 탭에서, 섀시에서 생성된 메시지를 저장할 수 있는 최대 3개의 외부 로그에 대해 다음 필드를 입력합니다.

원격 대상에 syslog 메시지를 전송하여 외부 syslog 서버의 사용 가능한 디스크 공간에 따라 메시지를 보관할 수 있으며, 저장한 후에 로그 데이터를 조작할 수 있습니다. 예를 들어 특정 유형의 syslog 메시지가 기록될 때 실행할 작업을 지정하고, 로그에서 데이터를 추출하고 보고를 위해 기록을 다른 파일에 저장하거나, 사이트별 스크립트를 사용하여 통계를 추적할 수 있습니다.

이름	설명
Admin State(관리 상태**)** 필드	원격 로그 파일에 syslog 메시지를 저장하려는 경우 **Enable(**활성화**)** 확인란을 선택합니다.

이름	설명
Level(수준**)** 드롭다운 목록	시스템에서 저장할 가장 낮은 메시지 수준을 선택합니다. 시스템은 원격 파일에 해당 수준 이상의 메시지를 저장합니다. 다음 중 하나일 수 있습니다. • **Emergencies** • **Alerts** • **Critical** • **Errors** • **Warnings** • **Notifications** • **Information(**정보**)** • **Debugging**
Hostname/IP Address(호스트 이름**/IP** 주소**)** 필드	원격 로그 파일이 있는 호스트 이름 또는 IP 주소입니다. 참고 IP 주소 대신 호스트 이름을 사용하는 경우 DNS 서버를 구성해야 합니다.
Facility(기능**)** 드롭다운 목록	파일 메시지의 기반으로 사용할 syslog 서버에 대한 시스템 로그를 선택합니다. 다음 중 하나일 수 있습니다. • **Local0** • **Local1** • **Local2** • **Local3** • **Local4** • **Local5** • **Local6** • **Local7**

c) **Save(**저장**)**를 클릭합니다.

단계 **4** 로컬 소스를 구성합니다.

a) **Local Sources(**로컬 소스**)** 탭을 클릭합니다.

b) **Local Sources(**로컬 소스**)** 탭에서 다음 필드를 입력합니다.

이름	설명
Faults Admin State(결함 관리 상태**)** 필드	시스템 결함 로깅의 활성화 여부. **Enable(**활성화**)** 체크 박스를 선택한 경우, 섀시 로그가 모든 시스템 오류를 로깅합니다.
Audits Admin State(감사 관리 상태**)** 필드	감사 로깅의 활성화 여부. **Enable(**활성화**)** 체크 박스를 선택한 경우 섀시가 모든 감사 로그 이벤트를 로깅합니다.
Events Admin State(이벤트 관리 상태**)** 필드	시스템 이벤트 로깅의 활성화 여부. **Enable(**활성화**)** 체크 박스를 선택한 경우, 섀시가 모든 시스템 이벤트를 로깅합니다.

c) **Save(**저장**)**를 클릭합니다.

DNS 서버 구성

시스템에서 호스트의 IP 주소를 확인해야 하는 경우 DNS 서버를 지정해야 합니다. 예를 들어 DNS 서버를 구성하지 않으면 섀시에서 설정을 구성할 때 www.cisco.com 과 같은 이름을 사용할 수 없습니다. IPv4 또는 IPv6 주소 중 하나로 서버의 IP 주소를 사용해야 합니다. 최대 4개까지 DNS 서버를 구성할 수 있습니다.

참고 여러 DNS 서버를 구성할 경우 임의의 순서로만 서버를 검색합니다. 로컬 관리 명령에 DNS 서버 조회가 필요한 경우, 임의 순서로 DNS 서버 3개만 검색할 수 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **DNS**를 선택합니다.

단계 **2** **Enable DNS Server(DNS** 서버 활성화**)** 체크 박스를 선택합니다.

단계 **3** 추가하려는 각 DNS 서버에 대해 최대 4개까지 **DNS Server(DNS** 서버**)** 필드에 DNS 서버의 IP 주소를 입력하고 **Add(**추가**)**를 클릭합니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

FIPS 모드 활성화

Firepower 4100/9300 섀시에서 FIPS 모드를 활성화하려면 다음 단계를 수행하십시오.

