

# Dlaczego nie należy już używać WEPa?

---

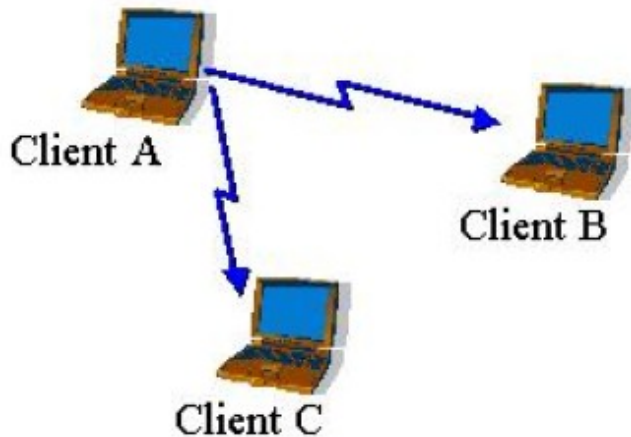
Piotr Stolarek

na podstawie opracowania:

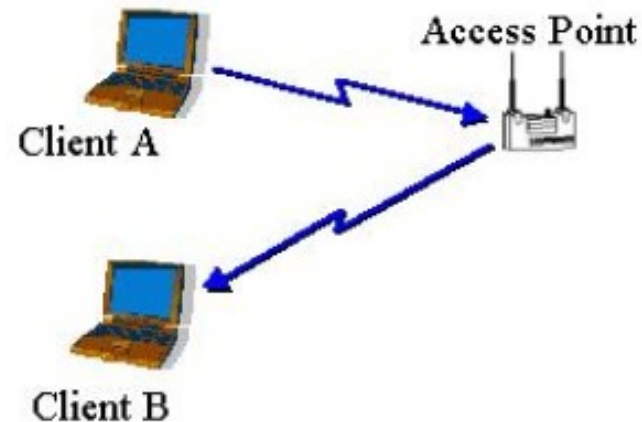
Vitaly Shmatikova

# Przegląd standardu 802.11b

- ◆ Standard sieci bezprzewodowych
  - Zatwierdzony w 1999
- ◆ 2 tryby pracy: **infrastructure** i **ad hoc**



IBSS (ad hoc) mode



BSS (infrastructure) mode

# Access Point SSID

---

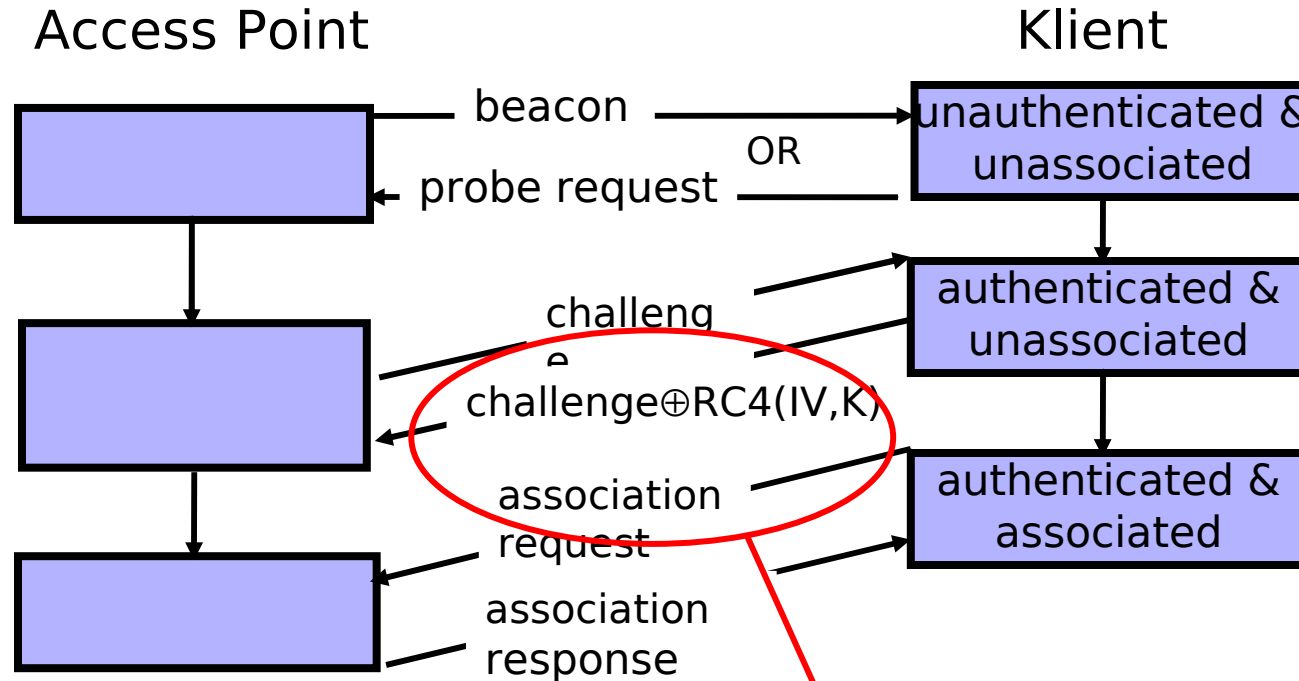
- ◆ Service Set Identifier (SSID) zapewnia unikalność nazw poszczególnych Access Pointów
  - Domyślnie AP przesyła swój SSID w tzw. “beacon frames” co każde kilka sekund
- ◆ Domyślne SSIDy są powszechnie znane
  - Domyślny SSID Linksysa to “linksys”, Cisco to “tsunami”, etc.
  - To ujawnia fakt aktywności AP
- ◆ Można skonfigurować AP tak aby nie rozsyłał swojego SSID oraz zmienić jego domyślną nazwę
  - Jednak wtedy każdy użytkownik musi wcześniej znać SSID

