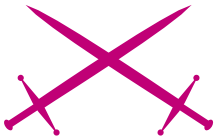



# Atak na Bluetooth

Ugo Lopez

stopień trudności

**Bluetooth jest technologią, która powstała by ułatwić nasze zdolności komunikowania się. Okazał się jednak także technologią nadającą się do kradzieży danych. W tym artykule przedstawimy Wam jak wykorzystać jego słabe punkty.**

Któż nie pamięta czasów, nie tak znowu odległych, kiedy to telefon komórkowy był postrzegany z pewną dozą nieufności bądź zazdrości? Uznawany pierwotnie bardziej za symbol statusu niż za narzędzie komunikacji, telefon komórkowy stanowi już od dłuższego czasu nieodłączny element codziennego życia każdego z nas.

Technologia Bluetooth, powstała by pokonać ograniczenia portów na podczerwień IRDA (Infra Red Device Adapter) i FastIRDA, gdzie urządzenia peryferyjne musiały być wzajemnie widoczne i oferowały niskie prędkości przesyłowe, rozwija się ponad wszelkie przewidywania. Jak w przypadku każdej rzeczy na polu informatycznym, gdy postęp dokonuje się w sposób tak gwałtowny, bezpieczeństwo pozostaje daleko w tyle. Rzeczywiście, liczne są słabości obecne w Bluetooth. Lecz po kolei, najpierw postarajmy się opisać w miarę szczegółowo tę technologię.

## Technologia Bluetooth

Bluetooth powstał w 2003 roku. Z inicjatywy wielu producentów stanowiących konsorcjum SIG (Ericsson, Nokia, Microsoft, Intel, Motorola, Apple i innych), Bluetooth jest obecny niemalże we wszystkich urządzeniach przenośnych (telefony komórkowe, słuchawki, nawigatory satelitarne, drukarki, itd.). Technologia ta pozwala tworzyć prawdziwe PAN (Personal Area Network) w sposób umożliwiający wzajemne korzystanie z zasobów pomiędzy urządzeniami, nie koniecznie jednakowymi, czyniąc maksymalnie prostym współdziałanie z użytkownikiem.

Aktualnie istnieją 2 standardy stosowane w urządzeniach peryferyjnych obecnych na rynku:

### Z artykułu dowiesz się...

- jakie są słabe punkty niektórych urządzeń Bluetooth i jak je wykorzystać.

### Co powinieneś wiedzieć...

- znajomość na poziomie użytkownika systemów Windows,
- znajomość na poziomie użytkownika systemów Linux, zastosowanie powłoki shell,
- znajomość na poziomie użytkownika zaawansowanego mobilnych urządzeń Bluetooth.

- Core Specification 1.2 – 5 listopada 2003,
- Core Specification 2.0 + EDR – 15 października 2004.

Oprócz nich istnieją również standardy już zaniechane. A dokładniej:

Bluetooth 1.0 i 1.0B: wersje 1.0 i 1.0B nęka wiele problemów, przede wszystkim związanych ze współdziałaniem produktów odmiennych konstruktorów. Pomiędzy tymi dwoma standardami dokonano zmian w procesie weryfikacji adresu fizycznego przypisanego każdemu urządzeniu: stara metoda uniemożliwiła pozostawanie anonimowym podczas komunikacji, stąd jakiś złośliwy użytkownik wyposażony w skaner częstotliwości mógł przechwycić ewentualne poufne informacje. Wersja B przyniosła także zmiany związane z obsługą środowiska Bluetooth usprawniając możliwość współdziałania,

Bluetooth 1.1: naprawia wiele błędów, które pojawiły się w wersji 1.0B. Pozwala na komunikację przy wykorzystaniu kanałów niekodowanych.

Obecnie standard 1.2, kompatybilny z wersją 1.1, przewiduje: *Adaptive Frequency Hopping* (AFH): ta technika zapewnia większą odporność na interferencje elektromagnetyczne, gdyż unika stosowania kanałów podatnych na silne interferencje, zwiększa szybkość transmisji, *extended Synchronous Connections* (eSCO): oferuje tryb transmisji audio wysokiej jakości, przesyłając ponownie dane w razie ich utraty. Został także wprowadzony czujnik jakości sygnału. Posiada interfejs pozwalający na obsługę aż trzech *UART* (standard symulujący obecność połączenia kablowego) i na dostęp do informacji związanych z synchronizacją transmisji Bluetooth.

Natomiast standard 2.0, kompatybilny ze wszystkimi wcześniejszymi standardami, zapewnia następujące usprawnienia: z motywów bezpieczeństwa unika przeskakiwania pomiędzy kanałami. Bezpieczeństwo zostaje zapewnione poprzez algorytmy kryptograficzne, obsługuje zarówno *multicast* jak i *broadcast*, zwiększa szybkość transmisji do 2,1 Mbit/s (3 we wspomnianej wcześniej wersji z 15 października 2004), wprowadza usługę obsługi jakości. Wprowadza także protokół umożliwiający dostęp do urządzeń współdzielonych, redukuje zauważalnie czasy odpowiedzi i zmniejsza o połowę wykorzystywaną moc.

