POLITECHNIKA OPOLSKA
WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI

INSTRUKCJA LABORATORYJNA DO PRZEDMIOTU OCHRONA DANYCH W SYSTEMACH I SIECIACH KOMPUTEROWYCH

Bezpieczeństwo WLAN

„Wspaniale powiedziane: standardowym poziomem bezpieczeństwa w przewodowej sieci LAN jest brak jakichkolwiek mechanizmów zabezpieczających, więc zadanie postawione przed protokołem WEP nie jest specjalnie wygórowane, protokół ten radzi sobie z nim całkiem nieźle."

Andrew S. Tanenbaum

1. Wstęp

Należy zaznaczyć, że autor tego ćwiczenia nie popiera ani nie zachęca do łamania zabezpieczeń sieci bezprzewodowych, a tym bardziej do nieuprawnionego korzystania z cudzych łącz. Czynności takie są przede wszystkim nieetyczne, a nieuprawnione korzystanie z cudzych łącz jest nielegalne. Autor nie ponosi odpowiedzialności za sposób wykorzystania przytoczonej tu wiedzy.

Dokument ten jest instrukcją do ćwiczenia, które odbywa się w ramach przedmiotu „Bezpieczeństwo danych w systemach i sieciach komputerowych" na Politechnice Opolskiej, kierunek – Informatyka. Ma ci uświadomić, aby nie zabezpieczać sieci bezprzewodowych szyfrowaniem *WEP*. A to co jest tu napisane, wykonujesz tylko w obrębie sieci stworzonej specjalnie na potrzeby laboratorium.

2. Wireless Fidelity, czyli Wi-Fi

Mianem *Wi-Fi* określamy potocznie zestaw standardów stworzonych do budowy bezprzewodowych sieci komputerowych. Szczególnym zastosowaniem *Wi-Fi* jest budowanie sieci lokalnych (*LAN*) opartych na komunikacji radiowej czyli *WLAN*. Standard *Wi-Fi* opiera się na *IEEE 802.11.*

W sieciach bezprzewodowych zgodnych ze standardem *IEEE 802.11*, urządzenia przesyłają dane drogą radiową. Tu niestety pojawia się problem. Mianowicie, każda stacja znajdująca się w zasięgu stacji nadającej odbiera wysyłane dane, co umożliwia podsłuch – brzmi to groźnie. Brak mechanizmów bezpieczeństwa powodowałby możliwość przetwarzania danych przez każdą stację znajdująca się w promieniu nadajnika.

Aby zapewnić minimalny poziom bezpieczeństwa należy zapewnić sygnałom poufność przy pomocy algorytmów szyfrujących oraz zastosować kontrolę dostępu do sieci *WLAN* poprzez użycie mechanizmów uwierzytelniania. Pierwszym rozwiązaniem opracowanym przez grupę *IEEE*, mającym zapewnić sieciom WLAN bezpieczeństwo, był algorytm *WEP* (*Wired Equivalent Privacy*). Niestety z powodu wielu słabych punktów, omówionych w dalszej części laboratorium wraz z przedstawieniem jego karty chorobowej, szybko okazało się, że sieci radiowe wykorzystujące algorytm *WEP* nie są bezpieczne.

Ważnym wydarzeniem, z punktu widzenia ochrony danych, było utworzenie stowarzyszenia *Wireless Ethernet Compatibility Alliance* - znane jako *Wi-Fi Alliance* - promującego technologie WLAN, zajmującego się także standaryzowaniem profili zabezpieczeń znanych jako *WPA (Wi-Fi Protected Access*) i *WPA2*. Produkty zgodne z *Wi-Fi* mają na sobie odpowiednie logo, które świadczy o zdolności do współpracy z innymi produktami tego typu. Logo *Wi-Fi* jest znakiem handlowym należącym do stowarzyszenia *Wi-Fi Alliance*. Ostatecznie standard *IEEE 802.11i* - nazwa marketingowa *WPA2* - został opublikowany pod koniec 2004 roku. Obecnie większość urządzeń *WLAN* dostępnych na rynku wspiera standard *IEEE 802.11i.*

3. Formy zabezpieczeń sieci standardu 802.11

3.1. Ukryty SSID

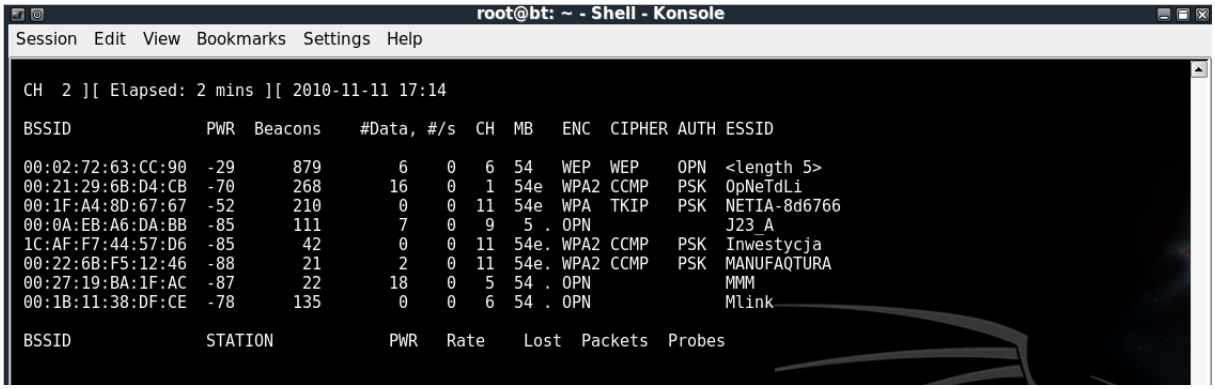
Ukrywanie nazwy (identyfikatora) sieci jest jednym z najczęściej stosowanych „zabezpieczeń" w sieciach bezprzewodowych. Funkcja ta jest prawdopodobnie oferowana przez wszystkie obecnie działające na rynku *Access pointy* (*AP*). Skoro jest taki myk, to czy warto z niego korzystać?

