**Using the NETCONF Protocol over Secure Shell (SSH)**

**RFC 6242: Using the NETCONF Protocol over Secure Shell (SSH) 개요**

이 문서는 \*\*NETCONF(Network Configuration Protocol)\*\*을 **SSH(Secure Shell) 세션** 내에서 \*\*서브시스템(subsystem)\*\*으로 호출하고 실행하는 방법을 설명합니다.  
**RFC 4742**를 대체하는 표준 문서이며, \*\*IETF(Internet Engineering Task Force)\*\*에서 공식 승인되었습니다.

**🔹 핵심 내용**

1️ **NETCONF와 SSH 연동**

* NETCONF를 SSH 세션 내에서 서브시스템으로 실행하는 방법을 정의
* 보안성과 신뢰성을 확보하기 위해 **SSH 프로토콜을 전송 계층**으로 사용

2️ **RFC 4742를 대체**

* 이전 **RFC 4742**에서 정의된 NETCONF-SSH 연동 방식의 문제점을 개선
* 최신 보안 및 운영 요구 사항을 반영

3️ **IETF 표준 승인**

* **IETF 커뮤니티의 합의(consensus)를 통해 공식 표준으로 승인**됨
* \*\*IESG(Internet Engineering Steering Group)\*\*의 검토 및 공개 리뷰를 거쳐 출판

4️ **법적 고지 및 저작권 보호**

* 문서의 모든 내용은 **IETF Trust** 및 **저자**의 저작권 보호를 받음
* **BCP 78** 및 **IETF Trust의 법적 조항**에 따라 문서를 사용할 수 있음
* 문서에서 추출된 코드 구성 요소는 **Simplified BSD License** 하에 제공됨

**📌 결론**

**RFC 6242**는 NETCONF를 SSH 환경에서 안전하게 실행하기 위한 표준을 정의하는 문서입니다.  
이를 통해 **네트워크 장비 및 서비스의 원격 관리**가 보다 **안전하고 효율적으로 수행**될 수 있습니다.

# 목차

[목차 2](#_Toc190161780)

[1. Introduction 3](#_Toc190161781)

[2. Requirements Terminology 4](#_Toc190161782)

[3. Starting NETCONF over SSH 5](#_Toc190161783)

[3.1 Capabilities Exchange 5](#_Toc190161784)

[4. Using NETCONF over SSH 7](#_Toc190161785)

[4.1 Framing Protocol 7](#_Toc190161786)

[4.2 Chunked Framing Mechanism 7](#_Toc190161787)

[4.3 End-of-Message Framing Mechanism 8](#_Toc190161788)

[5. Exiting the NETCONF Subsystem 10](#_Toc190161789)

[6. Security Considerations 11](#_Toc190161790)

[7. IANA Considerations 12](#_Toc190161791)

[8. Acknowledgements 13](#_Toc190161792)

[9. References 13](#_Toc190161793)

[9.1 Normative References 13](#_Toc190161794)

[9.2 Informative References 13](#_Toc190161795)

[Appendix A. Changes from RFC 4742 14](#_Toc190161796)

# 1. Introduction

\*\*NETCONF(Network Configuration Protocol)\*\*는 **XML 기반**의 네트워크 장비 구성 관리 프로토콜입니다.  
NETCONF는 **세션 계층(session-layer) 및 전송 계층(transport-layer)과 독립적**으로 설계되어, 다양한 전송 프로토콜과 연동될 수 있습니다.

이 문서는 NETCONF를 **SSH(Secure Shell) 세션 내에서 실행**하는 방법을 정의합니다.  
이를 위해 \*\*SSH 연결 프로토콜([RFC 4254](#_13.1_Normative_References))\*\*과 \*\*SSH 전송 프로토콜([RFC 4253](#_13.1_Normative_References))\*\*을 활용합니다.  
즉, **SSH 세션을 통해 NETCONF를 안전하게 실행**할 수 있도록 매핑(mapping)하는 방식입니다.

**🔹 핵심 내용**

1️ **NETCONF + SSH 연동**

* NETCONF를 **SSH 세션 내에서 실행**하는 방법 정의
* \*\*SSH 연결 프로토콜(RFC 4254)\*\*과 \*\*SSH 전송 프로토콜(RFC 4253)\*\*을 사용

2️ **보안성 강화**

* SSH 기반 전송을 사용하여 **암호화 및 인증** 지원
* 안전한 원격 네트워크 구성 관리 가능

3️ **메시지 전송의 유연성**

* 문서에서 NETCONF 메시지 전송 예제를 제공
* 하지만 특정 메시지 유형에 제한되지 않으며, **모든 NETCONF 메시지 전송 가능**

**📌 결론**

이 문서는 **SSH를 사용하여 NETCONF 프로토콜을 안전하게 실행하는 방법을 표준화**합니다.  
이를 통해 **네트워크 장비의 원격 관리 및 자동화가 더욱 안전하고 효율적으로 수행**될 수 있습니다.

