AUTENTYKACJA W SIECI APRSIS (lub kompletny jej brak)

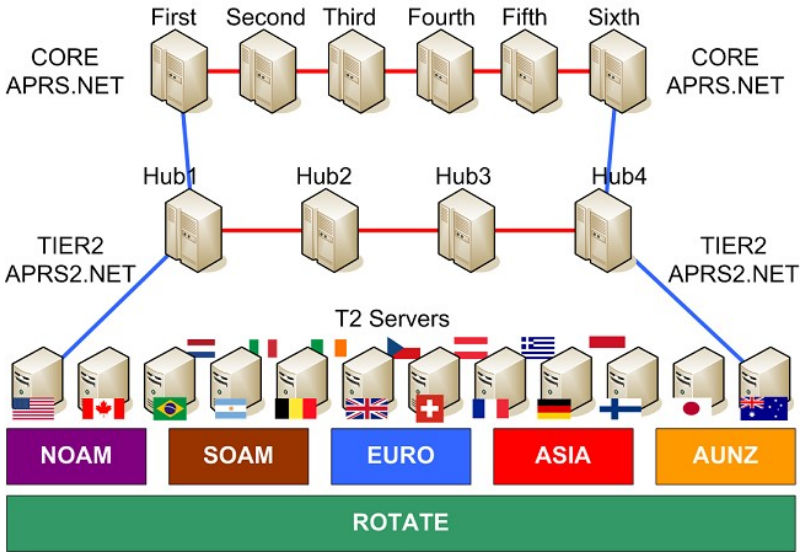
Aktualny sposób autoryzacji użytkownika w sieci APRSIS

Sieć APRSIS zbudowana jest za pomocą 9 serwerów w rdzeniu sieci, 5 koncentratorów oraz około 90 serwerów regionalnych.

W chwili obecnej w calej sieci APRSIS do połączeń między-serwerowych i klienckich używany jest wyliczany passcode.

Passcode ten jest wyliczany ogólnie znanym algorytmem na podstawie znaku (loginu) użytkownika lub serwera i może przyjąć wartość pomiędzy 1000 a 32767.

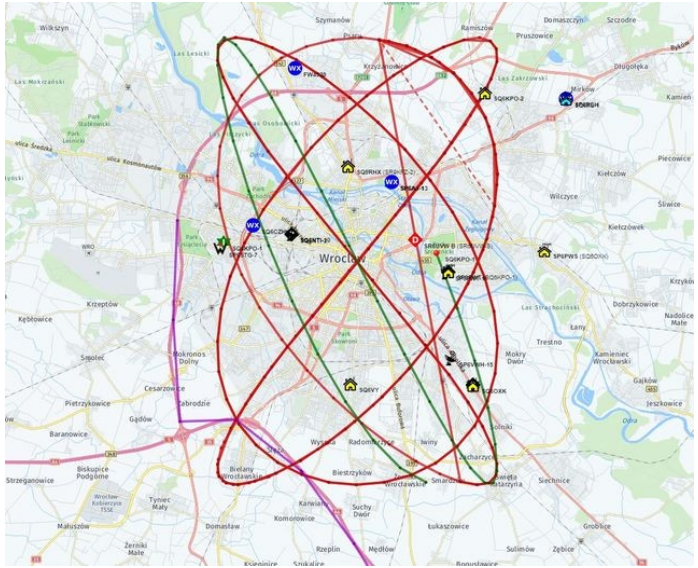
```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import time

def calculate_passcode(callsign='nolic'):
    assert isinstance(callsign, str)
    callsign = callsign.split('-')[0].upper()
    code = 0x73e2
    for i, char in enumerate(callsign):
        code ^= ord(char) << (8 if not i % 2 else 0)
    return code & 0x7fff

while(1):
    call = raw_input('enter your callsign: ')
    print(call.upper() + ': ' + str(calculate_passcode(call)))
```



Objawy problemu

Sieć jest atakowana przez nieuprawnionych użytkowników którzy mogą podszyć się pod dowolnego użytkownika.

Ataki te charakteryzują się tym że atakujący łączy się z dowolnym serwerem i używając dowolnego znaku oraz wyliczonego do niego hasła przesyła gigantyczne ilości informacji do serwerów APRSIS.



Trudności związane z implementacją innych sposobów logowania

W chwili obecnej kliencka część sieci APRS to dużo różnych rozwiazań które nie są przystosowane do innej metody logowania.

Każde urządzenie czy to pracujące w części radiowej czy też w części APRSIS znajduje się na liście TOCALLS, w chwili pisania tej prezentacji było to 232 urządzenia / programy.

Po odfiltrowaniu z tej listy urządzeń stricte radiowych pozostała część to różnej maści oprogramowanie na różne platformy,

które powstawało przez wiele lat od momentu uruchomienie sieci APRS.

Problematyczne jest wymuszenie zmian dla oprogramowania które nie jest w chwili obecnej wspierane.

Propozycja prostego rozwiązania problemu.