Jest już w fazie przygotowania nowa wersja Bluetooth (o nazwie *Lisbon*) przewidująca: *Atomic Encryption Chance*: okresowa zmiana hasła dla połączeń kodowanych, *Extended Inquiry Response*: dostarcza więcej informacji o urządzeniach usiłujących ustanowić połączenie, faworyzując tym samym filtering urządzeń niepewnych, *Sniff Subrating*: system redukcji zużycia w stanie sniffingu, poprawa *QoS* (Quality of Service), *Simple Pairing*: poprawa kontroli strumieni bitów poprzez mechanizmy parowania.

Istnieje także kolejna wersja (o nazwie *Seattle*) przewidująca wprowadzenie jako znaczącą innowację *Ultra Wide Band* (UWB), co pozwoli na znaczące zwiększenie prędkości.

Wreszcie, urządzenia Bluetooth można podzielić na 3 klasy (zobacz Tabela 1).

W celu zapoznania się z innymi szczegółami tych standardów można odwiedzić oficjalną stronę poświęconą technologii Bluetooth i ściągnąć dokumenty specyfikacji (*http://www.Bluetooth.com*).

Rzućmy teraz okiem jak funkcjonuje, zgrubsza, stos protokołów Bluetooth (Rys. 1)

W gruncie rzeczy możemy wyróżnić w nim dwie części:

**Tabela** 1. Trzy klasy Bluetooth

| Klasa | Moc(mW) | Moc (dBm) | Odległość(Przybliżona) |
|---|---|---|---|
| Klasa 1 | 100 mW | 20 dBm | ~ 100 metrów |
| Klasa 2 | 2,5 mW | 4 dBm | ~ 10 metrów |
| Klasa 3 | 1 mW | 0 dBm | ~ 1 metr |

- *Host protocols*: implementowane na poziomie oprogramowania komunikują się, poprzez API, z aplikacjami; obsługują funkcje wyższego poziomu,
- *Controller protocols*: obsługują moduł radiowy.

Oba składniki komunikują się ze sobą poprzez *HCI* (Host Controller Interface), który jest odpowiedzialny za definiowanie zbioru wiadomości i zbioru sposobów ich transportowania.

Teraz przyjrzyjmy się bardziej szczegółowo stosowi protokołów (Rys. 2).

Jest to schemat *ostateczny* stosu protokołów. W celu zapewnienia wymiany z innymi protokołami jego zasadniczymi składnikami są:

- *L2CAP* (Logical Link Control and Adaptation Protocol): kapsułkuje pakiety i zapewnia mechanizm abstrakcyjny przypominający koncepcję portów w TCP/IP,
- *SDP* (Service Discovery Protocol): upublicznia usługi oferowane przez poszczególne urządzenia i wyszukuje usługi oferowane przez urządzenia z którymi chce się komunikować.

Spróbujmy teraz zaprezentować diagram funkcjonalny protokołu Bluetooth (Rys. 3).

W skrócie, można wyróżnić następujące stany: *Standby*: oczekiwanie na połączenie, *Inquiry*: wykrywanie urządzeń znajdujących się w pobliżu, *Page*: próba połączenia z urządzeniem, *Connected*: urządzenie aktywne w sieci, *Transmit* data: dane w trakcie transmisji, *Park/Hold*: tryb niskiego zużycia.

Miejmy w pamięci ten diagram funkcjonalny, ponieważ przyda się nam do zrozumienia niektórych typów ataków.

## Zasady bezpieczeństwa

Rzuciwszy okiem na specyfikacje można zauważyć, że te przewidują trzy *rodzaje* (nie)bezpieczeństwa:

- *mode 1*: brak bezpieczeństwa,
- *mode 2*: ochrona na poziomie usługa/aplikacja,

- *mode 3*: ochrona na poziomie urządzenia.

Te trzy rodzaje są implementowane w ramach czterech poziomów: *pairing*: zostaje aktywowany pomiędzy dwoma urządzeniami, które chcą aktywować procedury bezpieczeństwa i kontaktują się pomiędzy sobą po raz pierwszy; w praktyce, użytkownik wprowadza pin (identyczny) na obu urządzeniach, a z kolei każde z nich generuje pseudolosową liczbę; w tym momencie otrzymuje się *shared secret* (sekret współdzielony), który jest wykorzystywany do komunikacji, *autentyfikacja*: tzw. mechanizm *challenge* (wyzwanie); w praktyce bazuje na liczbie pseudolosowej (challenge) i na shared secret, *kodyfikacja*: ma miejsce, ewentualnie, po autentyfikacji, *autoryzacja*: określa czy żądanie urządzenia ma zostać zaspokojone czy też nie; każda aplikacja może posiadać listę urządzeń, które mogą mieć do niej dostęp (*trusted device*), jeśli dane urządzenie nie znajduje się na tej liście wymagane jest potwierdzenie ze strony użytkownika.

W ramach tych poziomów stosowanych jest pięć głównych elementów: *adres BD_ADDR*: adres fizyczny poszczególnego urządzenia (swego rodzaju MAC Address), *klucz kodujący* (8-128 bit), *klucz połączenia* (128 bit), *liczby pseudolosowe* (128 bit), *algorytmy służące do generowania kluczy* (E0, E21, E22, itd.).

Jak widzicie, łatwo jest odnaleźć wszystkie te elementy na czterech poziomach.

