Abstract

The NETCONF configuration protocol provides mechanisms to install, manipulate, and delete the configuration of network devices. This document describes how to use TLS to secure NETCONF exchanges.

1 Introduction

The NETCONF protocol [NETCONF] defines a simple mechanism through which a network device can be managed. NETCONF is connection-
oriented, requiring a persistent connection between peers. This connection must provide reliable, sequenced data delivery, integrity and confidentiality and peers authentication. This document describes how to use TLS [TLS] to secure NETCONF connections.

1.2 Requirements language and Terminologies

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [KEYWORDS].

1.3 Terminology

This document uses the following terms:

manager
It refers to the end initiating the NETCONF connection. It issues the NETCONF RPC commands.

agent
It refers to the end replying to the manager’s commands during the NETCONF connection.

2. NETCONF over TLS

Since TLS is application protocol-independent, NETCONF can operate on top of the TLS protocol transparently. This document defines how NETCONF can be used within a Transport Layer Security (TLS) session.

2.1. Connection Initiation

The peer acting as the NETCONF manager SHOULD also act as the TLS client. It SHOULD connect to the server that passively listens for the incoming TLS connection on the IANA-to-be-assigned TCP port <TBC>. It SHOULD therefore send the TLS ClientHello to begin the TLS handshake. Once the TLS handshake has been finished, the manager and the agent MAY then send their NETCONF exchanges. In particular, the manager will send complete XML documents to the server containing <rpc> elements, and the agent will respond with complete XML documents containing <rpc-reply> elements. All these exchanges are encapsulated into TLS records of type "application data". These records are protected using the TLS material keys.

Current NETCONF messages don’t include a message’s length. This document uses consequently the same delimiter sequence defined in [NETSSH] and therefore the special character sequence, ]]>]], to delimit XML documents.
2.2. Connection Closure

Either NETCONF peer MAY stop the NETCONF connection at any time and therefore notify the other NETCONF peer that no more data on this channel will be sent and that any data received after a closure request will be ignored. This MAY happen when no data is received from a connection for a long time, where the application decides what "long" means.

TLS has the ability for secure connection closure using the Alert protocol. When the NETCONF peer processes a closure request of the NETCONF connection, it MUST send a TLS close_notify alert before closing the connection. Any data received after a closure alert is ignored.

The party that sends a close_notify MAY choose not to wait for the other party’s close_notify alert and simply close the connection, thus generating an incomplete close on the other party’s side. Once the other party gets the close_notify alert, it MUST reply with a close_notify unless it becomes aware that the connection has already been closed.

When a party has received the close_notify alert from the other party and still has pending data to send, it SHOULD send the pending data before sending the close_notify alert.

3. Endpoint Authentication and Identification

Usually, TLS uses public keys, Kerberos [TLSKERB], or preshared keys [TLSPSK] for authentication.

When public key is used for authentication, TLS supports three authentication modes: authentication of both parties, server authentication with an unauthenticated client, and total anonymity. User authentication in unauthenticated or authenticated client mode is outside the scope of this document. User authentication should be handled by either an extension of TLS (such as the TLS Inner Application Extension [IATLS]) or an authentication extension of NETCONF.

3.1. Server Identity

During the TLS negotiation, the client MUST carefully examine the certificate presented by the server to determine if it meets their expectations. Particularly, the client MUST check its understanding of the server hostname against the server’s identity as presented in the server Certificate message, in order to prevent man-in-the-middle attacks.
Matching is performed according to these rules [RFC4642]:

- The client MUST use the server hostname it used to open the connection (or the hostname specified in TLS "server_name" extension [TLSEXT]) as the value to compare against the server name as expressed in the server certificate. The client MUST NOT use any form of the server hostname derived from an insecure remote source (e.g., insecure DNS lookup). CNAME canonicalization is not done.

- If a subjectAltName extension of type dNSName is present in the certificate, it SHOULD be used as the source of the server’s identity.

- Matching is case-insensitive.

- A "*" wildcard character MAY be used as the left-most name component in the certificate. For example, *.example.com would match a.example.com, foo.example.com, etc., but would not match example.com.

- If the certificate contains multiple names (e.g., more than one dNSName field), then a match with any one of the fields is considered acceptable.

If the match fails, the client SHOULD either ask for explicit user confirmation or terminate the connection and indicate the server’s identity is suspect.

Additionally, clients MUST verify the binding between the identity of the servers to which they connect and the public keys presented by those servers. Clients SHOULD implement the algorithm in Section 6 of [PKICERT] for general certificate validation, but MAY supplement that algorithm with other validation methods that achieve equivalent levels of verification (such as comparing the server certificate against a local store of already-verified certificates and identity bindings).

If the client has external information as to the expected identity of the server, the hostname check MAY be omitted.

3.2. Client Identity

Typically, the server has no external knowledge of what the client’s identity ought to be and so checks (other than that the client has a certificate chain rooted in an appropriate CA) are not possible. If a server has such knowledge (typically from some source external to NETCONF or TLS) it SHOULD check the identity as described above.
4. Security Considerations

The security considerations described throughout [TLS] apply here as well.

5. IANA Considerations

IANA is requested to assign a TCP port number that will be the default port for NETCONF over TLS sessions as defined in this document.

IANA has assigned port <TBD> for this purpose.

6. Acknowledgment

The author would like to acknowledge Juergen Schoenwaelder for his detailed reviews of the content of the document. The author appreciates also David Harrington and Miao Fuyou for their effort on issues resolving discussion.

7. References

7.1. Normative References


Configuration Protocol over Secure Shell (SSH)", RFC 4742, December 2006.

7.2. Informative References


Author’s Addresses

Mohamad Badra
LIMOS Laboratory - UMR (6158), CNRS
France                    Email: badra@isima.fr

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an
attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).