# Wired Equivalent Privacy (WEP)

- ◆ Sporządzony specjalnie na potrzeby standardu 802.11b
  - Miał zapewnić sieci bezprzewodowej taką samą ochronę jak sieci ethernetowej
- ◆ Cele: poufność, integralność, uwierzytelnianie
- ◆ Polega na dzieleniu klucza przez AP i klienta
- ◆ Używa szyfratora strumieniowego jako zarodka dla RC4 wraz z 24-bitowym Initialization Vector'em i 40-bitowym kluczem
  - Fatalny wybór dla środowiska bezprzewodowego
  - W SSL ta technologia została zaimplementowana znacznie lepiej

# Shared-Key Authentication

Przed wymianą danych, AP żąda autentykacji klienta



Pasywny podsłuchiwacz, który uzyska RC4(IV, K), może odpowiedzieć na każdy "challenge" bez znajomości K



# Dlaczego RC4 to zły wybór dla WEP?

- ◆ Szyfratory strumieniowe wymagają synchronizacji na obydwu końcach połączenia
  - Niedobre ze względu na liczne sytuacje zgubienia pakietów
- ◆ Rozwiązanie w WEPie: osobny zarodek na każdy pakiet
  - Można jednak odszyfrować pakiet nawet jeśli poprzedni został utracony
- ◆ Niestety liczba seedów nie jest wystarczająco duża!
  - RC4 seed = 24-bit initialization vector + klucz
  - Zakładając 1500 bitowe pakiety przy prędkości 11 Mbps, mając  $2^{24}$  możliwych różnych IVsów ich liczba zostanie 'wyczerpana' w około 5h
- ◆ Powtarzalność seeda jest zabójcza dla szyfratora strumieniowego

# Odzyskiwanie klucza

- ◆ Spraw by AP zaszyfrował znaną porcją danych
  - Wyślij np. spam aby AP go zaszyfrował
  - Spraw ofiara przesłała wiadomość o znanej treści
- ◆ Jeśli osoba atakująca zna treść wiadomości przed zaszyfrowaniem oraz po, bardzo łatwo jest odzyskać klucz
  - $C \oplus M = (M \oplus RC4(IV, \text{klucz})) \oplus M = RC4(IV, \text{klucz})$
  - Nie problem jeśli klucz kodujący tzw. 'keystream' jest unikalny - niestety nie ma to miejsca w WEP
- ◆ Nawet jeśli atakujący nie zna wiadomości oryginalnej, może korzystać z pewnych regularności (przecież zwykłe wiadomości nie mają charakteru losowego)
  - Np. struktura pakietów IP jest regularna

# Powtarzalny keystream – główny problem WEP

- ◆ W WEPie powtórzony IV to tak naprawdę powtórzony keystream
- ◆ Zajęta sieć będzie bardzo często powtarzać IVsy
  - Wiele kart resetuje IV na 0 po restarcie, kolejno zwiększając tę wartość o 1  $\Rightarrow$  dlatego spodziewajmy się IVsów o małych wartościach
  - Nawet jeśli IVsy są rzeczywiście o losowych wartościach możemy się spodziewać ich kolizji  $O(2^{12})$
- ◆ Odzyskaj keystream dla każdego IV, przechowaj w tabeli
  - $(Z_{\text{zaneM}} \oplus \text{RC4}(\text{IV}, \text{klucz})) \oplus Z_{\text{zaneM}} = \text{RC4}(\text{IV}, \text{klucz})$
  - Gdy nieznane jest M korzystamy z regularności
- ◆ Poczekaj na powtórzone IVsy, dokonaj deszyfrowania i ciesz się oryginalną wiadomością:)

# Dłuższy klucz nie pomoże...

- ◆ Niedostosowanie kodera RC4 do transmisji pakietowej nie otrzymało żadnej poprawki
  - Dłuższe klucze nie pomogą!
    - Problem to powtarzalne IVsy, ich rozmiar jest stały (24 bits)
  - Ataki są pasywne i praktycznie niemal niemożliwe do wykrycia
- ◆ Atak Fluhrer'a na RC4
  - Wymaga zgromadzenia IVsów w odpowiedniej formie
  - WEP wysyła IVsy w postaci niezaszyfrowanej
  - Wygenerowanie 'odpowiednich' IVsów to obecnie kwestia minut
- ◆ Jako rezultat otrzymujemy klucz

# Część praktyczna...

[Brian Lee]

Składniki: Laptop (z kartą 802.11b, GPS, Netstumbler, Aircsnort, Wireshark) i samochód dowolnej marki...

- ◆ Laptop, palmtop np. Zurus (z wbudowanym Linuxem)
- ◆ Kismet – linuxowy odpowiednik Netstumblera, po odpowiednim podłączeniu modułu GPS i konfiguracji umożliwia generowanie trójwymiarowych map
- ◆ Zamiast Aircsnorta użyć pakietu Aircrack i samodzielnie sterować poszczególnymi etapami łamania klucza WEP (airodump, aireplay, aircrack). Byłby on także niezbędny do próby ataku słownikowego na WPA
- ◆ Alternatywa: użyć np. Wiresharka lub innego sniffera i samodzielnie analizować pakiety

# Rozwiązania...nie do końca najlepsze

---

- ◆ Utworzenie tunelu VPN
  - Traktujemy sieć bezprzewodową jak zwykłą niezabezpieczoną sieć ethernetową
- ◆ Ukrywajmy SSID naszych APków
  - Niestety, zbiór surowych pakietów wyjawi to odrobinę bardziej doświadczonemu intruzowi (te pakiety nie są szyfrowane)
- ◆ Przydzielajmy dostęp do sieci tylko klientom o konkretnym adresie MAC oraz ściśle z nim powiazanym adresie IP
  - Praktycznie niewykonalne lub przynajmniej kłopotliwe w większych sieciach
  - Zabazpieczenie to bardzo łatwo jest obejść (MAC oraz IP spoofing)

# Obowiązujące standardy

---

- ◆ Extensible Authentication Protocol (EAP)
  - Kilka metod uwierzytelniania bazujących na EAP
    - Cisco EAP-LEAP (hasło), EAP-TLS (certyfikaty z kluczem publicznym), PEAP (hasła lub certyfikaty), etc.
- ◆ 802.11i – częściowo naprawia błędy 802.11b
  - Patch: TKIP. Wprowadzie ciągle obecny RC4, ale IVsy są szyfrowane oraz ustanawiane nowe klucze transmisji co każde 10KB
    - Tym razem keystream się już nie powtarza!
    - Możliwy upgrade firmwaru starszych kart
  - WPA2-PSK (Pre-Shared Key)
    - Obecny standard bezpieczeństwa sieci bezprzewodowych
    - Potrzebne już 'nowe' karty obsługujące standard WPA2

# Four-way key handshake

**802.11 Station**

MAC Address=**SA**

Pairwise Master Key (**PMK**) Is  
Known; Generate **SNonce**

1. EAPOL-Key (**ANonce**)

Derive Pairwise Transient Key (**PTK**)  
using **ANonce** and **SNonce**

2. EAPOL-Key (**SNonce, MIC**)

Derive **PTK** using **ANonce** and **SNonce**  
Supply Group Transient Key (**GTK**)

3. EAPOL-Key (**MIC, Encrypted GTK**)

4. EAPOL-Key (**MIC**)

Install **PTK** and **GTK**

Install **PTK**

Encrypted data can now be sent / received

$$\text{PTK} \leftarrow \text{PRF-X}(\text{PMK}, \text{"Pairwise key expansion"},$$
  
$$\text{Min}(\text{AA}, \text{SA}) \parallel \text{Max}(\text{AA}, \text{SA}) \parallel \text{Min}(\text{ANonce}, \text{SNonce}) \parallel \text{Max}(\text{ANonce}, \text{SNonce}))$$

# Podsumowanie oraz kilka ewentualnych opcji...

- ◆ AES zamiast RC4
- ◆ CCMP zamiast CRC-32
- ◆ Stosować WPA-PSK lub RADIUS
- ◆ Nie rozsyłać SSID!
- ◆ Ustawić firewall
- ◆ Jeśli już WEP to + IPSec
- ◆ Ustawić silne hasło
- ◆ Postawić tzw. 'Fake Access Point'

# Naprawdę przydatne linki



- ◆ <http://www.remote-exploit.org/>
- ◆ <http://wirelessdefence.org/>
- ◆ <http://www.wardriving.pl/>
- ◆ <http://www.wi-fiplanet.com/tutorials/article.php/3667586>