Debug Secure Shell (SSH) auf NCS1K

Inhalt

Einleitung

In diesem Dokument werden grundlegende Verfahren zur Fehlerbehebung für Secure Shell (SSH) auf der NCS1K-Plattform beschrieben.

Voraussetzungen

In diesem Dokument wird davon ausgegangen, dass Sie mit XR-basierten Betriebssystemen auf Geräten wie dem Network Convergence System (NCS) 1002 vertraut sind.

Anforderungen

Cisco empfiehlt, dass Sie bezüglich der SSH-Verbindungsanforderungen über die folgenden Themen Bescheid wissen:

- Das relevante k9sec-Paket für das XR-Image
- SSH-Konfiguration auf Cisco Gerät vorhanden
- Erfolgreiche Schlüsselgenerierung, Schlüsselaustausch und Chiffrieraushandlung zwischen Host und Server

Verwendete Komponenten

Die Informationen in diesem Dokument basierend auf folgenden Software- und Hardware-

Versionen:

- NCS 1002 mit XR 7.3.1
- NCS 1004 mit XR 7.9.1

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Installierte Pakete überprüfen

Die Befehle show install active und show install committed das Vorhandensein des k9sec-Pakets identifizieren. Ohne dieses Paket können Sie keine Kryptografieschlüssel generieren, um eine SSH-Sitzung zu initiieren.

<#root>

```
RP/0/RP0/CPU0:NCS1002_1#
```

show install active

```
Wed Jul 19 09:31:18.977 UTC
Label : 7.3.1

Node 0/RP0/CPU0 [RP]
Boot Partition: xr_lv58
Active Packages: 4
ncs1k-xr-7.3.1 version=7.3.1 [Boot image]
ncs1k-mpls-te-rsvp-3.1.0.0-r731
ncs1k-mpls-2.1.0.0-r731
ncs1k-k9sec-3.1.0.0-r731

RP/0/RP0/CPU0:NCS1002_1#
```

show install committed

```
Wed Jul 19 09:31:37.359 UTC
Label : 7.3.1

Node 0/RP0/CPU0 [RP]
Boot Partition: xr_lv58
Committed Packages: 4
ncs1k-xr-7.3.1 version=7.3.1 [Boot image]
ncs1k-mpls-te-rsvp-3.1.0.0-r731
ncs1k-mpls-2.1.0.0-r731
ncs1k-k9sec-3.1.0.0-r731
```

Konfiguration

Zumindest das NCS1K erfordert die Konfiguration. ssh server v2 um SSH-Verbindungen zuzulassen.

Eingabe show run ssh um sicherzustellen, dass diese Konfiguration vorhanden ist:

```
<#root>

RP/0/RP0/CPU0:NCS1004_1#

show run ssh


Wed Jul 19 13:06:57.207 CDT
ssh server rate-limit 600
ssh server v2
ssh server netconf vrf default
```

Generierte Schlüssel identifizieren

Um eine SSH-Sitzung einzurichten, muss das NCS1K über einen öffentlichen kryptografischen Schlüssel verfügen. Identifizieren Sie generierte Schlüssel mithilfe von show crypto key mypubkey { dsa | ecdsa | ed25519 | rsa }. Der Standardschlüsseltyp ist rsa. Der Schlüssel wird als Hexadezimalzeichenfolge angezeigt, die aus Sicherheitsgründen hier weggelassen wird.

```
<#root>

RP/0/RP0/CPU0:NCS1002_1#

show crypto key mypubkey rsa


Wed Jul 19 10:30:09.333 UTC
Key label: the_default
Type : RSA General purpose
Size : 2048
Created : 11:59:56 UTC Tue Aug 23 2022
Data : <key>
```

Um einen Schlüssel eines bestimmten Typs zu generieren, geben Sie den Befehl crypto key generate { dsa | ecdsa | ed25519 | rsa } und einen Schlüsselmodul auswählen. Die Modulgröße variiert je nach Algorithmus.

Schlüsseltyp	Zulässige Modultypen/Kurventypen	Standardmodullänge (Bit)
DSA	512, 768, 1024	1024
ECDSA	nistp256, nistp384, nistp521	none

ed25519	256	256
RSA	512 bis 4096	2048

Überprüfen Sie, ob der Schlüssel erfolgreich generiert wurde mit show crypto key mypubkey.

