SecurityTube Python Scripting Expert (SPSE)

SecurityTube Python Scripting Expert

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 1: File Handling

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

File I/O in Python

- open(file_name, access_mode, buffering)
- read(byte_count)
- write(data)
- close()
- os.rename() os.delete()

Exercise

- Read /var/log/messages
- find all the logs in it which pertain to USB and print them out selectively

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 1: File Handling

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 2: Directory Navigation

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Directory Details

- Methods for traversing directories
- Listing files and their information
- creating and deleting directories + files
- test to check if something is a file or directory

Exercise

- Create a program which can recursively traverse directories and print the file listing in a hierarchical way

```
A
----a.txt
----b.txt
----B
---------c.out
```

Exercise

- For any given filename list out all the stats related to the file such as size, creation time, path etc.

Module 2: System Programming

SecurityTube Python Scripting Expert

End of Part 2: Directory Navigation

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 3: Process Creation

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Forking

- Cloning of a process
- Forking creates an identical process as the parent
- the thread of execution is duplicated exactly at the point of call to fork()
  - returns 0 in the child
  - returns pid of child in the parent
- PID is different for parent / child

Use of fork()

- Dedicate child to a task given by the parent
- parent and child can communicate if required using IPC
- parent / child binary remains the same

Spawning New Processes

- os.exec* functions
  - os.execl
  - os.execle
  - …

- Overlays parent process with the child

Module 2: System Programming

SecurityTube Python Scripting Expert

End of Part 3: Process Creation

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 2: System Programming

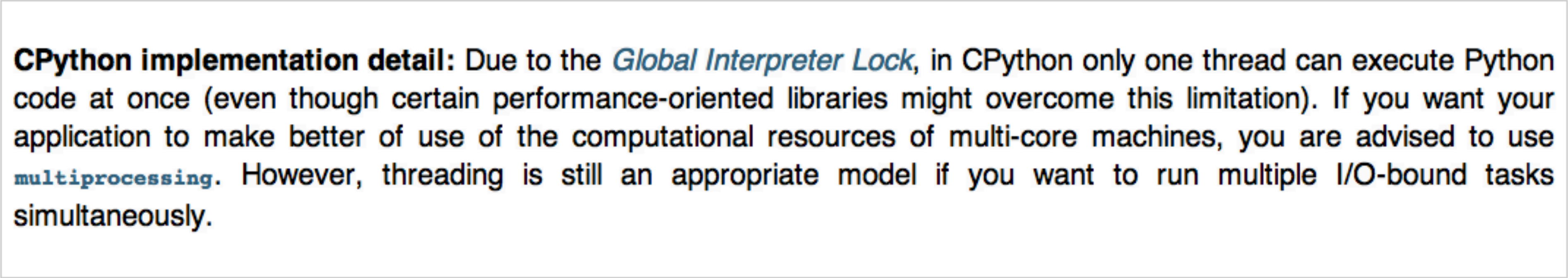
SecurityTube Python Scripting Expert

Part 4: Python Threads

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Global Interpreter Lock



CPython implementation detail: Due to the *Global Interpreter Lock*, in CPython only one thread can execute Python code at once (even though certain performance-oriented libraries might overcome this limitation). If you want your application to make better of use of the computational resources of multi-core machines, you are advised to use `multiprocessing`. However, threading is still an appropriate model if you want to run multiple I/O-bound tasks simultaneously.

Apart from I/O Tasks such as Network reads, Writing to disk etc. Python Threads are not too useful.

Threads in Python

- Simple threads using the thread module

- More complicated ones using the threading module

Exercise

- Based on the knowledge you have gained in the network programming module, create a multi-threaded port scanner in Python which uses SYN Scanning

Module 2: System Programming

SecurityTube Python Scripting Expert

End of Part 4: Threading

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 5: Threading and Queues

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Threading and Queues

- Create task queues
- Threads receive tasks
- Threads complete tasks and inform the queue
- All threads exit once queue is empty

Exercise

- Create a list of FTP sites
- Create a WorkerThread and Queue which can login to these sites and list the root directory and exit
- use 5 threads for this job and 10 FTP sites

Exercise: Threads and Locks

- There is a locking mechanism available in the Thread class which you can use to lock resources for dedicated use

- Create a sample code to illustrate this concept

Exercise: Multiprocessing

- Explore the multiprocessing module in Python
- How does it leverage multi-core setups?
- Program the TCP SYN scanner using multiprocessing

Module 2: System Programming

SecurityTube Python Scripting Expert

End of Part 5: Threading and Queues

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 6: Signals and IPC

http://www.securitytube.net
http://www.securitytube-training.com

Vivek Ramachandran
Course Instructor

Signals

- Allows handling of Asynchronous events
- SIGKILL is what gets sent when you use “kill -9”
- programming with Signals is easy

Exercise

- Create a TCP server which listens to a port
- Implement signals to ensure it automatically shuts down after a pre-configured duration, which is given via command line
- e.g. tcp-server –s 100

shutdown after listening to port for 100 seconds

Module 2: System Programming

SecurityTube Python Scripting Expert

End of Part 6: Signals and IPC

http://www.securitytube.net
http://www.securitytube-training.com

Vivek Ramachandran
Course Instructor

Module 2: System Programming

SecurityTube Python Scripting Expert

Part 7: Subprocess

http://www.securitytube.net
http://www.securitytube-training.com

Vivek Ramachandran
Course Instructor

subprocess

- subprocess.call([‘ps’, ‘aux’])

- subprocess.check_output([‘ls’, ‘-al’])

shell = False

Why is shell = True a security issue?

Advanced Usage

- Mapping of
  - STDIN
  - STDOUT
  - STDERR

- subprocess.Popen(…)

Module 2: System Programming

SecurityTube Python Scripting Expert

End of Part 7: Subprocess

http://www.securitytube.net
http://www.securitytube-training.com

Vivek Ramachandran
Course Instructor