New Efficient Intrusion Detection and Prevention System for Bluetooth Networks

Keijo M.J. Haataja
Department of Computer Science
University of Kuopio
P.O.Box 1627
FIN-70211 Kuopio, Finland
haataja@cs.uku.fi

ABSTRACT

In this paper, we investigate how various Bluetooth security attacks in progress can be prevented and stopped by monitoring communication for discovery of such attacks. Moreover, we propose an idea of the new efficient Intrusion Detection and Prevention System for Bluetooth networks to prevent attacks in progress. Proposed system is based on the set of rules that are used to identify strange communication behaviour of Bluetooth devices.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Communications Applications

Keywords

Ad-hoc Networks, Bluetooth, Intrusion Detection, Intrusion Prevention, Security Attacks, Wireless Security

1. INTRODUCTION

The use of wireless communication systems and their interconnections via networks have grown rapidly in recent years. Because radio frequency (RF) waves can penetrate obstacles, wireless devices can communicate with no direct line-of-sight between them. This makes RF communication easier to use than wired or infrared communication, but it also makes eavesdropping easier. Moreover, disrupting and jamming of wireless RF communication is easier than that of wired communication. Because wireless RF communication can suffer from these new threats, additional countermeasures are needed to protect against them.

Bluetooth [1] is a technology for short range wireless data and realtime two-way voice transfer. It operates at 2.4 GHz frequency in the free ISM-band (Industrial, Scientific, and Medical) by using frequency hopping. Bluetooth devices that communicate with each other form a *piconet*. The device that initiates a connection is the piconet *master*. One piconet can have maximum of seven active *slave* devices and one master device. All communication within a piconet goes through the piconet master.

Connection types define the possible ways Bluetooth devices can exchange data. Bluetooth has three connection types: ACL (Asynchronous Connection-Less), SCO (Synchronous Connection-Oriented) and eSCO (Extended SCO).

ACL links are for symmetric (maximum of 1306.9 kb/s for both directions) or asymmetric (maximum of 2178.1 kb/s for send and 177.1 kb/s for receive) data transfer. Retransmission of packets is used to ensure the integrity of data.

SCO links are symmetric (maximum of 64 kb/s for both directions) and they are used for transferring realtime two-way voice. Retransmission of voice packets is not used. Therefore, when the channel BER (Bit-Error-Rate) is high, voice can be distorted.

eSCO links are also symmetric (maximum of 864 kb/s for both directions) and they are used for transferring realtime two-way voice. Retransmission of packets is used to ensure the integrity of data (voice). Because retransmission of packets is used, eSCO links can also carry data packets, but they are mainly used for transferring realtime two-way voice. Only Bluetooth 1.2 (or later) devices can use eSCO links, but SCO links must also be supported to provide backward-compatibility.

Many kinds of Bluetooth devices, such as mobile phones, laptops, PCs, headsets, mice, keyboards and printers, are widely used all over the world. On November 14th 2006, the one billionth Bluetooth device was shipped [2], and the volume is expected to increase rapidly in the near future. According to the Bluetooth SIG (Special Interest Group), the target volume for 2010 is as high as two billions Bluetooth devices. Therefore, it is very important to keep Bluetooth security issues up-to-date.

Our results: In this paper, we investigate how various Bluetooth security attacks in progress can be prevented and stopped by monitoring communication for discovery of such attacks. Moreover, we propose an idea of the new efficient Intrusion Detection and Prevention System for Bluetooth networks to prevent attacks in progress.

The rest of the paper is organized as follows. Section 2 provides an overview of Bluetooth security. Set of rules used to identify strange communication behaviour of Bluetooth devices are devised in Section 3. Section 4 proposes an idea of the new efficient Intrusion Detection and Prevention System. Finally, Section 5 concludes the paper.

Mobilware'08, February 12-15, 2008, Innsbruck, Austria.

2. AN OVERVIEW OF BLUETOOTH SECURITY

The basic Bluetooth security configuration is done by the user who decides how a Bluetooth device will implement its connectability and discoverability options. The different combinations of connectability and discoverability capabilities can be divided into three categories, or *security levels*:

1. *Silent*: The device will never accept any connections. It simply monitors Bluetooth traffic.