# 2. Requirements Terminology

이 문서에서 사용되는 **"MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", "OPTIONAL"** 등의 용어는 **RFC 2119**에서 정의된 대로 해석해야 합니다.

🔹 RFC 2119의 의미 정리

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 3. Starting NETCONF over SSH

NETCONF을 SSH를 통해 실행하려면 다음과 같은 절차를 따릅니다.

1️ **SSH 연결 설정**

* SSH 클라이언트가 **SSH 전송 프로토콜(**[**RFC 4253**](#_13.1_Normative_References)**)**을 사용하여 SSH 연결을 설정합니다.
* 메시지 무결성과 암호화를 위해 키 교환을 수행합니다.

2️ **사용자 인증**

* SSH 클라이언트는 **SSH 인증 프로토콜(**[**RFC 4252**](#_13.1_Normative_References)**)**을 사용하여 **"ssh-userauth"** 서비스를 호출합니다.
* 인증이 성공하면 **"ssh-connection"** 서비스를 호출하여 SSH 세션을 설정합니다.

3️ **NETCONF 사용자명 확인**

* SSH가 제공하는 사용자명은 NETCONF 프로토콜에서 그대로 사용됩니다.
* 만약 **NETCONF에서 허용되지 않는 형식의 사용자명**(예: XML로 표현할 수 없는 문자 포함)이 들어오면 **SSH 세션을 종료(MUST)** 해야 합니다.

4️ **SSH 세션 및 NETCONF 서브시스템 실행**

* SSH 클라이언트는 **"session"** 타입의 채널을 열어 SSH 세션을 생성합니다.
* 이후 NETCONF 클라이언트는 **"netconf"** 서브시스템을 실행합니다.
* 이 과정은 **SSH v2(SSHv2)** 에서만 지원되며 **SSHv1에서는 사용할 수 없습니다.**
* NETCONF를 SSH 서브시스템으로 실행하면 **불필요한 시스템 메시지 처리 문제** 없이 바로 실행할 수 있습니다.

5️ **기본 포트 설정 (TCP 830)**

* NETCONF 서버는 **IANA에서 할당한 TCP 포트 830**을 기본 포트로 사용해야 합니다 (MUST).
* 다만, 서버는 **다른 포트에서 NETCONF SSH 서브시스템을 허용하도록 설정할 수도 있습니다(SHOULD)**.

✅ **NETCONF SSH 실행 예시**



* -s 옵션은 "netconf" 서브시스템을 호출하는 역할을 합니다.

## 3.1 Capabilities Exchange

NETCONF 세션이 시작되면 **서버와 클라이언트는 서로 지원하는 기능(캡처빌리티, Capabilities)을 교환**합니다.

🔹 **서버 → 클라이언트**

* NETCONF 서버는 **<hello> 메시지**를 XML 문서로 전송하여 자신의 기능을 알립니다.
* 예제:  
  텍스트, 스크린샷, 폰트이(가) 표시된 사진

  AI가 생성한 콘텐츠는 부정확할 수 있습니다.

🔹 **클라이언트 → 서버**

* 클라이언트도 자신의 **<hello> 메시지**를 서버에 전송하여 지원하는 기능을 알립니다.
* 예제:  
  텍스트, 스크린샷, 폰트이(가) 표시된 사진

  AI가 생성한 콘텐츠는 부정확할 수 있습니다.

⚠ **참고 사항**

* 위 예제에서는 서버가 먼저 **<hello>** 메시지를 보내고, 이후 클라이언트가 응답하는 순서로 보이지만,  
  실제 동작에서는 **서버와 클라이언트가 거의 동시에 <hello> 메시지를 보낼 수도 있습니다.**
* **<session-id>**: 서버가 할당한 NETCONF 세션의 고유 ID입니다.

**📌 정리**

✅ **NETCONF는 SSH를 통해 안전하게 실행할 수 있으며, 기본적으로 TCP 830 포트를 사용**  
✅ **NETCONF SSH 실행 시, 반드시 "netconf" 서브시스템을 호출해야 함**  
✅ **NETCONF 세션이 시작되면 <hello> 메시지를 통해 서버와 클라이언트의 기능을 교환**  
✅ **NETCONF에서 허용되지 않는 사용자명이 SSH에서 전달되면 세션을 종료해야 함**

# 4. Using NETCONF over SSH

NETCONF over SSH는 **NETCONF 클라이언트와 서버 간에 XML 문서를 교환하는 방식으로 동작**합니다.

