

Заметки кодера. Отладка скриптов под Windows.

Введение

Сразу скажу. Полноценно нок под Windows не запустить. Это связано с переходом на [SupervisorD](#), он не поддерживает Windows и, в планах, о реализации работы под Windows не упоминается.

Если нужен полноценный НОК по Windows - поставьте Linux/FreeBSD.... Нет, я серьёзно! Поддержка виртуализации, docker контейнеров позволяет не заниматься извращениями в обозримой перспективе. К тому же, не конвенциональная политика Microsoft всё равно заставит переходить на Linux.

Ну или пользоваться старым добрым CygWin - работоспособность SupervisorD под него заявлена ([Руководство1](#), [Руководство2](#)), плюс, не возникнет нижеизложенных проблем...

Зачем?

Не все пылают трепетной и нежной любовью к Linux, да и обстоятельства, зачастую, складываются так, что инженеру приходится работать с чем дадут. Да и удобно это - отлаживать скрипты на рабочем месте. К тому же PyCharm Professional, который поддерживает удалённую отладку, платен.... а возможность использовать PyCharm Community выглядит соблазнительной.

Почему?

Стандартным ответом будет - почему бы и нет. Ибо Python предназначен писать переносимые программы! т.е. не должно быть проблем при работе под разными ОС... данное обстоятельство и, позволяет, нам рассчитывать на успех сего мероприятия 😊 Ну, го.

Требования и Ограничения

Список ограничений будет примерный. Будет указано почему ограничение актуально. Вдруг, кому-то, удастся его преодолеть:

1. Невозможно запускать сервисы (процессы) НОКа. Поддерживается отладка только скриптов (профилей) оборудования.
2. Не работает SSH доступ к железкам! (только **SNMP** и **TELNET**). Связано это с пакетом **pylibssh2** - он не собирается под Windows....
Подробности в разделе [проблемы](#).
3. Невозможно использовать Virtualenv. Только системный Python. Подробности в разделе Проблемы.
4. Возможно, не работает ещё что-то...

Требования

1. Нам потребуется Python. Тестировались под последней версии 2.7. Подходит и Active Python и обычный.
2. PyCharm Community Edition - ради его возможности оладки всё и затевалось. (если у вас есть Professional, рекомендую забыть на всё это и пользоваться режимом удалённой отладки).
3. Терпение, терпение и ещё раз терпение.
4. Желание чтобы это заработало.



Нок постоянно дописывается. Поэтому, не исключена ситуация когда что-то не заработает. Если такое случилось и инструкция уже не актуальна, можно:

- Поправить проблему самому
- Откатиться на версию без проблемы
- Написать составителю статьи)

Писать разработчикам смысла нет. Т.к. конфигурация является не поддерживаемой.

Установка необходимого ПО.

Python

Подойдёт версия 2.7 Либо [ActivePython](#), или обычный [Python](#) необходима версия x86.

PyCharm

Скачиваем и устанавливаем [PyCharm Community Edition](#).

Microsoft Compiler for Python

Урезанный компилятор от Microsoft для сборки пакетов по Python из исходных кодов. Скачиваем [тут](#). После установки (устанавливается он в "") необходимо добавить его в путь поиска Python, иначе, при попытке собрать пакте, будет выдаваться сообщение: "unable to find vcvarsall.bat". Для этого в файле "<python_folder>\Lib\distutils\msvc9compiler.py" находим метод "def find_vcvarsall(version):" и в конце, перед строчкой "if not productdir:", добавляем "productdir= "C:/Users/<CurrentUser>/AppData/Local/Programs/Common/Microsoft/Visual C++ for Python/9.0" (где <CurrentUser> необходимо заменить на текущего пользователя, под которым происходила установка компилятора)

```
if not productdir or not os.path.isdir(productdir):
    toolskey = "VS%0.f0COMNTOOLS" % version
    toolsdir = os.environ.get(toolskey, None)
    if toolsdir and os.path.isdir(toolsdir):
        productdir = os.path.join(toolsdir, os.pardir, os.pardir, "VC")
        productdir = os.path.abspath(productdir)
        if not os.path.isdir(productdir):
            log.debug("%s is not a valid directory" % productdir)
            return None
    else:
        log.debug("Env var %s is not set or invalid" % toolskey)
productdir= "C:/Users/Andrey/AppData/Local/Programs/Common/Microsoft/Visual C++ for Python/9.0"
if not productdir:
    log.debug("No productdir found")
    return None
vcvarsall = os.path.join(productdir, "vcvarsall.bat")
if os.path.isfile(vcvarsall):
    return vcvarsall
```