Proponuję zmianę sposobu logowania by hasło było zarządzane przez użytkownika. Aby to osiągnąć należy wykonać poniższe kroki:

- zainstalować na serwerze APRS dodatkową bazę danych użytkowników wraz z interfejsem rejestracyjnym i administracyjnym

- zmiana w oprogramowaniu serwera APRS sposobu sprawdzania hasła, zamiast wyliczać co być powinno, sprawdź znak w bazie danych

Dzięki takiej zmianie zyskujemy:

- użytkownik może samodzielnie zmienić skompromitowane hasło

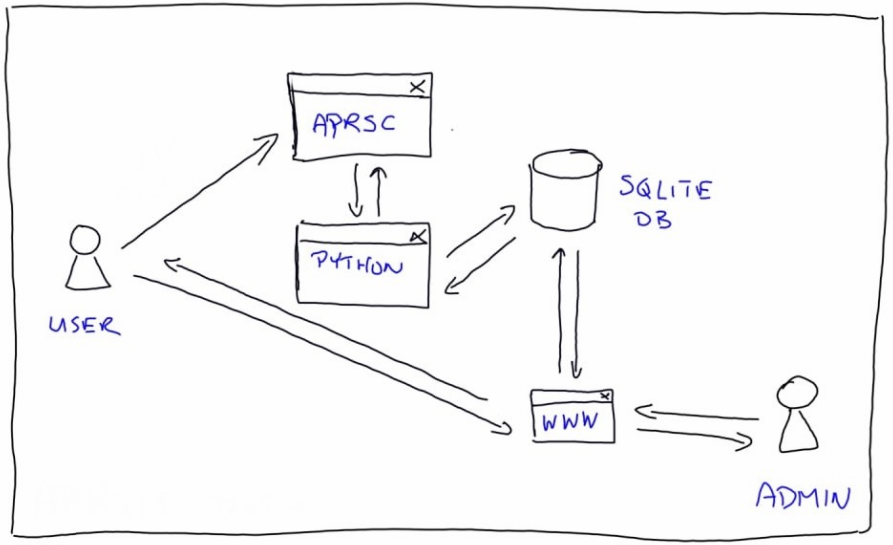
- administrator może zablokować danego użytkownika jeśli jego znak i hasło zostanie wykorzystane do spamowania sieci.

Jak to działa ?

Za pomocą drobnej zmianie w kodzie źródłowym, serwer aprsc został zmuszony by sprawdzanie hasła odbywało się w zewnętrznej bazie danych za pomocą dołączonego skryptu napisanego w pythonie.

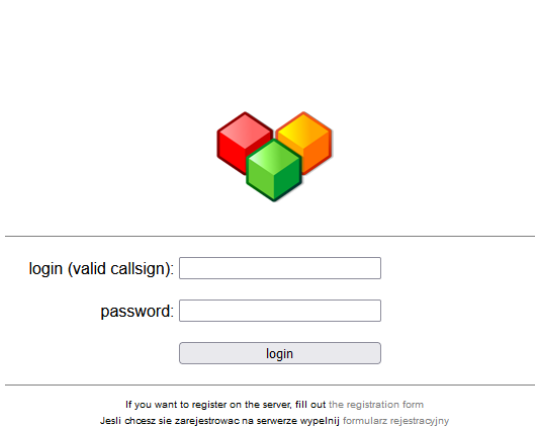
Zewnętrzna baza danych posiada interfejs dzięki któremu możliwe jest dodawanie i zarządzanie użytkownikami.

Administrator ma możliwość aktywacji lub deaktywacji konta użytkownika.



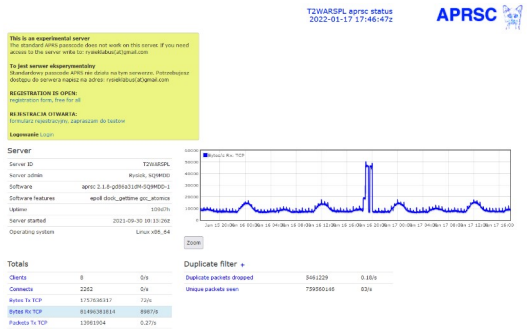
Sprawdźmy jak to działa

Działające rozwiązanie w postaci przekompilowanej wersji serwera aprsc oraz dodatkowej bazy danych SQLITE z interfejsem napisanym w pythonie znajduje się na serwerze T2WARSPL



Sprawdźmy jak to działa

Od kilku miesięcy testuję rozwiązanie na swoim serwerze, rozwiązanie jest stabilne.



Clients

Port	Username	Address	Verified	Up	Last in	Software	Packets Tx	Packets Rx	Bytes Tx	Bytes Rx	Tx/Rx bytes/s	OutQ	MsgRcpts	Filter
14580	SQ9SIM-2	88.220.84.111:32589	Yes	34m	1m	aprx 2.9.0-28-g13c6e79	304	7/5/0	39786	713	0 / 0	0	3	m/20
14580	SR9JSR-1	87.101.79.146:50962	Yes	4h13m	28s	aprx 2.9.0	2462	1866/1139/0	286949	180165	9.7 / 0	0	86	m/20
14580	SP5XSL	77.242.236.52:59104	Yes	5h29m	30s	aprx 2.9.0	1138	1161/978/0	192739	118335	16 / 0	0	48	m/20
14580	SR5NEB	213.216.74.157:26448	Yes	8h14m	2s	aprx 2.9.0	2748	6231/5315/0	387356	615334	19 / 24	0	63	m/20
14580	SP5ME-1	5.185.15.217:44634	Yes	9h31m	3s	aprx 2.9.0	2437	3398/3321/0	383126	341461	16 / 12	0	56	m/5
14580	SQ9MDD-4	78.11.136.222:59910	Yes	11h5m	3s	aprx 2.9.0	2337	5433/3694/0	392355	530508	6.9 / 12	0	59	m/1
14580	EA8CXN	37.132.108.199:50760	Yes	18h39m	7s	aprx 2.9.0	15013	6116/2744/0	1822789	717674	81 / 43	0	9	m/200
14580	SQ9IWE-2	89.70.110.177:47574	Yes	13d11h	1m	aprx 2.9.0	43310	9758/4491/0	8989436	835811	0 / 0	0	4	m/8