Um einen vorhandenen Schlüssel zu entfernen, geben Sie den Befehl crypto key zeroize { authentication | dsa | ecdsa | ed25519 | rsa } [label]. Stellen Sie sicher, dass Sie auf andere Weise auf das Gerät zugreifen können, da die Trennung von einem Gerät ohne Verschlüsselungsschlüssel den Zugriff mit SSH blockiert.

Identifizieren von SSH-Serverfunktionen

Der Server und der Host müssen sich auf einen Schlüsselaustausch, einen Hostschlüssel und eine Verschlüsselung einigen, bevor eine SSH-Sitzung eingerichtet wird. Um die Funktionen der NCS1K-Plattform zu identifizieren, geben Sie den Befehl show ssh server.

```
<#root>

RP/0/RP0/CPU0:NCS1004_1#

show ssh server


Wed Jul 19 13:28:04.820 CDT
---------------------
SSH Server Parameters
---------------------

Current supported versions := v2
SSH port := 22
SSH vrfs := vrfname:=default(v4-acl:=, v6-acl:=)
Netconf Port := 830
Netconf Vrfs := vrfname:=default(v4-acl:=, v6-acl:=)

Algorithms
---------------
Hostkey Algorithms := x509v3-ssh-rsa,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-nistp256,rsa-sh
Key-Exchange Algorithms := ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-grou
Encryption Algorithms := aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com
Mac Algorithms := hmac-sha2-512,hmac-sha2-256,hmac-sha1

Authentication Method Supported
------------------------------------
PublicKey := Yes
Password := Yes
Keyboard-Interactive := Yes
Certificate Based := Yes

Others
------------
DSCP := 16
Ratelimit := 600
Sessionlimit := 64
```

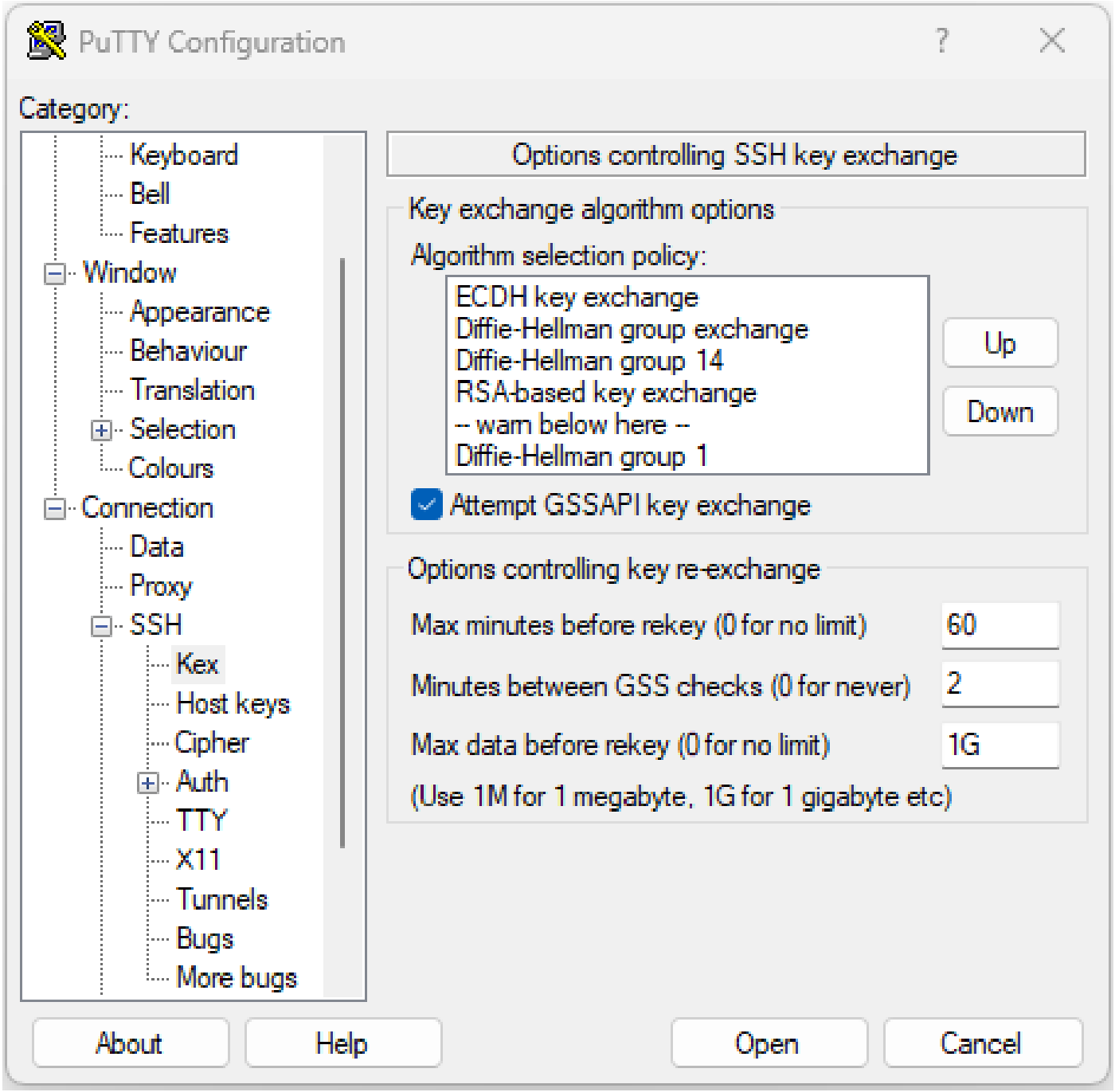
```
Rekeytime := 60
Server rekeyvolume := 1024
TCP window scale factor := 1
Backup Server := Disabled
Host Trustpoint :=
User Trustpoint :=
Port Forwarding := Disabled
Max Authentication Limit := 20
Certificate username := Common name(CN)
```

