

Catalyst 9000プラットフォームでのGuestShellのトラブルシューティング

内容

[はじめに](#)

[前提条件](#)

[要件](#)

[使用するコンポーネント](#)

[背景説明](#)

[ゲストシェルの概要](#)

[トラブルシューティング](#)

[コンフィギュレーション](#)

[ゲストシェルのライフサイクル](#)

[検証](#)

[リソースのリサイズ](#)

[一般的なシナリオ](#)

[DHCP 設定。](#)

[DNF Packet Managerの更新に失敗しました](#)

[アップグレード後にゲストシェルへのアクセスが失われる](#)

[IPv6アドレッシングの制限](#)

[Pythonスクリプト実行中のディスクスペースエラー](#)

[Syslogのロギング](#)

[トレースログ](#)

[関連情報](#)

はじめに

このドキュメントでは、Cat9Kスイッチのゲストシェルに関する問題をトラブルシューティングする方法について説明します。

前提条件

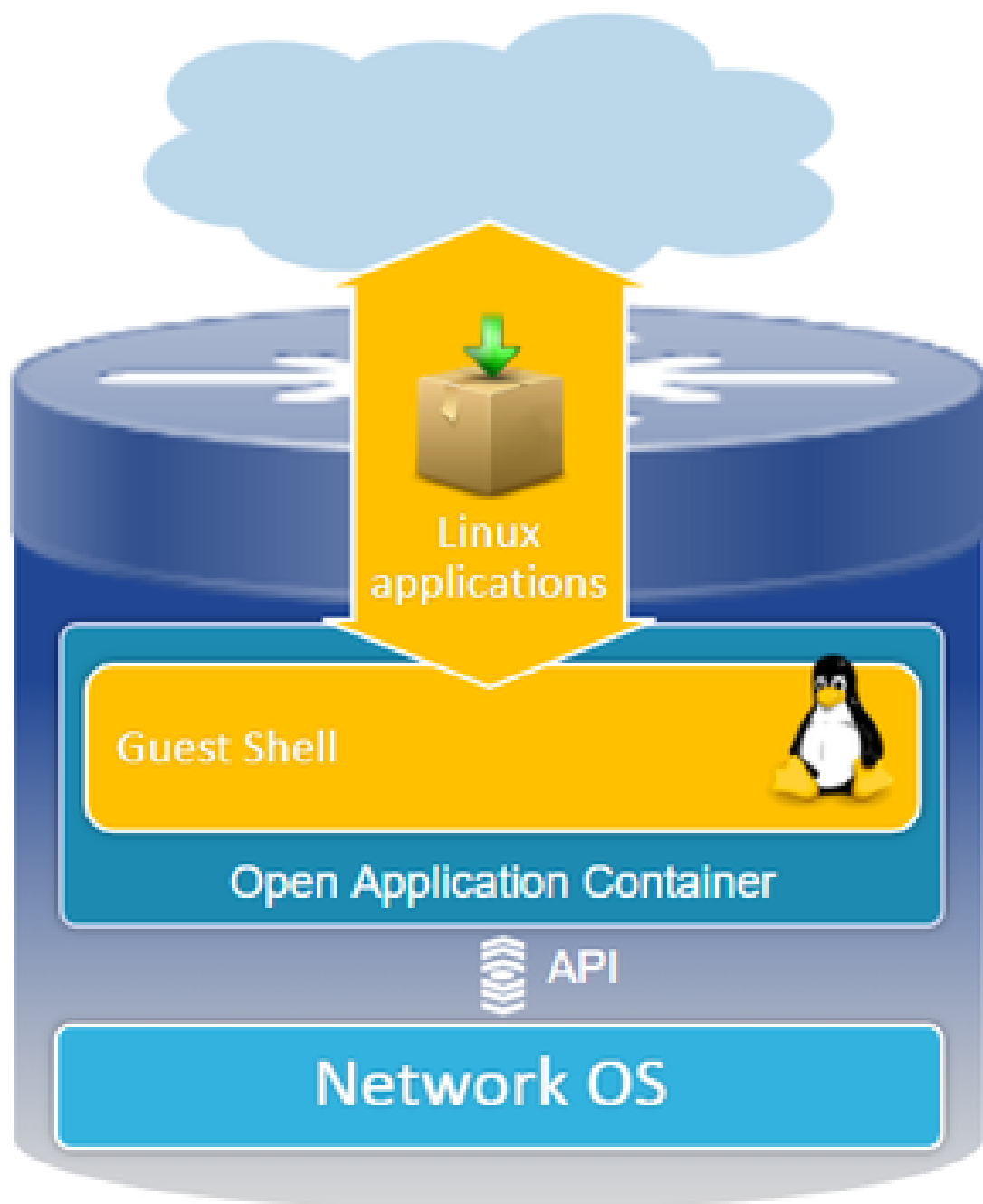
要件

- Cisco IOS® XEソフトウェアの基本知識
- Linuxファイルシステム
- Linuxコマンド

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- Catalyst 9200
- Catalyst 9300
- Catalyst 9400
- Catalyst 9500
- Catalyst 9600
- Cisco IOS XE 17.9.1以降のバージョン



このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認して

ください。

背景説明

ゲストシェルの概要

- ゲストシェルは、Linuxコンテナ(LXC)内で動作する独立した実行環境を提供します。
- ネットワークアクセス：管理者はLinuxネットワークインターフェイスを介してネットワークに接続でき、堅牢な接続と管理を実現します。
- ブートフラッシュアクセス：ブートフラッシュストレージに直接アクセスできるので、効率的なファイル管理とシステム運用が容易になります。
- Cisco IOS CLIアクセス：管理者はCisco IOSコマンドラインインターフェイスと直接対話する機能を備えており、シームレスな統合と制御が可能です。
- スクリプトの実行：この環境では、Pythonスクリプトのインストールと実行がサポートされており、自動化とカスタマイズが可能です。
- アプリケーションサポート：32ビットと64ビットの両方のLinuxアプリケーションをインストールして実行できるため、柔軟性と幅広い機能を提供します。

トラブルシューティング

.

コンフィギュレーション

1. IOXを有効にします。

```
<#root>
```

```
Switch#
```

```
conf terminal
```

```
Switch(config)#
```

```
iox
```

```
Switch(config)#
```

IOXが初期化されるまで1 ~ 5分待ちます。IOXが動作していることを確認します。CLIセッション中に表示されたメッセージを確認するか、showコマンドの出力を確認できます。

```
<#root>
```

```
*Mar 10 15:35:40.206: %UICFGEXP-6-SERVER_NOTIFIED_START: Switch 1 R0/0: psd: Server iox has been notified
```

```
*Mar 10 15:35:51.186: %IOX-3-PD_PARTITION_CREATE: Switch 1 R0/0: run_ioxn_caf: IOX may take upto 5 mins
```

```
*Mar 10 15:37:56.643: %IOX-3-IOX_RESTARTABILTILITY: Switch 1 R0/0: run_ioxn_caf: Stack is in N+1 mode, di
```

```
*Mar 10 15:38:05.835: %IM-6-IOX_ENABLEMENT: Switch 1 R0/0: ioxman: IOX is ready
```

