Robert Prast and Sean Coneys
Asis CTF Writeup
04/30/2018

Problem: Neighbour (62)

Description: For a given large integer **n**, find the **nearest perfect power** that is equal or smaller than **n**.

N is given once a string(X) is found where sha256(X)[-6:] = *Randomized hex code*

Overview:
The crux of the problem revolved around finding the nearest perfect square(e.g 4, 9, 16...etc) to a large given number. In its simplest form, a perfect power is simply m^k = n, where m and k have to be positive integers and therefore n will be a natural number. However instead of focusing on just one exponent, our search would have to be over a series of different exponents and bases.

Beginning Steps:
Upon connecting to the server, you are presented with the following output.

```
penguin@penguin:~/Desktop$ nc 37.139.22.174 11740
******************************************************************************
| Welcome to the Near Neighbor Problem, only mathematicians know the flag!! |
| Your mission is to find smallest natural number r for given n, such that  |
| there exist integers x, y: n - x**y = r > 0, x > 1 and y > 1. good luck!! |
******************************************************************************
Submit a printable string X, such that sha256(X)[-6:] = d37879
```

From this we can extract our formula, which is n-x**y=r, where r>0, x,y>1, and our goal is to find the smallest possible value for r. However, in order to get a value for N and begin the optimization problem, we must first tackle finding a string, such that sha256(x)[-6:]="series of psuedo random hex bytes" (always 6 long). We decided on going for a simple brute force approach. Essentially, we designed a for loop to test a large number of values. Each iteration of the loop would take the iteration number, and then take the sha256 hash of that i'th number (as a string). Then there is a simple comparison between the last 6 digits of the i'th hash digest and the given digest on the server. Since there are many solutions due the short length, we simply break the for loop after the first value is found. This gave us the output of 268684. After the server spits back a large value for a n. Allowing us the begin the hard math.

```python
import hashlib
for i in range(100000000):
    m=hashlib.sha256()
    m.update(str(i))
    code=m.hexdigest()
    if(code[-6:]=="d37879"):
        print("HERE")
        print(i)
        x=rawinput()
```

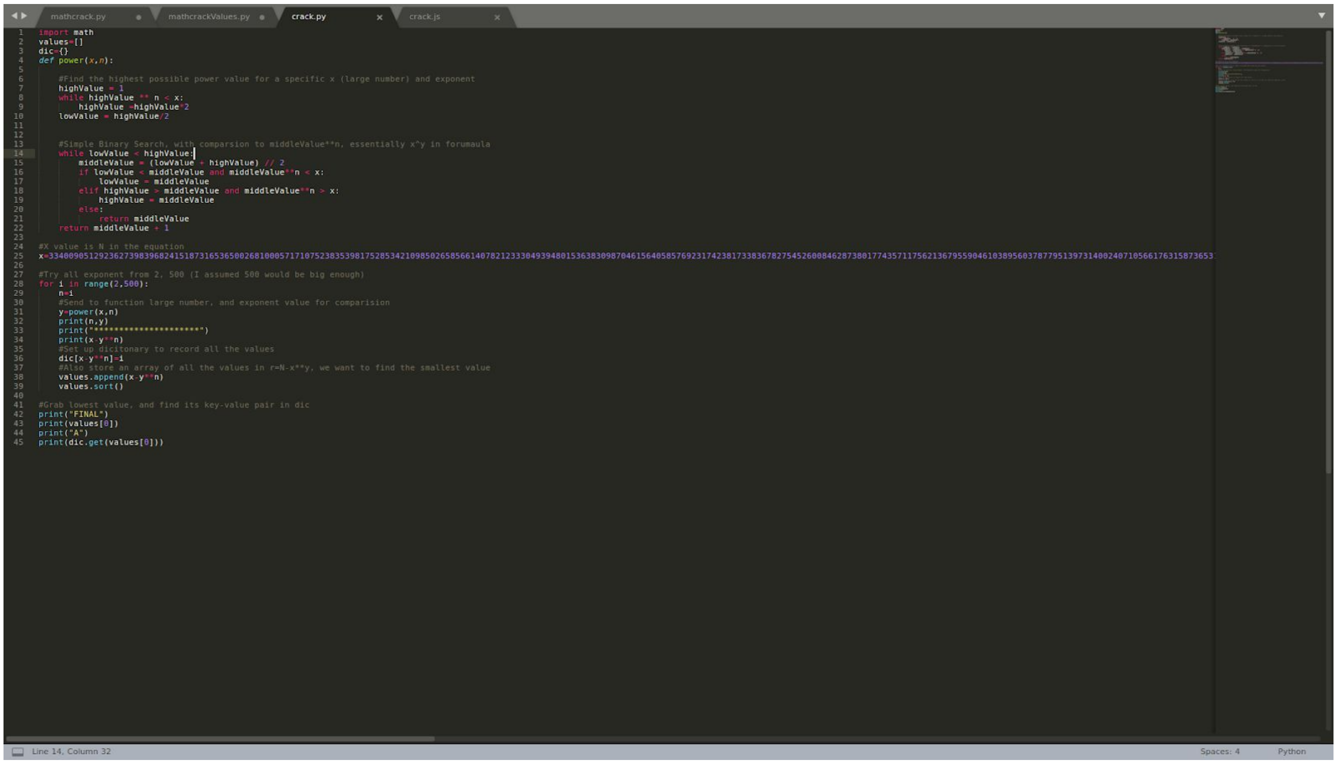
Part 2:

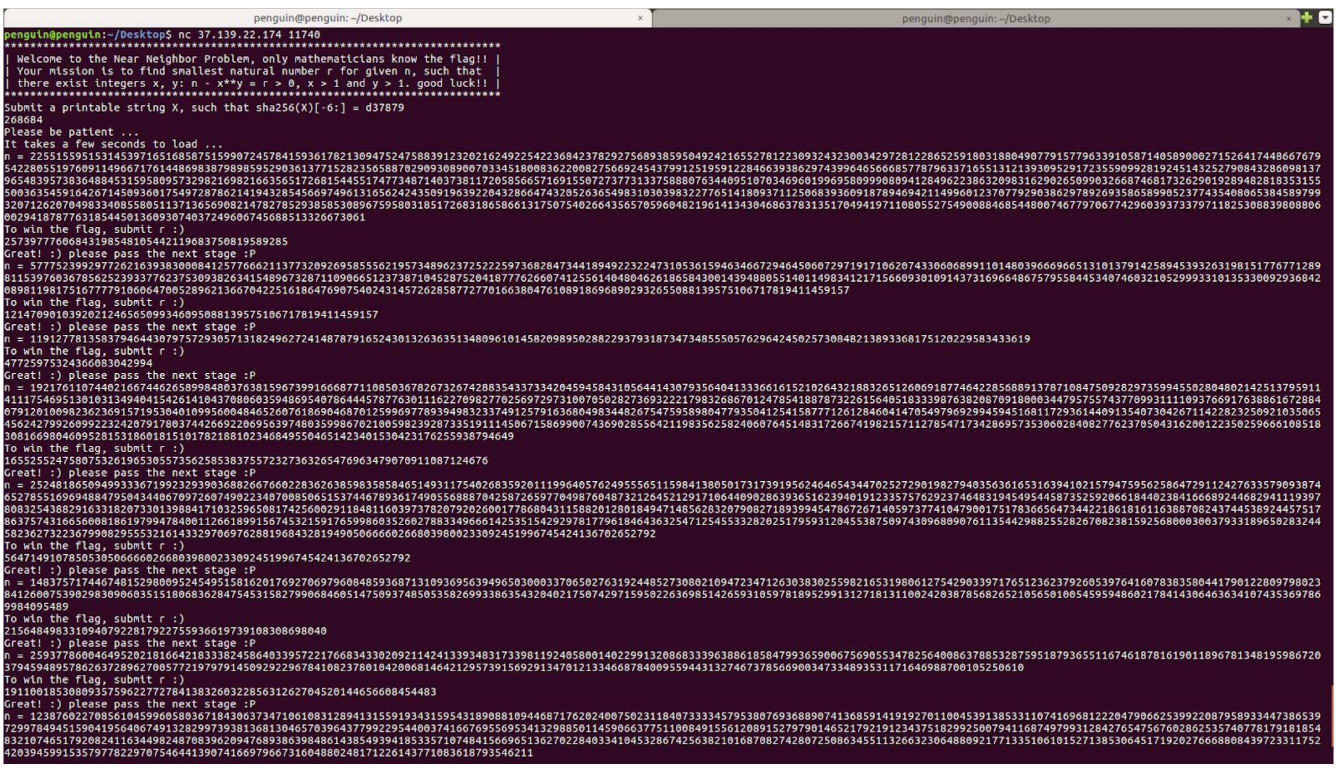
At first we planned on just doing a simple brute force, with some logical parameters. Since the given n was upwards of 100 digits, (we even got one that 500+ digits), we erroneously assumed the value for y (the exponent) would also have to be relatively large. Therefore, we wrote a quick for loop to have a set x (base) and then an ever increasing exponent value. We would then look at N-x**y and quickly saw that this method had some fatal errors. For starters, the base would also have to change. This can be seen with the following example:

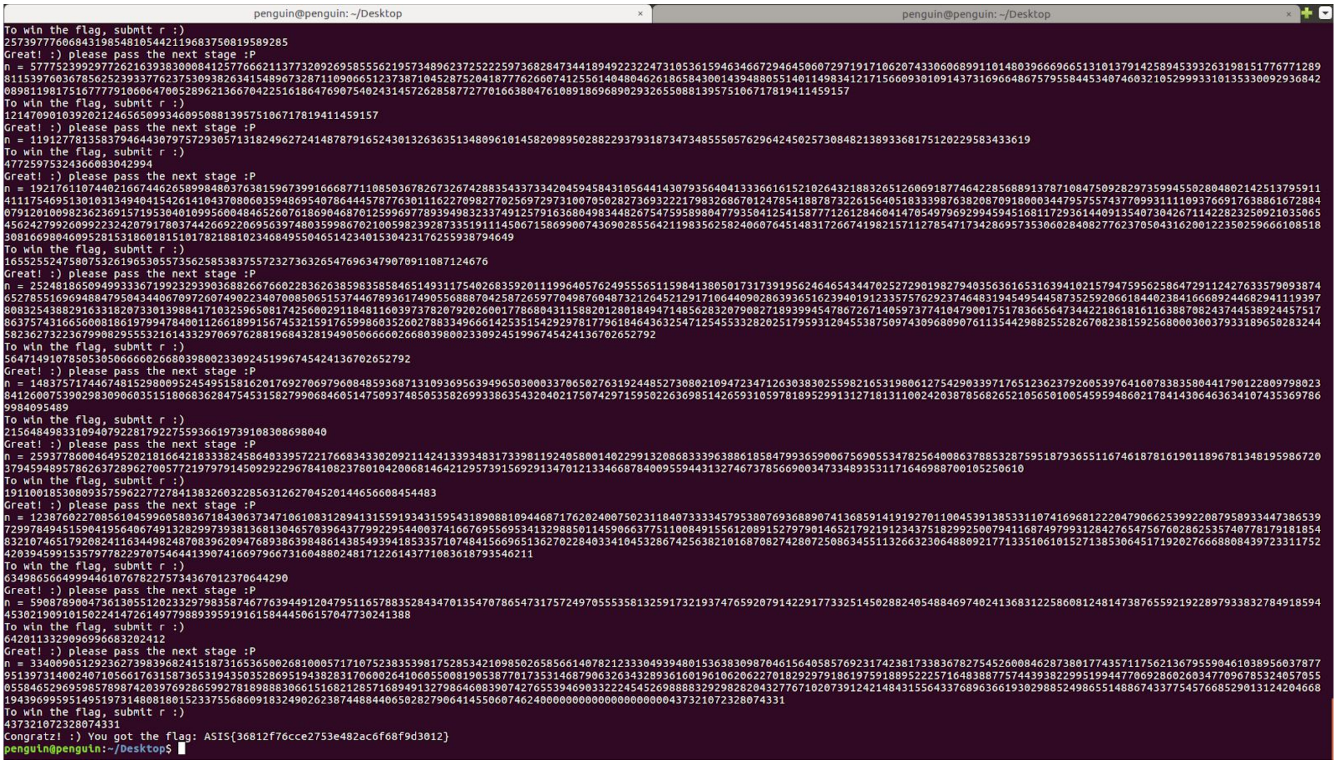
N=12 x=2 → y ={2,3} . This gives the smallest value for R being 4.
N=12 x=3 → y= {2} . This gives the smallest value for R being 3, and thus smaller than base of 2.