Identifizieren von Host-SSH-Funktionen

Der Host, der versucht, eine Verbindung herzustellen, muss mit mindestens einem Hostschlüssel, Schlüsselaustausch und Verschlüsselungsalgorithmus vom Server übereinstimmen, um eine SSH-Sitzung einzurichten.

PuTTY

PuTTY listet die unterstützten Schlüsselaustausch-, Host-Schlüssel- und Verschlüsselungsalgorithmen auf unter Connections > SSH. Der Host handelt die Algorithmen automatisch auf Grundlage seiner Funktionen aus. Dabei bevorzugt er den Schlüsselaustauschalgorithmus in der Reihenfolge der Benutzereinstellungen. Die Option Attempt GSSAPI key exchange ist für die Verbindung mit einem NCS1K-Gerät nicht erforderlich.



Screenshot der PuTTY SSH-Optionen

Linux

Linux-Server behalten in der Regel die unterstützten Algorithmen im /etc/ssh/ssh_config Datei. Dieses Beispiel stammt von Ubuntu Server 18.04.3.

```
Host *
# ForwardAgent no
# ForwardX11 no
# ForwardX11Trusted yes
# PasswordAuthentication yes
# HostbasedAuthentication no
# GSSAPIAuthentication no
# GSSAPIDelegateCredentials no
```

```
# GSSAPIKeyExchange no
# GSSAPITrustDNS no
# BatchMode no
# CheckHostIP yes
# AddressFamily any
# ConnectTimeout 0
# StrictHostKeyChecking ask
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# IdentityFile ~/.ssh/id_ecdsa
# IdentityFile ~/.ssh/id_ed25519
# Port 22
# Protocol 2
# Ciphers aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc
# MACs hmac-md5,hmac-sha1,umac-64@openssh.com
# EscapeChar ~
# Tunnel no
# TunnelDevice any:any
# PermitLocalCommand no
# VisualHostKey no
# ProxyCommand ssh -q -W %h:%p gateway.example.com
# RekeyLimit 1G 1h
SendEnv LANG LC_*
HashKnownHosts yes
GSSAPIAuthentication yes
```

Fehlerbehebung bei SSH-Verbindungen

Diese Befehle können helfen, Fehler bei SSH-Verbindungen zu isolieren.

Aktuelle ein- und ausgehende SSH-Sitzungen anzeigen mit show ssh session details.