프로시저

단계 **1** 관리자 사용자로 Firepower 4100/9300 섀시에 로그인합니다.

단계 **2** **Platform Settings**를 선택하여 Platform Settings(플랫폼 설정) 창을 엽니다.

단계 **3** **FIPS/CC mode**를 선택하여 FIPS and Common Criteria(FIPS 및 Common Criteria) 창을 엽니다.

단계 **4** FIPS에 대한 **Enable** 체크 박스를 선택합니다.

단계 **5** **Save** 를 클릭하여 구성을 저장합니다.

단계 **6** 프롬프트에 따라 시스템을 리부팅합니다.

FIPS 모드가 활성화되면, 허용되는 키 크기 및 알고리즘이 제한됩니다. MIO는 암호화 요구에 CiscoSSL 및 FOM(FIPS Object Module)을 사용합니다. 이를 통해 ASA의 독점 암호화 라이브러리 구현 및 HW 가속에 비해 FIPS 검증이 더 쉬워집니다.

다음에 수행할 작업

FXOS 릴리스 2.0.1 이전에는, 디바이스의 최초 설정 중 생성된 SSH 호스트 키가 1024비트로 하드 코딩되었습니다. FIPS 및 Common Criteria 인증 요구 사항을 충족하려면 이러한 과거의 호스트 키를 삭제하고 Generate theSSH Host Key(SSH 호스트 키 생성)에 설명된 절차를 사용하여 새 호스트 키를 생성해야 합니다. 이 추가 단계를 수행하지 않으면, FIPS 모드가 활성화되어 디바이스가 리부팅된 후 SSH를 사용하여 Supervisor에 연결할 수 없습니다. FXOS 2.0.1 이상을 사용하여 초기 설정을 수행한 경우 새 호스트 키를 생성할 필요가 없습니다.

Common Criteria 모드 활성화

Firepower 4100/9300 섀시에서 Common Criteria 모드를 활성화하려면 다음 단계를 수행하십시오.

프로시저

단계 **1** 관리자 사용자로 Firepower 4100/9300 섀시에 로그인합니다.

단계 **2** **Platform Settings**를 선택하여 Platform Settings(플랫폼 설정) 창을 엽니다.

단계 **3** **FIPS/CC mode**를 선택하여 FIPS and Common Criteria(FIPS 및 Common Criteria) 창을 엽니다.

단계 **4** Common Criteria에 대한 **Enable** 확인란을 선택합니다.

단계 **5** **Save** 를 클릭하여 구성을 저장합니다.

단계 **6** 프롬프트에 따라 시스템을 리부팅합니다.

Common Criteria는 컴퓨터 보안에 대한 국제 표준입니다. CC는 인증서, 감사, 로깅, 비밀번호, TLS, SSH 등에 중점을 둡니다. 기본적으로 FIPS 규정 준수를 가정합니다. FIPS와 마찬가지로 Cisco는 NIST 공인 랩 벤더와 계약을 체결하여 테스트를 수행하고 NIAP에 제출합니다.

CC 모드가 활성화되면 지원해야 하는 알고리즘, 암호 그룹 및 기능의 목록이 제한됩니다. MIO는 NDcPP(Network Device Collaborative Protection Profile)를 기준으로 평가됩니다. CiscoSSL은 대부분의 CC 규정 준수 가이드에서 다루는 요구사항의 일부만 시행할 수 있습니다.

다음에 수행할 작업

FXOS 릴리스 2.0.1 이전에는, 디바이스의 최초 설정 중 생성된 SSH 호스트 키가 1024비트로 하드 코딩되었습니다. FIPS 및 Common Criteria 인증 요구사항을 충족하려면 이러한 과거의 호스트 키를 삭제하고 Generate theSSH Host Key(SSH 호스트 키 생성)에 설명된 절차를 사용하여 새 호스트 키를 생성해야 합니다. 이 추가 단계를 수행하지 않으면, Common Criteria 모드가 활성화되어 디바이스가 리부팅된 후 SSH를 사용하여 Supervisor에 연결할 수 없습니다. FXOS 2.0.1 이상을 사용하여 초기 설정을 수행한 경우 새 호스트 키를 생성할 필요가 없습니다.