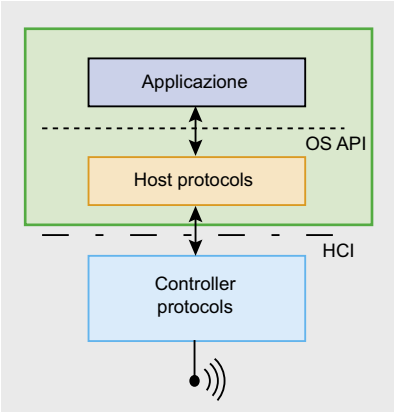
Teraz zobaczymy gdzie znajdują się słabości tego mechanizmu. Pierwsza i najbardziej oczywista tkwi w mechanizmie pairingu, który, jak zostało to opisane, przewiduje wprowadzenie pinu. Jeśli pin został określony w samym urządzeniu można wręcz przeprowadzić atak *online* typu *brute-force* (tj. bezpośrednio przeciwko urządzeniu ofiary). Natomiast w przypadku słabego pinu, można dokonać ataku *offline* (nie bezpośrednio przeciwko urządzeniu ofiary), tzw. *ataku przeciwko E22*: w skrócie, „rejestruje" się ruch na wszystkich 79 kanałach wykorzystując odpowiednie narzędzie, a następnie, offline, testuje się rozmaite klucze połączeniowe wygenerowane przy zastosowaniu rozmaitych pinów. Taki atak jest dość kosztowny za sprawą potrzebnego oprzyrządowania.

Aby bronić się przed tego typu atakiem jest dobrą normą stosowanie długich i trudnych do odgadnięcia pinów oraz wykonywanie pairingu w miejscach uznawanych za bezpieczne. Jednakże i bezpieczeństwo osiągnięte w taki sposób jest względne, gdyż, jak zostało to udowodnione, dzięki prostej modyfikacji *Bluetooth dongle* można dokonać ataku nawet z odległości przekraczającej 1,5 km. Wspaniały przykład daje nam grupa trifinite, której URL został zamieszczony w sekcji linków tego artykułu.

Poza tym istnieje szereg słabości na poziomie aplikacji, o których będziemy mówić w dalszej części tego artykułu. Te zależą bardziej od specyficznych implementacji zastosowanych przez producentów, niż od bugów w projektowaniu protokołów.

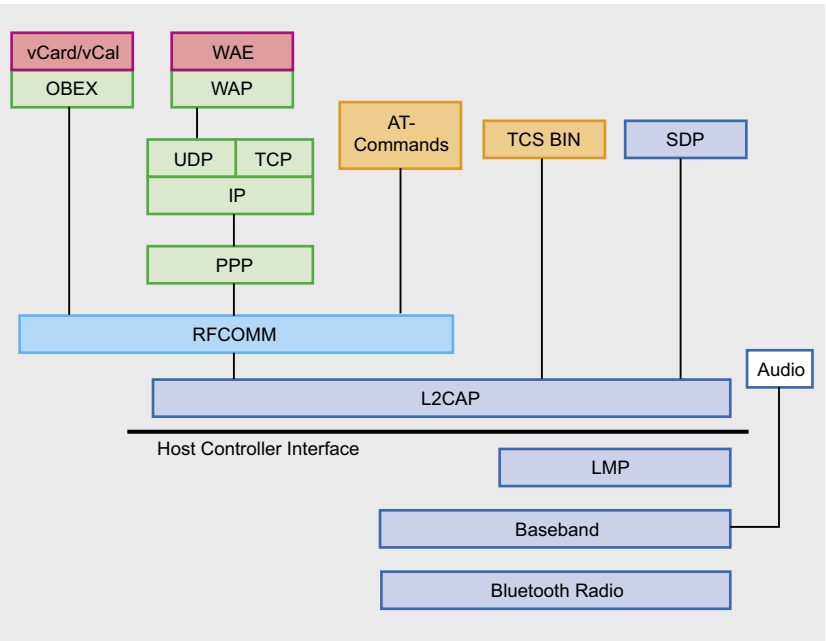
## Techniki ataku na Bluetooth

Mimo że jestem pasjonatem niemalże wszystkiego co przyniosła technologia w ciągu minionych lat, nie




**Rys. 1.** *Funkcjonowanie stosu protokołów Bluetooth*

zaliczam się do grona pierwszych uzależnionych od urządzeń przenośnych. Mówiąc szczerze, moja ciekawość została obudzona lekturą artykułu opublikowanego w jakimś dzienniku noszącego tytuł *Kiedy boli ząb...*, który traktował o *toothingu*, czyli o tym jak wykorzystując protokół Bluetooth można wysyłać wiadomości do osób kompletnie nam nieznanych, a posiadających Bluetooth device włączony i *dyspozycyjny*, w promieniu kilku metrów. Początkowo najbardziej uderzył mnie aspekt społeczny tego artykułu, lecz po chwili refleksji otworzyły się przede mną scenariusze znacznie bardziej rozległe, w których mało ostrożni użytkownicy są oszukiwani na tysiąc




**Rys. 2.** *Ostateczny schemat stosu protokołów*

i jeden sposobów przez jakiegoś przygodnego maniaka. To właśnie od tego momentu zacząłem zgłębiać dokumentacje związane z rozmaitymi typologiami ataku i obrony. Spróbujmy je wspólnie przeanalizować.