Moim zdaniem jak najbardziej tak. Wystarczającym powodem jest fakt, iż wiele oprogramowania, opartego o system operacyjny Windows, dołączonego do kart sieciowych nie wykrywa (tj. nie pokazuje) sieci, której nazwy nie jest w stanie wykryć.

Konkludując, czego oczy nie widzą, sercu nie żal. Skoro potencjalny hacker nie widzi naszej sieci – nie ma pokusy, żeby się do niej dobrać. Oczywiście wyłączenia rozgłaszania *SSID* nie należy pod żadnym pozorem traktować jako zabezpieczenia, jest to... taki sobie dodatek. Dlaczego?

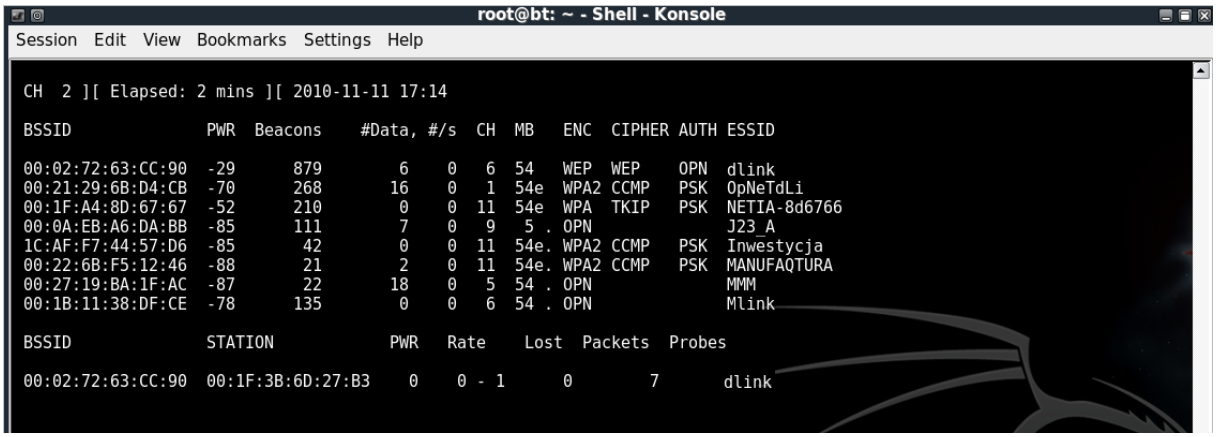
Każdorazowo, gdy do sieci bezprzewodowej podłącza się nowy klient, mający prawo podłączyć się do tej sieci – *SSID* między jego urządzeniem a *AP* przesyłany jest czystym tekstem. Co to oznacza? Otóż, wystarczy program nasłuchujący oraz odrobina cierpliwości, aby całe to pseudo-zabezpieczenie szlag trafił. Smutne, prawda? A jakie prawdziwe…

Bardzo szybko możemy się o tym przekonać uruchamiając pakiet *airodump-ng*. Tak wygląda wykryta sieć z wyłączonym rozgłaszaniem *SSID*:



Rysunek 3.1 Pakiet airodump-ng obrazujący sieć z wyłączonym rozgłaszaniem SSID

Wystarczy jednak chwila cierpliwości i nowy klient podłączony do sieci, aby naszym oczom pokazał się taki oto wspaniały widok:



Rysunek 3.2 Ukryty SSID już nie taki ukryty - <length 5> -> dlink

Sieć, która krzyczy we wszystkie strony o tym, że istnieje, przyciąga uwagę włamywaczy - dlatego wyłącz dla własnego spokoju lepiej rozgłaszanie *SSID*.

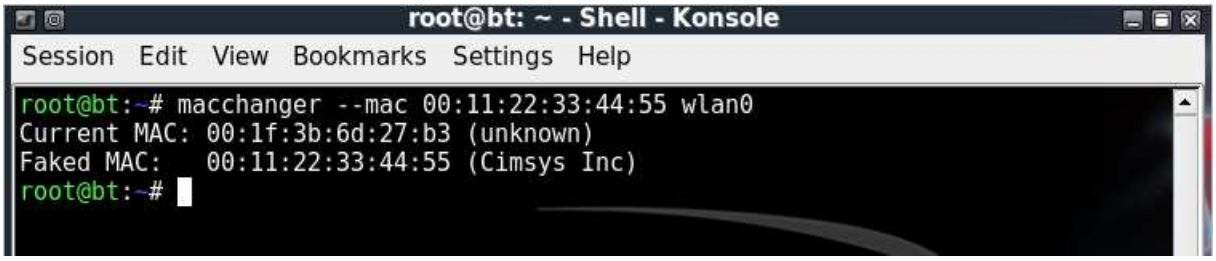
3.2. Filtracja MAC

Zacznijmy od podstaw – Co to jest *MAC*? Adresem *MAC* określa się sprzętowy adres karty sieciowej. Każda karta sieciowa posiada niepowtarzalny, 6-cio bajtowy adres mogący wyglądać np. tak: 00:A0:C9:21:B2:93. Pierwsze trzy bajty (tutaj 00:A0:C9) definiują producenta karty sieciowej (w tym przypadku Intel), kolejne trzy - nr seryjny nadany przez niego.

