サードパーティ製アプリケーション

この章の内容は、次のとおりです。

- サードパーティ アプリケーションについて，1 ページ
- サードパーティのネイティブ RPM/パッケージのインストール，1 ページ
- 署名付き RPM のインストール，3 ページ
- サードパーティ RPM の永続化，8 ページ
- VSH からの RPM のインストール，8 ページ
- サードパーティ製アプリケーション，13 ページ

サードパーティ アプリケーションについて

サードパーティ アプリケーション向けの RPM は、https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64 のリポジトリから入手できます。これらのアプリケーションは、Bash シェル内または NX-OS CLI 経由で **yum** コマンドを使用することで、ネイティブ ホストにインストールされます。

yum install*rpm* コマンドを入力すると、Cisco **YUM** プラグインが実行されます。このプラグインは、RPM を非表示の場所にコピーします。スイッチのリロード時、システムによって RPM が再インストールされます。

`/etc` での設定内容について、Linux プロセスの **incrond** が、そのディレクトリで作成されたアーティファクトをモニタリングし、それらを非表示の場所にコピーします。これらのコピーは `/etc` に戻されます。

サードパーティのネイティブ RPM/パッケージのインストール

パッケージのインストールの詳細なワークフローは次のとおりです。

手順

エージェントが保存されているシスコのリポジトリをポイントするようにスイッチのリポジトリを設定します。

```
bash-4.2# cat /etc/yum/repos.d/open-nxos.repo
[open-nxos]
name=open-nxos
baseurl=https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64/
enabled=1
gpgcheck=0
sslverify=0
```

yum を使用した rpm のインストール例を、すべてのインストール ログとともに示します。

例：

```
bash-4.2# yum install splunkforwarder
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package splunkforwarder.x86_64 0:6.2.3-264376 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=================================================================================================

 Package              Arch                 Version           Repository             Size
=================================================================================================
Installing:
splunkforwarder       x86_64               6.2.3-264376      open-nxos              13 M

Transaction Summary
=================================================================================================
Install       1 Package

Total size: 13 M
Installed size: 34 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : splunkforwarder-6.2.3-264376.x86_64
                                                                                          1/1
complete

Installed:
  splunkforwarder.x86_64 0:6.2.3-264376


Complete!
bash-4.2#
```

パッケージのインストールが成功したかをスイッチにクエリし、そのプロセスやサービスが起動され、実行中であることを確認する例を示します。

例：

```
bash-4.2# yum info splunkforwarder
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
Fretta              |  951 B     00:00 ...
groups-repo         | 1.1 kB     00:00 ...
localdb             |  951 B     00:00 ...
patching            |  951 B     00:00 ...
thirdparty          |  951 B     00:00 ...
Installed Packages
Name        : splunkforwarder
Arch        : x86_64
Version     : 6.2.3
Release     : 264376
Size        : 34 M
Repo        : installed
From repo   : open-nxos
Summary     : SplunkForwarder
License     : Commercial
Description : The platform for machine data.
```

署名付き RPM のインストール

署名付き RPM の確認

次のコマンドを実行し、特定の RPM が署名されているかどうかを確認します。

Run, **rpm -K *rpm_file_name***

署名付き RPM ではない場合

```
bash-4.2# rpm -K bgp-1.0.0-r0.lib32_n9000.rpm

bgp-1.0.0-r0.lib32_n9000.rpm: (sha1) dsa sha1 md5 OK
```

署名付き RPM の場合

```
bash-4.2#
rpm -K puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm

puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm: RSA sha1 MD5 NOT_OK

bash-4.2#
```

サードパーティ製の署名付き RPM では、パッケージをインストールする前に GPG 公開キーをインポートする必要があります。このキーがインポートされていない場合、**yum** を実行すると、次のエラーが返されます。

```
bash-4.2#
yum install puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm -q

Setting up Install Process

warning: rpmts_HdrFromFdno: Header V4 RSA/SHA1 signature: NOKEY, key ID 4bd6ec30

Cannot open: puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm. Skipping.

Error: Nothing to do
```