.

```
Switch#show iox-service
```

```
IOx Infrastructure Summary:
```

```
-----
```

```
IOx service (CAF) :
```

```
Running
```

```
IOx service (HA) :
```

```
Running
```

```
IOx service (IOxman) :
```

```
Running
```

```
IOx service (Sec storage) :
```

```
Running
```

```
Libvirt 5.5.0 :
```

```
Running
```

```
Dockerd v19.03.13-ce :
```

```
Running
```

```
Sync Status : Disabled
```

2. ネットワーク構成。

この例では、AppGigabitEthernetインターフェイスを使用してネットワークアクセスを提供します。

```
<#root>
```

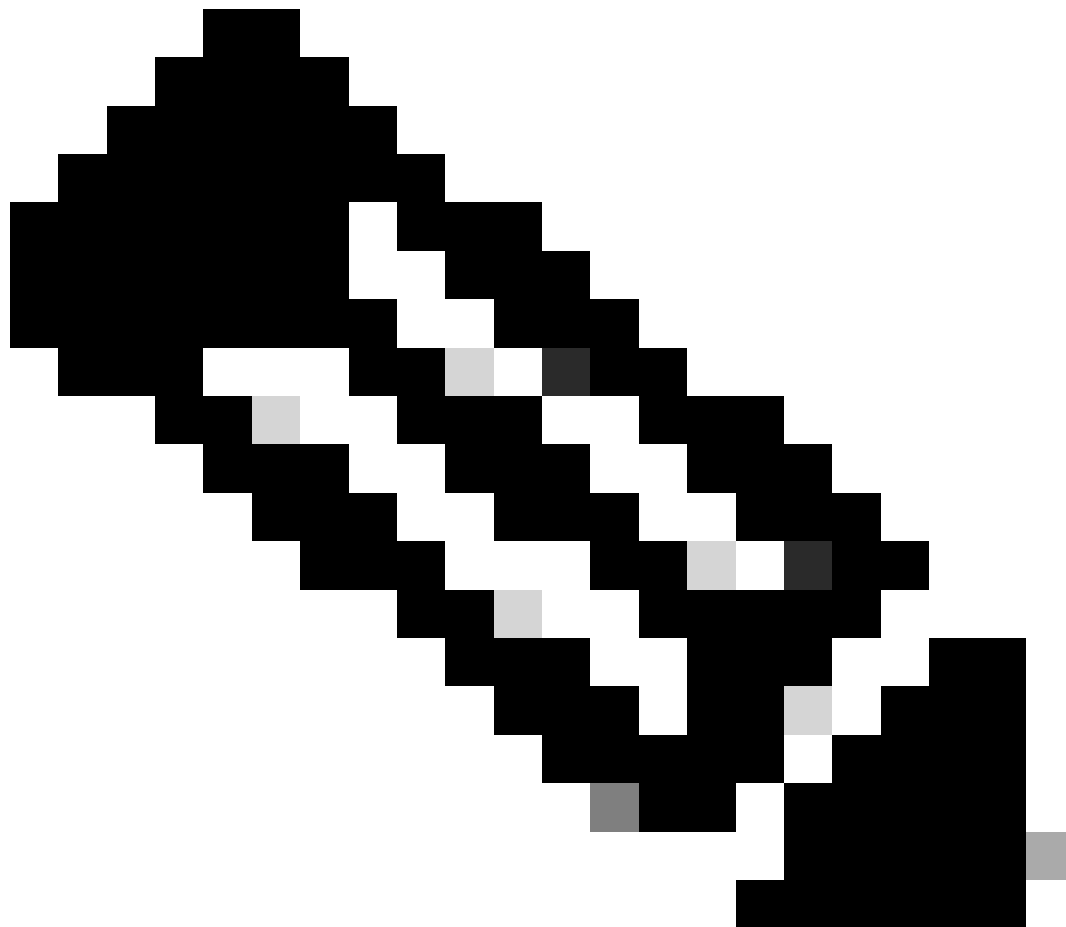
```
!  
interface AppGigabitEthernet1/0/1  
  switchport trunk allowed vlan 50  
  switchport mode trunk  
!  
app-hosting appid guestshell  
  app-vnic AppGigabitEthernet trunk  
  vlan 50 guest-interface 0
```

```
guest-ipaddress 192.168.10.10 netmask 255.255.255.0
```

```
  name-server0 192.168.10.254
```

```
end
```

```
!
```



注:GuestShellはデフォルトでDHCPクライアントサービスを含まないため、静的IPアドレスを使用します。DHCPクライアントサービスをインストールすると、IPアドレスを動的に取得できます。

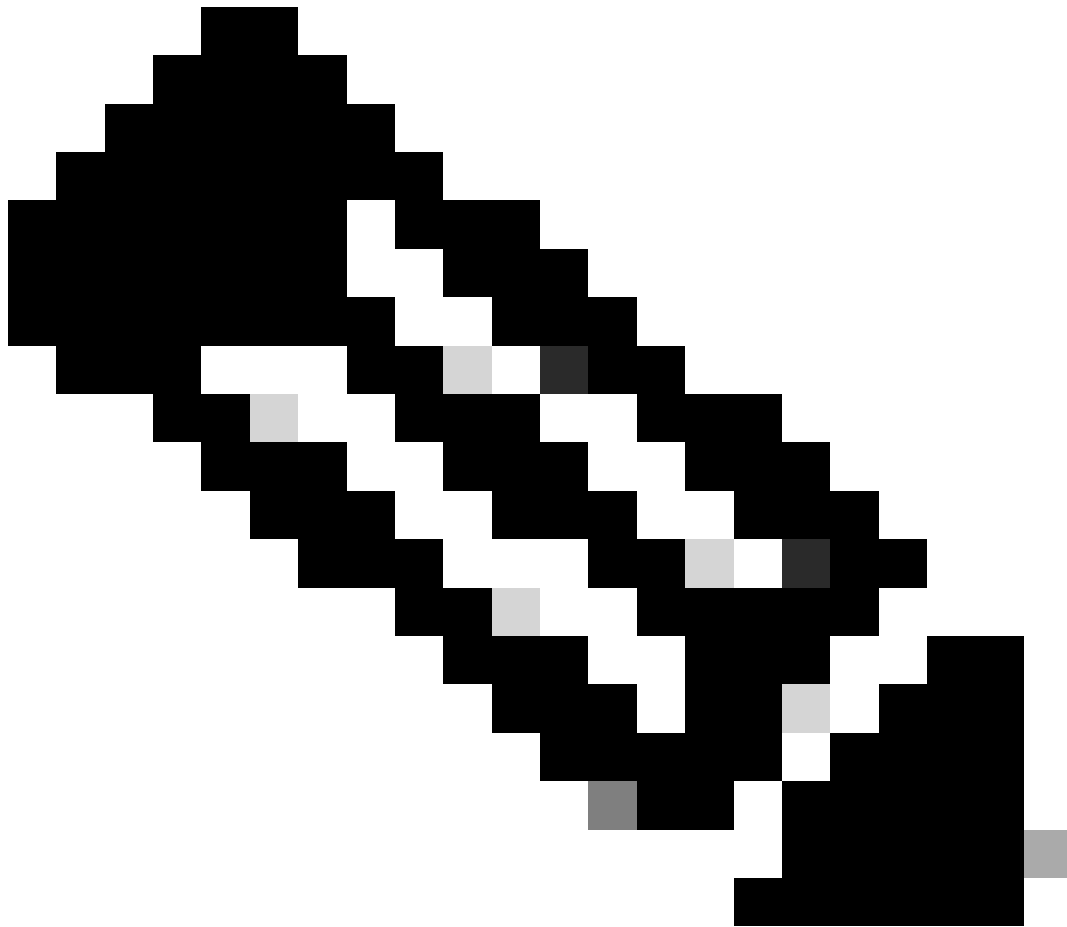
この例では、管理インターフェイス(Gi0/0)を使用してネットワークアクセスを提供します。管理モードでは、guestshellはCisco IOSで設定された管理ポートのIPアドレスがゲートウェイとして機能するネットワークに管理ポートを使用します。

```
<#root>
```

```
!  
app-hosting appid guestshell
```

```
app-vnic management guest-interface 0
```

```
name-server0 8.8.8.8  
!
```