Czym jest więc filtracja *MAC* na *AP*? Jest to zdefiniowanie w pamięci *AP* adresów *MAC* bezprzewodowych kart sieciowych klientów, które mogą komunikować się z danym *AP*. Wszystkie inne adresy są odrzucane. Tak więc, jeśli już „zdobyłeś hasło umożliwiające się podpiąć po *Wi-Fi* do *AP* dawcy", a „Internet ci nie działa", oznacza to jedno – jest włączona filtracja *MAC* na *AP*.

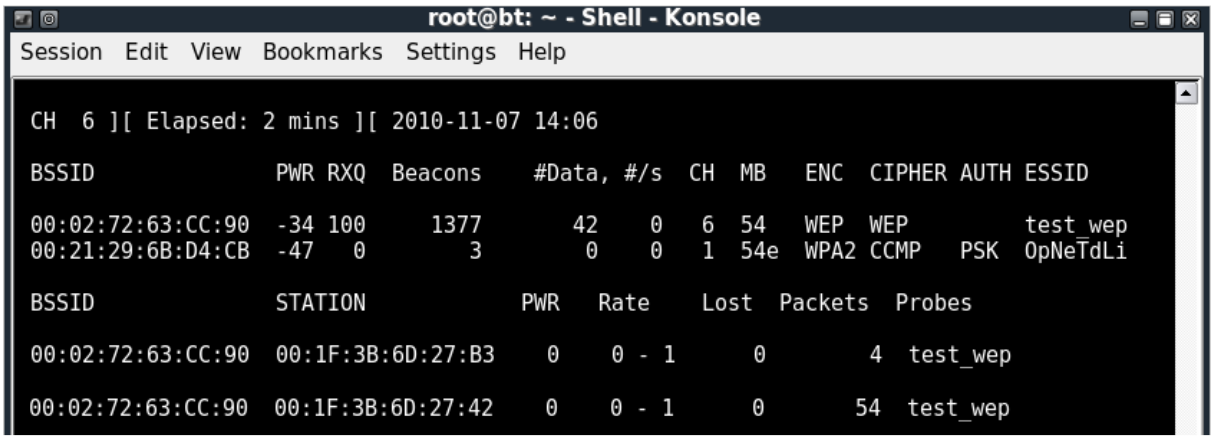
Nasuwa się więc pytanie – czy warto stosować filtrację *MAC*? Uważam że tak - zwłaszcza w małych sieciach. Po pierwsze - często wszystkie komputery w małych sieciach są włączone a to uniemożliwia wykorzystanie przez hackera adresu *MAC* naszej karty sieciowej upoważnionej do komunikacji z *AP*. Po drugie w przypadku zdublowania się adresów *MAC* w danej sieci - jeden z nich przestaje działać, a to oznacza, że ktoś wpięty jest na lewo. Windows w takiej sytuacji sygnalizuje każdemu użytkownikowi komunikatem na ekranie. Po trzecie - żeby zwolnił się któryś z używanych adresów trzeba trochę (czasami bardzo długo) poczekać. Po czwarte – nie jesteśmy spokrewnieni z Nostradamusem i nigdy nie wiemy, kiedy „legalny" klient ponownie włączy swój komputer, co prowadzi nas do punktu drugiego. No i po piąte - filtracja *MAC* jest - szczerze - bardzo uciążliwa, ponieważ jeśli przyjdzie do nas kolega z laptopem i będziemy chcieli go „puścić po *Wi-Fi*" w naszej sieci, to musimy przejść skomplikowany proces klikologii.

Sterowniki większości kart sieciowych umożliwiają zmianę adresu *MAC*. Pod *Linuxem* można posłużyć się programem *macchanger*, ma on wiele opcji. Pojawia się tylko pytanie: na jaki? Prawidłową odpowiedzią jest „na adres któregoś z legalnych klientów". A więc wystarczy poczekać aż którykolwiek komputer odłączy się od sieci, zmienić nasz adres *MAC* - na jego.



Rysunek 3.3 Zmiana adresu MAC

Poniższy zrzut ekranu pokazuje dwa komputer podłączone do naszego *AP*. Wystarczy więc je spisać (zrobić zrzut ekranu, skopiować & wkleić...) i czekać. Albo wrócić wieczorem. Albo nad ranem. Wcześniej czy później któryś z nich na pewno zniknie:



Rysunek 3.4 Do sieci test_wep mamy dwóch podłączonych klientów

3.3. Protokół WEP

Zostanie omówiony szczegółowo w kolejnym rozdziale.

3.4. Protokół WPA

Mechanizm szyfrowania *WPA* jest jedynym rozwiązaniem, które nie zostało zdefiniowane przez *IEEE*, a jest stosowane przez niemal wszystkich producentów urządzeń *Wi-Fi*.

Podstawą identyfikacji użytkowników są standard 802.1x oraz protokół *EAP* (*Extensible Authentication Protocol*). Zdefiniowano tu dwa tryby uwierzytelniania. W pierwszym, zwanym *Enterprise* (przeznaczonym do sieci korporacyjnych), uwierzytelnianie użytkowników odbywa się na serwerze *RADIUS (Remote Authentication Dial-in User Service).* Natomiast dla małych sieci domowych utworzono tryb *Personal* z kluczem współdzielonym *WPA-PSK (Pre-Shared Key*), wymagającym jedynie podania hasła w opcjach konfiguracyjnych każdego z urządzeń tworzących sieć bezprzewodową *Wi-Fi*.