キーの手動インポートによる署名付き RPM のインストール

- GPG キーを /etc rootfs にコピーして、これらがリブート後も永続化されるようにします。

```
bash-4.2# mkdir -p /etc/pki/rpm-gpg

bash-4.2# cp -f RPM-GPG-KEY-puppetlabs /etc/pki/rpm-gpg/
```

- 次のコマンドを使用して、キーをインポートします。

```
bash-4.2# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-puppetlabs

bash-4.2#

bash-4.2# rpm -q gpg-pubkey

gpg-pubkey-4bd6ec30-4c37bb40
bash-4.2# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-puppetlabs

bash-4.2#

bash-4.2# rpm -q gpg-pubkey

gpg-pubkey-4bd6ec30-4c37bb40
```

- *yum* コマンドを使用して署名付き RPM をインストールします。

```
bash-4.2#
yum install puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages

groups-repo        | 1.1 kB     00:00 ...

         .
localdb            |  951 B     00:00 ...


patching           |  951 B     00:00 ...


thirdparty         |  951 B     00:00 ...


Setting up Install Process

Examining puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm:
puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64

Marking puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm to be installed

Resolving Dependencies

--> Running transaction check

---> Package puppet-enterprise.x86_64 0:3.7.1.rc2.6.g6cdc186-1.pe.nxos will be installed


--> Finished Dependency ResolutionDependencies Resolved

==================================================================================


 Package           Arch     Version                          Repository
Size
```

```
=================================================================================

Installing:

puppet-enterprise  x86_64   3.7.1.rc2.6.g6cdc186-1.pe.nxos    /puppet-enterprise-
46 M
                                                                3.7.1.rc2.6.g6cdc186-1.
                                                                pe.nxos.x86_64


Transaction Summary

=================================================================================


Install       1 Package

Total size: 46 M

Installed size: 46 M

Is this ok [y/N]: y

Downloading Packages:

Running Transaction Check

Running Transaction Test

Transaction Test Succeeded

Running Transaction

  Installing : puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64
                                                                                  1/1


Installed:

  puppet-enterprise.x86_64 0:3.7.1.rc2.6.g6cdc186-1.pe.nxos


Complete!

bash-4.2#
```

キーの自動インポートによる署名付きサードパーティ RPM のインストール

キーおよび RPM をポイントするように yum repo をセットアップします。

```
root@switch# cat /etc/yum/repos.d/puppet.repo

[puppet]

name=Puppet RPM

baseurl=file:///bootflash/puppet

enabled=1

gpgcheck=1

gpgkey=http://yum.puppetlabs.com/RPM-GPG-KEY-puppetlabs
```

```
metadata_expire=0

cost=500

bash-4.2# yum install puppet-enterprise

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages

groups-repo                                | 1.1 kB     00:00 ...

localdb                                    |  951 B     00:00 ...

patching                                   |  951 B     00:00 ...

puppet                                     |  951 B     00:00 ...

thirdparty                                 |  951 B     00:00 ...

Setting up Install Process

Resolving Dependencies

--> Running transaction check

---> Package puppet-enterprise.x86_64 0:3.7.1.rc2.6.g6cdc186-1.pe.nxos will be installed

--> Finished Dependency Resolution

Dependencies Resolved

=================================================================================================

 Package             Arch     Version                              Repository     Size

=================================================================================================

Installing:

puppet-enterprise    x86_64    3.7.1.rc2.6.g6cdc186-1.pe.nxos    puppet          14 M

Transaction Summary

=================================================================================================

Install       1 Package

Total download size: 14 M

Installed size: 46 M

Is this ok [y/N]: y

Retrieving key from file:///bootflash/RPM-GPG-KEY-puppetlabs

Importing GPG key 0x4BD6EC30:

 Userid: "Puppet Labs Release Key (Puppet Labs Release Key) <info@puppetlabs.com>"

 From  : /bootflash/RPM-GPG-KEY-puppetlabs

Is this ok [y/N]: y

Downloading Packages:

Running Transaction Check

Running Transaction Test
```

```
Transaction Test Succeeded
Running Transaction
Warning! Standby is not ready. This can cause RPM database inconsistency.
If you are certain that standby is not booting up right now, you may proceed.
Do you wish to continue?
Is this ok [y/N]: y
Warning: RPMDB altered outside of yum.
Installing : puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64
                                                                          1/1
/sbin/ldconfig: /usr/lib/libboost_regex.so.1.49.0 is not a symbolic link
Installed:
puppet-enterprise.x86_64 0:3.7.1.rc2.6.g6cdc186-1.pe.nxos

Complete!
```

