TDDC03 Projects, Spring 2006

Säkerheten och brister i Bluetooth

Rickard Hyllenstam, Fredrik Johansson

Supervisor: David Bayers

Säkerheten och Brister i Bluetooth

Rickard Hyllenstam Fredrik Johansson

Linköpings universitet, Sweden
Email: {richy342,frejo063}@student.liu.se
Handledare: David Bayers

Abstract

Bluetooth är en teknologi för trådlös kommunikation på korta avstånd. Säkerheten har funnits i åtanke redan från början, men brister har upptäckts i både standarden och implementationen hos produkter. I denna rapport ger vi en bakgrund samt sammanfattar kända säkerhetsbrister och beskriver hur vi utnyttjat några av dessa brister. Det finns idag realistiska hot men vi hoppas att tillverkarna tar sitt ansvar och att användarna får upp ögonen för problematiken. Ett enkelt sätt att undvika hotet är att stänga av Bluetoothfunktionaliteten när den inte används.

1. Inledning

Allt fler konsumentprodukter innehåller idag Bluetooth-funktion. Enligt Bluetooth SIG (Special Interest Group), som äger varumärket Bluetooth, fanns det år 2004 cirka 250 miljoner Bluetooth-enheter. Dessutom räknar de med att detta ökar med 5 miljoner enheter i veckan [1].

Säkerhetsföretaget F-secure sökte efter Bluetooth-enheter på Cebitmässan i Hannover och hittade 12500 ”sökbara” enheter under den veckan som mässan pågick [2]. Detta tillsammans med de säkerhetsbrister som upptäckts, gör att Bluetooth på senare tid väckt stort intresse bland såväl säkerhetsföretag som personer vars intresse är att utnyttja dessa brister.

1.1 Syfte

Vi vill med denna rapport ge en bakgrund samt sammanfatta de kända säkerhetsbristerna som finns idag och beskriva hur vi själva utnyttjat några av dessa bister.

2. Bluetooth specifikation

Detta stycke refererar från Bluetooth Application Developers Guide [4].

Bluetooth är en specifikation för trådlös kommunikation på korta avstånd som är optimerad för data- och ljudöverföring. Den använder sig av ISM-bandet (Industrial, Scientific, Medical) som använder frekvenser mellan 2.402-2.480 Ghz. Detta band är upplåtet för licensfri användning i stora delar av världen. Då många applikationer, som till exempel trådlösa telefoner, trådlös videoöverföring och w-lan, utnyttjar detta band är risken för störningar stor. Detta tillsammans med lagstiftning om maximal uteffekt inom detta band har medfört att Bluetooth utnyttjar sig av FHSS[1]. Detta innebär att enheterna byter frekvens ofta och sänder lite information i taget. Skulle någon signal störas ut, kan den återfås eller återskapas genom en annan sändning eller med hjälp av felrättande koder. Bluetooth använder sig av en teknik som kallas Gaussian Frequency Shift Keying och byter frekvens 1600 gånger per sekund i normalfallet. Antalet frekvenser som används är 79.

Bluetooth är från början ingen standard utan en specifikation från Special Interest Group (SIG) Bluetooth. På senare tid har den blivit en erkänd standard i form av IEEE802.15.1 [3]. En tillverkare är tvingad att följa dessa specifikationer och lämna över sina produkter för utvärdering innan de får tillstånd att använda sig av varumärket Bluetooth tillsammans med sina produkter. I vissa fall är själva Bluetooth-funktionaliteten implementerad på en egen modul och nås via ett av tillverkaren specificerat gränssnitt. Då är det modulen som testas och godkänns. Dessa moduler är ett enkelt sätt att lägga till Bluetooth-funktionalitet hos en produkt utan att behöva genomgå testerna som krävs.

Den version av Bluetoothspecifikationen som kommer att avhandlas här är version 1.2, utom där vi uttryckligen jämför den med den nya versionen 2.0. Bluetooth specificerar inte bara det fysiska lagret och länklagret utan även i vissa fall applikationslagret, genom så kallade profiler. Detta för att säkerställa interoperabiliteten mellan Blutooths-produkter från olika tillverkare.

Version 1.2 stöder en asymmetrisk datahastighet av upp till 732.2 Kbps i ena riktningen och 57.6 Kbps i den andra. Detta går att ändra till ett antal olika lägen. Värt att nämna är att för symetrisk överföring, blir överföringshastigheten 432 Kbps i båda riktningarna. Version 2.0 EDR (Enhanced Data Rate) stöder överföringar strax under 3 Mbps.