Исходный код НОКа

Скачиваем с сайта [НОС Project](#). Либо используем команду "hg clone <https://bitbucket.org/nocproject/noc> нос". Распаковывать нос, необходимо на том-же диске, на котором находится python. После распаковки НОКа необходимо скопировать с рабочего экземпляра папку с настройками (etc /noc.yml, etc/noc.conf) в папку с НОКом.



Распаковывать нос, необходимо на том-же диске, на котором находится python!



Папка, в которую распакован/клонирован НОС должна называться **нос**. Иначе не заработает.

Requirements

Требования НОС'а в части модулей Python'а. Установка, практически, не доставляет проблем. Список пакетов находится в файлике "node.txt", папке requirements НОКа. Но, некоторые пакеты придётся установить вручную:

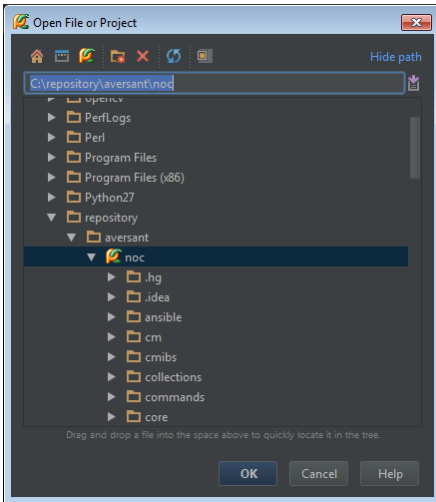
1. Копируем файл <нос_dir>\requirements\node.txt в какую-нибудь папку. Переименовываем его в node-windows.txt.
2. Исключаем из него следующие пакеты (удаляем строчки с ними):
 - atomiclong==0.1.1 - не компилируется под Windows
 - pyproj==1.9.4
 - нос-pkg-pyclips==1.0.7
 - numpy==1.9.2
3. После этого даём команду "pip -r node-windows.txt"
4. Доустанавливаем пакеты pyproj и numpy командами "pip install pyproj" и "pip install numpy"
5. Если возникли ошибки, то ищем проблемные пакеты [здесь](#), скачиваем и устанавливаем вручную.

Пакеты atomiclong и нос-pkg-pyclips для отладки скриптов не требуются, поэтому, их отсутствие на работоспособности не скажется.