注：管理インターフェイスを使用する場合、Guestshell eth0インターフェイスのデフォルト設定は、IPアドレス[192.168.30.2](https://www.ietf.org/rfc/rfc1918.txt)にハードコードされています。

仮想ポートグループ(VPC)インターフェイスを設定できます。

- 仮想ポートグループの設定：固定IPアドレスを使用して仮想ポートグループを確立します。
- インターネット接続：前面ポートインターフェイスがインターネットに接続され、外部アクセスが可能であることを確認します。
- NAT設定：ネットワークアドレス変換を容易にするために、仮想ポートグループと前面ポートの間にNATを設定します。
- GuestShell IP Assignment：仮想ポートグループインターフェイスと同じサブネット内のGuestShellにIPアドレスを割り当てます。

```
!  
interface VirtualPortGroup0  
  ip address 192.168.35.1 255.255.255.0  
  ip nat inside  
!
```

```
interface GigabitEthernet1/0/3
  no switchport
  ip address 192.168.100.10 255.255.255.0
  ip nat outside
!
ip route 0.0.0.0 0.0.0.0 192.168.100.254
ip route vrf Mgmt-vrf 0.0.0.0 0.0.0.0 192.168.10.254
ip nat inside source static tcp 192.168.35.2 7023 192.168.100.10 7023 extendable
!
!
ip access-list standard NAT_ACL
  10 permit 192.168.0.0 0.0.255.255
!
app-hosting appid guestshell
  app-vnic gateway1 virtualportgroup 0 guest-interface 0
  guest-ipaddress 192.168.35.2 netmask 255.255.255.0
  app-default-gateway 192.168.35.1 guest-interface 0
  name-server0 8.8.8.8
end
```

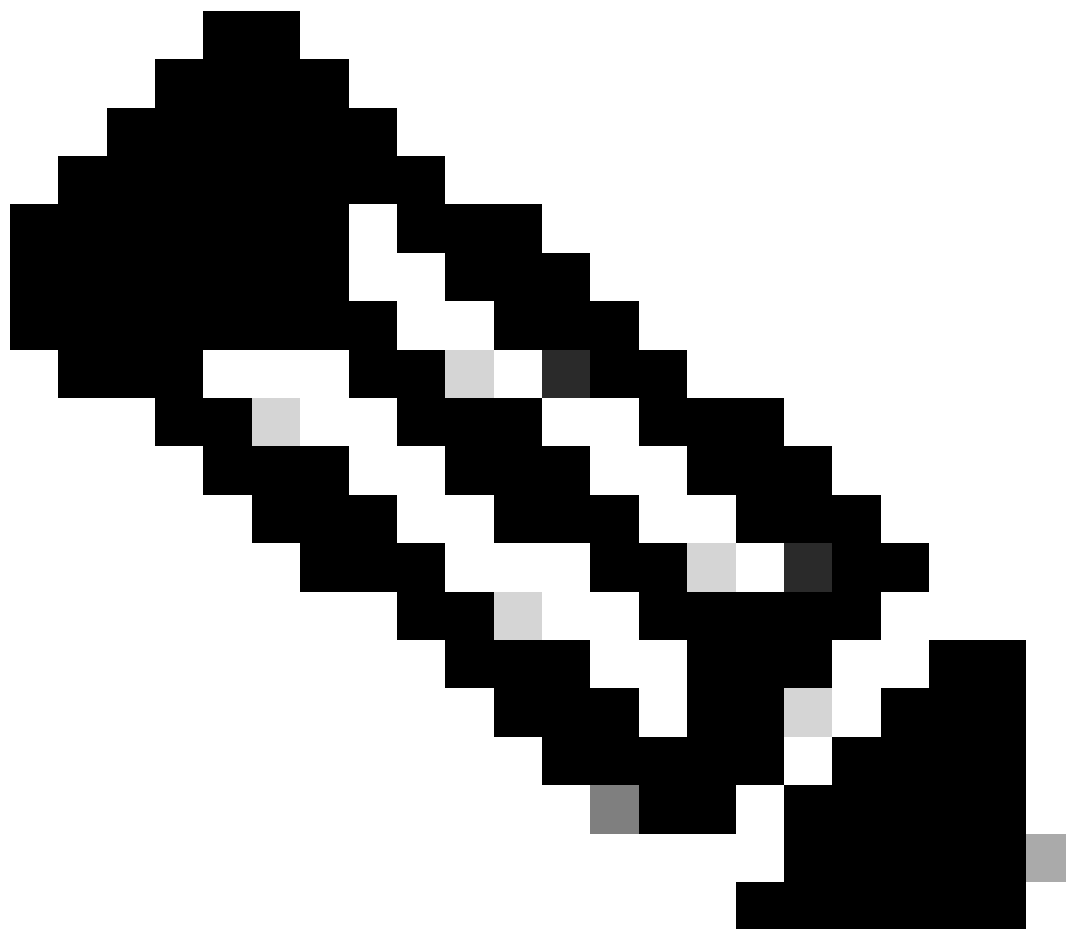
3. ゲストシェルを有効にします。

<#root>

Switch#

guestshell enable

```
Interface will be selected if configured in app-hosting
Please wait for completion
guestshell installed successfully
Current state is: DEPLOYED
guestshell activated successfully
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully
```



注：guestshell enableコマンドは、guestshellをインストールする唯一の方法です。App-hosting CLIを使用すると、ゲストシェルの制御にactivate/connect/deactivateを使用できません。代わりにguestshell exec CLIを使用することを推奨します。

ゲストシェルのライフサイクル

Guestshell disable:guestshellを無効にすると、guestshellへのアクセスが削除され、現在のセッションが強制終了されます。ファイルやデータは保持されるので、guestshell enableを使用してアクセスを復元できます。

```
<#root>
```

```
Switch#
```

```
guestshell disable
```

```
Guestshell disabled successfully
```