Bluetoothenheters räckvidd bestäms av vilken klass de tillhör, det finns tre klasser.

- Klass 1 <100m räckvidd, max 100mW effekt
- Klass 2 <10m räckvidd, max 2.5mW effekt
- Klass 3 <1m räckvidd, max 1mW effekt.

Dessa avstånd skall ses som maximala avstånd för normalanvändning av produkterna. Men för den som har rätt utrustning, är räckvidden betydligt större.

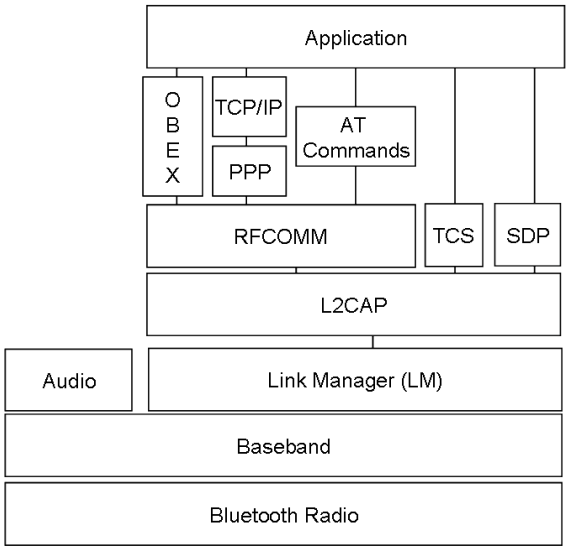
[1] Frequency-hopping spread spectrum.

2.1 Hur Bluetooth fungerar

Innan vi går igenom hur säkerhetsfunktionerna fungerar måste vi förklara hur Bluetooth fungerar ur ett användarperspektiv. Bluetooth-funktionaliteten hos en produkt kan vara i fyra olika lägen: avstängd, sökläge, sökbar och osynlig. När Bluetooth är avslaget går det inte att använda Bluetooths tjänster. När den är sökbar så lyssnar den och svarar på speciella anrop som skickas ut av en enhet som är i sökläge. Den svarar inte på dessa anrop om den är i osynligt läge, men enheten fungerar och går att använda för att initiera trafik och svarar även på direkta anrop till enheten. För att hitta andra enheter används sökläget. Då sänder enheten ut förfrågningar som enheter i sökbart läge svarar på.

2.1.1 Parning av enheter

Här följer en närmare beskrivning av vad som händer när två enheter skall koppla ihop sig. Anledningen till att en parning behövs är att enheterna skall synkronisera sig och utbyta nycklar. När den ena enheten är i sökbart läge innebär detta att den byter frekvens ungefär en gång i sekunden. Detta för att enheten skall kunna hittas även om en viss frekvens råkar vara utstörd. Den andra enheten som är i sökläge byter frekvens 3200 gånger i sekunden och skickar ut förfrågningar på dessa frekvenser. På detta sätt kommer de förr eller senare att mötas på en gemensam frekvens. Den sökbara enheten väntar en slumpmässig tid för att undvika att krocka med andra sökbara enheter och skickar sedan ett Frekvency Hop Synchronisation (FHS) meddelande. Detta innehåller tillräckligt med information för att enheterna skall kunna komma överens om ett hoppmönster och synkronisera detta. För att garantera att en enhet skall hittas kräver Bluetooth-specifikationen att en sökning skall pågå i minst 10.24 sekunder [5].



Figur 1, Bluetoothstacken

2.1.2 Skapandet av en förbindelse

Nu har enheterna kontakt med varandra på radionivån i protokollstacken. Men för att enheterna skall kunna skapa en förbindelse, Asyncronous Connection-oriented Logical Transport (ACL) måste de genomföra något som kallas paging. Detta sätter upp en kanal mellan enheterna. Detta gör att enheterna har kontakt med varandra på Link manager nivån, se figur 1.

Den enhet som initierar en uppkoppling blir alltid master och det är masterns BD_ADDR[2] som bestämmer hoppordningen och synkroniseringen. Efter att paging är klart, finns det i vissa enheter stöd att byta roller. Detta är viktigt för enheter som till exempel accesspunkter för ett nätverk. Där är det klienterna som initierar anslutningen, men det är givetvis bäst om accesspunkten är master och styr alla tillkopplade klienter.