## Bluejacking

Jak wcześniej wspomniałem, zaciekawiła mnie możliwość darmowego komunikowania się za pośrednictwem systemu wiadomości z osobami kompletnie obcymi, które jednak posiadają aktywne urządzenie Bluetooth w promieniu kilku metrów. Jak to możliwe? Trzeba pamiętać, że w fazie *discovery* innych urządzeń, zostaje przekazana nazwa identyfikująca urządzenia. Nie jest ona niczym innym jak polem tekstowym. Wyobraźmy sobie teraz, że pole tekstowe zawiera coś w rodzaju: *Problemy sieciowe, proszę wybierz 1234*. Oczywiście 1234 to pin który uprzednio wystukaliśmy na naszym urządzeniu. Nieświadomy użytkownik wystuka pin i tym samym dopełni pairingu. A my dokonamy ataku bluejacking (termin ten początkowo był wykorzystywany na określenie wszystkich ataków na protokół Bluetooth). Ten typ ataku, mający swe źródło w *inżynierii socjalnej*, jest pod wieloma względami bardzo podobny do ataku określanego mianem phishing.

Poza tą *empiryczną* procedurą, istnieje wiele darmowych narzędzi, które także pozwalają na dokonanie tego ataku.

Oto niektóre z nich: Freejack (*http://www.bluejackq.com/freejack.jar*), SMAN (*http://www.bluejackq.com/sman13a-eng.zip*), Mobiluck (*http://www.mobiluck.com/download-Bluetooth-software-all-phones-en.php*), Easyjack (*http://www.getjar.com/products/2758/EasyJack*)

Internet pełen jest narzędzi przystosowanych do tego celu, tu zostały wymienione tylko niektóre z nich. Trzeba pamiętać, że rodzaj narzędzia zależy także od stosowanego telefonu komórkowego.

## Discovery mode abuse

W przypadku większości urządzeń dostępnych na rynku możliwe jest, po włączeniu, ustalenie czy usługa Bluetooth ma być *widoczna* czy *ukryta*. To co się wówczas dzieje w trybie ukrytym polega na niczym innym jak na odrzucaniu przez urządzenie wszelkich żądań inquiry pochodzących w broadcast od innych aparatów Bluetooth. Tak więc usługi nie zostają całkowicie dezaktywowane, lecz jedynie odrzucane są żądania kierowane do SDP. Poza tym, jak już powiedzieliśmy, każde urządzenie posiada swój BD_ADDR: składa się z 48 bitów, z których pierwsze 24 zależą wyłącznie od producenta (swego rodzaju *vendor code*), są więc stałe. Z pozostałych jedynie ostatnich 6 bitów identyfikują w sposób jednoznaczny urządzenie, jako że służą do identyfikacji typu urządzenia (komórka, dongle, słuchawka, itd.). Dlatego nie jest wcale rzeczą trudną wykrycie urządzenia Bluetooth, także w trybie ukrytym. Rzeczywiście, z punktu widzenia obliczeniowego, można odnaleźć bity zmienne w czasie niewiele dłuższym niż jedna godzina.

Narzędzia służące do takiej operacji, w chwili redagowania niniejszego artykułu, są dostępne tylko pod Linuksem. Są to:

- Redfang (*http://www.securitywireless.info/Downloads-index-req-getit-lid-41.html*),
- Bluesniff (*http://bluesniff.shmoo.com/bluesniff-0.1.tar.gz*).

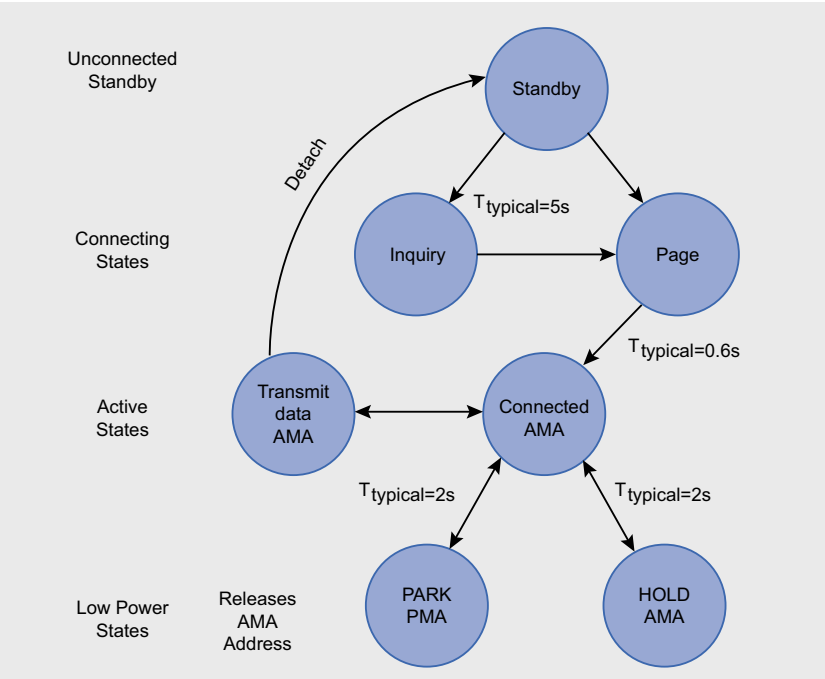
Redfang jest narzędziem linii poleceń zrealizowanym przez @Stake, aktualnie stanowiącym część Symanteca, i stanowi POF (Proof Of Concept) techniki ataku już opisanej. Bluesniff to front-end graficzny do Redfang.