<#root>

```
RP/0/RP0/CPU0:NCS1002_1#
```

show ssh session details

```
Wed Jul 19 13:08:46.147 UTC
SSH version : Cisco-2.0

id key-exchange pubkey incipher outcipher inmac outmac
------------------------------------------------------------------------------------------------
Incoming Sessions
128733 ecdh-sha2-nistp256 ssh-rsa aes256-ctr aes256-ctr hmac-sha2-256 hmac-sha2-256
128986 diffie-hellman-group14 ssh-rsa aes128-ctr aes128-ctr hmac-sha1 hmac-sha1
128988 diffie-hellman-group14 ssh-rsa aes128-ctr aes128-ctr hmac-sha1 hmac-sha1

Outgoing sessions
```

Verlaufssitzungen von SSH beinhalten fehlgeschlagene Verbindungsversuche mit dem Befehl show ssh history detail.

<#root>

RP/0/RP0/CPU0:NCS1002_1#

show ssh history details

```
Wed Jul 19 13:13:26.821 UTC
SSH version : Cisco-2.0

id key-exchange pubkey incipher outcipher inmac outmac start_time end_time
--------------------------------------------------------------------------------------------------------------
Incoming Session
128869diffie-hellman-group14-sha1ssh-rsa aes128-ctr aes128-ctr hmac-sha1 hmac-sha1 19-07-23 11:28:55 19
```

SSH-Ablaufverfolgungen liefern eine feine Detailgenauigkeit des Verbindungsprozesses mit show ssh trace all.

<#root>

RP/0/RP0/CPU0:NCS1002_1#

show ssh trace all

```
Wed Jul 19 13:15:53.701 UTC
3986 wrapping entries (57920 possible, 40896 allocated, 0 filtered, 392083 total)
Apr 29 19:13:19.438 ssh/backup-server/event 0/RP0/CPU0 t6478 [SId:=0] Respawn-count:=1, Starting SSH Se
Apr 29 19:13:19.438 ssh/backup-server/shmem 0/RP0/CPU0 t6478 [SId:=0] Shared memory does not exist duri
```

SSH-Re-Key-Werte konfigurieren

Die SSH-Konfiguration zum erneuten Schlüsselaustausch bestimmt die Zeit und die Anzahl der Bytes, bevor ein neuer Schlüsselaustausch stattfindet. Aktuelle Werte anzeigen mit show ssh rekey.

<#root>

RP/0/RP0/CPU0:NCS1004_1#

show ssh rekey

```
Wed Jul 19 15:23:06.379 CDT
SSH version : Cisco-2.0

id RekeyCount TimeToRekey(min) VolumeToRekey(MB)
------------------------------------------------------------
Incoming Session
1015      6      6.4              1024.0
1016      0      58.8             1024.0

Outgoing sessions
```

Um die Lautstärke für die erneute Eingabe festzulegen, verwenden Sie den Befehl ssh server rekey-volume [size]. Die Standardgröße für die erneute Schlüsselerstellung ist 1024 MB.

<#root>

```
RP/0/RP0/CPU0:NCS1004_1(config)#
ssh server rekey-volume 4095

RP/0/RP0/CPU0:NCS1004_1(config)#
commit
```

Legen Sie den Wert für den Zeitgeber für die erneute Schlüsselausgabe ebenfalls wie folgt fest: ssh server rekey-time [time]. Der Standardwert ist 60 Minuten.

```
RP/0/RP0/CPU0:NCS1004_1(config)# ssh server rekey-time 120
RP/0/RP0/CPU0:NCS1004_1(config)# commit
```