2.1.3 Muliplexerande lagret

Logical Link Control and Adaptation Protcol, L2CAP är nästa lager i protokollstacken. Det ansvarar för multiplexering av olika överliggande protokoll. Vilket protokoll som väljs bestäms av värdet på protocol service multiplexor (PSM).

2.1.4 Tillgängliga tjänster

För att se vilka tjänster som finns tillgängliga. används service discovery protocol (SPD), se figur 1. Det har PSM 0x0001. I den tillfrågade enheten finns en databas som beskriver de tjänster enheten erbjuder. Denna databas går antingen att leta igenom manuellt eller söka efter tjänster med hjälp av nyckelord. När en önskad tjänst hittats så kan den frågande enheten få information om hur den skall ansluta sig. Informationen innehåller PSM och data till högre lager, som behövs för anslutning till tjänsten. Beroende på vilken säkerhetsnivå som är aktiv och vilken PSM som används så kommer säkerhetsfunktioner att aktiveras.

2.2 Bluetooths säkerhetsfunktioner

Om inget annat angers refereras detta stycke från Bluetooth Specification [21].

Det finns tre olika säkerhetslägen som Bluetooth-utrustning kan arbeta i.

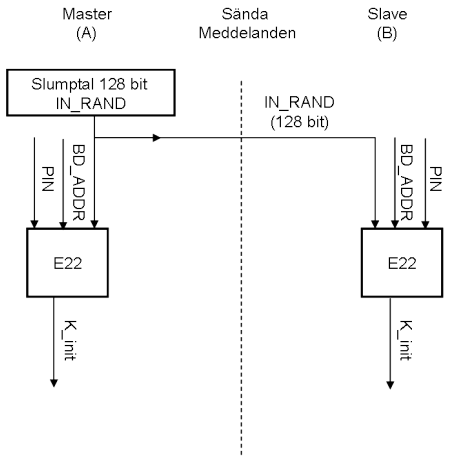
- Läge 1 - Inga säkerhetsfunktioner är aktiva och behörigheten är helt upp till applikationslagret.
- Läge 2 - Säkerheten är baserad på vilken tjänst som används. I praktiken vilket PSM som används.
- Läge 3 - Godkännande krävs redan på länklagret.

[2] Blutooth device adress, unik enhetsadress på varje enhet.

Vilket läge som används är upp till tillverkaren av produkten. Om en Bluetooth-modul används bestäms säkerhetsnivån av modulens tillverkare, inte av produktens tillverkare.

De tre olika lägena är inte säkerhetsåtgärder i sig, utan en nivå när säkerhetsfunktionerna skall träda i kraft. För säkerhetsfunktionen finns det något som kallas security-manager. Den har till uppgift att konfigurera säkerheten, anropa rutiner för att interagera med användaren, samt förse lägre lager med PIN[3]-koder eller nycklar. Security-managern innehåller en databas som beskriver hur den skall reagera när olika tjänster anropas, det vill säga vilket PSM som används. Dessutom finns en databas som håller reda på enheter som är auktoriserade eller som tidigare varit ihopkopplade och därför har en gemensam nyckel sparad.

En enhet kan ha olika förhållande till andra enheter, enheten kan lita på en annan enhet. Detta kallas auktorisation och därmed tillåts enheten att ansluta utan några senare förfrågningar. En enhet som inte är auktoriserad kan få åtkomst till tjänsterna som erbjuds genom att verifiera sig vid varje tillfälle. Detta sker med hjälp av en gemensam PIN-kod, vilket kallas autentisering.



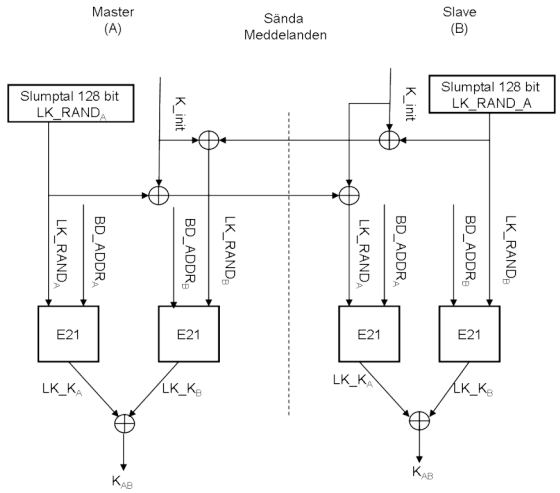
Figur 2, skapandet av K_{init}.