2. *Private*: The device cannot be discovered, i.e. it is a so-called *non-discoverable device*. Connections will be accepted only if the *BD_ADDR (Bluetooth Device Address)* of the device is known to the prospective master. A 48-bit BD_ADDR is unique and refers globally to only one individual Bluetooth device.

3. *Public*: The device can be both discovered and connected to. It is therefore called a *discoverable device*.

Because Bluetooth is a wireless communication system, there is always a possibility that the transmission could be deliberately jammed or intercepted, or that false or modified information could be passed to the piconet devices. To provide protection for the piconet, the system can establish security at several protocol levels. Bluetooth has built-in security measures at the link level.

Security threats in distributed networks, such as Bluetooth, can be divided into three categories: disclosure, integrity and DoS (Denial-of-Service) threat. *Disclosure threat* means that information can leak from the target system to an eavesdropper that is not authorized to access the information. *Integrity threat* concerns the deliberate alteration of information in an attempt to mislead the recipient. *DoS threat* involves blocking of access to a service, making it either unavailable or severely limiting its availability to an authorized user. [3]

Powerful directional antennas can be used to increase the scanning, eavesdropping and attacking range of almost any kind of Bluetooth attack considerably. One very good example of a long-distance attacking tool is the BlueSniper Rifle [4, 5]. It is a rifle stock with a powerful directional antenna attached to a small Bluetooth compatible computer. Scanning, eavesdropping and attacking can be done over a mile away from the target devices. Moreover, anyone with some basic skills and a few hundred dollars can build his own BlueSniper Rifle. Therefore, the possibility that an attacker is using range enhancement for improving the performance of the attacks should be taken seriously.

Nowadays it is also possible to transform a standard 30 USD Bluetooth dongle into a full-blown Bluetooth sniffer [6, 7]. We have also verified this fact in our research laboratory with many different CSR-based (Cambridge Silicon Radio) Bluetooth USB dongles supporting Bluetooth versions up to 2.0+EDR (Enhanced Data Rate). In addition, tools for reverse engineering the firmware of CSR-based Bluetooth dongles are available [8]. The tools include a disassembler for the official firmware, and an assembler that can be used for writing custom firmware. With these tools anyone can now write custom firmware for CSR-based Bluetooth dongles to include raw access for Bluetooth sniffing. The tools also include the source code for sniffing Bluetooth under Linux. Moreover, it is expected that in the near future techniques for finding hidden (non-discoverable) Bluetooth devices within a few minutes [9, 10] will be ported onto a standard CSR dongle via a custom firmware. This will open new doors for practical Bluetooth security research and it will also provide a cheap basic weapon to all attackers for Bluetooth sniffing. It is expected that Bluetooth sniffing will soon become a very popular sport among attackers and hackers, thus making Bluetooth security concerns even more alarming.

There are also four different *security modes* that a device can implement. In Bluetooth technology, the device can be only in one of the following security modes at a time:

1. *Nonsecure*: Bluetooth device does not initiate any security measures.

2. *Service-level enforced security mode*: Two Bluetooth devices can establish a nonsecure ACL link. Security procedures, namely authentication, authorization and optional encryption, are initiated when an L2CAP (Logical Link Control and Adaptation Protocol) CO (Connection-Oriented) or an L2CAP CL (Connection-Less) channel request is made.

3. *Link-level enforced security mode*: Security procedures are initiated when the ACL link is established.

4. *Service-level enforced security mode*: This mode is similar to mode 2, except that only Bluetooth devices using SSP (Secure Simple Pairing) can use it, i.e. only Bluetooth 2.1+EDR (or later) devices can use this security mode.

Authentication is used for proving the identity of one piconet device to another. The results of authentication are used for determining client's *authorization level*. *Encryption* is used for encoding the information being exchanged between Bluetooth devices in a way that eavesdroppers cannot read its contents.

