Blueprinting™

Remote Device Identification based on Bluetooth Fingerprinting Techniques

White Paper (Version 0.3)

Martin Herfurt and Collin Mulliner

{martin,collin}@trifinite.org

December 20, 2004

Abstract

We introduce a method to efficiently determine a Bluetooth device's properties as needed for a variety of purposes. Blueprinting aims to set a standard for Bluetooth fingerprinting devices. The idea is similar to IP fingerprinting techniques as used in tools like `nmap` where it is possible to determine a hosts operating system by specific behavior of the IP stack. With Blueprinting it is possible to determine the manufacturer, the device model and the firmware version of the respective device. The complexity of the introduced method is intentionally simple so that this procedure can be executed on constrained devices that are not capable of calculating common hashes such as MD5: the J2ME Connected Limited Device Configuration (CLDC) Version 1.0 (as used in many mobile handsets) can perform it.

trifinite.group

1 Introduction

During the last years, Bluetooth has become a well-recognized wire replacement standard for all different kinds of devices. Mainly in the consumer electronics and the automotive industry, the Bluetooth standard has gained acceptance and is deployed in a growing number of products. Currently, the actual number of Bluetooth radios in use is four times higher than the number of Wi-Fi radios deployed.

Herein, we introduce a method to efficiently determine a Bluetooth device's properties as needed for a variety of purposes. Blueprinting aims to set a standard for Bluetooth fingerprinting devices. The idea is similar to IP fingerprinting techniques as used in tools like `nmap` where it is possible to determine a hosts operating system by specific behavior of the IP stack. With Blueprinting it is possible to determine the manufacturer, the device model and the firmware version of the respective device. The complexity of the introduced method is intentionally simple so that this procedure can be executed on constrained devices that are not capable of calculating common hashes such as MD5: the J2ME Connected Limited Device Configuration (CLDC) Version 1.0 (as used in many mobile handsets) can perform it.

There are many different reasons that justify a method that allows the identification of Bluetooth-enabled devices by the characteristics of their radio interface.

1.1 Device Statistics

One of the purposes that Blueprinting could be used for is statistical examination of different environments. This way, it is possible to create statistics over manufacturer and device models in special places as it was done in the CeBIT field trial report [1]. There are more scenarios where the determination of Bluetooth device properties is making sense.

1.2 Automated Application Distribution

There are many different mobile handsets that all have different operating system platforms running. One of the most popular platforms is Symbian [2] but there is a number of other platforms. Mobile device manufacturers are developing applications for many different purposes. In order to deliver the application for the right platform, the application distributor needs to know about the requesting device model, so that the application that is pushed to the device might be a version that supports e.g. the bigger display of a certain device. Unfortunately, there are also malicious applications like the proof-of-concept virus Cabir [3] that could profit from an identification method like Blueprinting.

1.3 Security Audits

Early implementations of the Bluetooth standard in devices of various device manufacturers are subject to more or less severe security issues. Attacks like the BlueSnarf attack [4], the BlueBug attack [5] or the BlueSmack attack [6], which enable the extraction of sensitive information, the abuse of telecommunications services or the denial of service are subject to the firmware and the model of some phones. In order to communicate eventual security issues to the respective manufacturers it is important to know about the properties of the concerned device. Blueprinting contributes to the efforts done in order to make Bluetooth devices more secure.

2 Device Information

Blueprinting encapsulates the necessary information in order to determine device specific properties such as the manufacturer, the model information and the firmware version. Since mobile phones and PDAs make up the biggest group of Bluetooth enabled devices, Blueprinting mainly focuses on these devices. The method relies on device specific information that has been collected in experiments such as the CeBIT experiment [1], and, therefore, is not as detailed as it could be.

Every Bluetooth enabled device has some characteristics that are either unique (Bluetooth device address), manufacturer specific (the first part of the Bluetooth device address) or model-specific (service description records). Blueprinting is combining the different information that Bluetooth-enabled devices reveal in order to identify the manufacturer as well as the model of the device. The firmware version that runs on certain devices can be derived based upon a devices different characteristics.

2.1 Bluetooth Device Address

As mentioned above the Bluetooth device address (BD_ADDR) is unique and globally refers to one single device. This BD_ADDR address consists out of 48 bits (6 bytes) that are usually notated like MAC addresses (e.g. MM:MM:MM:XX:XX:XX). The address is programmed into the Bluetooth radio. The first three bytes of this address (the bytes that are denoted by M's above) refer to the manufacturer of the chipset. An actual list of all these codes that refer to different manufacturers can be found in the OUI database hosted by IEEE [7]. Unfortunately, it is not possible to tell anything about the device model by interpretation of the remaining three bytes. These bytes (denoted by X's above) are used randomly in different models. Therefore, for identifying a manufacturer's model, Blueprinting takes the SDP [8] profiles, which can be queried from devices that offer services, into account.

