**Lab1 - Jakob Roberts - v00484900**

**Question 1:** What 3 quantities does an attacker need to know to determine Netscape's generated seed?

the time of day, the process ID, and the parent process ID

**Question 2:** How does a predictable PRNG seed reduce its level of security?

An attacker who can guess the PRNG seed value can easily determine the encryption keys used in Netscape's secure transactions.

**Question 3:** How many numbers per second were you able to generate with each method?

Urandom generated approximately 5832704/second and random generated about 3.13/second.

**Question 4:** Why are PRNGs called "Pseudo"?

Because they are not truly random like random numbers based on a naturally occurring substance.

**Question 5:** What are two properties that CSPRNGs should have? (Hint: Check [Wikipedia](https://en.wikipedia.org/wiki/Cryptographically_secure_pseudorandom_number_generator))

1. Every CSPRNG should satisfy the next-bit test
2. Every CSPRNG should withstand "state compromise extensions"

**Question 6:** Describe the results of your 4 generated png files.

Both Random and Urandom produced very spread out results when not ordered, but when ordered, they form a very nice linear line. On both the sorted results, the linear line has some slight inconsistencies but nothing extravagant.

**Question 7:** Why is reading from /dev/urandom is significantly faster than /dev/random/?

Urandom uses the existing entropy pool to generate pseudo-randomness, whereas Random generates randomness from actual entropy gathered by the environment of the computer. Because Random relies on the environment, it can produce results sporadically and is therefore quite slow in comparison to Urandom