IP 액세스 목록 구성

기본적으로 Firepower 4100/9300 섀시는 로컬 웹 서버에 대한 모든 액세스를 거부합니다. 각 IP 블록에 대해 허용된 서비스 목록으로 IP 액세스 목록을 구성해야 합니다.

IP 액세스 목록은 다음 프로토콜을 지원합니다.

- HTTPS
- SNMP
- SSH

IP 주소(v4 또는 v6) 각 블록에서 각 디바이스에 대해 최대 100개의 서로 다른 서브넷을 구성할 수 있습니다. 서브넷 0과 접두사 0은 서비스에 대한 무제한 액세스를 허용합니다.

프로시저

단계 **1** 관리자 사용자로 Firepower 4100/9300 섀시에 로그인합니다.

단계 **2** **Platform Settings**를 선택하여 Platform Settings(플랫폼 설정) 페이지를 엽니다.

단계 **3** **Access List**를 선택하여 Access List(액세스 목록) 영역을 엽니다.

단계 **4** 이 영역에서 IP 액세스 목록에 나열된 IPv4 및 IPv6 주소를 보고 추가하고 삭제할 수 있습니다.

IPv4 블록을 추가하려면 유효한 IPv4 IP 주소 및 [0-32] 길이의 접두사를 입력하고 프로토콜을 선택해야 합니다.

IPv6 블록을 추가하려면 유효한 IPv6 IP 주소 및 [0-128] 길이의 접두사를 입력하고 프로토콜을 선택해야 합니다.

컨테이너 인스턴스 인터페이스에 대해 **MAC** 풀 접두사 추가 및 **MAC** 주소 확인

FXOS 섀시는 컨테이너 인스턴스 인터페이스용 MAC 주소를 자동으로 생성하며 각 인스턴스의 공유 인터페이스가 고유한 MAC 주소를 사용하도록 보장합니다. FXOS 섀시는 다음 형식을 사용하여 MAC 주소를 생성합니다.

A2*xx.yyzz.zzzz*

여기서 *xx.yy*는 사용자 정의 접두사 또는 시스템 정의 접두사이고 *zz.zzzz*는 섀시에서 생성하는 내부 카운터입니다. 시스템 정의 접두사는 IDPROM에 프로그래밍되는 번인된 MAC 주소 풀의 첫 번째 MAC 주소의 하위 2바이트와 일치합니다. MAC 주소 풀을 확인하려면 **connect fxos**, **show module**을 차례로 사용합니다. 예를 들어 모듈 1에 대해 표시되는 MAC 주소 범위가 b0aa.772f.f0b0~b0aa.772f.f0bf이면 시스템 접두사는 f0b0입니다.

자세한 내용은 컨테이너 인스턴스 인터페이스용 자동 MAC 주소를 참조하십시오.

이 절차에서는 MAC 주소를 확인하고 필요에 따라 생성에 사용되는 접두사를 정의하는 방법을 설명합니다.

참고 논리적 디바이스를 구축한 후에 MAC 주소 접두사를 변경하는 경우 트래픽 중단이 발생할 수 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **MAC Pool(MAC** 풀**)**을 선택합니다.

이 페이지에는 생성된 MAC 주소와 해당 MAC 주소를 사용하는 컨테이너 인스턴스 및 인터페이스가 표시됩니다.

단계 **2** (선택 사항) MAC 주소 생성에 사용되는 MAC 주소 접두사를 추가합니다.

a) **Add Prefix(**접두사 추가**)**를 클릭합니다.

Set the Prefix for the MAC Pool(MAC 풀에 대한 접두사 설정**)** 대화 상자가 나타납니다.

a) 1~65535 사이의 10진수 값을 입력합니다. 이 접두사가 4자리 16진수로 변환되어 MAC 주소의 일부로 사용됩니다.