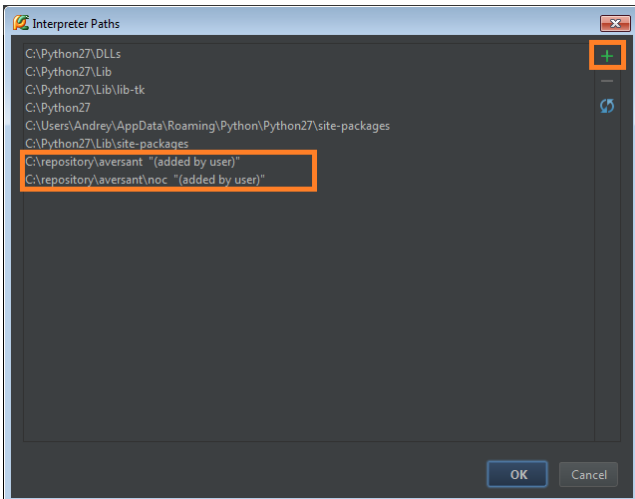
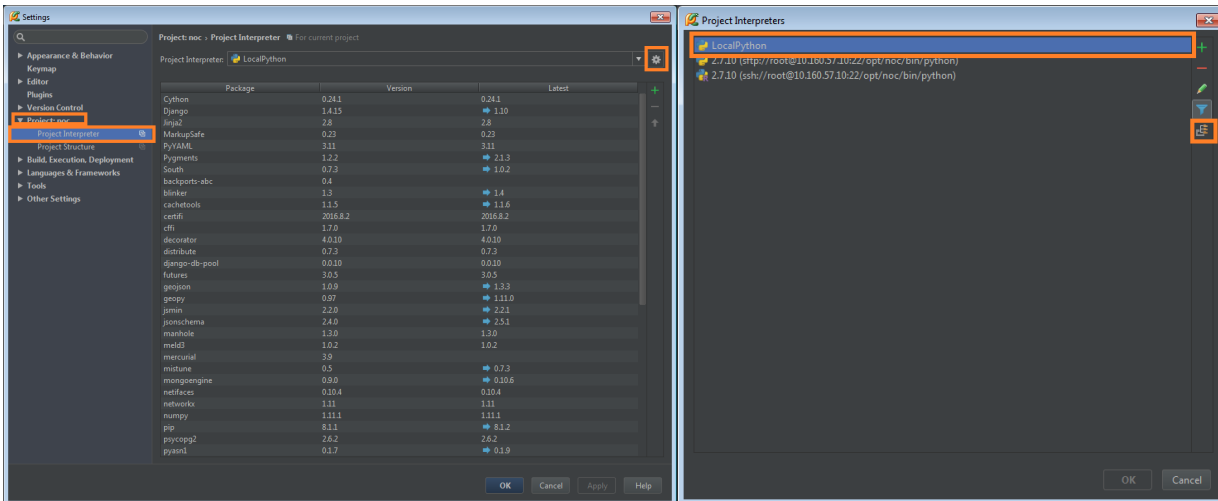
```
>c:
> mkdir repository
> cd repository
> hg clone https://bitbucket.org/nocproject/noc noc
> copy noc\requirements\node.txt noc\requirements\node-windows.txt
> pip install -r noc\requirements\node-windows.txt
> pip install pyimport numpy pyproj
```

Настройка PyCharm

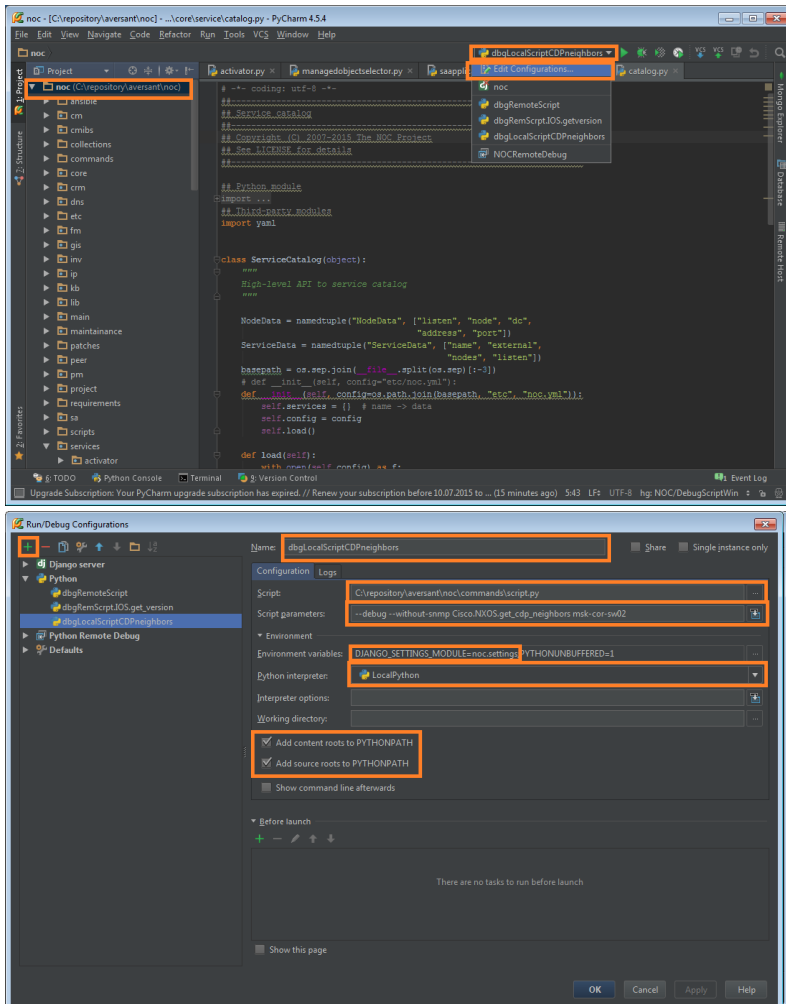
Открываем проект с НОКом (н-р он лежит в папке <нос_path> ("C:\repository\aversant\noc")).