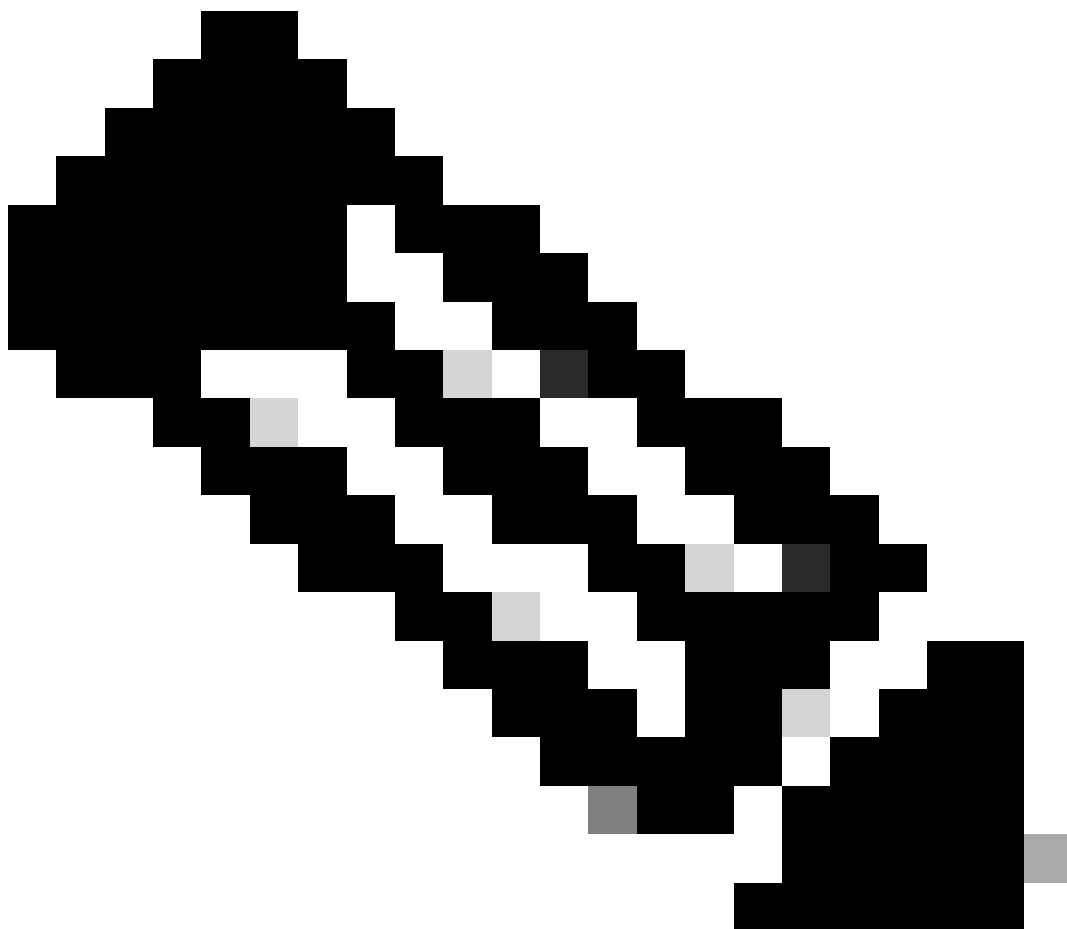
Guestshell destroy : ゲストシェルフファイルシステムを不可逆的に削除します。すべての files/data/scripts/settinguestshell/インストールされたパッケージとモジュール。

```
<#root>
```

```
Switch#
```

```
guestshell destroy
```

```
Guestshell destroyed successfully
```



注 : このコマンドを実行すると、すべてのデータが不可逆的に失われます。

Guestshell Run:Guestshellはコマンドguestshell run bashを実行してゲストシェル内にシェルを作成します。これにより、/binおよび/sbinにある任意のLinuxバイナリを使用できます。

```
<#root>
```

Switch#

guestshell run bash

```
[guestshell@guestshell ~]$ ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=2 ttl=254 time=0.517 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=254 time=0.552 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=254 time=0.447 ms
64 bytes from 192.168.10.1: icmp_seq=5 ttl=254 time=0.549 ms
```

Guestshell run python：このコマンドは、インタラクティブなPythonインタプリタを起動するために使用します。

<#root>

Switch#

guestshell run python3

```
Python 3.6.8 (default, Dec 22 2020, 19:04:08)
[GCC 8.4.1 20200928 (Red Hat 8.4.1-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

検証

次のコマンドを使用して、ゲストシェルを検証できます。

<#root>

Switch#

show app-hosting detail appid guestshell

```
App id           : guestshell
Owner            : iox
State            :
```

RUNNING

```
Application
Type           : lxc
Name           : GuestShell
Version        : 3.3.0
Description    : Cisco Systems Guest Shell XE for x86_64
Author         : Cisco Systems
Path           : /guestshell/guestshell.tar
URL Path       :
Multicast      : yes
Activated profile name : custom
```

```
Resource reservation
Memory         : 256 MB
```

Disk : 1 MB
CPU : 800 units
CPU-percent : 11 %
VCPU : 1

Platform resource profiles

Profile Name	CPU(unit)	Memory(MB)	Disk(MB)

Attached devices

Type	Name	Alias

serial/shell	iox_console_shell	serial0
serial/aux	iox_console_aux	serial1
serial/syslog	iox_syslog	serial2
serial/trace	iox_trace	serial3

Network interfaces

eth0:

MAC address : 52:54:dd:5b:c4:b8
IPv4 address : 192.168.30.2
IPv6 address : ::
Network name : mgmt-bridge200

Port forwarding

Table-entry	Service	Source-port	Destination-port

Switch#show app-hosting list			
App id	State		

guestshell			

RUNNING

Switch#

guestshell run sudo ifconfig

eth0: flageuestshell=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.10.10 netmask 255.255.255.0 broadcast 192.168.10.255
inet6 fe80::5054:ddff:fece:a7c9 prefixlen 64 scopeid 0x20
ether 52:54:dd:ce:a7:c9 txqueuelen 1000 (Ethernet)
RX packets 3 bytes 266 (266.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 9 bytes 726 (726.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flageuestshell=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10
loop txqueuelen 1000 (Local Loopback)
RX packets 338 bytes 74910 (73.1 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 338 bytes 74910 (73.1 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

リソースのリサイズ

```
<#root>
```

```
!  
app-hosting appid guestshell  
  app-vnic management guest-interface 0
```

```
app-resource profile custom
```

```
  cpu 1000  
  memory 512  
  persist-disk 200  
!
```

変更を有効にするには、ゲストシェルを無効にしてから有効にする必要があります。