## Blueprinting

Jest rodzajem nmap w odniesieniu do Bluetooth. Technika ta pozwala na uzyskanie informacji technicznych o badanym urządzeniu, wykonując matching uzyskanych charakterystyk z tymi obecnymi w uaktualnionej bazie danych. Także w tym przypadku istnieje narzędzie pod Linuksa obsługiwane z linii poleceń:

- Blueprint (*http://trifinite.org/Downloads/bp_v01-3.zip*)

Także Redfang, przedstawiony we wcześniejszym paragrafie, zajmuje się blueprintingiem.




**Rys. 3.** *Diagram funkcjonalny protokołu Bluetooth*

## Bluesnarf

Jest atakiem wywodzącym się z wadliwej implementacji specyfikacji wielu telefonów komórkowych (liczne modele Ericsson, Sony-Ericsson, Nokia, Siemens, Motorola). Z bardziej szczegółowym wykazem urządzeń peryferyjnych w to zamieszanych można zapoznać się tu: *http://www.thebunker.net/security/Bluetooth.htm.*

Lecz na czym polega ten atak? Na niczym innym jak na łączeniu się z usługą OBEX Push (często wykorzystywana w celu wymiany elektronicznych wizytówek). Wadliwa implementacja w niektórych telefonach komórkowych pozwala, poza otrzymywaniem wizytówek, także na *OBEX Get*, czyli na żądanie pliku. Tzn., jeśli wiem że na komórce ofiary jest obecny plik *chcęgo.jar*, to mogę go ściągnąć *przeskakując* fazę autentyfikacji. Często nie trzeba nawet znać ścieżki pliku w systemie, ponieważ wiele aparatów zapamiętuje informacje odnoszące się do systemu plików w pliku tekstowym, którego lokalizacja jest znana a priori, gdyż zależy od systemu. Na przykład, komórki Ericsson i Sony-Ericsson pierwszej generacji zapisują książkę telefoniczną w telecom/pb.vfc, a kalendarz w *telecom/calc.vcs*.

W celu dokonania tego typu ataku wystarczy jakikolwiek client OBEX. Oto niektóre:

- obexftp (*http://openobex.triq.net/obexftp/installing*),
- obex-commander (*http://intradarma.com/OBEXCommander.html*).

Pierwszy z tych dwóch clientów jest pod Linuksa, drugi pod Windows.

## Bluesnarf++

Bardzo podobny do Bluesnarf, lecz pozwala na pełen dostęp w zakresie odczytu/zapisu do systemu plików, bez konieczności pairingu.

Tu znajdziecie narzędzie do realizacji tego rodzaju ataku:

- Bluediving (*http://bluediving.sourceforge.net/*)

To narzędzie jest prawdziwą suite, jest niezmiernie użyteczne w celu wykonania bardzo złożonych pentestów. Na stronie znajduje się cały szereg innych, na prawdę użytecznych narzędzi.

## Bluebug

Także ta słabość wynika ze złej implementacji niektórych specyfikacji i jest obecna tylko w niektórych modelach przenośnych urządzeń peryferyjnych. W odróżnieniu od Bluesnarf i Bluesnarf++, pozwala na wykonanie poleceń AT na urządzeniu ofiary. Tym razem problem dotyczy usług na kanale RFCOMM nie zgłoszonych przez SDP, lecz mimo to wykorzystywanych.

W praktyce, atakujący może uzyskać pełen dostęp do telefonu komórkowego, zwłaszcza może: telefonować, wysyłać, czytać i eliminować SMS/MMS, czytać i pisać w książce telefonicznej, zmieniać parametry konfiguracyjne.

Aby wykazać istnienie tej dziury, poza wcześniej wspomnianym Bluediving, możemy wykorzystać BlooooverII (*http://trifinite.org/trifinite_stuff_bloooverii.html*). Pozwala on na przeprowadzenie także innych ataków, m.in. *Helomoto*, w gruncie rzeczy będącego połączeniem ataków Bluesnarf i Bluebug: przerywa otrzymywanie Vcard i, za sprawą błędu implementacyjnego, urządzenie pozostaje w trybie trusted. Ciekawostka, nazwa wywodzi się z faktu, że jest to słabość typowa dla systemów Motorola.

## Bluesmack

Jest to atak typu DOS, i nie jest niczym innym jak *Ping of Death* w odniesieniu do Bluetooth. Polega na zwiększaniu ponad miarę echo request (L2CAP ping) mającego być wysłanym w kierunku urządzenia ofiary. Niektóre terminale odbierają dane lecz jednocześnie generują błędy blokując zupełnie komórkę (niektóre Compaq iPaq, na przykład).