Algorytm *RC4* nie odszedł do lamusa. Szyfrowanie danych nadal jest na nim oparte, jednak klucz na podstawie którego generowana jest sekwencja szyfrująca, to już nie duet *IV* oraz *WEP*, lecz 128-bitowy klucz, dynamicznie generowany i rozprowadzany przez protokół *TKIP (Temporal Key Integrity Protocol).*

Kolejną nowością jest kontrola integralności komunikatów *MIC (Message Integrity Check),* realizowana za pomocą zaawansowanych funkcji matematycznych, a nie prostego algorytmu *CRC-32*, przy czym *MIC* jest stosowane oprócz *CRC-32*, a nie zamiast niego.

3.5. Protokół WPA2

WPA2 – jest to nowelizacja o nazwie kodowej 802.11i protokołu *WPA*. Zasadniczą zmianą w stosunku do specyfikacji *WPA* jest rezygnacja z algorytmu *RC4* na rzecz protokołu *AES* (*Advanced Encryption Standard*). Za zarządzanie kluczami i integralność komunikatów odpowiada pojedynczy składnik używający protokołu *CCMP* (*Counter mode Cipher Block Chaining (CBC) - Message Authentication Code (MAC) Protocol).* Tutaj możemy już definitywnie stwierdzić, mechanizmy zawarte w metodzie *WPA2* gwarantują najwyższy poziom bezpieczeństwa przez uwierzytelnianie użytkowników, dobre szyfrowanie dynamicznie generowanym kluczem oraz kontrolę integralności przesyłanych danych.

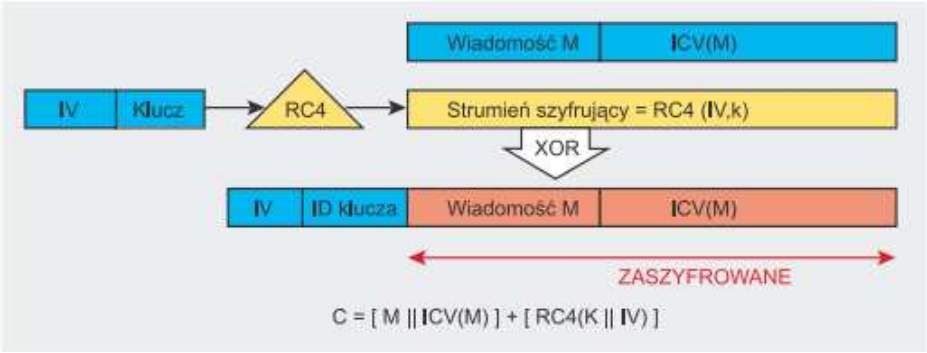
4. Wired Equivalent Privacy

4.1. Krótka historia WEP

WEP, czyli *Wired Eqivalent Privacy*, powstał w 1999 roku jako domyślny protokół pierwszego standardu *IEEE 802.11*. Opiera się on na algorytmie szyfrującym RC4, w którym rozróżniamy dwie części: tajny klucz (o długości 40 lub 104 bitów) oraz przyłączony do niego 24-bitowy *wektor inicjalizacyny* (*IV*). Tworzą one razem ciąg używany do szyfrowania tekstu jawnego *M* oraz jego sumy kontrolnej *ICV (Integrity Check Value*). Wynikowy szyfrogram *C* wyliczany jest według następującego wzoru.

$$C = [\, M \,||\, ICV(M) \,] + [\, RC4(K \,||\, IV) \,]$$

gdzie || jest operatorem konkatenacji, a + jest operatorem XOR.



Rysunek 4.1 Protokół szyfrujący WEP (źródło: hakin9)

Zauważalny jest fakt, iż bezpieczeństwo transmisji *WEP* zależy od wektora inicjalizacyjnego. W celu zachowania odpowiedniego poziomu bezpieczeństwa i zminimalizowania ujawnień powinien być zwiększany dla każdego pakietu tak, by każdy kolejny pakiet był szyfrowany już innym kluczem. Niestety, ale *IV* jest przesyłany otwartym tekstem, a na domiar złego standard *802.11* nie przewiduje obowiązkowej jego inkrementacji. W efekcie dostępność tego zabezpieczenia zależy od implementacji standardu na konkretnej stacji bezprzewodowej (punkcie dostępowym lub karcie bezprzewodowej).

4.2. Algorytm szyfrujący

Twórcami protokołu *WEP* nie byli specjaliści w dziedzinie bezpieczeństwa i kryptografii. Wkrótce po jego wprowadzeniu okazało się, że opisane cztery lata wcześniej słabości algorytmu *RC4* są aktualne także tutaj. W 2001 roku został opublikowany głośny artykuł o *WEP*, traktujący o wrażliwości i podatności algorytmu szyfrującego *RC4* na dwa rodzaje ataków: atak na niezmienność klucza i atak ze znanym *IV* (autorami artykułu byli Scott Fluhrer, Itsik Mantin oraz Adi Shmir).

Oba ataki wykorzystywały fakt, iż dla niektórych wartości klucza początkowe bajty strumienia mogły być zależne od kilku bitów klucza szyfrującego – choć teoretycznie każdy bit strumienia powinien był różnić się od poprzedniego z prawdopodobieństwem 50%. Klucz szyfrujący w przypadku *WEP* jest tworzony poprzez scalenie klucza tajnego z *IV*, zatem w rzeczy samej dla niektórych wartości *IV* istnieją klucze słabe.

Wyżej wymienione podatności na ataki zostały wykorzystane w praktyce przez narzędzia takie jak *Aircrack-ng*, potrafiące odtworzyć klucze *WEP* na podstawie analizy dostatecznie dużej liczby pakietów (*#Data*). W ruchliwej sieci taki atak był kwestią kilku minut, natomiast w sieci o małym ruchu wymagało to dużo cierpliwości. Została jednak opracowana zoptymalizowana wersja ataku. David Hulton (h1kari) uwzględnił w wyliczeniach nie tylko pierwszy bajt wyniku *RC4* (jak to miało miejsce w metodzie *FMS*), ale również kolejne bajty, co pozwoliło na zmniejszenie ilości danych niezbędnych do odtworzenia klucza.