```
<#root>
```

```
Switch#
```

```
guestshell disable
```

```
Guestshell disabled successfully  
Switch#
```

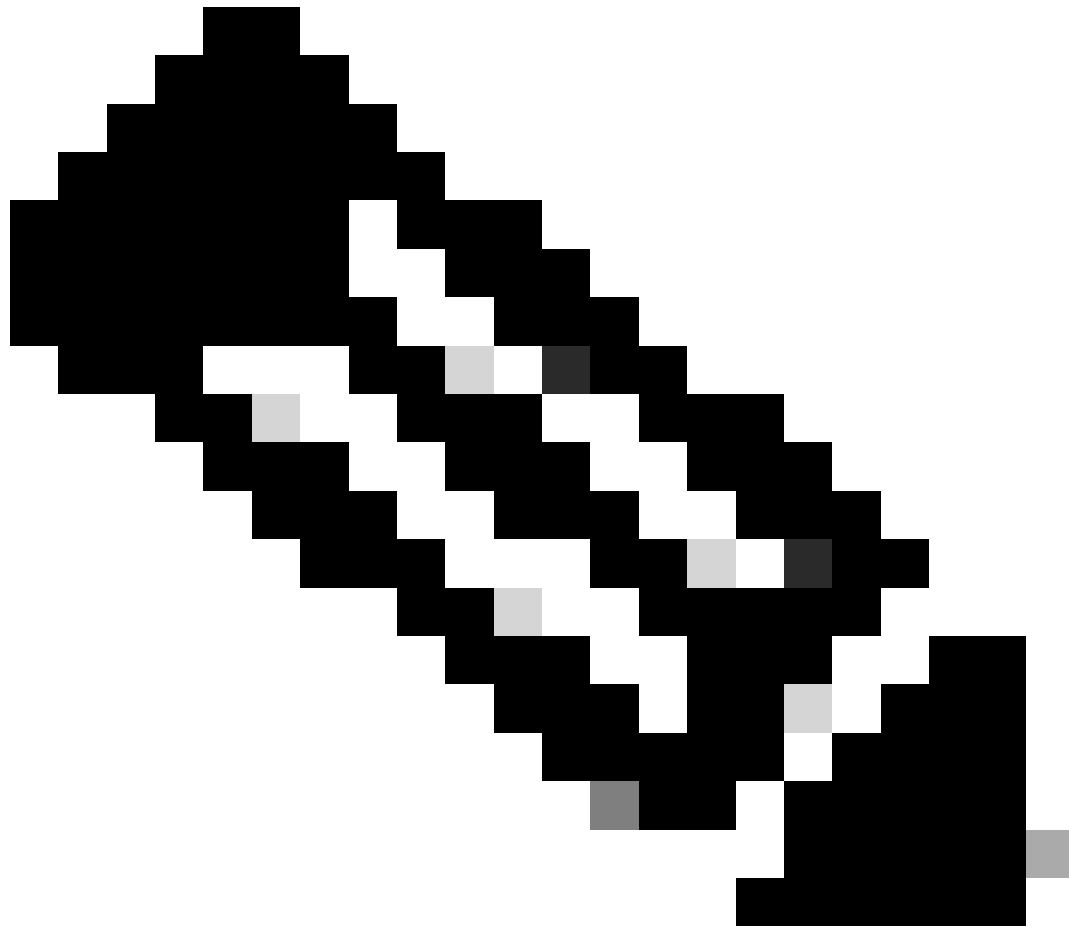
```
guestshell enable
```

```
*Mar 11 01:17:46.841: %SYS-5-CONFIG_I: Configured from console by coguestshell enable  
Interface will be selected if configured in app-hosting  
Please wait for completion  
guestshell activated successfully  
Current state is: ACTIVATED  
guestshell started successfully  
Current state is: RUNNING  
Guestshell enabled successfully  
Switch#
```

```
show app-hosting detail appid guestshell | sec Resource reservation
```

```
Resource reservation
```

```
Memory           : 512 MB  
Disk              : 200 MB  
CPU               : 1110 units  
CPU-percent       : 15 %  
VCPU              : 1
```



注：スイッチでは、プラットフォームによる制限の最大値までリソースのサイズ変更が許可されます。アプリケーションについては、「[Cisco Catalyst 9000プラットフォームハードウェアリソース](#)」を参照してください。

一般的なシナリオ

DHCP 設定。

問題：DHCPクライアントバイナリ(DHCLIENT)が存在しません。

解決方法

DHCPクライアントをインストールするには、Yumユーティリティで`sudo yum install dhcp-client`コマンドを使用します。ただし、CentOS Stream 8のリポジトリは使用できなくなっています。

1. DHCPが機能するようにアプリケーションホスティングでIPアドレスを設定することは避けて

ください。

```
!  
interface AppGigabitEthernet1/0/1  
    switchport trunk allowed vlan 50  
    switchport mode trunk  
!  
app-hosting appid guestshell  
    app-vnic AppGigabitEthernet trunk  
        vlan 50 guest-interface 0  
name-server0 8.8.8.8  
!
```

2. ミラーリストをyum repoのbaseurl vaultに変更します。

<#root>

Switch#

guestshell run bash

[guestshell@guestshell ~]\$

sudo find /etc/yum.repos.d/ -type f -exec sed -i 's/mirrorlist=/#mirrorlist=/g' {} +

[guestshell@guestshell ~]\$

sudo find /etc/yum.repos.d/ -type f -exec sed -i 's/#baseurl=/baseurl=/g' {} +

[guestshell@guestshell ~]\$

sudo find /etc/yum.repos.d/ -type f -exec sed -i 's/mirror.centos.org/

/g' {} +

[guestshell@guestshell ~]\$ cat /etc/yum.repos.d/CentOS-Stream-RealTime.repo

CentOS-Stream-RealTime.repo

The mirrorlist system uses the connecting IP address of the client and the
update status of each mirror to pick current mirrors that are geographically
close to the client. You should use this for CentOS updates unless you are
manually picking other mirrors.
#

If the mirrorlist does not work for you, you can try the commented out
baseurl line instead.

[rt]

name=CentOS Stream \$releasever - RealTime

#mirrorlist=http://mirrorlist.centos.org/?release=\$stream&arch=\$basearch&repo=RT&infra=\$infra

baseurl=http://

/\$contentdir/\$stream/RT/\$basearch/os/

gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial

3. パッケージをインストールします。

<#root>

guestshell@guestshell ~]\$

sudo yum install dhcp-client

ast metadata expiration check: 0:50:34 ago on Wed Mar 12 17:44:46 2025.
Dependencies resolved.

Package	Architecture	Version	Repository	Size
Installing:				
dhcp-client	x86_64	12:4.3.6-50.e18	baseos	319 k
Installing dependencies:				
bind-export-libs	x86_64	32:9.11.36-13.e18	baseos	1.1 M
dhcp-common	noarch	12:4.3.6-50.e18	baseos	208 k
dhcp-libs	x86_64	12:4.3.6-50.e18	baseos	148 k

Transaction Summary

Install 4 Packages

Total download size: 1.8 M

Installed size: 3.9 M

Is this ok [y/N]: y

Downloading Packages:

(1/4): dhcp-client-4.3.6-50.e18.x86_64.rpm	284 kB/s 319 kB	00:01
(2/4): dhcp-common-4.3.6-50.e18.noarch.rpm	171 kB/s 208 kB	00:01
(3/4): dhcp-libs-4.3.6-50.e18.x86_64.rpm	572 kB/s 148 kB	00:00
(4/4): bind-export-libs-9.11.36-13.e18.x86_64.r	577 kB/s 1.1 MB	00:02

Total	908 kB/s 1.8 MB	00:02
-------	-------------------	-------

CentOS Stream 8 - BaseOS	1.6 MB/s 1.6 kB	00:00
--------------------------	-------------------	-------

Importing GPG key 0x8483C65D:

 Userid : "CentOS (CentOS Official Signing Key) <CentOS Official Signing Key>"

 Fingerprint: 99DB 70FA E1D7 CE22 7FB6 4882 05B5 55B3 8483 C65D

 From : /etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial

Is this ok [y/N]: y

Key imported successfully

Running transaction check

Transaction check succeeded.

Running transaction test

Transaction test succeeded.

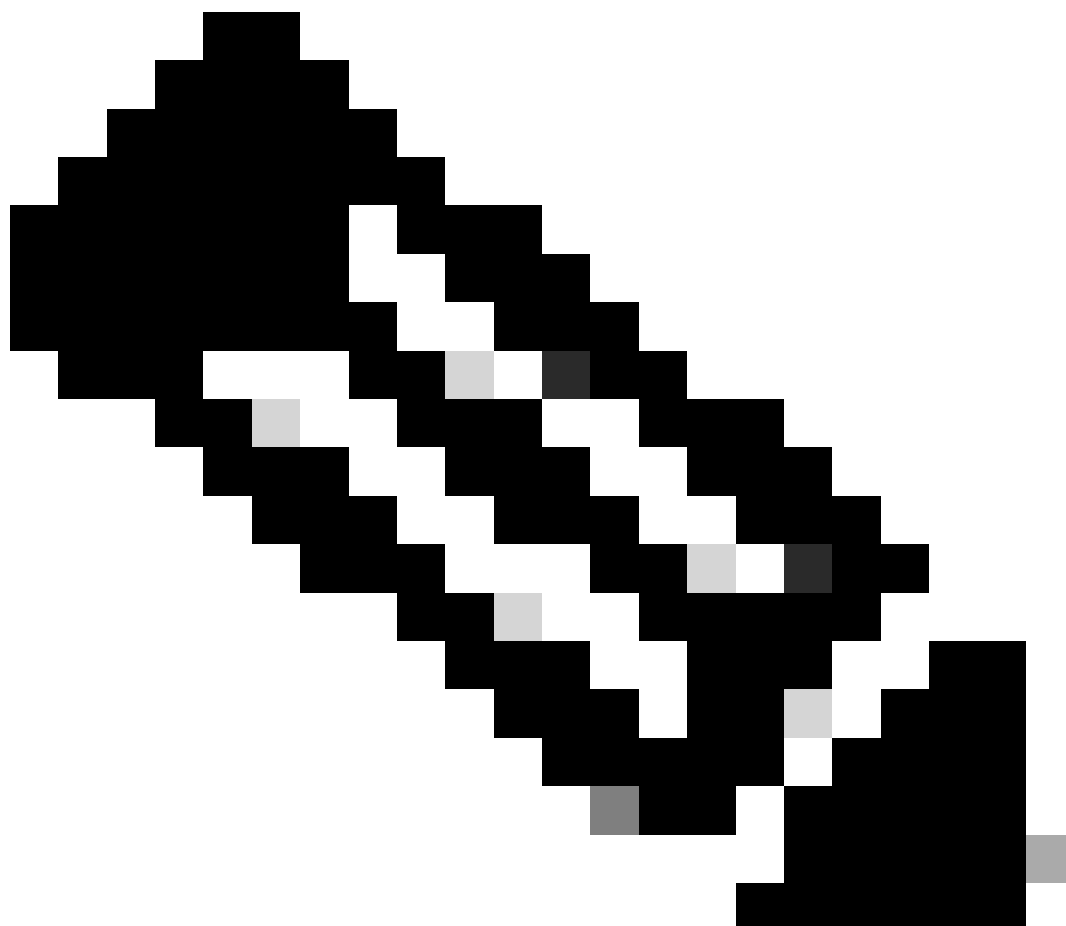
Running transaction

Preparing	:		1/1
Installing	:	dhcp-libs-12:4.3.6-50.el8.x86_64	1/4
Installing	:	dhcp-common-12:4.3.6-50.el8.noarch	2/4
Installing	:	bind-export-libs-32:9.11.36-13.el8.x86_64	3/4
Running scriptlet:		bind-export-libs-32:9.11.36-13.el8.x86_64	3/4
Installing	:	dhcp-client-12:4.3.6-50.el8.x86_64	4/4
Running scriptlet:		dhcp-client-12:4.3.6-50.el8.x86_64	4/4
Verifying	:	bind-export-libs-32:9.11.36-13.el8.x86_64	1/4
Verifying	:	dhcp-client-12:4.3.6-50.el8.x86_64	2/4
Verifying	:	dhcp-common-12:4.3.6-50.el8.noarch	3/4
Verifying	:	dhcp-libs-12:4.3.6-50.el8.x86_64	4/4

Installed:

bind-export-libs-32:9.11.36-13.el8.x86_64	dhcp-client-12:4.3.6-50.el8.x86_64
dhcp-common-12:4.3.6-50.el8.noarch	dhcp-libs-12:4.3.6-50.el8.x86_64

Complete!



注: (CALO用) Yum用のプロキシを設定することは可能です。これにより、Yumはパッケージのダウンロードのためにプロキシサーバを介して接続できます。