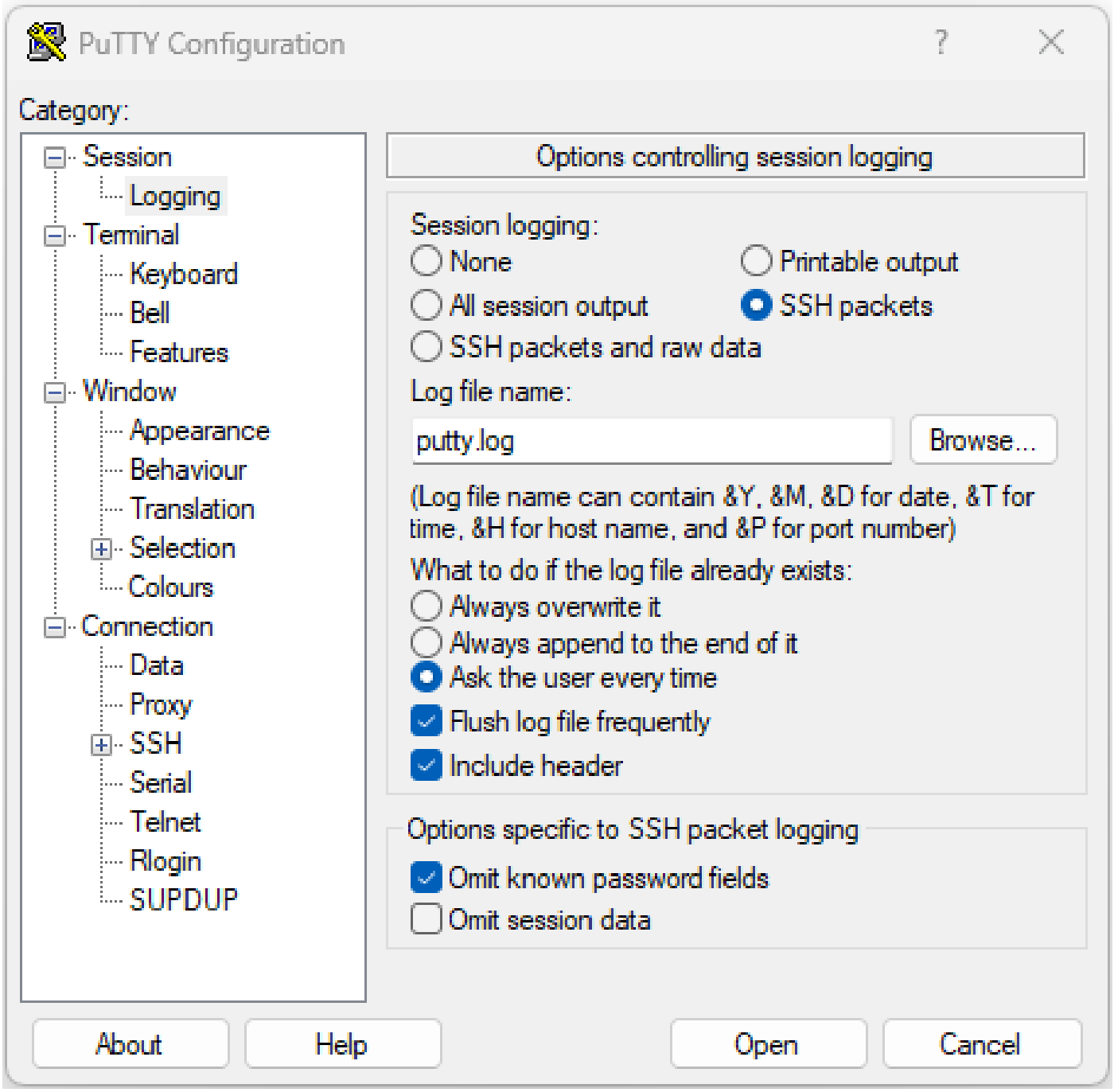
SSH-Fehlersuche

Die Fehlermeldung debug ssh server zeigt Echtzeitausgaben für aktive SSH-Sitzungen und Verbindungsversuche an. Um eine fehlerhafte Verbindung zu beheben, aktivieren Sie den Debugging-Befehl, versuchen Sie die Verbindung, und beenden Sie den Debugging-Befehl mit undebug all. Protokollieren Sie die Sitzung mithilfe von PuTTY oder einer anderen Terminalanwendung für die Analyse.

<#root>

```
RP/0/RP0/CPU0:NCS1002_1#
debug ssh server
```

PuTTY umfasst eine Funktion zum Protokollieren von SSH-Paketen unter Session > Logging.



Screenshot der PuTTY SSH-Protokollierung

Unter Linux `ssh -vv` (sehr ausführlich) enthält detaillierte Informationen zum SSH-Verbindungsprozess.

<#root>

```
ubuntu-18@admin:/$
```

```
ssh -vv admin@192.168.190.2
```

Zusätzliche Protokolle

Mehrere Show-Techniker erfassen nützliche Informationen zu SSH.

- **show tech { ncs1k | ncs1001 | ncs1004 } detail**
- **show tech crypto session**
- **show tech ssh**
- **admin show tech { ncs1k | ncs1001 | ncs1004 }-admin**

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.