Spring 2021 Cryptography and Network Security

Homework 3

Release Date: 2021/5/18

Due Date: 2021/6/7, 23:59

Instruction

- **Submission Guide:** Please submit all your codes and report to NTU COOL. You need to put all of them in a folder named by your student id, compress it to hw3_{student_id}.zip. For example, hw3_r09922456.zip. The report must be in **PDF** format, and named report.pdf.
- You may encounter new concepts that haven't been taught in class, and thus you're encouraged to discuss with your classmates, search online, ask TAs, etc. However, you must write your own answer and code. Violation of this policy leads to serious consequence.
- You may need to write programs in the Capture The Flag (CTF) problems. Since you can use any programming language you like, we will use a pseudo extension code**.ext** (e.g., code.py, code.c) when referring to the file name in the problem descriptions.
- This homework set are worthy of 120 points, including bonus.
- You are recommended to provide a brief usage of your code in **readme.txt** (e.g., how to compile, if needed, and execute). You may lose points if TAs can't run your code.
- In each of the Capture The Flag (CTF) problems, you need to find out a flag, which is in `CNS{...}` format, to prove that you have succeeded in solving the problem.
- Besides the flag, you also need to submit the code you used and a short write-up in the report to get full points. The code should be named **code{problem_number}.ext**. For example, code3.py.
- In some CTF problems, you need to connect to a given service to get the flag. These services only allow connections from 140.112.0.0/16, 140.118.0.0/16 and 140.122.0.0/16.

Capture The Flag

1. IoT (20%)

You are given a deliberately insecure machine, download it from this link. It is a virtual machine, use Oracle Virtual Box to import this machine and get the flag. Some of the vulnerabilities in this machine are based on OWASP IoT Top 10, find the vulnerabilities in this virtual machine.

Hint: The point of this challenge is to get your way in and escalate your privileges, so you are not given the login username and password right away. You could start with ***nmap****.*

2. Randomness Casino (15%)

Welcome to my casino! In my casino, The result is generated by the randomness from the public, including you! I let you to be the last one to submit the randomness, so you can rest assured that the result is not rigged. Now try to win enough money from me and buy the flag! The challenge source is included in `hw3/randomness`.

a) (5%) (easy) The challenge source is `naive.py`. You can access the system by `nc cns.csie.org 7680`.

b) (5%) Now you should give me the commitment of your randomness first, and then I will ask you to submit your randomness. No cheating! If your randomness does not match your commitment, you will lose. The challenge source is `commitment.py`. You can access the system by `nc cns.csie.org 7681`.

c) (5%) I enhanced the implementation of the commitment, so there should be no chance for you to cheat. Can you win this time? The challenge source is `commitment2.py`. You can access the system by `nc cns.csie.org 7682`.

3. VDF (15%)

You have learned Verifiable Delay Function (VDF) in class, now let's try to implement it. We use the class group VDF proposed by Wesolowski. The challenge source is included in `hw3/VDF`. The implementation of class group is provided in `ClassGroup.py`, and you only have to implement the VDF part.

a) Given the class group, iteration T, and proof parameter L, give me the output $y = x^{(2^T)}$ and the proof $\pi = x^{\lfloor \frac{2^T}{L} \rfloor}$ such that $y = \pi^L x^{(2^T \mod L)}$, where x is the generator of the class group. The challenge source is `getVDFproof.py`. You can access the system by `nc cns.csie.org 7685`.

The implementation of class group operation ***** *would be faster for the same group element. That is,* ***a*a*** *would be faster than* ***a*b****. The timeout is 60 seconds, and TA's implementation takes about 40 seconds on linux2.*

b) When a bad class group is chosen, the security requirements of VDF may not hold. Try to break it! The challenge source is `getVDFproof2.py`. You can access the system by `nc cns.csie.org 7686`.

4. DDoS (30% + Bonus 10%)

We have learned a lot of attack/defense techniques in class. Here, we provided each student a machine that may be a victim or an attacker of a DDoS attack. Your job is to keep your machine away from a DDoS attack. For more information, please visit https://cnshw3.csie.org/. You can sign in with your student ID and the password we announced in NTU COOL (Assignments → DDoS Password). We also created an attacker that will attack your machine every five minutes. Try your best to protect your machine from our attack! In the report, please describe the vulnerability in detail and how to patch/remove it.

Note: Currently, there are only 4 challenges. Other challenges will be released before 2021/05/25 23:59.

The server is at `cnshw3.csie.org`*, a ssh secret key is provided in the released file, you can connect to your server by* `ssh cns2021@cnshw3.csie.org -i id_ed25519_cns`*, then input your account / password, which are the same as that of the website (https://cnshw3.csie.org/)*

For several problems, you may need to upload your patch (more precisely, the problem in `/home/cns/code` *in your server), please use* `nc cnshw3.csie.org 9090` *to upload your patch, more instruction will be listed in README.md in provided files*

5. Smart Contract (20% + Bonus 10%)

(a) (15 points + 10 bonus) Here we prepared a `HW3` smart contract as an easy hands-on practice for hacking smart contract. Enjoy!
Reminders:

1. The `HW3` contract is deployed on `Kovan` testnet, not Ethereum mainnet.

   HW3 contract address: 0x31877e75ad477579a885ca5006a208e0058ce58b

   CNS token address: 0xa2bc8b4EB61d84F76742D9Ca5FBBca48A436f790

2. You can get Kovan ether from: Kovan faucet

3. Remix is recommended tool for this homework.

4. Please pass in your student ID (lower case, ex: b02902000) as a parameter while solving the challenges. Make sure your score is increased and recorded on the smart contract. DO NOT use others' student ID. Using others' student ID will lead to serious consequences. If your student ID is taken by others, please contact CNS TAs.

5. Please describe your solutions in the report and submit your codes (if any) as code5-*.ext for any challenge you solved.

6. FAQ: `https://hackmd.io/@hRwqVLRKR-C_OlJpkO8xxA/S177Mg3uu`

(Note: All transaction can be searched from etherscan so do not threaten others to give you CNS token!)

(b) (5 points) One of the biggest hacking events by far was `the DAO hack`, in which the hacker took away 50 millions worth of Ether. It was a Re-Entrancy Attack, causing Ethereum to hard fork.

What is `Re-Entrancy Attack`? How to prevent it?