## Enumeration and Exploitation

### Python 3 (Medium)

This challenge evaluates the contestant's ability to understand and analyze vulnerabilities in a compiled Python program. The uncompyle program can be used to convert the compiled program back into Python code. The result of running uncompyle can be seen below:

```
import sys

def main():
    if len(sys.argv) != 2:
        print 'Invalid args'
        return
    password = sys.argv[1]
    builder = 0
    for c in password:
        builder += ord(c)

    builder = builder << 2
    builder = ~builder
    builder = builder ^ 12648430
    builder = ~builder
    if builder == 12645638 and ord(password[0]) == 78 and len(password) == 11:
        print 'correct'
    else:
        print 'incorrect'


if __name__ == '__main__':
    main()
```

An analysis of the code reveals that the sum of the ASCII codes for the characters in the password will have a specific value after several transformations. Question 1 can be solved by hand by reversing the transformations. Below is a solution created with the aid of some JavaScript code.

```
> ~(12645638)
-12645639
> ~(12645638) ^ 12648430
-2793
> ~(~(12645638) ^ 12648430)
2792
> ~(~(12645638) ^ 12648430) >> 2
698
> 698 - 78
620
> 620 / 10
62
> String.fromCharCode(62)
'>'
> String.fromCharCode(78)
'N'
> 'N' + Array(10).join('>')
'N>>>>>>>>>'
```



| Question | Answer |
|---|---|
| **What is a secret key that will pass validation?** | N>>>>>>>>>> |

© 2017 Cyber Skyline. All Rights Reserved.
Unauthorized reproduction or distribution of this copyrighted work is illegal.