2.2.1 Nyckelhantering

När en enhet för första gången försöker ansluta till en tjänst som är skyddad med läge 2, sker ett antal olika steg. Först skapas en initialiseringsnyckel K_{init}. Sedan skapas en länknyckel K_{ab}. Därefter kommer enheterna att verifieras. Skulle kryptering krävas så kommer detta att ske först när verifieringen är avslutad. Dessa steg beskrivs närmare nedan.

För skapandet av K_{init} används PIN-koden, den ena enhetens BD_ADDR och ett slumpmässigt tal. Dessa inparametrar körs hos båda enheterna genom E22- algoritmen[4] och K_{init} skapas. Se figur 2.

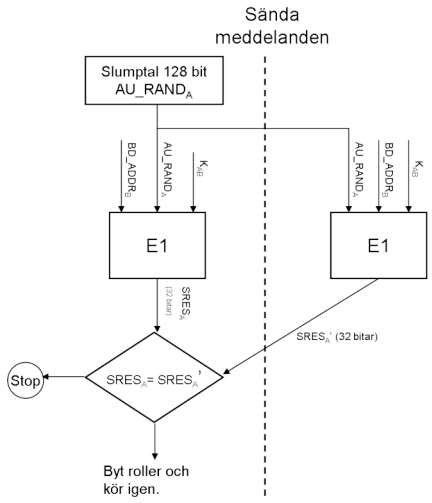
Sedan skall länknyckeln K_{ab} skapas, den är en unik nyckel som endast är tänkt att de två hopkopplade enheterna skall dela. K_{init} används för att kryptera överföringen av två nya slumptal. På båda sidorna kommer sedan respektive slumptal och BD_ADDR köras genom algoritmen E21 och sedan kombineras samman till länknyckeln K_{ab}. Se figur 3.



Figur 3, Tillverkning av länknyckel K_{ab}.

I enheter som har begränsad beräknings- eller minneskapacitet så används ibland antingen K_a eller K_b som länknyckel istället för K_{ab}.

Till sist används länknyckeln för att utföra verifiering av enheterna med hjälp av E1-algoritmen. Denna del körs två gånger där enheterna byter roller. Se figur 4.



Figur 4, Del ett av verifieringsfasen.

[3] Personal Identification Number

[4] Alla tre E-algoritmer har sitt ursprung i blockchiffret SAFER+ (Secure And Fast Encryption Routine) som var en av kandidaterna till AES-standarden. [22]

Om denna kontroll är godkänd så sparas K_{ab} i minnet och kommer att användas vid alla framtida kopplingar mellan de båda enheterna. Då hoppas de första stegen över och enheterna går direkt till verifiering. Det är tillåtet enligt specifikationen att glömma bort gemensamma nycklar. Då måste processen upprepas vid nästa tillfälle. För djupare genomgång av dessa moment se Yaniv Shaked och Avishai Wools artikel *Cracking the Bluetooth PIN* [8].

2.2.2 Kryptering

Om kryptering används så skapas krypteringsnyckeln K_C med hjälp av länknyckeln. Nyckeln är mellan 8 till 128 bitar lång. Denna används sedan tillsammans med BD_ADDR_{Master} och $Clock_{Master}$ för att skapa en bitström som adderas modulo 2 med informationen som skall krypteras. Värt att nämna är att endast nyttotrafiken krypteras.

2.3 Brister i Bluetoothstandarden

När standarden togs fram fanns ett säkerhetstänkande med från början. Dock har en del brister i standarden upptäckts i efterhand. Några av dessa tas upp i detta stycke.

2.3.1 Avlyssning av parningsförfarandet

När två enheter skall para sig skapas ett slumptal av den ena enheten, detta slumptal skickas sedan över i klartext till den andra enheten. En gemensam initieringsnyckel K_{init} skapas sedan enligt förfarandet i kapitel 2.2.1. Om slumptalet mellan dessa enheter kan avlyssnas och enheternas BD_ADDR är kända, behöver endast PIN-koden knäckas för att kunna räkna fram K_{init}. En PIN-kod är vanligen fyra siffror lång och enligt Yaniv Shaked och Avishai Wool i [8] tar det mellan 0.06 och 0.3 sekunder att knäcka en sådan. För att lagra trafiken mellan enheterna medan nycklarna knäcks krävs endast en buffer på 40kB då PIN-koden är fyra siffror långt. Det är sedan trivialt att räkna fram K_{ab} från den avlyssnade trafiken mellan enheterna.