Добавляем пути к НОКУ в настройку Python. Идём в **Settings**(Ctrl+Alt+S) -> **Project:noc** -> **"Project Interpreter"**. Нажимаем на **шестерню справа** от выпадающего списка с интерпретаторами -> **больше (More)**. В списке выбираем локальный python и переходим в **настройку путей** (Path settings) (нижний значок на панели справа). Добавляем путь к папке с НОКом <нос_path> и путь к папке на один выше папки с НОКом. После, нажимаем везде **OK** до выхода в основное окно.



Добавляем скрипт для запуска. Для этого в основном окне проекта выбираем в выпадающем списке справа сверху "Edit configuration". В открывшемся окне, нажимаем кнопку плюс, слева и выбираем в списке Python. Настраиваем согласно скриншоту ниже. Придумываем имя. В пути (Path) указываем script.py - он запускается для отладки скриптов. В параметры вписываем необходимые (синтаксис совпадает с ./noc script". В переменные окружения (Environment Variables) обязательно добавляем строку "DJANGO_SETTINGS_MODULE" = "noc.conf". Интерпретатор выбираем локальный. Проставляем галочки внизу.



Далее скрипт можно запускать на выполнение, смотреть, какие проблемы возникают, и решать их.

Проблемы

В этом разделе я пошагово опишу попытки запуска и какие ошибки, при этом, возникают. В конце приведена ссылка на репозиторий с уже выполненными исправлениями, поэтому, можно пропустить этот раздел и переходить к [последней главе](#). И, если что-то не заработало, уже искать здесь причины. Если причину найти не удалось - напишите мне, я посмотрю.

Глобальная проблема запуска НОКа на Windows - это пути. В коде НОКа очень много мест, где указываются пути и прописаны они в UNIX формате. Т.е. на Windows работать не будут. Основная задача, как раз, связана с поиском этих мест и исправления ситуации. Исправлять можно 2 способами - заменять их на формат Windows. Использовать независимый формат (os.path).

Поскольку, я не владею Python'ом в совершенстве и не знаю ООП, просьба сильно не пинать за кривость подхода! будет лучше если высказать предложения, как сделать правильно. И мне будет полезно и кому-то ещё.

Не работает SSH. Ошибка (EXCEPTION: <type 'exceptions.ImportError'> Cannot load handler 'noc.core.script.cli.ssh.SSHCLI': No module named libssh2)

Проблема в libssh2, при компиляции всплывает ошибка: "libssh2.h(133) : error C2371: 'ssize_t' : redefinition; different basic types"

```
C:\repository\libssh2\include\libssh2\libssh2.h(133) : error C2371: 'ssize_t' : redefinition; different basic types
c:\python27\include\pyconfig.h(205) : see declaration of 'ssize_t'
```

error: command 'C:\\Users\\Andrey\\AppData\\Local\\Programs\\Common\\Microsoft\\Visual C++ for Python\\9.0\\VC\\Bin\\cl.exe' failed with exit status 2

Судя по ошибке, проблема в разнице типов данных. Есть патч для Python'a, но его отказались добавлять в апстрим: [Дискуссия1](#), [Дискуссия2](#).
Если получится победить проблему - просьба отписаться 😊

Не работает SNMP.

