Random Number Generators

In-Class Exercise: Comparing Randomness

Cryptography Course

January 17, 2026

Exercise Overview

Goal

Compare three different types of random number generators by generating and analyzing bit sequences.

- **Duration:** 10–15 minutes
- **Task:** Generate three sequences of 100 bits
- **Analysis:** Visual inspection and simple randomness tests

The Three Generators

1. **True Random Number Generator (TRNG)**
   - Uses physical phenomena (atmospheric noise)
   - Source: `random.org`
2. **Linear Congruential Generator (LCG)**
   - Deterministic algorithm: $X_{n+1} = (aX_n + c) \mod m$
   - Fast but predictable
3. **Cryptographically Secure PRNG (CSPRNG)**
   - Designed for cryptographic applications
   - Unpredictable even with partial knowledge

Step 1: True RNG (random.org)

1. Visit: `https://www.random.org/integers/`
2. Configure the generator:
   - Number of integers: **100**
   - Range: **0** to **1**
   - Format: **One per line**
3. Click **"Get Numbers"**
4. Copy the sequence and save it as `trng_sequence.txt`

Note

This uses atmospheric noise for true randomness!

Step 2: Linear Congruential Generator

Simple Python Implementation

```
a, c, m = 1103515245, 12345, 2**31
seed = 42
sequence = []
for _ in range(100):
    seed = (a * seed + c) % m
    sequence.append(seed % 2)
print(''.join(map(str, sequence)))
```

Save the output as `lcg_sequence.txt`

Step 3: Cryptographically Secure PRNG

Python's secrets Module

```
import secrets
sequence = [secrets.randbelow(2)
            for _ in range(100)]
print(''.join(map(str, sequence)))
```

Save the output as csprng_sequence.txt

Security Note

The secrets module is suitable for security-sensitive applications.

Step 4: Visual Analysis

Simple Tests to Perform

1. **Count 0s and 1s**
   - Should be approximately 50/50
2. **Look for patterns**
   - Long runs of same bit?
   - Repetitive sequences?
3. **Compare sequences**
   - Which one "looks" more random?
   - Which one has unexpected patterns?

Discussion Questions

- Can you visually distinguish between the three sequences?
- What makes a sequence "look random"?
- Why might the LCG show patterns?
- When would you use each type of generator?
- What are the security implications of using weak RNGs in cryptography?

Key Takeaways

Important Points

- **TRNGs** use physical randomness (unpredictable)
- **LCGs** are fast but unsuitable for cryptography
- **CSPRNGs** balance speed and security
- Visual inspection is *not sufficient* for security
- Proper statistical tests are needed for cryptographic use

Questions?