```
[guestshell@guestshell ~]$ echo "proxy=http://<IP_address:port>/" | sudo tee -a /etc/yum.conf > /dev/null
```

4. eth0のDHCP IPアドレスを要求します。

<#root>

```
[guestshell@guestshell ~]$  
sudo dhclient eth0
```

5. IPアドレスの割り当てをチェックして、DHCPクライアントが動作しているかどうかを確認します。

<#root>

Switch#

guestshell run ifconfig

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
```

```
inet 192.168.10.2  netmask 255.255.255.0  broadcast 192.168.10.255
```

```
    inet6 fe80::5054:ddff:fea0:4aef  prefixlen 64  scopeid 0x20  
    ether 52:54:dd:a0:4a:ef  txqueuelen 1000  (Ethernet)  
    RX packets 1516  bytes 2009470 (1.9 MiB)  
    RX errors 0  dropped 0  overruns 0  frame 0  
    TX packets 687  bytes 54603 (53.3 KiB)  
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
```

```
    inet 127.0.0.1  netmask 255.0.0.0  
    inet6 ::1  prefixlen 128  scopeid 0x10  
    loop txqueuelen 1000  (Local Loopback)  
    RX packets 773  bytes 90658 (88.5 KiB)  
    RX errors 0  dropped 0  overruns 0  frame 0  
    TX packets 773  bytes 90658 (88.5 KiB)  
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

DNF Packet Managerの更新に失敗しました

問題：次のエラーが原因で、Guestshellでsudo dnf update -yを完了できません：

```
[guestshell@guestshell ~]$ sudo dnf upgrade --refresh  
Warning: failed loading '/etc/yum.repos.d/CentOS-Base.repo', skipping.
```

解決方法

1. RPMパッケージを再インストールしてアップグレードします。

<#root>

! Clean packages

```
[guestshell@guestshell ~]$ sudo dnf clean all
```

! Reinstall and update the tpm2-tss package:

```
[guestshell@guestshell ~]$
```

```
sudo dnf install tpm2-tss-2.3.2-3.el8
```

```
[guestshell@guestshell ~]$
```

```
sudo dnf upgrade rpm
```

2. gitパッケージを個別にインストールします。

<#root>

```
[guestshell@guestshell ~]$
```

```
sudo dnf install git -y
```

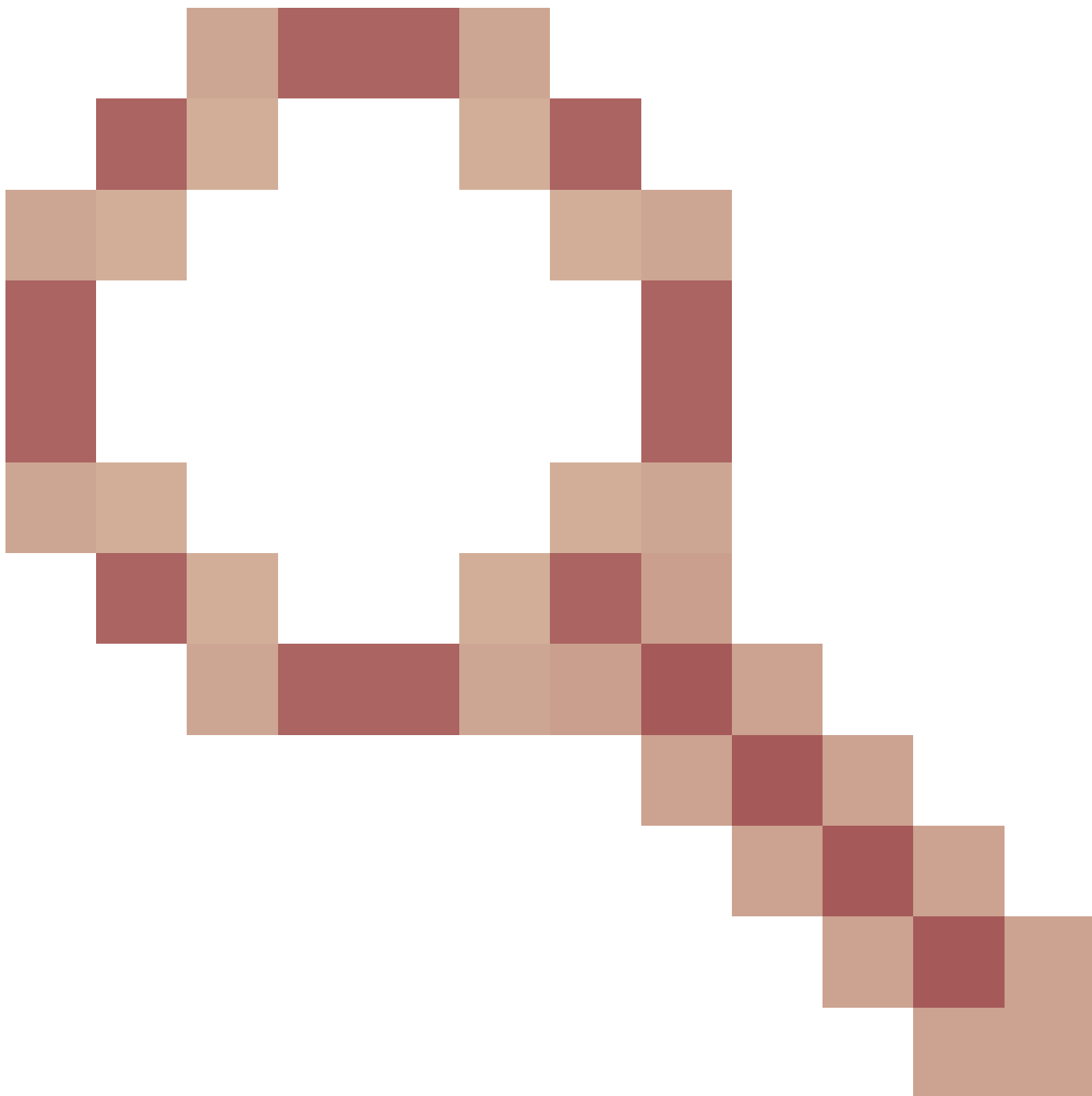
アップグレード後にゲストシェルへのアクセスが失われる

問題：バージョン17.08.01以降にアップグレードすると、ゲストシェルにアクセスできなくなり、スクリプトを実行できなくなります。

```
Switch#guestshell run bash  
Switch#
```

解決方法

この問題は、Cisco Bug ID [CSCwi63075](#)



これは、スイッチでFIPSが有効になっている場合にアップグレード中にトリガーされます。

1. FIPSを無効にします。

```
<#root>
```

```
Switch#
```

```
configure terminal
```

```
Switch(config)#
```

```
no fips authorization-key
```

FIPS: Authorization-key erased ONLY from the Flash.
But the authorization-key is still operational. Use

"reload"

command for complete removal of key and to enter into non fips-mode.
Make sure to remove fips key from all the members of the stack individually

2. 修正を含むいずれかのバージョン、バージョン17.12.04以降のリリースにアップグレードできます。

IPv6アドレッシングの制限

問題：Cisco Guestshell環境内のインターフェイスに意図しないIPv6アドレスが表示される問題があります。ただし、意図的に設定されているわけではありません。

IPv4を使用するスイッチで確認された設定は次のとおりです。

<#root>

```
!  
app-hosting appid guestshell  
app-vnic AppGigabitEthernet trunk  
vlan 50 guest-interface 0
```

guest-`ipaddress` 192.168.20.10 netmask 255.255.255.0 <-- IPv4 address configured

```
app-default-gateway 192.168.20.1 guest-interface 0  
app-resource profile custom  
name-server1 192.168.20.1  
!
```

Guestshell環境内でifconfigコマンドを実行すると、IPv4アドレスとIPv6アドレスの両方が表示されるようになります。

<#root>

```
lat1-2-ssw01.gts#guestshell run bash  
[guestshell@guestshell ~]$
```

sudo ifconfig

```
eth0: flags=4163 mtu 1500  
inet 192.168.20.10 netmask 255.255.255.0 broadcast 192.168.20.255
```

```
inet6 2620:119:5022:515:5054:ddff:fe41:c643 prefixlen 64 scopeid 0x0  
inet6 fe80::5054:ddff:fe41:c643 prefixlen 64 scopeid 0x20
```

```
ether 52:54:dd:41:c6:43 txqueuelen 1000 (Ethernet)  
RX packets 7829 bytes 1750981 (1.6 MiB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 5551 bytes 744320 (726.8 KiB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
lo: flags=73 mtu 65536  
inet 127.0.0.1 netmask 255.0.0.0  
inet6 ::1 prefixlen 128 scopeid 0x10
```

```
loop txqueuelen 1000 (Local Loopback)
RX packets 292 bytes 63812 (62.3 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 292 bytes 63812 (62.3 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

解決方法

手順1:GuestshellでIPv6ルーティングとアドレッシングを無効にします。

<#root>

```
lat1-2-ssw01.gts#
```

```
guestshell run bash
```

```
[guestshell@guestshell ~]$
```

```
sudo sysctl -w net.ipv6.conf.all.disable_ipv6=1
```

```
[guestshell@guestshell ~]$
```

```
sudo sysctl -w net.ipv6.conf.default.disable_ipv6=1
```

```
[guestshell@guestshell ~]$
```

```
sudo sysctl -w net.ipv6.conf.lo.disable_ipv6=1
```

手順2:IPv6が無効になっていることを確認します。

```
[guestshell@guestshell ~]$ /sbin/ifconfig
```

```
eth0: flags=4163 mtu 1500
    inet 192.168.20.10 netmask 255.255.255.0 broadcast 192.168.20.255
    ether 52:54:dd:41:c6:43 txqueuelen 1000 (Ethernet)
    RX packets 7829 bytes 1750981 (1.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5551 bytes 744320 (726.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
lo: flags=73 mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 292 bytes 63812 (62.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 292 bytes 63812 (62.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Pythonスクリプト実行中のディスクスペースエラー

問題：この問題には、Guestshell内のEmbedded Event Manager(EEM)で実行されているPythonスクリプトが関係しています。次のエラーでディスク領域が不足しているため、スクリプトが失敗

します。

<#root>

```
guestshell run python3 /flash/guest-share/monitoring.py -rt True -bgp True
---- pushing bgp status ----
OSError: [Errno 28] No space left on device
```

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