2.3.2 Svagheter i krypteringen

Miia Hermelin och Kaisa Nyberg i [19] visar att strömkryptot som används av Bluetooth har en svaghet som kan halvera tidsåtgången för att knäcka kryptot med uttömmande sökning.

2.3.3 Svagheter i länknyckeln

Vid skapandet av en länknyckel finns möjligheten att använda den ena enhetens nyckel istället för att skapa en kombinerad, se kapitel 2.2.1.

Låt säga att två enheter A och B kommunicerar med varandra och av någon anledning väljer att använda K_a som länknyckel. Låt sedan enhet C kommunicera med A och också använda sig av K_a. Nu kan enhet B lyssna på trafiken mellan A och C, ty de har samma länknyckel. Enhet B har nu möjligheten att utge sig för att vara C i kommunikation med A. För att lyckas med detta måste B ändra sin BD_ADDR till C´s adress. Det krävs en speciell hårdvara för att lyckas byta BD_ADDR [20].

3. Kända Bluetooth attacker

Det finns ett flertal olika verktyg publicerade på Internet för att kunna utnyttja diverse säkerhetsbrister hos Bluetooth-enheter. Enligt säkerhetsföretaget F-secure låter många personer som äger en Bluetooth-enhet, Bluetooth vara inställt i sökbart läge [2]. Denna inställning öppnar en möjlighet för en hacker att utnyttja säkerhetsbrister hos enheten.

Nedan följer en genomgång av några kända attacker som använder sig av Bluetooth för att bland annat komma åt privat information från en enhet eller utnyttja enhetens tjänster.

3.1 Bluesnarfing

Hos en del Bluetoothenheter är det möjligt att ansluta till enheten utan att ägaren märker det. Attacken använder sig av OPP[5] som är ett protokoll för att utbyta visitkort och liknande objekt. Vanligtvis krävs ingen verifiering av användarna vid användning av denna tjänst. I vissa fall finns det brister i implementationen av dessa tjänster hos enheten, vilket möjliggör en attack. Genom att gissa, eller ha vetskap om, till exempel telefonbokens filnamn kan objektet hämtas utan ägarens vetskap.

Bluesnarfing är enklast att utnyttja på enheter som är i sökbart läge. Men genom att känna till enhetens BD_ADDR är det möjligt att ansluta direkt. Om andressen inte är känd kan till exempel verktyget redgfag användas för att söka igenom ett stort antal adresser. Detta för att hitta enheter som är påslagna men ej i sökbart läge [6,8].

Bluebugging

Den här attacken utnyttjar modemfunktionen i mobiltelefoner. Attacken innebär att någon skapar en seriell anslutning till en enhet och kan sedan använda sig av AT-kommandon[6] för att få tillgång till viss information och vissa tjänster. Med AT-kommandon är det möjligt att skicka och ta emot sms, ringa upp ett valfritt telefonnummer på ägarens bekostnad eller ansluta till Internet. Anslutningen behöver bara vara aktiv i några sekunder för att kunna skicka över AT-kommandon. Under denna tid kan exempelvis en vidarekoppling av telefonsamtal skapas. Attacken har fått sitt namn från att en telefons mikrofon kan användas för avlyssning, ”buggning”. Genom att ringa upp ett nummer från offrets telefon med hjälp av AT-kommandon, kan telefonen och dess omgivning avlyssnas [7].

[5] OBEX Push Profile

[6] AT står för attention och är Hayes-kommandon för att styra ett modem.

3.2 Bluejacking

Denna typ av attack kan användas för att skicka anonyma meddelanden till Bluetooth-enheter. Attacken utnyttjar Bluetooth-protokollets parningsfas. Genom att utnyttja att protokollet tillåter enhetens namn att vara 248 tecken långt, kan enhetsnamnet ersättas av ett meddelande. När hackern sedan ansluter till en annan enhet syns meddelandet som enhetsnamn hos mottagaren [11].