Możemy przeprowadzić nasze próby bądź przy pomocy niezwykle użytecznego *szwajcarskiego scyzoryka* Bluediving bądź ściągając jedynie biblioteki Bluetooth dla Linux Bluez (*http://www.bluez.org/download.html*, wszelako niezbędne także dla Bluediving) i formatując odpowiednio polecenie l2ping. W przypadku wielu modeli iPaq wystarczy napisać:

```
l2ping -s <num_byte>
z <num_byte> większym lub równym 600.
```

## Bluebump

Jest to atak inżynierii socjalnej. Atakujący ustanawia połączenie trusted z jakimś urządzeniem, na przykład wysyłając Vcard i zmuszając odbiorcę do autentyfikacji (*Mode-3-Abuse*). Atakujący podtrzymuje otwarte połączenie i mówi ofierze by ta przerwała połączenie ze swoim urządzeniem peryferyjnym. Oczywiście ofiara nie jest świadoma że połączenie jest jeszcze aktywne. W tym momencie atakujący prosi by ponownie został wygenerowany klucz połączeniowy. Tym samym posiada ofiarę na swojej liście, nie musząc ponownie przechodzić autentyfikacji. Atakujący może połączyć się z ofiarą dopóki ta nie wykasuje także tego nowego klucza.

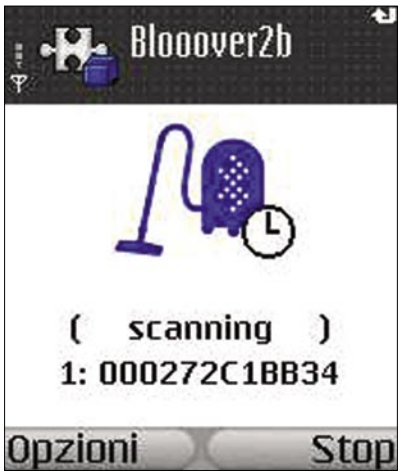
## Bluedump

Wykorzystując sniffer Bluetooth, można wykonać dumping pinów i niektórych kluczy podczas sesji Bluetooth. Atakujący musi znać BD_ADDR jakiejś pary urządzeń będących w pairingu; w tym momencie atakujący *spoofuje* (wciąż poprzez Bluediving, jeśli to możliwe) BD_ADDR jednego z dwóch urządzeń i łączy się z drugim. Kiedy ofiara przechodzi do autentyfikacji, zważywszy że atakujący nie posiada klucza połączeniowego, jego urządzenie odpowiada poprzez '*HCI_Link_Key_Request_Negative_Reply*' co, w niektórych przypadkach, prowadzi do wykasowania klucza połączeniowego na urządzeniu ofiary i do kolejnego pairingu z atakującym.

## Bluechop

Atak pozwalający obalić całą piconet. Urządzenie peryferyjne względem piconet spoofuje urządzenie


**Rys. 4.** *Uruchamiamy BloooverII*




**Rys. 5.** *Find devices*


**Rys. 6a.** *Ustawianie głównych parametrów*

slave spoza piconetu a następnie kontaktuje się z masterem tejże piconet. Ponieważ protokół przewiduje że slave przystąpi do piconetu sam w następstwie żądania mastera, ten gubi się i powoduje upadek piconetu. Także do tego ataku jest przydatne narzędzie do spoofingu jak Blueediving. To prawdopodobnie jedyny atak na protokół Bluetooth który nie wykorzystuje złych implementacji stosu Bluetooth, ponieważ uderza w urządzenia wszystkich producentów.

## Car whisperer

Atak na urządzenie peryferyjne Bluetooth audio wewnątrz samochodu. Grupa trifinite.org przeprowadziła go aby wyczulić producentów aut na kwestie bezpieczeństwa urządzeń peryferyjnych Bluetooth wewnątrz pojazdu. Jako narzędzie do przeprowadzenia ataku carwhisperer (*http://trifinite.org/trifinite_stuff_carwhisperer.html*) mogą posłużyć niektóre spośród narzędzi dotychczas przez nas poznanych, ewentualnie odpowiednio zmodyfikowanych. Technika ataku opiera się na fakcie, że urządzenia peryferyjne Bluetooth wewnątrz pojazdu posiadają gotowe klucze, często wręcz znane ponieważ standardowe ('0000' lub '1234' w przypadku wielu słuchawek i/lub zestawów głośnomówiących). Efektem ataku jest rejestrowanie rozmów ofiary bądź transmisja audio fake (fałszywe wiadomości o natężeniu ruchu, itp.).

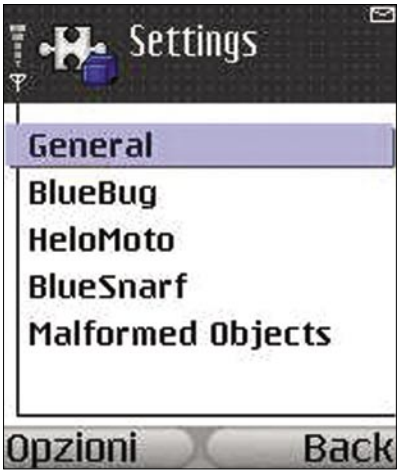
## Worm

Kolejnym problemem Bluetooth jest fakt jego wykorzystywania jako środka rozprzestrzeniania się niektórych robaków. Przykładem jest *Inqtana.A*, robak proof-of-concept wykorzystujący do rozprzestrzeniania się technologię Bluetooth systemów Mac OS X 10.4 (Tiger). Ten robak kopiuje się na wszystkich urządzeniach widocznych poprzez funkcjonalności OBEX i się samowykonuje przy kolejnym uruchomieniu systemu.

Istnieją także inne robaki (z których wielu można uniknąć przy odrobinie ostrożności) jak *Cabir, Mabir* i inne.