2.2 SDP Profiles

Service Description Protocol (SDP) [8] profiles are a concept that is used by Bluetooth in order to identify a certain service to other devices. This is done for autoconfiguration purposes and to help a user setup a connection to the specific device. SDP Profiles are served by the device's sdp server and provide information on how to access the offered profiles. Every SDP profile entry has some properties that can be used to identify the device.
Figure 1 shows a Obex Push profile as it is retrieved by the BlueZ [9] utility called `sdptool`.

3 Blueprinting

Blueprinting uses specific information from SDP profiles of a device to create a hash for the respective device. According to the standard[8], there is always a field that holds the Service

```
Service Name:  OBEX Object Push
Service RecHandle:  0x1000c
Service Class ID List:
"OBEX Object Push" (0x1105)
Protocol Descriptor List:
"L2CAP" (0x0100)
"RFCOMM" (0x0003)
Channel:  9
"OBEX" (0x0008)
Language Base Attr List:
code_ISO639:  0x656e
encoding:  0x6a
base_offset:  0x100
Profile Descriptor List:
"OBEX Object Push" (0x1105)
Version:  0x0100
```

Figure 1: OPUSH Profile from a Nokia 6310i

Record Handle, which is a 32 bit number that is assigned by the SDP server when a service is registered during startup of the device (e.g. 0x1000c in figure 1). In the case of mobile phones, the RecordHandles for the profile entries at the SDP server are not dynamically assigned but statically coded in the phone's firmware. The other value that is taken into the hash is the RFCOMM channel or the L2CAP psm number that the service can be accessed under. In the above profile, this would be RFCOMM channel 9.

One part of a device's Blueprinting hash is the sum of the RecHandle times the Channel for all running services. The following example shows this by the example of a Nokia 6310i SDP profile export:

RecHandle	Channel	Product
0x1000b	2	131094
0x1000c	9	589932
0x1000d	1	65549
0x1000e	15	983250
0x1000f	3	196653
0x10010	13	852176
0x10011	12	786636
		3605290

3.1 Blueprinting Software

The Blueprint [10] software is a proof-of-concept implementation of the herein described Bluetooth fingerprinting technique. For simplicity, it was implemented in Perl and reads the output of `sdptool` [9]. Blueprint uses a simple text based database which contains fingerprints and information about the associated device. The implementation also combines the actual fingerprint with the manufacturer part of the BD_ADDR to achieve a higher matching rate.

```
00:60:57@2621543
```

Figure 2: Example of a Blueprint fingerprint

```
00:60:57@2621543
device:  Nokia 6310i
version:  V 5.22 15-11-02 NPL-1
date:  n/a
type:  mobile phone
note:  vulnerable to BlueBug attack
```

Figure 3: Example of a Blueprint database entry

4 Related Work

4.1 Bluetooth Security Device Database

The Bluetooth Device Security Database [11] was created after various security related bugs where found on embedded Bluetooth devices. The *btdsd* projects goal is to collect information on (default) security settings of Bluetooth enabled devices. The collection shows that nearly all manufacturers have different default security settings and security features implemented. The database was used in the evaluation of the Blueprinting technique.

5 Future Work

The work described here is the basis for ongoing work in this area. The trifinite.group is inviting everyone to contribute in all future efforts. Continued progress relies on developing a more comprehensive set of SDP profiles, which can be sent via email. For information on how to contribute, check the Bluetooth Device Security Database page [11].

5.1 Non-SDP Fingerprinting

Blueprinting, so far, only uses the Service Discovery Protocol (SDP) [8] information for identifying devices. In the future, data from higher and lower level protocols should be used for identification as well. Examples could be: LinkManager (LM) commands (when connecting to a specific service) or Obex [12] behavior.

6 Conclusions

Blueprinting is a novel method for the identification of Bluetooth-enabled devices by means of their radio interface and the Bluetooth stack of the operating system. The information gathered so far about the SDP profiles demonstrates a decreasing diversity in mobile phone operating systems; the prevalent usage of e.g. Symbian [2]. The increasing uniformity is evident from similar Blueprinting hashes even when the hardware and the manufacturer of the products differ. In the future, current trends dictate the variety of Blueprinting hashes will most likely decrease. The fact that many phones have the same operating system could result in serious trouble once a security flaw is discovered for a common operating system.

Blueprint Device Hashes

This section lists the hashes that have been collected so far. Some of the devices have multiple entries. The explanation for this is that these devices have different firmware versions that result in a different Blueprinting hash.

