Politechnika Warszawska

Wydział Fizyki

Kryptografia i bezpieczeństwo informacji dla fizyków

Sprawozdanie z zadania nr. 2 na temat:

**„Metryki losowości, testy FIPS 140-2 generatorów liczb pseudolosowych”**

Data wykonania zadania w laboratorium: 22.10.2019r.

**Wykonali:**

Maciej Czarnecki

Denys Morokov

Fizyka techniczna II stopień, 2 rok

1. **Wstęp**
   1. **Cel:** celem zadania jest 1) oprogramowanie liniowego generatora kongruentnego, a następnie wykonanie testu FIPS na wygenerowanych liczbach; 2) zbadać jak poprawić entropię systemowego generatora od chwili startu.
   2. **Wykorzystany język programowania:** Python.
   3. **Specyfikacja komputera:** testy wykonano na dwóch komputerach, gdzie na jednym system operacyjny Linux jest zainstalowany natywnie, natomiast na drugim – przez użycie wirtualnej maszyny.
      1. **Natywny Linux:**

- Wersja systemu: Ubuntu 18.04 LTS

- Powłoka: GNOME 3.28.2

- Procesor: Intel Core i5

- Dysk: SSD

- Pamięć: 8 GiB

- Dostęp do internetu: poprzez Wi-Fi

- Brak hwrng

- Do komputera były podłączone klawiatura i mysz

* + 1. **Wirtualna maszyna:**

- Wersja maszyny wirtualnej: VirtualBox 6.0.12

- Wersja systemu: Ubuntu 18.04 LTS

- Powłoka: GNOME 3.28.2

- Procesor: Intel Core i5

- Dysk: SSD

- Pamięć: 3 GiB

- Dostęp do internetu: brak

- Brak hwrng

- Do komputera były podłączone klawiatura i mysz

1. **Opracowanie wyników**
   1. **Liniowego generatora kongruentnego, test FIPS.**

Oprogramowano liniowy generator kongruentny w języku Python. Skrypt generuje 10 ciągów losowych o długości 4096 bitów, które zapisano do pliku txt. Następnie w celu wykonania testu FIPS zainstalowano pakiet Rng-tools. Za pomocą polecenia „rngtest < nazwa\_pliku.txt” wykonano test FIPS. Rysunek 1 przedstawia wynik testu otrzymany na maszynie wirtualnej, a rysunek 2 na natywnym Linuxie.



Rys. 1. Wynik testu FIPS dla 10 ciągów losowych otrzymany na maszynie wirtualnej



Rys. 2. Wynik testu FIPS dla 10 ciągów losowych otrzymany na natywnym Linuxie

* 1. **Sposoby na poprawianie entropii systemowej.**

**Po pierwsze,** można użyć specjalnego sprzętu (TPM, Trusted Platform Module) lub instrukcji procesora, takich jak RDRAND (dostępny w procesorach Intel IvyBridge i Haswell). Za pomocą komendy „rngd -v” można sprawdzić, czy są takie urządzenia w systemie. Jeżeli są, można wykorzystać „rngd” z pakietu rng-tools, który odczytuje entropie z TPM i wypełnia tę entropią pulę entropii jądra. Odbywa się to poprzez specjalne wywołanie ioctl (RNDADDENTROPY) interfejsu /dev /random.

**Po drugie,** można skorzystać z pakietu Haveged. Ten pakiet wykorzystuje algorytm Havage, który generuje entropię na podstawie liczników i stanów procesora. Ze względu na złożoną, wielopoziomową konstrukcję procesorów ten sam kod jest zawsze wykonywany z różną prędkością, a ta niespójność jest podstawą algorytmu Havage. Domyślnie podtrzymuje wartość entropii jądra nie niżej 1024.

**Trzecim** sposobem jest wykorzystanie BitFolk, który ma kilka kluczy Entropy Electronics Simtec podłączonych do różnych hostów. Klucze generują entropię z losowego przepływu elektronów, a demon odczytuje entropię z każdego klucza przez USB i podaje go na porcie TCP. Aby zdalni gospodarze mogli skorzystać z entropii, uruchamiają innego demona, „ekeyd-egd-linux . ekeyd-egd-linux” kontaktuje się z serwerem entropijnym BitFolk, pobiera pewną entropię i podaje ją do jądra maszyny, na której działa, aby później mogła zostać udostępniona z /dev/random.

1. **Podsumowanie**

Generator nie przeszedł testów FIPS, ponieważ generator jest liniowy i liczba pseudolosowa w kroku i generowana przez generator zależy od liczby losowanej w kroku i-1.