3.3 Bluebumping

Namnet på denna attack kommer från den fysiska världen där en relativt novis person kan knäcka ett vanligt lås med en modifierad nyckel. Bluebumping handlar även det om nycklar, därav namnet. Attacken går ut på att försöka tvinga två parade enheter att para sig en gång till. En hacker försöker först parar sig helt legitimt med ett offer och skapa en länknyckel. Hackern ber sedan offret att ta bort länknyckeln för hackerns enhet, men hackern talar aldrig om att han fortfarande håller uppkopplingen öppen mot offret. Nu begär hackern en ny generering av länknyckel, genom detta får hackern en ny länknyckel utan att behöva verifiera sig igen. Hackern har nu tillgång till offrets enhet så länge som länknyckel inte raderas av offret, som är omedveten om att den finns [9].

3.4 Carwhispering

Attacken utnyttjar bilar som är utrustade med Blutoothhandsfree. Attacken kan utföras både mot bilar som är stillastående eller i rörelse. För bilar i rörelse används en riktantenn och en dator för att skicka information till och från bilens handsfree. Vanligen använder tillverkare av dessa enheter en standardnyckel (exempelvis 1234) för att ansluta sig till den. Detta möjliggör avlyssning av samtal i bilen eller uppspelning av ett meddelande. När en anslutning har upprättats med standardnyckeln kan enhetens alla tjänster utnyttjas. I vissa fall ger detta en möjlighet att få tillgång till en användares telefonbok [10].

3.5 Skadlig kod

Bluetooth kan även användas för att sprida skadlig kod till andra enheter. Senast i år rapporterades det om nya trojaner för Symbian OS som drabbat Nokias 60-serie av mobiltelefoner [11].

Det finns även exempel på denna attack som sprider liknande kod mellan datorer och mobiltelefoner. Ett exempel är Inqtana, vilket är ett javabaserat virus för Mac OS X [14].

3.6 Denial of service attack

Med en Bluetooth-enhet går det att störa ut en annan Bluetooth-utrustad enhet med hjälp av en så kallad denial of service-attack (DoS). Genom att konstant sända parningsförfrågningar till den attackerade enheten går det att till exempel uttömma en enhets batterikapacitet och temporärt störa ut enhetens funktioner. Det är idag inte möjligt att komma över någon information med hjälp av denna attack, utan endast möjligt att störa ut en enhet. Tyvärr finns det än så länge få dokumenterade mjukvaruskydd mot en DoS-attack. Ett rekommenderad skydd är att avlägsna sig från det område där hackern befinner sig i [12,13].

4. Resultat

En del i detta projekt var att undersöka kända bluetoothbrister och sedan försöka använda någon av dessa i ett eget experiment. Nedan följer en genomgång av vårt program.

4.1 Bluebear

För detta experiment valde vi att använda oss av en dator utrustad med linux. Vi valde distributionen backtrack [15]. I efterhand kan vi konstatera att detta inte var en nödvändighet då vi inte använder oss av någon specifik funktion i den distributionen. Det går bra att använda vårt program, som har fått namnet Bluebear[18], på vilken Linuxdistribution som helst, förutsatt att den officiella Bluetooth-stacken BlueZ[7] finns installerad. Vi har valt att göra ett program som kan utnyttja säkerhetsbrister i vissa mobiltelefonmodeller.

För att skapa ett program som hämtar eller lämnar information på en Bluetooth-enhet använder vi oss av befintliga kommandon för Bluetooht-stacken Bluez. För att skapa själva programmet har vi valt att använda oss av skriptspråket Perl. Till Perl finns en del färdiga moduler för kommunikation med mobiltelefoner. I modulen Device::Modem hittar vi bland annat metoder för att ansluta oss till telefonens modem. I modulen Device::Gsm hittar vi sedan metoder för att kunna skicka AT-kommandon.

Bluebears huvudfunktion är uppbyggt efter ett känt säkerhetshål i Nokias mobiltelefon 6310i. Nokias telefon erbjuder OBEX-tjänster via kanal 17 utan några krav på verifiering hos användaren. Nokias implementation på den här kanalen möjliggör att det även går att komma åt andra tjänster, vilket är ett misstag. När anslutningen har upprättats till RFCOMM-lagret, går det att ansluta sig till telefonens modem och använda sig av AT-kommandon utan någon vidare verifiering. Om denna anslutning lyckas går det sedan, via vår programmeny, att hämta adressboken, läsa SMS, ringa samtal och hämta enhetsinformation som till exempel batteristatus. För att välja vilken enhet attacken skall utföras på kan användaren mata in en känd BT_ADDR eller använda sökfunktionen och hitta sökbara enheter.