```
File "/flash/guest-share/monitoring_periodic_tasks.py", line 18, in
    print(bgp_status())
File "/bootflash/guest-share/monitoring_bgp_status.py", line 15, in bgp_status
    vrf = cli.cli('show vrf')
File "/usr/lib/python3.6/site-packages/cli/__init__.py", line 311, in cli
    _log_to_file("CLI execution invoked for '" + command + "'")
File "/usr/lib/python3.6/site-packages/cli/__init__.py", line 87, in _log_to_file
    logfile.close()
```

OSError: [Errno 28] No space left on device

! This error indicates that the disk space allocated for logging command executions within the Guestshell is insufficient.

解決方法

ディスク領域が不足する問題を解決するには、Guestshell環境のpersist-diskサイズを増やす必要があります。

1.アプリケーションリソースプロファイルを変更して、永続ディスクサイズを増やします。

<#root>

```
Switch(config-app-hosting)#
app-resource profile custom
Switch(config-app-hosting-profile)#
persist-disk 100
Switch(config-app-hosting-profile)#
cpu 800
Switch(config-app-hosting-profile)#
memory 256
Switch(config-app-hosting-profile)#
end
```

2. 設定を保存し、guestshellを有効/無効にします。

```
Switch# write memory
Switch#guestshell disable
Guestshell disabled successfully
Switch#guestshell enable
Interface will be selected if configured in app-hosting
Please wait for completion
guestshell installed successfully
Current state is: DEPLOYED
guestshell activated successfully
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully
```

Syslogのロギング

次のコマンドをguestshellで実行すると、syslogメッセージを表示できます。

```
<#root>
```

```
[guestshell@guestshell ~]$
```

```
sudo logger -p 1 "Priority 1"
```

```
[guestshell@guestshell ~]$
```

```
sudo cat /var/log/messages
```

```
Mar 11 02:05:24 localhost systemd[248]: user@0.service: Failed at step PAM spawning /usr/lib/systemd/sy
Mar 11 02:05:24 localhost systemd[1]: user@0.service: Failed with result 'protocol'.
Mar 11 02:05:24 localhost systemd[1]: Failed to start User Manager for UID 0.
Mar 11 02:05:24 localhost systemd[1]: Stopping /run/user/0 mount wrapper...
Mar 11 02:05:24 localhost systemd[1]: run-user-0.mount: Succeeded.
Mar 11 02:05:24 localhost systemd[1]: user-runtime-dir@0.service: Succeeded.
Mar 11 02:05:24 localhost systemd[1]: Stopped /run/user/0 mount wrapper.
Mar 11 02:05:24 localhost root[250]: Priority 1
[guestshell@guestshell ~]$ sudo cat /var/log/secure
Mar 11 02:05:24 localhost systemd[248]: pam_unix(systemd-user:account): expired password for user root
Mar 11 02:05:24 localhost sudo[246]: pam_systemd(sudo:session): Failed to create session: Start job for
Mar 11 02:05:24 localhost sudo[246]: pam_unix(sudo:session): session opened for user root by (uid=0)
```

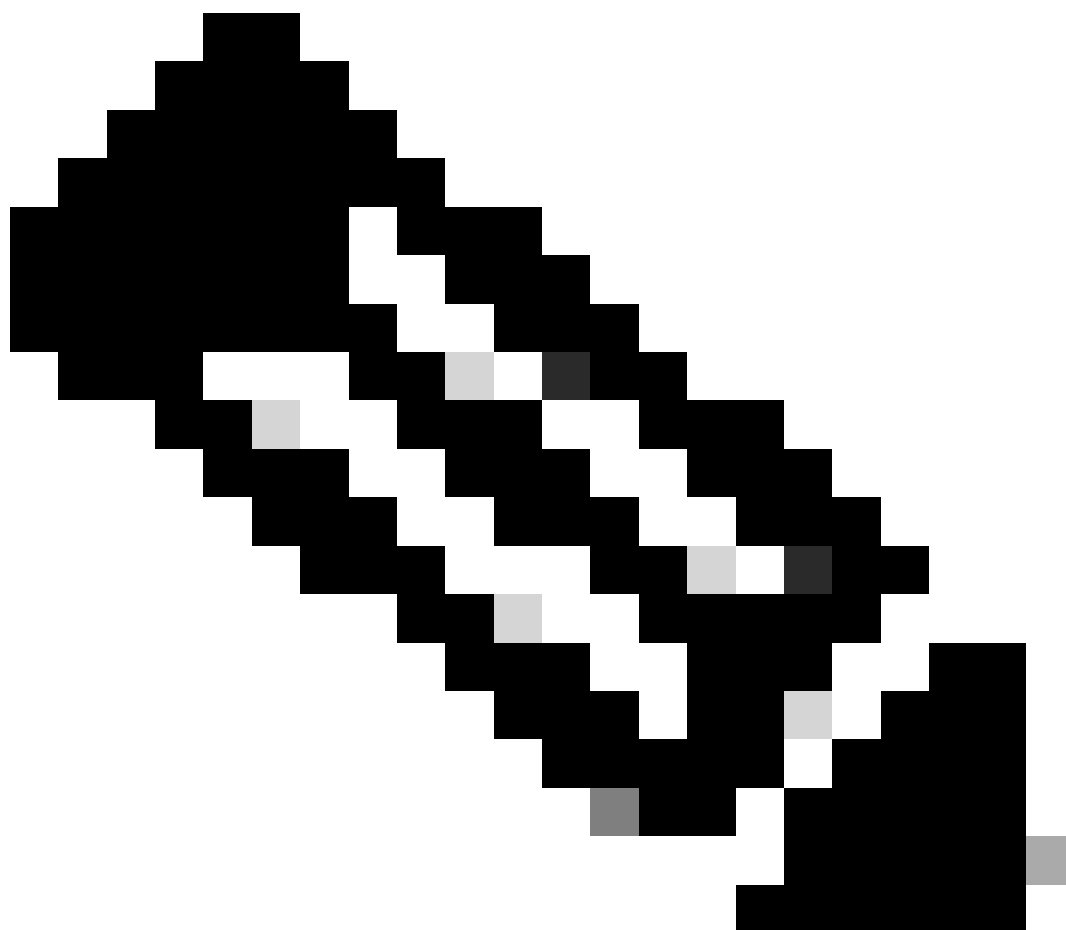
トレースログ

ファイルをローテーションするには、app-hosting move appid guestshell log to bootflash:folder_nameを使用できます。これはターゲットディレクトリにローテーションするために使用できますが、traceloguestshellディレクトリにもローテーションされます。

```
<#root>
```

```
Switch#
```

```
app-hosting move appid guestshell log to bootflash:
```



注:IOX guestshell.logは常に/tmp/rp/trace内のアクティブなtracelogファイルです。このファイルが1 MBに達すると、自動的にタイムスタンプとともにbootflash:tracelogShell/にローテーションされ、新しいファイルが開始されます。

関連情報

- [Cisco Catalyst 9000シリーズスイッチでのアプリケーションホスティングに関するホワイトペーパー](#)
- [Cisco Catalyst 9300 シリーズ スイッチ ハードウェア設置ガイド](#)
- [プログラマビリティ設定ガイド、Cisco IOS XE 17.9.x](#)
- [Catalyst 9300 StackWise システムアーキテクチャに関するホワイトペーパー](#)
- Cisco Bug ID [CSCwi63075](#):FIPSモードが有効な場合、17.08.01以降にアップグレードした後、CLIでGuestshellにアクセスできない

- [テクニカル サポートとドキュメント - Cisco Systems](#)

翻訳について

シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。