Bluetooth uses *SAFER+ (Secure And Fast Encryption Routine +)* with a 128-bit key as an algorithm for authentication and key generation. SAFER+ is a block cipher developed by Massey et al. in 1998 [11]. Although some optimizations for faster breaking of SAFER+ exist, such as in [12], it is considered secure. [1, 12]

Bluetooth security is based on building a chain of events, none of which provides meaningful information to an eavesdropper, and all events occur in a specific sequence for security to be set up successfully. Two Bluetooth devices begin with the same *PIN (Personal Identification Number)* code that is used for generating several 128-bit keys as Figure 1 illustrates. Each master-slave pair can have a different PIN code for providing trusted relationship between the devices.

An *initialization key* (K_{init}) is generated when Bluetooth devices meet for the first time and it is used for securing the generation of other more secure 128-bit keys, which are generated during the next phases of the security chain of events. The K_{init} is derived from a 128-bit pseudorandom number IN_RAND, an L-byte ($1 \leq L \leq 16$) PIN code, and a BD_ADDR. It is worth noting that the IN_RAND is sent via air in unencrypted form. If one device has the fixed PIN code, the BD_ADDR of the another device is used. If both devices can support a variable PIN code, the BD_ADDR of the device that received the IN_RAND is used. The K_{init} is used to encrypt a 128-bit pseudorandom number (RAND

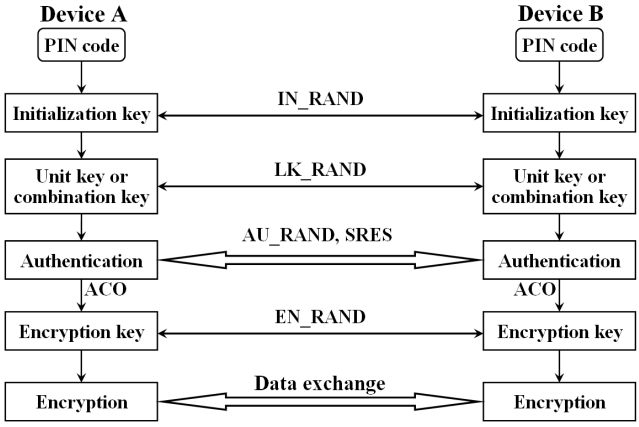

Figure 1: Summary of Bluetooth security operations

or LK_RAND) exchanged in the next phase of the security chain of events when a link key (a unit key or a combination key) is generated.

A *unit key (K_A)* is derived from the information of only one device (BD_ADDR_A, $RAND_A$). Only devices that have limited resources, i.e. no memory to store several keys, should use the K_A, because it provides only a low level of security. Therefore, Bluetooth specifications [1] do not recommend to use the K_A anymore.

A *combination key (K_{AB})* is always dependent on two devices and therefore it is derived from the information of both devices (BD_ADDR_A, LK_RAND_A, BD_ADDR_B, LK_RAND_B). It is worth noting that producing the K_{AB} is nothing more than a simple bitwise XOR between two unit keys, i.e. $K_{AB} = K_A \oplus K_B$. Each device can produce its own unit key and each device also has the BD_ADDR of other device. Therefore, two devices have to exchange only their respective pseudorandom numbers in order to produce each other's unit key.

The next phase of security chain of events is the *challenge-response authentication* in which claimant's knowledge of a secret link key is checked. During each authentication, a new 128-bit pseudorandom number AU_RAND is exchanged via air in unencrypted form. The claimant returns a 32-bit result (SRES, Signed Response) to the verifier via air in unencrypted form. The verifier also calculates the same SRES value and compares it to the received SRES. If SRES values match, the authentication is completed successfully, and a 96-bit result (ACO, Authenticated Ciphering Offset) is produced in both devices.

An *encryption key (K_C)* is derived from the ACO, the current link key, and a 128-bit pseudorandom number EN_RAND. The master generates the EN_RAND and sends it to the slave via air in unencrypted form. The K_C is one input to the keystream generator that makes symmetric encryption possible by generating the same cipher bit stream, or the *keystream*, in both devices. Other inputs to the keystream generator are the BD_ADDR of the master (BD_ADDR_A) and 26 bits of the master realtime clock (CLK_{26-1}).

It is worth noting that only payload of the Bluetooth Baseband packet is encrypted (not an access code or a header), and therefore an attacker cannot use the regularly repeating information (that is easy to guess by the attacker) of the access code and the header in order to break the cipher faster.