Blueprinting Hash	**Manufacturer**	**Model**	**Firmware**
00:0A:95@1114112	Apple	Wireless Keyboard	unknown
00:01:EC@2359452	Ericsson	T39m	unknown
00:30:6E@269099048	HP	bt130	unknown
08:00:28@3342638	HP	iPAQ h6315	initial firmware
08:00:17@2949325	HP	iPAQ 5500	PocketPC (4.20.1081)
00:0C:55@983040	Microsoft	Windows XP	SP2
C6:F7:4A@655407	Motorola	A1000	unknown
00:0A:28@1769675	Motorola	V600	unknown
00:60:57@1704044	Nokia	3650	unknown
00:60:57@1704020	Nokia	3650	unknown
00:60:57@1704022	Nokia	3650	unknown
00:60:57@1704023	Nokia	3650	unknown
00:60:57@3605290	Nokia	6310i	unknown
00:60:57@3607710	Nokia	6310i	unknown
00:60:57@3604685	Nokia	6310/6310i	unknown
00:60:57@2621543	Nokia	6310/6310i	unknown
00:0E:ED@4391166	Nokia	6320	unknown
00:60:57@1704035	Nokia	6600	unknown
00:60:57@1704034	Nokia	6600	unknown
00:02:EE@4391166	Nokia	6820	unknown
00:60:57@4128974	Nokia	7600	unknown
00:60:57@1507391	Nokia	7650	unknown
00:02:EE@1507908	Nokia	7650	V 3.16 / 15-08-02 / NHL-2NA
00:02:EE@5112150	Nokia	7820	unknown
00:60:57@1704022	Nokia	N-Gage	unknown
00:60:57@1704023	Nokia	N-Gage	unknown
00:60:57@1507402	Nokia	N-Gage	V_3.30_28-08-2003_NEM-4
08:00:46@196613	Sony	Clie PEG TH55	unknown
00:0A:D9@4063698	Sony Ericsson	T610	R1L013
00:0E:07@4063698	Sony Ericsson	T630	unknown
00:0A:D9@917518	Sony Ericsson	P800	CXC12529 R2C

00:0A:D9@1179718	Sony Ericsson	P900	unknown
00:0A:D9@1180018	Sony Ericsson	P900	unknown
00:0A:D9@1179678	Sony Ericsson	P900	unknown
00:0A:D9@4063698	Sony Ericsson	Z600	unknown
00:01:E3@1188286	Siemens	S55	unknown
00:01:E3@1537756	Siemens	S55	PDate: 2003-03-31 SW-Version: 12 SW-Date: 2003-03-28 Variant: A 159 Std-MAp/SW: 76/14
00:01:E3@1957354	Siemens	S65	unknown
01:90:71@1957354	Siemens	SK65	unknown
00:01:E3@1704023	Siemens	SX1	Product: SX1 P-Date: 2003-12-14 SVN: 05 Appl SW Date: 21112003 Appl SW: 12:2_05Date: 2003-11-21 Modem Variant: B 101 Std-Map/SW: 1/5 D-Map/Prov.: 1/6 Variant Name: SX1_TMOD-uk-de-nl_05_0003 Lang T9: uk-de-nl/uk-de-nl Rolf Variant Name: SX1_TMOD-uk-de-nl_05_0003 Rolf lang T9: uk-de-nl/uk-de-nl Codecs: FR/EFR/HR Audio-Par.: NfV 16 Acc.: None
00:E0:00@983040	Siemens Fujitsu	LOOX 600	Operating System Version: 3.0 (PocketPC but version is unknown)

References

[1] Martin Herfurt. Bluesnarfing @ CeBIT 2004 – Detecting and Attacking bluetooth-enabled Cellphones at the Hannover Fairground. Technical report, trifinite.org, http://trifinite.org/trifinite_downloads.html, March 2004.

[2] Symbian Ltd. Symbian OS. http://www.symbian.com.

[3] Networks Associates Technology, Inc. Symbian Cabir. http://vil.nai.com/vil/content/v_126245.htm, June 2004.

[4] Ben Laurie Adam Laurie. Serioius flaws in bluetooth security lead to disclosure of personal data. Technical report, A.L. Digital Ltd., http://bluestumbler.org/, January 2004.

[5] Martin Herfurt. BlueBug. Technical report, trifinite.org, http://trifinite.org/trifinite_stuff_bluebug.html, April 2004.

[6] Martin Herfurt. BlueSmack. Technical report, trifinite.org, http://trifinite.org/trifinite_stuff_bluesmack.html, December 2004.

[7] IEEE OUI and Company_id Assignments. http://standards.ieee.org/regauth/oui/oui.txt, 2004.

[8] Bluetooth SIG Inc. Bluetooth-The Official Bluetooth Membership Site. https://www.bluetooth.org.

[9] BlueZ Project. BlueZ – Official Linux Bluetooth protocol stack. http://www.bluez.org.

[10] Martin Herfurt Collin R. Mulliner. Blueprint - proof-of-concept implementation for Bluetooth fingerprinting. http://www.trifinite.org, December 2004.

[11] Collin R. Mulliner. bluetooth device security database. http://www.betaversion.net/btdsd/, November 2003.

[12] Wikipedia - The Free Encyclopedia. IrOBEX. http://en.wikipedia.org/wiki/OBEX.