Для работы SNMP необходимо наличие компилятора Microsoft. Ошибка

Нет файла. Н-п IOError: [Errno 2] No such file or directory: 'etc/noc.yml'.

По этой проблеме будет несколько сообщений. Они будут касаться разных файлов. Я приведу список с исправлением, которое необходимо внести.

core/config/base.py

```
import os

class BaseConfig(object):
    CONFIG = os.path.join(os.sep.join(__file__.split(os.sep)[: -3]), "etc", "noc.yml")
```

ConfigParser.NoSectionError: No section: 'main'

Django не может прочитать настройки. Может быть вызвана 2 причинами:

1. Не задана переменная окружения DJANGO_SETTINGS_MODULE (см. пункт Настройка Pycharm)
2. Необходимо поправить пути в файле настроек Django (settings.py).

settings.py

```
# Load config
config = ConfigParser.SafeConfigParser()
basepath = os.sep.join(__file__.split(os.sep)[: -1])
config.read([os.path.join(basepath, "etc", "noc.defaults"), os.path.join(basepath, "etc", "noc.conf")])
```

AttributeError: 'module' object has no attribute 'getuid'

Связана с отсутствием метода getuid в Windows. Можно, просто убрать применение метода.

lib/debug.py

```
if 1 == 0:
    CP_SET_UID = os.stat("var/log")[stat.ST_UID]
```

ImportError: No module named ber

Данный модуль лежит в папке "noc\core\snmp", а ошибка связана с тем, что он подгружает откомпилированные модули (лежат в "speedup\ber"). Чтобы поправить ошибку необходим добавленный компилятор Microsoft для Python и добавить модуль "import pyximport" поправить файл "core/snmp/ber.py". Добавлять необходимо перед строчкой импорта "from noc.speedup.ber import parse_tlv_header, parse_p_oid".

core/snmp/ber.py

```
## Python modules
import math
import struct
## Third-party modules
import pyximport
pyximport.install()
```

AttributeError: 'module' object has no attribute 'link' (EXCEPTION: <type 'exceptions.IOError'> [Errno 2] No such file or directory: 'etc/noc.yml')

И снова где-то не правильный путь.... В конкретном случае он в файле "catalog.py" (указано под EXCEPTION). (25,26 строчки).

core/service/catalog.py

```
import os

class ServiceCatalog(object):
    basepath = os.sep.join(__file__.split(os.sep)[: -3])
    def __init__(self, config=os.path.join(basepath, "etc", "noc.yml")):
```

И в том же файле, но в 34, 35 строчках

core/service/catalog.py

```
def load(self):
    with open(self.config) as f:
        data = yaml.load(f)
    with open(os.path.join(self.basepath, "ansible", "config", "services.yml")) as f:
```

Failed to load script Cisco.IOS.get_version: No module named get_version. Failed to load script None

Ошибка связана с заданием "Working dir" [Настройка скрипта](#). Но вылечить можно поправив файл "core/script/loader.py", добавив определение относительного пути.

core/script/loader.py

```
basepath = os.sep.join(__file__.split(os.sep)[: -3])
if os.path.exists(
    os.path.join(
        basepath, "sa", "profiles", vendor, system,
        "%s.py" % sn
    )
):
```

Не запускается скрипт. Ошибка "ImportError: Import by filename is not supported."

Проверить, что указана переменная DJANGO_SETTINGS_MODULE, проверить что внесены все правки.

ВОЗМОЖНОСТИ

Преодолев все ошибки у нас появилась возможность отлаживать скрипты для оборудования не отходя от рабочего места:

