CHAPTER 1

概要

この章では、Cisco Nexus 3000 シリーズ スイッチで Python アプリケーション プログラミング インターフェイス（API）サポート機能を使用するための概要と、必要なインストール情報を示します。

この章は、次の内容で構成されています。

Python API の概要

Python は簡単に習得できる強力なプログラミング言語です。効率的で高水準なデータ構造を持ち、オブジェクト指向プログラミングに対してシンプルで効果的なアプローチを取っています。Python は、簡潔な構文、動的な型付け、およびインタープリタ型という性質によって、ほとんどのプラットフォームのさまざまな分野でスクリプティングと高速なアプリケーション開発を実現する理想的な言語です。

Python のインタープリタと広範な標準ライブラリは、Python Web サイトから、主要なプラットフォームに対応したソース形式またはバイナリ形式で自由に利用できます。

http://www.python.org/

また、このサイトには、サードパーティが無償で提供している多数の Python モジュール、プログラム、およびツールのディストリビューションとそれらへのリンク、さらに追加のドキュメンテーションが掲載されています。

Cisco Nexus 3000 シリーズ スイッチは、Python v2.7.2 で使用できるすべての機能をサポートします。

Cisco Nexus 3000 シリーズ スイッチの Python スクリプティング機能では、次の作業を行うことができます。

- スイッチ起動時に設定を確認するためのスクリプトを実行する。
- 設定をバックアップする。
- バッファ使用状況の特性をモニタし、対応することで、予防的な輻輳管理を行う。
- 電源投入時の自動プロビジョニングまたは EEM モジュールとの統合。
- 一定の時間間隔でジョブを実行する（ポートの自動定義など）。
- スイッチのコマンドライン インターフェイス（CLI）にプログラムからアクセスしてさまざまなタスクを実行する。

Python のインストール

Python インタープリタは Cisco NX-OS ソフトウェアにデフォルトで搭載されています。Python は **python** コマンドを入力することで起動できます。また、**import cisco.py** コマンドで cisco.py モジュールをインポートすることで、Cisco NX-OS API にアクセスするスクリプトを作成できます。

サードパーティ製の純粋な Python パッケージのインストール

mypkg.tgz をサーバにコピーすることで、サードパーティ製の純粋な Python パッケージをインストールできます。サードパーティ製パッケージを抽出してインストールするには、次の手順を実行します。

- **copy scp://user@server/path/to/mypkg.tgz bootflash:mypkg.tgz vrf management** コマンドを実行して、tar ファイルのセキュア コピーを行います。
- **tar extract bootflash:mypkg.tgz** コマンドを使用して、mypkg.tgz ファイルを展開します。
- **move bootflash:mypkg-1.2/* bootflash:** コマンドを使用して、抽出したファイルをブートフラッシュに移動します。
- **python setup.py instal** コマンドを使用して、パッケージをインストールできます。
- コピーしたファイルをブートフラッシュから削除します。
- サードパーティ製パッケージは、スクリプトまたは Python シェルで使用できます。

```
switch# python
>>> import mypkg
```

（注）　将来のリリースでは、**easy_install** コマンドを使用して、サードパーティ製パッケージをインストールできるようになります。

Python の使用

ここでは、パラメータを渡すことによって Python スクリプトを作成し、実行する方法について説明します。内容は次のとおりです。

Python シェルの開始

Python シェルは、パラメータを指定せずに **python** コマンドを使用することで開始できます。

```
switch# python
Python 2.7.2 (default, Oct 11 2011, 13:55:49)
[GCC 3.4.3 (MontaVista 3.4.3-25.0.143.0800417 2008-02-22)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Loaded cisco NX-OS lib!
>>> print 'helo world!'
helo world!
>>>exit()
switch#
```

スクリプトの実行

Python スクリプトは、**python** <*filename*> コマンドを使用して実行することができます。

```
switch# python test.py
['/bootflash/test.py']
doing 0/1
doing 0/2
doing 1/2
switch#
```

スクリプトへのパラメータの引き渡し

Python スクリプトは、**python** <*filename*> [*arg1, arg2, arg3,.....*] コマンドを使用して実行することができます。

```
switch# python test.py foo bar 1 2
['/bootflash/test.py', 'foo', 'bar', '1', '2']
doing 0/1
doing 0/2
doing 1/2
switch#
```

Embedded Event Manager のサポート

Embedded Event Manager（EEM）は、イベントに基づいた Python スクリプトの呼び出しをサポートします。**variable $** コマンドを使用して、イベントの syslog を Python スクリプトに渡すことができます。

次の例は、IF_DOWN イベントに対して Python スクリプトを呼び出す EEM を示しています。

EEM の設定:

```
switch(config)# event manager applet if-mon
switch(config-applet)# event syslog pattern "*IF_DOWN.*"
Configuration accepted successfully
switch(config-applet)# action 1.0 cli python if-mon.py eth1/1 $command
switch(config-applet)# end
```

Python スクリプト:

```
If-mon.py

import re
import sys

def findIf ():
    x = re.compile ('[Ee]thernet\d+\/\d+')
    for a in sys.argv[1:]:
        if x.match (a):
            print a
            return a
    return None

print 'Starting my script.. args:'
print sys.argv
intf = findIf ()
if not intf:
    intf = 'eth1/1'

print 'Detected shut on interface %s' % intf
```

```
from cisco import *
s, o = cli ('show run int %s' % intf)
print ('----------------------\n%s\n' % o)
print 'Restoring interface %s' % intf
s, o = cfg_if (intf, desc='++dont shut++', state='up')
print ('----------------------\n%s\n' % o)
s, o = cli ('show run int %s' % intf)
print ('sh ver\n----------------------\n%s\n' % o)

print ('\nbye\n')
```