Szyfrowanie komunikacji sieciowej metodą Reverse Interval Mapping

Aleksandra Żebrowska
Anna Śnieżek
Jan Dudek
Łukasz Dębski

Cel projektu:

Celem projektu jest zapoznanie się z metodą szyfrowania danych Reverse Interval Mapping wykorzystującą chaos do zakodowanie wiadomości przesyłanych przez komunikator.

Wstęp teoretyczny:

W około VIII wieku p. n. e. grecki epik Hesion w swoim logicznym eposie kosmologicznym "Theogony" po raz pierwszy użył sformułowania "na początku był chaos"; mityczne pojęcie chaosu zostało pozostawione samo sobie aż do początku XX wieku, kiedy pojawiła się Teoria Chaosu
Chaos jest ekscytujący ponieważ:
łączy codziennie obserwowane zjawiska z prawami przyrody pokazując delikatne związki pomiędzy prostością i złożonością;
prezentuje deterministyczny, stosujący się do fundamentalnych praw fizyki Wszechświat zdolny do nieporządku, złożoności i nieprzewidywalności
pokazuje, że przewidywalność zjawisk jest rzadkim fenomenem objawiającym się tylko w przypadku założeń filtrujących różnorodność złożonego świata
pozwala kwestionować tradycyjne procedury modelowania zjawisk przez naukę

Chaos był różnorodnie definiowany, na przykład:
"Rodzaj uporządkowania pozbawionego okresowości"
"Pozornie losowe zachowanie deterministycznego systemu"

“Jakościowe badania niestabilnych, aperiodycznych zachowań w deterministycznych nieliniowych dynamicznych systemach”
Oraz dana przez angielskiego matematyka Iana Stewarda:
“Zdolność prostych modeli, bez wprowadzonych właściwości losowych, do generowania wysoce nieregularnych zachowań”

Teoria Chaosu skupia się na zadawaniu pytań dotyczących długoterminowych zachowań systemu. Zamiast formułować założenia na temat przyszłego stanu, w jakim znajdzie się system, nauka ta skupia się jakościowym badaniu systemu który jest niestabilny i aperiodyczny. Ma to szerokie zastosowanie w m. in. ekonomii, przewidywaniu zjawisk atmosferycznych, mechanice płynów, chemii i biologii populacji. Teoria Chaosu nie jest w stanie odpowiedzieć na pytanie, w jakim stanie znajdzie się wyszczególniony element układu w sprecyzowanym czasie, jest w stanie natomiast badać długoterminowy wpływ całego systemu. Tego rodzaju badanie ma znacznie większe przełożenie na świat obserwowany dookoła, ponieważ nie wymaga pomijania, przede wszystkim, warunków zewnętrznych oraz sprzężenia zwrotnego funkcji; własności te powodują niesamowite komplikacje, gdyśmy chcieli wziąć je pod uwagę przy tradycyjnym modelu, funkcja nabrałaby ogrnomnej komplikacji.

Najbardziej podstawową cechą chaosu jest ogromna czułość na warunki początkowe;
“Czy trzepot skrzydeł motyla w Brazylii może wywołać tornado w Texasie?” - Edward Lorenz
Przykładem zjawiska jest rzut kostką; dynamika rzutu pozwala dokładnie obliczyć jego rezultat, jednak biorąc pod uwagę jak drastyczną zmianę przyniesie drobna zmiana w zmiennych początkowych (kierunku, sile, kącie wyrzutu itp) jest praktycznie niemożliwe wykonać dwa identyczne rzuty; rezultat sprawia wrażenie całkowicie losowego.
Powyższa cecha jest niezwykle przydatna w informatyce. Komputery na bazie bramek logicznych są w stanie jedynie wykonywać zaprogramowane przez człowieka polecenia; nie są zdolne do “wymyślenia” liczby, wobec czego nie są w stanie zwrócić wartości losowej. Substytutem tej właściwości są funkcje chaotyczne - kiedy delikatna zmiana w zmiennych przekłada się na bardzo drastyczną zmianę w wartości funkcji, jesteśmy w stanie generować liczby dające łudzące wrażenie losowości; mówimy wtedy o liczbach pseudolosowych.

Działanie programu

w pierwszym kroku uruchamiamy serwer oraz dwóch klientów komunikatora. Wpisana wiadomość w oknie jednego z klientów zostaje zakodowana przy pomocy mapy odwzorowania i wysłana na serwer. Wiadomość zostaje zaprezentowania na serwerze w postaci ciągu znaków. W oknie drugiego klienta wiadomość zostaje zdekodowana przy pomocy mapy odwrotnej i wyświetlona w postaci oryginalnego tekstu.

Kod Programu

Kod tworzący komunikator zaczerpnęliśmy ze strony podanej w opisie zadania.

Funkcja szyfrująca:

```
def koduj(tekst):
        tekst_bin1 = bin(int(binascii.hexlify(tekst.encode('utf-8')), 16))[2:]
        tekst_bin = tekst_bin1.zfill(8 * ((len(tekst_bin1) + 7) // 8))
        tekst_roz = [tekst_bin[i:i + 3] for i in xrange(0, len(tekst_bin), 3)]
        tekst_2 = []
        for txt in tekst_bin:
                x = (1.0 + p) / 2
                for t in txt:
                        if t == '0':
                                x = odwrotne(x/p)
                        else:
                                x = 1.0 - odwrotne(x/p)
                tekst_2.append(x)
        return ' '.join(map(str, tekst_2))
```

Funkcja deszyfrująca:

```
def dekoduj(tekst_kod):
    teksty = map(float, tekst_kod.split())
    tekst_bin = ''
    for t in teksty:
        x = ''
        while 0.0 <= t <= 1.0:
        if t < 0.5:
                        x = '0' + x
                else:
                        x = '1' + x
                t = p * odwzoruj(t)
       tekst_bin += x
    return bin_na_tekst(tekst_bin)
```

Bibliografia:

Z. Sardar- Chaos for Beginners
S. Sternberg- Dynamical systems

strona:
http://www.bogotobogo.com/python/python_network_programming_tcp_server_client_chat_server_chat_client_select.php