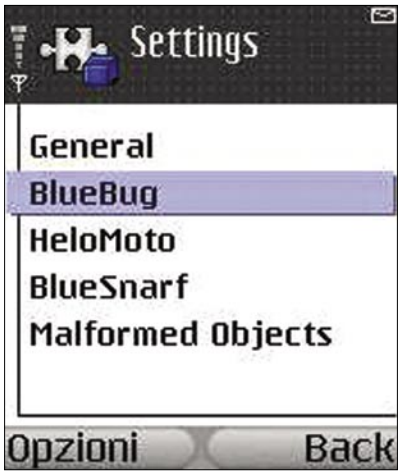
## Od teorii do praktyki

Zobaczymy teraz jak wprowadzić w czyn część tego wszystkiego co widzieliśmy na poprzednich stronach. Spośród rozmaitych egzaminowanych programów użyjemy BloooverII (*http://trifinite.org/trifinite_stuff_blo-*




**Rys. 6b.** *Ustawianie głównych parametrów*


**Rys. 6c.** *Ustawianie głównych parametrów*




**Rys. 7.** *Atak Bluebug*


**Rys. 8.** *Konfiguracje do ataku Bluebug*


**Rys. 9a.** *Przebieg ataku Bluebug*


**Rys 9b.** *Przebieg ataku Bluebug*

*ooverii.html*). Ataki, których możemy spróbować to: bluebug, helomoto, bluesnarf, bluesnarf++ (tylko przy zastosowaniu wersji Breeeder), wysyłanie malformed objects poprzez OBEX.

BloooverII jest programem do urządzeń przenośnych wykorzystujących MIDP 2.0 (Mobile Information Device Profile) i Bluetooth API JSR-82.

Najpierw instalujemy BloooverII (najlepiej wersję Breeeder, która jak widzieliśmy umożliwia także atak bluesnarf++). Przenosimy ją na naszą komórkę za pomocą IRDA, USB lub samego Bluetooth a następnie postępujemy zgodnie z procedurą instalacyjną. Aktywujemy Bluetooth na naszym telefonie komórkowym.

Ważna uwaga: jeśli nie zostało inaczej wskazane próbujmy zawsze jednego tylko typu ataku z jedną tylko funkcją, w przeciwnym razie nasza komórka mogłaby się zawiesić. Inna uwaga: niekiedy program, w czasie ataku, sygnalizuje nazwę komórki atakującej (tak na przykład zdarza się w przypadku Nokii 6310i). Tak więc, jeśli robicie kawał przyjacielowi, zmieńcie nazwę waszego telefonu komórkowego w taki sposób aby nie wskazywała bezpośrednio na was. Wreszcie, jeśli atak nie uda się za pierwszym razem, nie poddawajcie się i spróbujcie jeszcze raz. Niekiedy trzeba powtórzyć atak kilkakrotnie, zanim zadziała poprawnie.

Teraz możemy uruchomić BlooverII i naszym oczom ukaże się taki oto widok (Rys. 4).

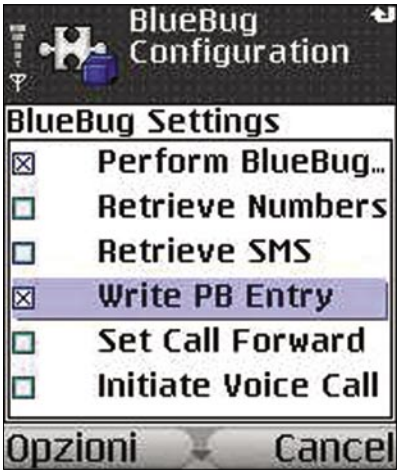
Próbujemy wykryć urządzenia peryferyjne za pomocą *Find devices* (Rys. 5).

Znalazłszy urządzenia zabieramy się za ustawienie głównych parametrów (pamiętajcie, że należy je ponownie ustawić po każdym uruchomieniu Bloooover 2), jak zostało to przedstawione na Rysunku 6.

Ustawiamy je dokładnie w taki sposób jak na Rysunku 6. Teraz spróbujemy kilku ataków. Najpierw Bluebug (zobacz Rysunek 7).

Na zrzutce ekranowej (Rys. 8) znajdują się konfiguracje do ataku Bluebug. Ustawmy je tak jak na rysunku. Teraz zdecydujmy co chcemy osiągnąć poprzez ten rodzaj ataku (Rys. 9).

Po kolei (Rysunki 9a-e), zostały zilustrowane ataki pozwalające na odczytanie książki telefonicznej, SMS-ów, na pisanie w książce telefonicznej, na przekierowywanie połączeń telefonicznych otrzymanych i na inicjalizowanie




**Rys. 9c.** *Przebieg ataku Bluebug*


**Rys. 9d.** *Przebieg ataku Bluebug*


**Rys. 9e.** *Przebieg ataku Bluebug*

nowych. W tym momencie należy coś doprecyzować. BloooverII nie jest programememem dla hackerów, przeciwnie, jest to POC (*Proof of Concept*). Należy poczynić założenie, że nie został pomyślany w celu wyrządzania szkód. Tak więc jedyne połączenie które można zrealizować to połączenie do darmowych numerów. Teraz można zmodyfikować tę funkcję wykorzystując edytor szesnastkowy (typu HHD Free Hex Editor, można go ściągnąć tu: *http://www.hhdsoftware.com/free-hex-editor.html*). W praktyce wystarczy odpowiednio zmodyfikować plik e.class znajdujący się wewnątrz Blooover2b.jar. Po otwarciu archiwum przy pomocy dowolnego programu do obsługi skompresowanych archiwów (WinRAR bardzo dobrze się do tego nadaje), e.class znajduje się w org\trifinite\bloover2b. Oczywiście informacje te nie zostają podane w celu popełniania przestępstw, tak więc zwracam uwagę na użytek jaki z nich zrobicie.