접두사가 어떻게 사용되는지 예를 들어 설명하자면, 접두사를 77로 설정하는 경우 섀시에서는 77을 16진수 값 004D(*yyxx*)로 변환합니다. 접두사가 MAC 주소에서 쓰일 때는 섀시 기본 형식에 부합하도록 역전됩니다(*xxyy*).

A2**4D.00***zz.zzzz*

접두사가 1009 (03F1)일 때 MAC 주소는 다음과 같습니다.

A2**F1.03***zz.zzzz*

b) **OK(**확인**)**를 클릭합니다.

이 접두사를 사용하는 새 MAC 주소가 생성되어 할당됩니다. 현재 접두사와 생성된 16진수 값이 테이블 위에 표시됩니다.

컨테이너 인스턴스에 대한 리소스 프로파일 추가

컨테이너 인스턴스당 리소스 사용량을 지정하려면 리소스 프로필을 하나 이상 생성합니다. 논리적 디바이스/애플리케이션 인스턴스를 구축할 때 사용할 리소스 프로필을 지정합니다. 리소스 프로파일은 CPU 코어 수를 설정합니다. RAM은 코어 수에 따라 동적으로 할당되며 디스크 공간은 인스턴스당 40GB로 설정됩니다.

- 최소 코어 수는 6입니다.

참고 코어 수가 적은 인스턴스는 코어 수가 더 많은 CPU 사용률보다 CPU 사용률이 상대적으로 높아질 수 있습니다. 코어 수가 적은 인스턴스는 트래픽 로드 변경에 더욱 민감합니다. 트래픽 삭제를 경험하는 경우 더 많은 코어를 할당해 보십시오.

- 코어는 최대값까지 짝수(6, 8, 10, 12, 14 등)로 할당할 수 있습니다.
- 사용 가능한 코어의 최대 수는 보안 모듈/섀시 모델에 따라 달라집니다. 컨테이너 인스턴스의 요구 사항 및 사전 요구 사항 섹션을 참조하십시오.

섀시에는 최소 코어 수가 포함된 "Default-Small"이라는 기본 리소스 프로필이 있습니다. 이 프로필의 정의를 변경할 수 있으며 해당 프로필을 사용하지 않으면 삭제할 수도 있습니다. 이 프로필은 섀시를 다시 로드할 때 생성되며, 시스템에 다른 프로필은 없습니다.

리소스 프로파일이 현재 사용 중이라면 해당 설정을 변경할 수 없습니다. 해당 프로파일을 사용하는 인스턴스를 비활성화하고 리소스 프로파일을 변경한 후에 마지막으로 인스턴스를 다시 활성화해야 합니다. 설정된 고가용성 쌍 또는 클러스터에서 인스턴스 크기를 조정하는 경우에는 최대한 빠른 시간 내에 모든 멤버를 같은 크기로 설정해야 합니다.

FTD 인스턴스를 FMC에 추가한 후 리소스 프로파일 설정을 변경하는 경우 FMC **Devices(**디바이스**)** > **Device Management(**디바이스 관리**)** > **Device(**디바이스**)** > **System(**시스템**)** > **Inventory(**재고 목록**)** 대화 상자에서 재고 목록을 업데이트합니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **Resource Profiles(**리소스 프로필**)**를 선택한 다음 **Add(**추가**)**를 클릭합니다.

Add Resource Profile(리소스 프로필 추가**)** 대화 상자가 나타납니다.

단계 **2** 다음 파라미터를 설정합니다.

- **Name(**이름**)** - 1~64자 사이의 프로필 이름을 설정합니다. 프로필을 추가한 후에는 이 프로필 이름을 변경할 수 없습니다.
- **Description(**설명**)** - 프로필에 대한 설명(최대 510자)을 설정합니다.
- **Number of Cores(**코어 수**)** - 섀시에 따라 프로필의 코어 수를 6~최대값 사이의 짝수로 설정합니다.

단계 **3** **OK(**확인**)**를 클릭합니다.