署名付き RPM の Repo への追加

手順

ステップ 1 署名付き **RPM** を **repo** ディレクトリにコピーします

ステップ 2 **create repo** が成功するために、対応するキーをインポートします

```
bash-4.2# ls
puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm  RPM-GPG-KEY-puppetlabs
bash-4.2#
bash-4.2# rpm --import RPM-GPG-KEY-puppetlabs
bash-4.2# createrepo .
1/1 - puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm
Saving Primary metadata
Saving file lists metadata
Saving other metadata
bash-4.2#
```

キーをインポートしない場合

```
bash-4.2# ls
puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm  RPM-GPG-KEY-puppetlabs
bash-4.2#
bash-4.2# createrepo .
warning: rpmts_HdrFromFdno: Header V4 RSA/SHA1 signature: NOKEY, key ID 4bd6ec30

Error opening package - puppet-enterprise-3.7.1.rc2.6.g6cdc186-1.pe.nxos.x86_64.rpm

Saving Primary metadata
Saving file lists metadata
Saving other metadata
```

ステップ 3 **repo config ファイルを /etc/yum/repos.d の下に作成し、この repo を指定します**

```
bash-4.2# cat /etc/yum/repos.d/puppet.repo
[puppet]
name=Puppet RPM
baseurl=file:///bootflash/puppet
enabled=1
gpgcheck=1
gpgkey=file:///bootflash/puppet/RPM-GPG-KEY-puppetlabs
#gpgkey=http://yum.puppetlabs.com/RPM-GPG-KEY-puppetlabs
metadata_expire=0
cost=500

bash-4.2# yum list available puppet-enterprise -q
Available Packages
puppet-enterprise.x86_64        3.7.1.rc2.6.g6cdc186-1.pe.nxos
                      puppet
bash-4.2#
```

サードパーティ RPM の永続化

次に、サードパーティ RPM の永続化の論理を示します。

- ローカルの **yum** リポジトリは、サードパーティの RPM の維持を目的としたものです。/etc/yum/repos.d/thirdparty.repo は /bootflash/.rpmstore/thirdparty をポイントしています。
- **yum install third-party.rpm** コマンドを入力するたびに、RPM のコピーが //bootflash/.rpmstore/thirdparty に保存されます。
- 再起動時、サードパーティ リポジトリ内のすべての RPM がスイッチに再インストールされます。
- /etc 設定ファイル内のすべての変更は、/bootflash/.rpmstore/config/etc で維持され、再起動時にこれらが /etc で再生されます。
- /etcディレクトリで作成されたスクリプトは、リロード後も維持されています。たとえば、サードパーティ サービス スクリプトが /etc/init.d/ で作成され、リロード中にアプリケーションを起動するために使用されます。

VSH からの RPM のインストール

パッケージの追加

NX-OS 機能 RPM は、VSH CLI を使用してインストールすることもできます。

手順

	コマンドまたはアクション	目的
ステップ 1	**show install package**	既存のインストール済みのバージョンとパッケージを表示します。
ステップ 2	**install add ?**	サポートされてる URI を確認します。
ステップ 3	**install add***rpm-packagename*	**install add** コマンドは、パッケージ ファイルをローカル ストレージ デバイスまたはネットワーク サーバにコピーします。