Postępowanie związane z przeprowadzeniem innych ataków jest w zasadzie podobne, tak więc możecie spróbować ich sami. Chciałbym tylko dodać, że niekiedy przy urachamianiu ataku Helomoto BloooverII blokuje się. W takim przypadku powtarzamy operację uruchamiając Helomoto razem z Bluebug.

## Terminologia

- *POF* (*Proof of Concept*): wykorzystywany w środowisku badawczym, jest to praktyczny eksperyment potwierdzający teorię przedstawioną w ramach traktatu lub artykułu,
- *Pentest* (*Penetration Test*): są to testy przeprowadzane w odniesieniu do wszelkiego rodzaju sieci (a więc także i Bluetooth) w celu sprawdzenia ich rzeczywistego bezpieczeństwa,
- *AT*: skrót od Attention, są to polecenia pozwalające na pełne przejęcie kontroli nad telefonem komórkowym,
- *DOS* (*Denial of Service*): to *zmasowany* atak na usługę stawiający sobie za cel wyłącznie jej crash i uczynienie jej nieosiągalną,
- *Ping of Death*: *ping śmierci* bierze swą nazwę od dawnej słabości Windows 95, który zawieszał się otrzymawszy polecenie ICMP (ping właśnie) źle sformatowane,
- *Piconet*: sieć Bluetooth w formie gwiazdy, w której jedno urządzenie zachowuje się jako master inne natomiast jako slave. To samo urządzenie może być masterem jednej sieci i jednocześnie slave innej sieci. Połączenie wielu piconetów nazywa się *scatternet*.

## W Sieci

- *http://www.Bluetooth.com/* - oficjalna strona projektu,
- *http://Bluetooth.interfree.it/* - strona na której wyjaśnia się w prosty sposób Bluetooth,
- *http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/* - atak przeciwko E22,
- *http://trifinite.org/* - strona grupy trifinite,
- *http://www.securiteam.com/tools/5JP0I1FAAE.html* - źródło bluefang,
- *http://www.betaversion.net/btdsd/* - Bluetooth Device Security Database,
- *http://www.niksula.hut.fi/~jiitv/bluesec.html* – pogłębione bezpieczństwo Bluetooth,
- *http://ftp.vub.ac.be/~sijansse/2e%20lic/BT/Tools/Tools.html* – narzędzia bezpieczeństwa Bluetooth,
- *http://it.wikipedia.org/wiki/Bluetooth* – strony wikipedii poświęcone Bluetooth,
- *http://www.remote-exploit.org/index.php/BlueTooth* – strona z bardzo dobrymi narzędziami i informacjami o słabościach Bluetooth.

## O autorze

W Sienie ukończył Inżynierię Informatyczną, od 2001 wykonuje wolny zawód. Zajmuje się kształceniem, systemami i bezpieczeństwem w środowisku korporacyjnym. Jest administratorem systemu, konsultantem w kwestiach prywatności, odpowiedzialnym za bezpieczeństwo i systemy informatyczne wielu włoskich firm; poza tym pracuje jako wykładowca dla kilku spółek zajmujących się kształceniem, m.in Elea-De Agostini i Percorsi, i za ich sprawą zajmuje się kształceniem przedstawicieli najważniejszych podmiotów w środowisku włoskim, jak Ministerstwo Sprawiedliwości, IBM, Alitalia i in. Przeprowadził kilka konferencji na temat e-government dla władz Reggio Calabria oraz posiada najprzeróżniejsze certyfikaty informatyczne, przeważnie na polu systemów i bezpieczeństwa, lecz także na polu Office Automation.

W wolnym czasie jest trenerem i międzynarodowym arbitrem tenisowym.

Kontakt z autorem: *www.ugolopez.it*

## Podsumowanie

Jak widzieliśmy w tym artykule, istnieje wiele różnorodnych technik ataku na protokół, bazują one w znacznej mierze na naiwności użytkowników lub na złych implementacjach stosu Bluetooth ze strony poszczególnych producentów. Oczywiście, wszystko czego nauczyliśmy się w tym artykule powinno służyć pomocą w obronie naszej prywatności, a nie w celu naruszania tejże w odniesieniu do osób trzecich! Pamiętajmy, roztropne zachowanie jest najlepszą obroną przeciwko zagrożeniom płynącym z każdej sieci, także z sieci Bluetooth. Wykaz wszystkich telefonów komórkowych posiadających zainstalowane Java Bluetooth API znajduje się tu:

http://www.j2mepolish.org/devices/devices-btapi.html

Szczególność tego oprogramowania polega na tym, że było pierwszym oprogramowaniem pozwalającym na ataki, wcześniej były one możliwe tylko przy wykorzystaniu laptopa. ●