Etap sprawdzania integralności posiada jeszcze jedną istotną słabość wynikającą z użycia *CRC32* jako algorytmu sumy kontrolnej. *CRC32* jest wprawdzie często używany do wykrywania błędów transmisji, ale liniowość przetwarzania dyskwalifikowała go jako algorytm kryptograficznie bezpieczny. Już cztery lata temu dowiedli tego Nikita Borysow, Ian Goldberg oraz David Wagner. Po tych odkryciach powszechnie zostało przyjęte, że oferowany przez *WEP* poziom bezpieczeństwa nadaje się jedynie dla użytkowników domowych i aplikacji bez znaczenia krytycznego.

Po pojawieniu się w 2004 roku ataków KoreKa (uogólnione ataki *FMS* korzystające z optymalizacji h1kariego) oraz odwrotnego ataku indukcyjnego Arbaugha (pozwalającego na deszyfrowanie dowolnych pakietów bez znajomości klucza z wykorzystaniem techniki wstrzykiwania pakietów), nawet powyższe zastrzeżenie straciło rację bytu. Narzędzia implementujące te techniki, na przykład *Aircrack-ng* autorstwa Christophe'a Devine'a czy WepLab José Ignacia Sáncheza, potrafią odtworzyć 128-bitowy klucz *WEP* w zaledwie

10 minut (czasem trochę dłużej, w zależności od konkretnego punktu dostępowego i karty sieciowej). Dodanie wstrzykiwania pakietów znacznie skróciło czas łamania zabezpieczeń *WEP*, gdyż odtworzenie klucza nie wymagało już milionów, a zaledwie tysięcy pakietów o różnych *IV* – około 150 000 dla 64-bitowego klucza *WEP* i 500 000 dla klucza 128-bitowego. Technika wstrzykiwania pozwala zebrać potrzebne dane dosłownie w kilka minut. Od tej chwili stało się oczywistym, iż protokół *WEP* jest zatem nieodwołalnie martwy (patrz Tabela 4.1) i nie należy go używać, nawet w przypadku stosowania rotacji kluczy.

Data	Opis
wrzesień 1995	Odkrycie potencjalnej słabości algorytmu RC4 (Wagner)
październik 2000	Pierwsza publikacja opisująca podatności WEP: *Unsafe at any key size; An analysis of the WEP encapsulation* (Walker)
maj 2001	Indukcyjny atak z wybranym tekstem jawnym na WEP/ WEP2 (Arbaugh)
lipiec 2001	Atak na CRC z przerzucaniem bitów – *Intercepting Mobile Communications: The Insecurity of 802.11* (Borisov, Goldberg, Wagner)
sierpień 2001	Ataki FMS – *Weaknesses in the Key Scheduling Algorithm of RC4* (Fluhrer, Mantin, Shamir)
sierpień 2001	Publikacja AirSnorta
luty 2002	Zoptymalizowane ataki FMS autorstwa *h1kari*
sierpień 2004	Ataki KoreKa z niepowtarzalnymi WI, publikacja narzędzi chopchop i chopper
lipiec/sierpień 2004	Narzędzia Aircrack (Devine) i WepLab (Sanchez) implementujące ataki KoreKa

Tabela 4.1 Karta chorobowa protokołu WEP (Źródło: hakin9)

Wady bezpieczeństwa protokołu *WEP* można podsumować następująco:

- metoda generowania klucza – choroba genetyczna przeniesiona z algorytmu *RC4*,
- wektor inicjalizacyjny jest zbyt krótki (dla wersji 24-bitowej wystarczy niecałe 5000 pakietów, by osiągnąć 50% prawdopodobieństwo kolizji) i umożliwienie powtórnego użycia identycznego *IV*
- brak rzetelnego testowania spójności (algorytm *CRC32* absolutnie się do tego nie nadaje, zdecydowanie lepiej spisuje się przy wykrywania błędów, ale ze względu na swą liniowość nie jest kryptograficznie bezpieczny),
- brak domyślnie zaimplementowanej metody aktualizacji kluczy.

5. Zadanie do samodzielnego wykonania

WEP jest najbardziej popularnym (i jednocześnie najbardziej dziurawym) szyfrowaniem stosowanym w sieciach bezprzewodowych. Właśnie z takim zabezpieczeniem będziecie mieli Państwo do czynienia w poniższym zadaniu. Korzystając z programu *aircrack-ng* zaimplementowanego w dystrybucję *BackTrack*, możemy bardzo szybko odtworzyć klucz *WEP*, o ile tylko posiadamy wystarczającą ilość słabych wektorów *IV* w przechwyconych pakietach. W naszym przypadku będzie to szyfrowanie 128-bitowe, do złamania którego będziemy potrzebowali ok. 50 000 *IV*.

5.1. Narzędzia

Zadanie będziemy realizowali przy użyciu *BackTrack 4 R2* z poziomu LiveDVD lub LivePendrive. Pakietem programów, z którego będziemy korzystać, jest *aircrack-ng*. *Aircrack-ng* to narzędzie sieciowe służące do detekcji, przechwytywania pakietów i analizy sieci *Wi-Fi*. Umożliwia łamanie zabezpieczeń *WEP, WPA/WPA2-PSK*. Do prawidłowego działania aplikacji potrzebna jest karta sieciowa obsługująca tryb monitor.