次の例は、Chef RPM をアクティブ化する方法を示しています。

```
switch# show install package
switch# install add ?
WORD        Package name
bootflash:  Enter package uri
ftp:        Enter package uri
http:       Enter package uri
modflash:   Enter package uri
scp:        Enter package uri
sftp:       Enter package uri
tftp:       Enter package uri
usb1:       Enter package uri
usb2:       Enter package uri
volatile:   Enter package uri
switch# install add
bootflash:chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5.x86_64.rpm
[####################] 100%
Install operation 314 completed successfully at Thu Aug  6 12:58:22 2015
```

次の作業

すでにパッケージをアクティブ化している場合は、パッケージのアクティブ化, （9 ページ）に移動します。

（注） RPM パッケージの追加とアクティブ化は、1 つのコマンドで行うことができます。

```
switch#
install add bootflash:chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5.x86_64.rpm
 activate
```

パッケージのアクティブ化

はじめる前に

RPM は、先に追加しておく必要があります。

手順

	コマンドまたはアクション	目的
ステップ 1	**show install inactive**	追加され、アクティブ化されていないパッケージのリストを表示します。
ステップ 2	**install activate***rpm-packagename*	パッケージをアクティブ化します。

次の例は、パッケージをアクティブ化する方法を示しています。

```
switch# show install inactive
Boot image:
        NXOS Image: bootflash:///yumcli6.bin

Inactive Packages:
        sysinfo-1.0.0-7.0.3.x86_64
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
Available Packages
chef.x86_64       12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5 thirdparty
eigrp.lib32_n9000 1.0.0-r0                                            groups-rep
o
sysinfo.x86_64    1.0.0-7.0.3                                         patching
switch# install activate chef-12.0-1.el5.x86_64.rpm
[####################] 100%
Install operation completed successfully at Thu Aug  6 12:46:53 2015
```

パッケージの非アクティブ化

手順

	コマンドまたはアクション	目的
ステップ 1	**install deactivate***package-name*	RPM パッケージを非アクティブ化します。

次の例は、Chef RPM パッケージを非アクティブ化する方法を示しています。

```
switch# install deactivate chef
```

パッケージの削除

手順

	コマンドまたはアクション	目的
ステップ 1	**install remove***package-name*	RPM パッケージを削除します。

次の例は、Chef RPM パッケージを削除する方法を示しています。

```
switch# install remove chef-12.0-1.el5.x86_64.rpm
```

インストール済みパッケージの表示

手順

	コマンドまたはアクション	目的
ステップ 1	**show install packages**	インストール済みパッケージの一覧が表示されます。

次の例は、インストール済みパッケージの一覧を表示する方法を示しています。

```
switch# show install packages
```

詳細ログの表示

手順

	コマンドまたはアクション	目的
ステップ 1	**show tech-support install**	詳細なログを表示します。

次の例は、詳細なログを表示する方法を示しています。

```
switch# show tech-support install
```

パッケージのアップグレード

手順

	コマンドまたはアクション	目的
ステップ 1	install add *package-name* activate upgrade	パッケージをアップグレードします。

次の例は、パッケージをアップグレードする方法を示しています。

```
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate ?
downgrade  Downgrade package
forced     Non-interactive
upgrade    Upgrade package
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate upgrade
[####################] 100%
Install operation completed successfully at Thu Aug  6 12:46:53 2015
```

パッケージのダウングレード

手順

	コマンドまたはアクション	目的
ステップ 1	install add *package-name* activate downgrade	パッケージをダウングレードします。

次の例は、パッケージをダウングレードする方法を示しています。

```
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate ?
downgrade  Downgrade package
forced     Non-interactive
upgrade    Upgrade package
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate downgrade
[####################] 100%
Install operation completed successfully at Thu Aug  6 12:46:53 2015
```

サードパーティ製アプリケーション

NX-OS

他のサードパーティ製アプリケーション向けの Cisco NX-OS リポジトリについては、次を参照してください。 https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64/

NX-API REST API オブジェクト モデルの仕様については、次を参照してください。 https://opennxos.cisco.com/public/api/nxapi-rest/