네트워크 제어 정책 구성

Cisco 이외 디바이스의 검색을 허용하기 위해 FXOS에서는 IEEE 802.1ab 표준에 정의된 벤더 중립적인 디바이스 검색 프로토콜인 *LLDP(Link Layer Discovery Protocol)*를 지원합니다. LLDP를 통해 네트워크 디바이스에서 네트워크의 다른 디바이스에 자신에 관한 정보를 광고할 수 있습니다. 이 프로토콜은 데이터 링크 레이어를 통해 실행되므로 서로 다른 네트워크 레이어 프로토콜을 실행하는 두 시스템에서 서로에 관한 정보를 얻을 수 있습니다.

LLDP는 디바이스와 해당 인터페이스의 기능 및 현재 상태에 관한 정보를 전송하는 단방향 프로토콜입니다. LLDP 디바이스는 다른 LLDP 디바이스로부터 정보를 얻을 때만 이 프로토콜을 사용합니다.

FXOS 섀시에서 이 기능을 활성화하기 위해 LLDP 전송 및 수신 동작을 지정하는 네트워크 제어 정책을 구성할 수 있습니다. 네트워크 제어 정책을 생성한 후에는 인터페이스에 할당해야 합니다. 고정 포트, EPM 포트, 포트 채널 및 breakout 포트를 비롯한 전면 인터페이스에서 LLDP를 활성화할 수 있습니다.

참고

- LLDP는 전용 관리 포트에서 구성할 수 없습니다.
- 블레이드에 연결되는 내부 백플레인 포트에서는 기본적으로 LLDP가 활성화되어 있으며, 비활성화를 위한 옵션은 없습니다. 다른 모든 포트에서는 기본적으로 LLDP가 비활성화되어 있습니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **Network Control Policy(**네트워크 제어 정책**)**를 선택합니다.

단계 **2** **Add(**추가**)**를 클릭합니다.

단계 **3** Network Control Policy(네트워크 제어 정책) 대화 상자에서 다음 필드를 수정합니다.

이름	설명
Name(이름**)** 필드	네트워크 제어 정책에 대한 고유한 이름입니다.
LLDP recieve(LLDP 수신**)** 확인란	LLDP 패킷을 수신하도록 FXOS를 활성화합니다.
LLDP transmit 확인란	LLDP 패킷을 전송하도록 FXOS를 활성화합니다.
설명 필드	네트워크 제어 정책에 대한 설명입니다.

단계 **4** **Save(**저장**)**를 클릭합니다. 네트워크 제어 정책을 생성한 후에는 인터페이스에 할당해야 합니다. 네트워크 제어 정책을 사용하여 인터페이스를 수정하고 구성하는 단계는 실제 인터페이스 구성의 내용을 참조하십시오.

섀시 URL 구성

FMC에서 직접 FTD 인스턴스를 위해 Firepower Chassis Manager를 쉽게 열 수 있도록 관리 URL을 지정할 수 있습니다. 섀시 관리 URL을 지정하지 않으면 섀시 이름이 대신 사용됩니다.

FTD 인스턴스를 FMC에 추가한 후 섀시 URL 설정을 변경하는 경우 **Devices(**디바이스**)** > **Device Management(**디바이스 관리**)** > **Device(**디바이스**)** > **System(**시스템**)** > **Inventory(**인벤토리**)** 대화 상자에서 각 유닛의 인벤토리를 업데이트합니다.

프로시저

단계 **1** **Platform Settings(**플랫폼 설정**)** > **Chassis URL(**섀시 **URL)**을 선택합니다.

단계 **2** 다음 파라미터를 설정합니다.

- **Chassis Name(**섀시 이름**)** - 1~60자 사이의 섀시 이름을 설정합니다.
- **Chassis URL(**섀시 **URL)** - FMC가 Firepower Chassis Manager 내에서 FTD 인스턴스에 연결할 때 사용해야 하는 URL을 설정합니다. URL은 https://로 시작해야 합니다. 섀시 관리 URL을 지정하지 않으면 섀시 이름이 대신 사용됩니다.

단계 **3** **Update(**업데이트**)**를 클릭합니다.