Pakiet *aircrack-ng* zawiera:

Nazwa	Opis
aircrack-ng	Łamie Wired Equivalent Privacy metodą brute force i Wi-Fi Protected Access metodą ataku słownikowego.
airdecap-ng	Służy do odszyfrowania ruchu sieciowego, zapisanego wcześniej w postaci pliku w formacie pcap
airmon-ng	Ustawia kartę w tryb monitora
aireplay-ng	Wstrzykiwacz pakietów
airodump-ng	Sniffer przechwytuje pakiety krążące w sieci WLAN
airolib-ng	Przechowuje i zarządza ESSID oraz hasłami
packetforge-ng	Tworzy zaszyfrowane pakiety do wstrzykiwania
airbase-ng	Udaje AP
airdriver-ng	Narzędzie do zarządzania sterownikami kart Wi-Fi
easside-ng	Narzędzie do komunikowania się z AP bez klucza WEP
tkiptun-ng	Atak WPA/TKIP
wesside-ng	Odzyskiwanie klucza WEP

Tabela 5.1 Możliwości pakietu aircrack-ng (Wikipedia)

5.2. Etapy zadania

Przebieg naszego laboratorium, możemy podzielić na:

1. uruchomienie dystrybucji z LiveDVD / LivePendrive
2. wybór celu
3. zbieranie Initialization Vectors
4. łamanie klucza

5.2.1. Uruchomienie dystrybucji z LiveDVD / LivePendrive

Po uruchomieniu komputera wyposażonego w kartę sieciową bezprzewodową bootujemy naszą dystrybucję Linuxa z płyty lub pendrive'a

Następnie wybieramy domyślny sposób wyświetlania obrazu – *DEFAULT* i klikamy *ENTER*.

Po zatrzymaniu się ekranu wpisujemy: *startx*, przechodząc tym samym do trybu graficznego.

```
startx
```

5.2.2. Wybór celu

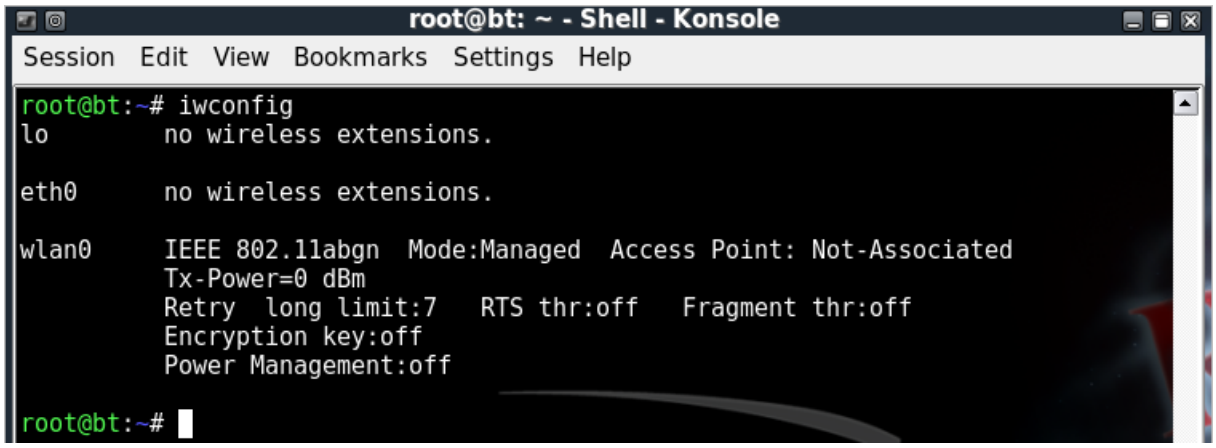
Uruchamiamy konsolę jak na Rysunku 5.1.

Rysunek 5.1 Uruchamiamy konsolę

Następnie po przejściu do okna konsoli poleceniem *iwconfig* sprawdzamy interfejs naszej bezprzewodowej karty sieciowej.

```
iwconfig
```

Powinniśmy otrzymać ekran zbliżony do poniższego (Rysunek 5.2).



Rysunek 5.2 Efekt polecenia *iwconfig*

W moim przypadku jest to *wlan0*. U każdego z wykonujących ćwiczeń, może być to inny interfejs. Do dalszych poczynań, należy zapamiętać swój interfejs.

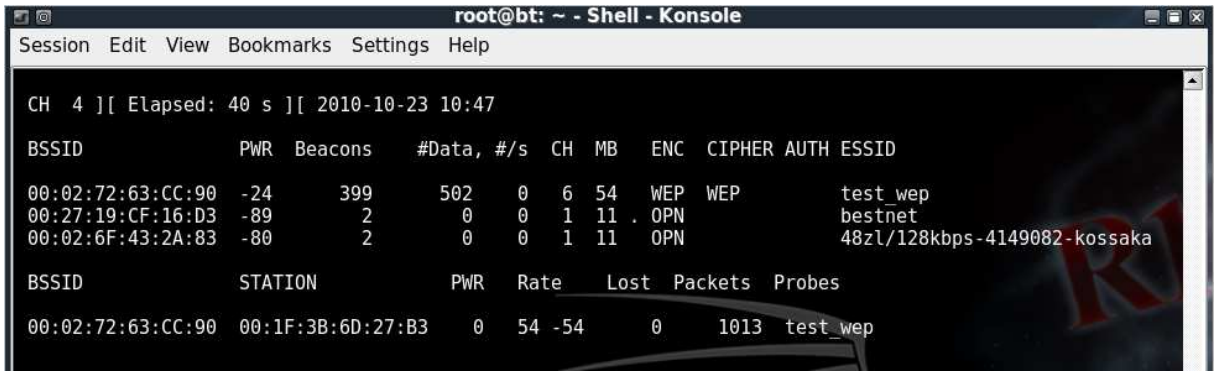
Następnie przełączamy naszą kartę w *tryb monitor* (monitor mode) i rozpoczniemy poszukiwanie naszego celu ataku.

```
airmon-ng start wlan0
```

Od teraz będziemy już operować nie na interfejsie *wlan0* tylko *mon0*.
Następnie, przeskanujemy naszą okolicę w celu znalezienia obiektu do ataku.

```
airodump-ng mon0
```

Otrzymamy ekran zbliżony do poniższego (Rysunek 5.3). Zaraz opiszę, co możemy z niego wyczytać.