DevOps 構成管理ツール

DevOps 構成管理ツールについては、以下のリンクを参照してください。

- Ansible 2.0 リリース（Nexus サポート）については http://releases.ansible.com/ansible/
- Ansible NX-OS モジュールの例については https://github.com/jedelman8/nxos-ansible
- Puppet については https://forge.puppetlabs.com/puppetlabs/ciscopuppet
- Cisco Puppet モジュール（Git）については https://github.com/cisco/cisco-network-puppet-module
- Chef については https://supermarket.chef.io/cookbooks/cisco-cookbook
- Cisco Chef クックブック（Git）については https://github.com/cisco/cisco-network-chef-cookbook

V9K（自動化アプリケーション統合テストとモデルング）

仮想 Nexus 9K にアクセスするには、get-n9kv@cisco.com に電子メールを送信し、CCO id を指定します。ユーザには、ダウンロード リンクが記載された電子メールが送信されます。

これにより、ESX5.1/5.5、VirtualBox、Fusion、KVM のサポートが提供されます。

自動化ツールの教育用コンテンツ

Open NX-OS アーキテクチャと自動化に関する無料のマニアルについては、次を参照してください。 http://www.cisco.com/c/dam/en/us/td/docs/switches/datacenter/nexus9000/sw/open_nxos/programmability/guide/Programmability_Open_NX-OS.pdf

collectd

collectd は、システム パフォーマンスの統計情報を定期的に収集し、さまざまな方法（たとえば、RRD ファイルなど）で値を保存する機能を提供するデーモンです。

collectd では、自身を稼働しているシステムに関する統計情報を収集し、この情報を保存します。これらの統計情報は、現在のパフォーマンスのボトルネックを特定し（パフォーマンス分析など）、将来的なシステム負荷（キャパシティ プランニングなど）を予測するために使用できます。

詳細については、https://collectd.orgを参照してください。

Ganglia

Ganglia は、クラスタやグリッドのようなハイパフォーマンス コンピューティング システムに対応する拡張性の高い分散監視システムです。これは、クラスタ間の連携を目的とした階層型設計に基づいています。これは、XML（データ表現）、XDR（コンパクトでポータブルなデータ伝送）、RRDtool（データ ストレージと仮想化）など、広く普及しているテクノロジーを活用します。これは、ノードごとのオーバーヘッドをきわめて小さくし、高い同時接続を実現するために、綿密に設計されたデータ構造とアルゴリズムを使用します。実装は堅牢であり、すでに広範なオペレーティングシステムおよびプロセッサアーキテクチャに移植されています。現在、世界中の数千台のクラスタで使用されています。大学のキャンパス全体および世界中のクラスタのリンクに使用されており、2000 ノードでクラスタを処理するように拡張できます。

詳細については、http://ganglia.infoを参照してください。

Iperf

Iperf は、TCP および UDP の帯域幅の最大パフォーマンスを測定するための最新の代替方式として、NLANR/DAST が開発したものです。Iperf を使用することで、さまざまなパラメータと UDP 特性の調整ができます。Iperf は、帯域幅、遅延ジッター、データグラム損失を報告します。

詳細については、http://sourceforge.net/projects/iperf/またはhttp://iperf.sourceforge.netを参照してください。

LLDP

LLDP は、EDP や CDP などの独占的なリンク層プロトコルの代替となるために設計された業界標準のプロトコルです。LLDP の目的は、リンク層通知を隣接ネットワーク デバイスに配信するメカニズムとしてベンダー間で互換性のある方式を提供することです。

詳細については、https://vincentbernat.github.io/lldpd/index.htmlを参照してください。

Nagios

Nagios は、ネットワーク サービス（ICMP、SNMP、SSH、FTP、HTTP など）やホスト リソース（CPU 負荷、ディスク使用状況、システム ログなど）のモニタリング、そして警告サービス（サーバ、スイッチ、アプリケーション、サービスを対象）を提供するオープン ソース ソフトウェアです。Nagios は、Nagios remote plugin executor（NRPE）と SSH または SSL トンネルを通じたリモート モニタリングを提供します。