* 클라이언트 → **<rpc> 요청**을 서버로 전송
* 서버 → **<rpc-reply> 응답**을 클라이언트로 반환

## 4.1 Framing Protocol

이전 버전에서는 **]]>]]>** 문자열을 **메시지 구분자**로 사용했지만,

* XML 속성, 주석, 처리 명령어 등에 **해당 문자열이 포함될 가능성이 있어 충돌 위험**이 존재
* 기존 구현과의 **호환성을 유지하면서도 문제를 해결하기 위해 새로운 프레이밍 방식이 도입**됨

**프레이밍 방식 선택 기준**

텍스트, 폰트, 라인, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

* **<hello> 메시지 뒤에는 ]]>]]>을 반드시 포함해야 함**
* 이후 **서버와 클라이언트가 모두 :base:1.1을 지원하면 Chunked 프레이밍 적용**

**그렇지 않으면 기존 ]]>]]> 기반 프레이밍 적용**

## 4.2 Chunked Framing Mechanism

이 방식은 **메시지를 여러 개의 청크(Chunk)로 나누어 전송하는 방법**입니다.

**📌 청크 메시지의 문법 (ABNF 표현)**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**📌 메시지 예시**

예제: <rpc> 메시지를 청크로 나누어 전송하는 경우

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

💡 **청크 단위로 변환된 메시지 (가독성을 위해 \n은 LineFeed로 표기)**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

💡 청크 구조 해석

텍스트, 폰트, 라인, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**⚠ Chunked Framing 오류 처리**

* **청크 크기 오류 발생** (예: 0으로 시작하는 값, 4,294,967,296 이상의 값)
* **디코딩 중 예기치 않은 에러 발생**  
  ➡ **NETCONF 세션을 종료해야 함 (MUST)**  
  ➡ **버퍼 오버플로우 방지를 위한 보안 고려 필요**

## 4.3 End-of-Message Framing Mechanism

**기존 ]]>]]> 기반 프레이밍**

✅ **이전 버전과의 호환성을 위해 제공됨**  
✅ :base:1.0을 지원하는 경우에만 사용

**메시지 구조**

* **각 XML 문서 끝에 ]]>]]> 구분자 추가**
* **SSH Transport 계층이 ]]>]]>을 감지하여 메시지를 구분**

**📌 예제 (구성 정보 조회 요청)**

**📌 클라이언트 → 서버 요청**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

📌 서버 → 클라이언트 응답

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

📌 정리

텍스트, 폰트, 번호, 라인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**📌 핵심 개념 요약**

✅ **NETCONF over SSH는 XML 기반으로 작동하며, 클라이언트와 서버가 <rpc> 및 <rpc-reply> 문서를 주고받음**  
✅ **기본적으로 ]]>]]> 메시지 구분자를 사용하지만, :base:1.1이 지원되면 "청크(Chunked) 프레이밍" 방식 적용**  
✅ **청크 크기 오류 또는 메시지 디코딩 오류 발생 시, 반드시 세션을 종료해야 함 (MUST)**

# 5. Exiting the NETCONF Subsystem

NETCONF 세션을 종료하려면 **<close-session>** RPC 요청을 사용합니다.

✅ **서버는 받은 순서대로 메시지를 처리하며, <close-session>이 처리되면 SSH 세션을 닫음**  
✅ **종료 요청 이후 들어온 메시지는 처리하지 않음**

**📌 예제 (청크 방식)**

**클라이언트 → 서버 요청**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

서버 → 클라이언트 응답

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

📌 **<ok/> 응답을 받은 후, 세션이 정상적으로 종료됨!** 🚀

# 6. Security Considerations

NETCONF는 **네트워크 장비의 구성 및 상태 정보를 조회 및 수정**하는 프로토콜이므로,  
접근 권한이 있는 사용자 및 시스템만 사용할 수 있도록 제한해야 합니다.

**🔒 SSH 서버 및 클라이언트 인증**

✅ **SSH 서버의 신원 검증** (클라이언트 → 서버)

* SSH 클라이언트는 **서버의 신원을 확인**한 후에만 인증 정보(비밀번호, 키 등) 및 구성 데이터를 전송해야 함

✅ **SSH 클라이언트의 신원 검증** (서버 → 클라이언트)

* SSH 서버는 **클라이언트의 정당성을 확인**한 후에만 구성 데이터 제공

❗ **알 수 없거나 예상하지 못한 SSH ID와 연결 금지**

**🔐 강력한 암호화 필요**

NETCONF 메시지에는 **민감한 정보**(예: 사용자 계정, 보안 키)가 포함될 수 있음 →  
📌 **강력한 암호화 채널**을 사용하여 데이터 보호 필요

**🌐 TCP 포트 830 및 보안 고려**

* **기본적으로 IANA가 지정한 포트(830)에서 NETCONF SSH 서브시스템 제공**
  + ✅ **방화벽에서 식별 및 필터링 가능**
  + ❗ 하지만 공격자가 쉽게 NETCONF 트래픽을 탐지할 수도 있음
* **다른 포트에서 NETCONF SSH 실행 가능하도록 설정 가능**
  + ❗ **방화벽 및 네트워크 설정을 변경하지 않으면, 외부 공격자가 접근 가능해질 위험**