Rysunek 5.3 Wykaz sieci które są w naszym zasięgu

Zatem, co tu widzimy i **co jest dla nas istotne**? Po kolei:

1. **BSSID** – czyli w praktyce adres *MAC AP*
2. PWR – (PoWeR) czyli siła sygnału *AP* (na kartach Intela obliczana wg wzoru n – 100, gdzie n – rzeczywista siła sygnału)
3. Beacons – liczba przechwyconych bezużytecznych pakietów
4. **#Data** – liczba przechwyconych *IV*
5. #/s – szybkość przechwytywania na sekundę
6. **CH** – informacja o kanale, na którym działa sieć
7. MB – maksymalna szybkość transmisji danych
8. **ENC** – rodzaj zabezpieczenia sieci (*OPEN/WEP/WPA/WPA2*)
9. CIPHER – ustawienia szyfrowania
10. AUTH – metoda uwierzytelniania
11. **ESSID** – czyli nazwa sieci (o ile jest dostępna)
12. STATION – urządzenie podłączone do *AP*, którego *MAC* jest po lewej
13. Probes – nazwa sieci, do której jest podpięte w/w urządzenie

Interesuje nas sieć o następujących parametrach, będziemy dalej właśnie na niej pracować:

1. BSSID –
2. CH – 13
3. ENC – WEP
4. ESSID – TEST_WEP

Przerywamy wyszukiwanie sieci poprzez kliknięcie kombinacji klawiszy CTRL+C.

Okazuje się, że do naszej sieci jest podłączony legalny użytkownik Jest to miła niespodzianka, ponieważ znacznie skróci nam to przechwytywanie pakietów.

Następnie wyłączamy tryb monitorowania.

```
airmon-ng stop mon0
airmon-ng stop wlan0
```

Po wyłączeniu trybu monitorowania, włączymy go ponownie, ale ustawiając naszą kartę sieciową już na konkretnym kanale – na kanale, na którym nadaje nasz *AP*.

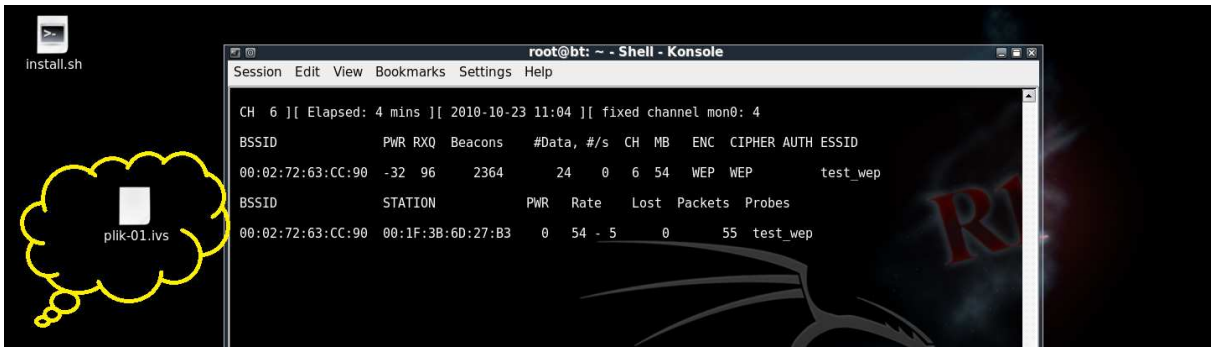
```
airmon-ng start wlan0 X ;gdzie za X podstawiamy nr kanału,
                          czyli w naszym przypadku
airmon-ng start wlan0 13
```

Po wykonaniu powyższego polecenia, nasza karta sieciowa jest ustawiona w *monitor mode* na *kanale 13.*

Niestety, ale w ramach naszego zadania, musimy trzymać się ram etycznych i prawnych, dlatego będziemy mogli jedynie przechwytywać wolno latające pakiety w eterze. **Zadanie ma na celu jedynie pokazanie, jak słabym zabezpieczeniem jest WEP, podczas gdy tylko przechwytujemy rozsiewane na lewo i prawo pakiety.** Nie możemy tworzyć fałszywych pakietów i wysyłać ich do router, nie możemy też przeprowadzić ataku polegającego na fałszywej autoryzacji i fikcyjnego podpięcia się do AP, dzięki czemu będziemy mogli się ciągle pod- oraz rozłączać, generując w ten sposób sztuczny ruch. Podkreślam, jest to nieetyczne i nie do końca zgodne z prawem. Zainteresowanych różnymi metodami ataków, odsyłam do strony zamieszczonych w rozdziale *Pomocne strony*.