Del två av Bluebear utnyttjar brister i implementationen hos vissa Sony-Ericssontelefoner. Men hjälp av Bluez-kommandot obexftp, som använder OBEX-protokollet för filöverföring, kan vi hämta kända filnamn hos några Ericssonmodeller. Vi kan med hjälp av kända filnamn hämta adressböcker, kalendrar, enhetsinformation och visitkort från en enhet. Allt sker utan att

[7] BlueZ – Officiell Linux Bluetooth protokollstack [16]

någon verifiering eller anslutning behöver skapas hos enheten.

Våra tester visar att AT-kommandon fungerar bra på en Nokia 6310i som inte uppgraderat telefonens mjukvara. Att hämta filer via obexftp fungerar bra på Ericsson T610 och T68 av de modeller vi har testat, så länge som de inte har uppgraderat sin mjukvara.

Vårt experiment visar att det relativt enkelt går att skriva sina egna program genom att endast känna till hur linux tillsammans med BlueZ fungerar. Vill man inte skriva egna program finns det otalet verktyg tillgängliga på Internet för nedladdning till olika operativsystem. Det finns till och med en hel Linuxdistribution på en CD-skiva, som går att starta från utan installation, vid namn Auditor tools. Skivan är full av olika säkerhetsverktyg, bland annat Bluetoothverktygen Bluesnarfer, BlueBug och Redgfag [17].

5. Diskussion

Bluetoothspecifikationen har en väl genomtänkt säkerhetsstruktur, men som vi beskriver i kapitel 2.3 finns det några svagheter som har upptäckts i efterhand. Dessa svagheter är dock relativt svåra att utnyttja och en ändring av standarden skulle innebära stora problem med kompabiliteten.

Däremot finns det ett antal exempel på säkerhetsbrister i implementationen av Bluetooth. De flesta brister har vi funnit hos mobiltelefoner, vilket också var en tidig produkt som använde sig av Bluetooth. Tillverkarna har tagit till sig kritiken och uppdaterat sina mjukvaror till nyare modeller. Det finns även möjligheter att uppgradera gamla telefoner med ny mjukvara. Problemet är att det är upp till användaren att se till så att produkten blir uppdaterad. En uppgradering av mjukvaran är oftast ingen enkel procedur, vilket vi tror medför att det inte utförs så ofta. En annat stort problem är att det ofta går att lura användaren trots att tekniken är väl skyddad. Ett exempel på detta är bluejacking där ett smart utformat meddelande kan få en användare att godkänna en uppkoppling. Då är telefonen sårbar trots att säkerhetsfunktionerna har används korrekt.

Hotet mot en Bluetoothutrustad telefon kan tyckas vara litet, men det finns motiv som gör hotet större. Det är till exempel möjligt att via attacker som bluebugging, ringa ett betalsamtal eller skicka meddelanden från någons telefon utan ägarens vetskap. Till exempel kan någon skicka ett kompromettерande meddelande som senare spåras till ägaren av telefonen och det finns ingen möjlighet att se hur det gick till när det sändes. Detta är ett problem då liknande meddelanden använts som bevis vid rättegångar, se till exempel Knutby-målet.

Dessutom har vi konstaterat att det är relativt enkelt att utnyttja dessa brister. Dels finns det en hel del färdiga verktyg att använda, som inte kräver någon specifik kunskap om Bluetooth. Vårat egna program visar också att det även är enkelt att själv skapa de verktyg som behövs för att utnyttja dessa brister. I och med detta skulle det vara fullt möjligt att automatisera en attack för att uppträcka sårbara enheter. Vi ser en risk i att detta kan utnyttjas i ekonomiskt syfte.

6. Slutsats

Standarden är ur säkerhetssynpunkt väl utformad och de få brister som finns är svåra att utnyttja. Ett större problem är produkter med säkerhetsbrister i implementationen av Bluetooth. Slutligen finns också problematiken med godtrogna användare. Hotet finns och är verkligt idag men vi hoppas att tillverkarna tar sitt ansvar och att användarna får upp ögonen för problematiken. Ett enkelt sätt att undvika hotet är att stänga av sin Bluetoothfunktionalitet när den inte behövs.