詳細については、https://www.nagios.org/を参照してください。

OpenSSH

OpenSSH は、インターネットの技術ユーザに信頼されている無料の SSH 接続ツールです。Telnet、rlogin、FTP では、パスワードが暗号化されずにインターネット経由で送信されますが、これらのユーザはこのことを認識していないことがあります。OpenSSH では、すべてのトラフィック（パスワードも含む）が暗号化され、盗聴や接続のハイジャックなどの攻撃が効果的に排除されます。また、OpenSSH は、セキュア トンネリング機能と複数の認証方式を提供し、すべての SSH プロトコル バージョンをサポートしています。

詳細については、http://www.openssh.comを参照してください。

Quagga

Quagga は、さまざまなルーティング プロトコルの実装を提供するネットワーク ルーティング ソフトウェア スイートです。Quagga デーモンは、ネットワーク アクセス可能な CLI（「vty」）を使用して設定されます。

（注） Quagga BGP のみ検証が完了しています。

詳細については、http://www.nongnu.org/quagga/を参照してください。

Splunk

Splunk は、データの収集、分析、およびモニタリングを行う Web ベースのツールです。Splunk Enterprise は、マシン生成データからの重要な運用インテリジェンスの取得に役立ちます。これには、強力な検索機能、視覚化機能、使用例のための事前にパッケージ化されたコンテンツなどが幅広く付属しており、すべてのユーザが迅速に情報を把握して共有できます。未加工のデータは、Splunk ユニバーサル フォワーダを使用して Splunk サーバに送信されます。ユニバーサル フォワーダは、リモート ソースから信頼性の高いセキュアなデータ収集を提供し、インデックス化と統合のためにそのデータを Splunk Enterprise に転送します。たとえば、数万台のリモート システムに拡張し、パフォーマンスへの影響を最小限を抑えながらテラバイト単位のデータを収集できます。

詳細については、http://www.splunk.com/en_us/download/universal-forwarder.htmlを参照してください。

tcollector

tcollector は、ローカル コレクタからデータを収集し、そのデータを OpenTSDB にプッシュするクライアント サイド プロセスです。すべてのホストで実行すれば、各ホストのデータを TSD に送信できます。

詳細については、http://opentsdb.net/docs/build/html/user_guide/utilities/tcollector.htmlを参照してください。

tcpdump

Tcpdump は、ネットワーク インターフェイス上のブール式に一致するパケットの内容説明を印刷します。説明の前にはタイムスタンプが付けられ、デフォルトでは、午前零時を起点とした時間、分、秒、秒数の小数第一位が印刷されます。また、-w フラグ付きで実行されると、後で分析できるように、パケット データがファイルに保存されます。-r フラグ付きで実行されると、パケットがネットワークインターフェイスから読み出されるのではなく、保存されたパケットファイルから読み出されます。さらに、-V フラグ付きで実行されると、保存されたパケットファイルの一覧が読み出されます。いずれの場合も、式に一致するパケットのみが tcpdump コマンドによって処理されます。

詳細については、http://www.tcpdump.org/manpages/tcpdump.1.htmlを参照してください。

Tshark

Tshark は、ネットワーク プロトコル アナライザです。これにより、稼働中のネットワークからパケットデータをキャプチャしたり、以前に保存したキャプチャ ファイルからパケットを読み出すことができます。復号化された形式のこれらのパケットを標準出力に印字するか、パケットをファイルに書き込むことができます。Tshark のネイティブのキャプチャ ファイル形式は pcap 形式です。これは、**tcpdump** やその他のさまざまなツールで使用されている形式でもあります。Tshark は、cap_net_admin ファイル機能を削除した後に、ゲスト シェル 2.1 内で使用できます。

```
setcap
 cap_net_raw=ep /sbin/dumpcap
```

（注） このコマンドは、ゲスト シェル内で実行する必要があります。

詳細については、https://www.wireshark.org/docs/man-pages/tshark.htmlを参照してください。