Następnie, nasz *airodump-ng* będzie nasłuchiwał *AP* o *BSSID* (--bssid), na naszym kanale (--channel X), przechwytywał tylko *IV* (--ivs), a wszystko zapisywał do pliku (--write nazwa_pliku).

```
airodump-ng --channel 13 --bssid zz:xx:cc:vv:bb:nn:mm --ivs --write plik mon0
```



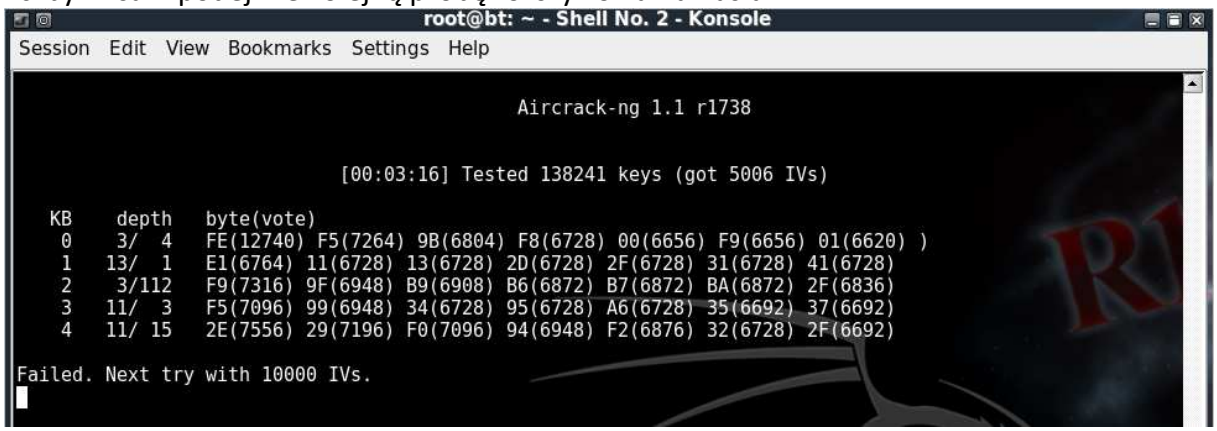
Rysunek 5.4 Rozpocznie się przechwytywanie pakietów, które nas interesują, a na pulpicie pojawi się plik-01.ivs, w którym będą przetrzymywane IV do dalszej analizy.

W trakcie zbierania pakietów (nie przerywając go), w drugim oknie konsoli uruchomimy program przeznaczony do łamania hasła – *aircrack-ng*.

Załóżymy na potrzeby tego zadania, że nasz klucz jest 128-bitowy. Polecenie będzie wyglądało następująco:

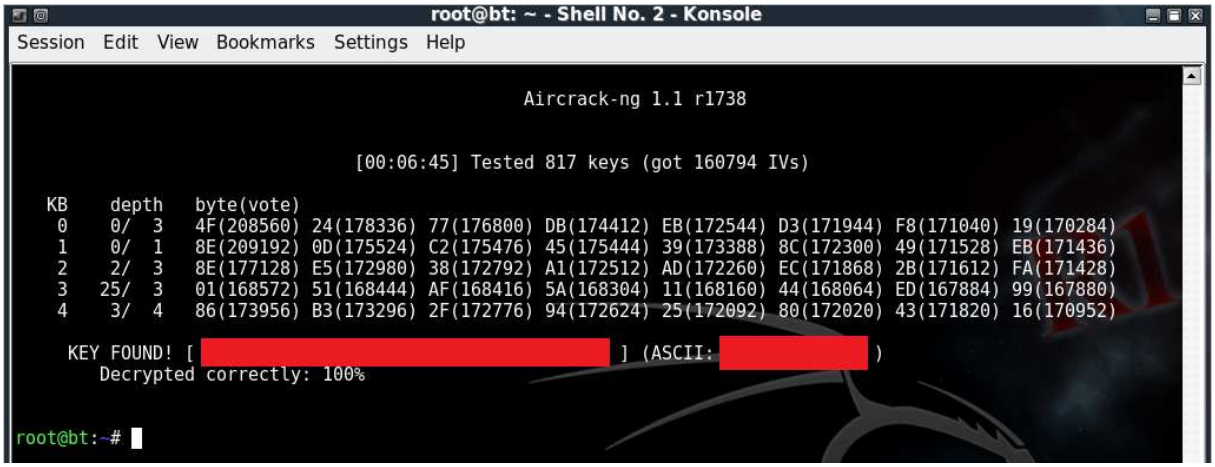
```
aircrack-ng –n 128 --bssid zz:xx:cc:vv:bb:nn:mm plik-01.ivs
```

W pierwszym oknie powinniśmy zauważyć, iż rośnie nam liczba przechwyconych *IV* (#Data). W drugim oknie konsoli, zauważamy, że *aircrack-ng* próbuje regularnie rozszyfrowywać hasło. Informuje, iż liczba zebranych *IV* jest niewystarczająca i następna próba zostanie podjęta przy zwiększonej ich liczbie. Nie musimy go restartować, po zebraniu wymaganej liczby *IV* sam podejmie kolejną próbę rozszyfrowania hasła.



Rysunek 5.5 *Aircrack-ng* informuje nas, że zebraliśmy zbyt mało IV

Jednak po chwili cierpliwości i osiągnięciu ok. 50 000 *IV*, powinniśmy ujrzeć komunikat:



Rysunek 5.6 Rozszyfrowany klucz WEP

Po wykonanym zadaniu w celu opuszczenia systemu, klikamy na *Log out*, następnie po zatrzymaniu się okna klikamy kombinacje klawiszy: *ALT+CTRL+DELETE*. Następnie usuwamy nośnik, z którego bootowaliśmy i klikamy *ENTER*.

6. Wnioski

Studenci zobowiązani są do sporządzenia wniosków po przeprowadzeniu zadania.

7. Pomocne strony

- http://aircrack-ng.org/doku.php?id=aireplay-ng#fragmentation_vs._chopchop
- http://aircrack-ng.org/doku.php?id=how_to_crack_wep_via_a_wireless_client
- http://aircrack-ng.org/doku.php?id=how_to_crack_wep_with_no_clients
- http://www.cyberbajt.pl/raport/377/
- http://www.drzewo-wiedzy.pl/?page=artykul&id=171