Referenser

[1] Bluetooth SIG, ”Bluetooth History”, 22:a mars 2006, http://www.Bluetooth.com/Bluetooth/SIG/Who/History/

[2] Mikael Ricknäs, ”F-Secure letar sårbara mobiler”, *ComputerSweden,* Nummer 34, Sidan 6, 24:e mars 2006.

[3] Ian Gifford, Dr. Chatschik Bisdikian, Bob Heile, Karen McCabe, ”IEEE Approves IEEE 802.15.1 standard for wirelwss perosnal area networks adapted frm the bluetooth specification”, *IEEE,* 2:a januari 2004, http://standards.ieee.org/announcements/802151app.html

[4] David Kammer, Gordon McNutt, Brian Senese, Jennifer Bray, ”Bluetooth Application Developers Guide”, *syngress,* kapitel 1, 2002

[5] David Kammer, Gordon McNutt, Brian Senese, Jennifer Bray, ”Bluetooth Application Developers Guide”, *syngress,* sida 21, 2002

[6] Adam Laurie, Ben Laurie, ”Serious flaws in bluetooth security lead to disclosure of personal data”, *A.L. Digital Ltd,* 14:onde oktober 2004, http://www.thebunker.net/security/Bluetooth.htm

[7] Trifinite , ”BlueBug”, *trifinite.group,* 23:e mars 2006, http://trifinite.org/trifinite_stuff_bluebug.html

[8] Yaniv Shaked, Avishai Wool, ”Cracking the Bluetooth PIN”, *School of Electrical Engineering Systems*, 2:a maj 2005, http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/index.html

[9] Adam Laurie, Marcel Holtmann, Martin Herfurt., ”BlueBumping”, *trifinite.group,* 23:e mars 2006, http://trifinite.org/trifinite_stuff_bluebump.html

[10] Martin Herfurt, ”Introducing the car whisperer at What the Hack”, *trifinite.group*, 31:a juli 2005, http://trifinite.org/blog/archives/2005/07/introducing_the.html

[11] Deborah Hale, ”Symbian operation system – Nokia series 60 mobile phones – 3 new Trojans”, *SANS – Internet Storm Center,* 20:onde januari 2006, http://isc.sans.org/diary.php?storyid=1056

[12] Carlos A. Soto, ”A menu of Bluetooth attacks”, *GCN,* 25:e oktober 2005, http://www.gcn.com/print/24_20/36437-1.html

[13] Bluetooth SIG, ”bluetooth.com | Security”, Bluetooth.com, 24:e mars 2006, http://www.Bluetooth.com/Bluetooth/Learn/Security/

[14] Jarno Niemelä, ”F-Secure Worm Information Pages: Inqtana.A”, 22:a februari 2006, http://www.f-secure.com/v-descs/inqtana_a.shtml

[15] Remote-exploit, ”BackTrack Downloads”, *Downloadpage for BackTrack,* 18:onde April 2006, http://www.remote-exploit.org/index.php/BackTrack_Downloads

[16] BlueZ, ”Official Linux Bluetooth protocol Stack”, *BlueZ Poject, 18:onde April 2006,* http://www.bluez.org/

[17] Remote-exploit, ”Auditor main”, *Auditor tools*, 18:onde April 2006, http://www.remote-exploit.org/index.php/Auditor_main

[18] Rickard Hyllenstam, Fredrik Johansson, ”Bluebear”, *Bluetooth-hacking program written in Perl,* 2006, http://www-und.ida.liu.se/~richy342/bluebear/

[19] Miia Hermelin, Kaisa Nyberg, ”Correlation Properties of the Bluetooth Combiner”, *Proceeding of ICISC ’99,* 1999, http://citeseer.ist.psu.edu/hermelin99correlation.html

[20] Creighton T. Hager and Scott F. Midkiff, ”Demonstrating Vulnerabilities in Bluetooth Security”, *GLOBECOM ’03*, 2003, http://ieeexplore.ieee.org/

[21] Bluetooth SIG, ”Bluetooth Specification Version 2.0 + EDR vol 3”, *Bluetooth SIG,* sid 770-811, 2004, http://www.bluetooth.com/NR/rdonlyres/1F6469BA-6AE7-42B6-B5A1-65148B9DB238/840/Core_v210_EDR.zip

[22] Cylink Corporation, ”Nomination of SAFER+ as Candidate Algortithm for the Advanced Encryption Standard (AES)”, *Cylink Corporation*, 1998, http://mcrypt.hellug.gr/docs/index.html