**🛡 EOM(End-of-Message) 시퀀스 취약점**

기존 **RFC 4742**에서는 **"]]>]]>"** 시퀀스가 XML에서 사용되지 않는다고 가정했으나,  
🔴 **실제로는 속성, 주석 등에 포함될 수 있음 → 공격에 악용 가능**

✅ **초기 <hello> 메시지에서는 여전히 EOM 사용**  
✅ **하지만 NETCONF base:1.1을 지원하는 경우, 청크(Chunked) 프레임 방식으로 변경하여 공격 방지**

**🚀 정리**

✔ **SSH 인증 강화** → 신원 검증 후만 데이터 교환  
✔ **강력한 암호화** → 민감한 데이터 보호  
✔ **TCP 포트 830 보안 설정** → 공격자 탐지 방지  
✔ **EOM 시퀀스 취약점 해결** → 청크 프레임 방식 사용

**📌 NETCONF over SSH는 보안에 민감한 프로토콜이므로, 강력한 인증 및 암호화 적용이 필수!** 🔐

# 7. IANA Considerations

IANA(Internet Assigned Numbers Authority)는 **NETCONF over SSH의 표준 포트 및 서브시스템 이름**을 할당함.

**🛠 TCP 포트 할당**

* **RFC 4742**(이전 버전)에서 **TCP 포트 830**을 **NETCONF over SSH의 기본 포트**로 지정
* 본 문서에서도 해당 포트 사용을 유지

**🛠 SSH 서브시스템 이름 할당**

* **IANA는 "netconf"를 SSH 서브시스템 이름**으로 등록
* 이 할당은 [**RFC 4250**](#_13.1_Normative_References)(SSH 프로토콜 할당을 관리하는 RFC)에 따라 이루어짐

📌 **업데이트 사항**: IANA는 기존 할당을 유지하면서 **참조 문서를 RFC 4742 → 현재 문서로 변경**

**🔎 정리**

✔ **NETCONF over SSH 기본 포트: TCP 830**  
✔ **SSH 서브시스템 이름: "netconf"**  
✔ **IANA 할당 정보가 본 문서로 업데이트됨**

# 8. Acknowledgements

감사의 글

# 9. References

## 9.1 Normative References

[RFC 2119](https://datatracker.ietf.org/doc/html/rfc2119) Key words for use in RFCs to Indicate Requirement Levels

[RFC 4250](https://datatracker.ietf.org/doc/html/rfc4250) The Secure Shell (SSH) Protocol Assigned Numbers

[RFC 4252](https://datatracker.ietf.org/doc/html/rfc4252) The Secure Shell (SSH) Authentication Protocol

[RFC 4253](https://datatracker.ietf.org/doc/html/rfc4253) The Secure Shell (SSH) Transport Layer Protocol

[RFC 4254](https://datatracker.ietf.org/doc/html/rfc4254) The Secure Shell (SSH) Connection Protocol

[RFC 5234](https://datatracker.ietf.org/doc/html/rfc5234) Augmented BNF for Syntax Specifications: ABNF

[RFC 6241](https://datatracker.ietf.org/doc/html/rfc6241) Network Configuration Protocol (NETCONF)

## 9.2 Informative References

[RFC 2629](https://datatracker.ietf.org/doc/html/rfc2629) Writing I-Ds and RFCs using XML

# Appendix A. Changes from RFC 4742

이 문서는 **RFC 4742**와 비교하여 다음과 같은 주요 변경 사항을 포함함:

📌 **보안 강화**

* 새로운 **청크 프레이밍 메커니즘**(chunked framing mechanism) 도입하여 **EOM 프레이밍의 보안 문제 해결**.
* \*\*보안 고려 사항(Security Considerations)\*\*을 확장하고, **EOM 관련 문제**에 대한 설명 추가.

📌 **기술적 개선**

* \*\*청크 인코딩(chunked encoding)\*\*의 예시 추가 및 줄바꿈 위치 강조.
* **NETCONF 사용자명(username) 처리 관련 요구 사항**을 **RFC 6241**에 맞춰 수정.
* **"client/server" 및 "manager/agent"** 용어를 각각 \*\*"SSH client/server" 및 "NETCONF client/server"\*\*로 일관되게 변경.
* **"command" 또는 "message" 대신 "operation"** 용어를 일관되게 사용.

📌 **기타 정리**

* **RFC 4742에서 보고된 오류(errata) 반영**. (공식 오류 목록은 [RFC 에디터 웹사이트](http://www.rfc-editor.org)에서 확인 가능)