After reviewing this information, we assumed the new correct path to take would be to use a theorem solver, specifically z3. However, this ended up giving inconclusive results since finding a minimum value using z3 is quite difficult. Therefore we abandoned this path, in favor of a more efficient brute force.

We initially planned to write an algorithm that checks N-x^y, where x is determined in an initial for loop, and y is nested within that – essentially trying every base with every exponent (where x and y are integers). Theoretically this would work, however with these extremely large N numbers the amount of iterations would have to be extraordinary. Therefore, we implemented a binary search to find the correct exponent and base, so as not to exceed the n value/ break the equation. After doing this, finding the nearest perfect power is found very quickly. Since the challenge would give you multiple n values to find different r's, we simply changed the n value in the script each round. This eventually led to the flag which is:
ASIS{36812f76cce2753e482ac6f68f9d3012}


mathcrack.py
mathcrackValues.py
crack.py
crack.js
import math
values=[]
dic={}
def power(x,n):
#Find the highest possible power value for a specific x (large number) and exponent
highValue = 1
while highValue ** n < x:
highValue =highValue*2
lowValue = highValue/2
#Simple Binary Search, with comparsion to middleValue**n, essentially x^y in forumaula
while lowValue < highValue:
middleValue = (lowValue + highValue) // 2
if lowValue < middleValue and middleValue**n < x:
lowValue = middleValue
elif highValue > middleValue and middleValue**n > x:
highValue = middleValue
else:
return middleValue
return middleValue + 1
#X value is N in the equation
#Try all exponent from 2, 500 (I assumed 500 would be big enough)
for i in range(2,500):
n=i
#Send to function large number, and exponent value for comparision
y=power(x,n)
print(n,y)
print("*********************")
print(x-y**n)
#Set up dicitonary to record all the values
dic[x-y**n]=i
#Also store an array of all the values in r=N-x**y, we want to find the smallest value
values.append(x-y**n)
values.sort()
#Grab lowest value, and find its key-value pair in dic
print("FINAL")
print(values[0])
print("A")
print(dic.get(values[0]))
Line 14, Column 32
Spaces: 4
Python


penguin@penguin: ~/Desktop
penguin@penguin: ~/Desktop
penguin@penguin:~/Desktop$ nc 37.139.22.174 11740
| Welcome to the Near Neighbor Problem, only mathematicians know the flag!! |
| Your mission is to find smallest natural number r for given n, such that |
| there exist integers x, y: n - x**y = r > 0, x > 1 and y > 1. good luck!! |
Submit a printable string X, such that sha256(X)[-6:] = d37879
268684
Please be patient ...
It takes a few seconds to load ...
To win the flag, submit r :)
Great! :) please pass the next stage :P
To win the flag, submit r :)
Great! :) please pass the next stage :P
To win the flag, submit r :)
Great! :) please pass the next stage :P
To win the flag, submit r :)
Great! :) please pass the next stage :P
To win the flag, submit r :)
Great! :) please pass the next stage :P
To win the flag, submit r :)
Great! :) please pass the next stage :P
To win the flag, submit r :)
Great! :) please pass the next stage :P


penguin@penguin: ~/Desktop
penguin@penguin: ~/Desktop
To win the flag, submit r :)
25739777606843198548105442119683750819589285
Great! :) please pass the next stage :P
To win the flag, submit r :)
1214709010392021246565099346095088139575106717819411459157
Great! :) please pass the next stage :P
To win the flag, submit r :)
47725975324366083042994
Great! :) please pass the next stage :P
To win the flag, submit r :)
165525524758075326196530557356258538375572327363265476963479070911087124676
Great! :) please pass the next stage :P
To win the flag, submit r :)
564714910785053050666602668039800233092451996745424136702652792
Great! :) please pass the next stage :P
To win the flag, submit r :)
21564849833109407922817922755936619739108308698040
Great! :) please pass the next stage :P
To win the flag, submit r :)
1911001853080935759622772784138326032285631262704520144656608454483
Great! :) please pass the next stage :P
To win the flag, submit r :)
63498656649994461076782275734367012370644290
Great! :) please pass the next stage :P
To win the flag, submit r :)
6420113329096996683202412
Great! :) please pass the next stage :P
To win the flag, submit r :)
437321072328074331
Congratz! :) You got the flag: ASIS{36812f76cce2753e482ac6f68f9d3012}
penguin@penguin:~/Desktop$