In Bluetooth versions up to 2.0+EDR, pairing is based exclusively on the fact that both devices share the same PIN code or passkey. The PIN is the only source of entropy for the shared secret. As the PINs often contain only four decimal digits, the strength of the resulting keys is not enough for protection against passive eavesdropping on communication. Even with longer 16-character alphanumeric PINs, full protection against active eavesdropping cannot be achieved: it has been shown that MITM (Man-In-The-Middle) attacks on Bluetooth communications can be performed [13, 14, 15].

Bluetooth version 2.1+EDR [1] adds a new specification for the pairing procedure, namely SSP. Its main goal is to improve the security of pairing by providing protection against passive eavesdropping and MITM attacks.

Instead of using (often short) passkeys as the only source of entropy for building the link keys, SSP employs Elliptic Curve Diffie-Hellman public-key cryptography. To construct the link key, devices use public-private key pairs, a number of nonces, and Bluetooth addresses of the devices. Passive eavesdropping is effectively thwarted by SSP, as running an exhaustive search on a private key with approximately 95 bits of entropy is currently considered to be infeasible in short time.

In order to provide protection against MITM attacks, SSP either uses an Out-Of-Band (OOB) channel (e.g., Near Field Communication, NFC), or asks for the user's help: for example, when both devices have displays and keyboards, the user is asked to compare two six-digit numbers. Such a comparison can be also thought as an OOB channel which is not controlled by the MITM. If the values used in the pairing process have been tampered with by the MITM, the six-digit integrity checksums will differ with the probability of 0.999999.

SSP uses four *association models*. In addition to the two association models mentioned previously (*OOB* and *Numeric Comparison*), models named *Passkey Entry* and *Just Works* are defined. The Passkey Entry model is used in the cases when one device has input capability, but no screen that can display six digits. A six-digit checksum is shown to the user on the device that has output capability, and the user is asked to enter it on the device with input capability. The Passkey Entry model is also used if both devices have input, but no output capabilities. In this case the user chooses a 6-digit checksum and enters it in both devices. Finally, if at least one of the devices has neither input nor output capability, and an OOB cannot be used, the Just Works association model is used. In this model the user is not asked to perform any operations on numbers; instead, the device may simply ask the user to accept the connection.

The choice of the association model depending on the device capabilities is shown in Table 1. DisplayYesNo indicates that the device has a display and at least two buttons that are mapped to "yes" and "no": using the buttons the user can either accept the connection or decline it. Other notation in the table is self-explanatory.

Secure Simple Pairing is comprised of six phases:

1. *Capabilities exchange*: The devices that have never met before or want to perform re-pairing for some reason, first exchange their IO (Input/Output) capabilities (see Table 1) to determine the proper association model to be used.

Table 1: Bluetooth device capabilities and SSP association models [1]

Device 1	Device 2	Association Model
DisplayYesNo	DisplayYesNo	Numeric comparison*
	DisplayOnly	Numeric comparison
	KeyboardOnly	Passkey Entry*
	NoInputNoOutput	Just Works
DisplayOnly	DisplayOnly	Numeric comparison
	KeyboardOnly	Passkey Entry*
	NoInputNoOutput	Just Works
KeyboardOnly	KeyboardOnly	Passkey Entry*
	NoInputNoOutput	Just Works
NoInputNoOutput	NoInputNoOutput	Just Works

* The resulting link key is considered *authenticated*.

2. *Public key exchange*: The devices generate their public-private key pairs and send the public keys to each other. They also compute the Diffie-Hellman key.

3. *Authentication stage 1*: The protocol that is run at this stage depends on the association model. One of the goals of this stage is to ensure that there is no MITM in the communication between the devices. This is achieved by using a series of nonces, commitments to the nonces, and a final check of integrity checksums performed either through the OOB channel or with the help of user.

4. *Authentication stage 2*: The devices complete the exchange of values (public keys and nonces) and verify the integrity of them.

5. *Link key calculation*: The parties compute the link key using their Bluetooth addresses, the previously exchanged values and the Diffie-Hellman key constructed in phase 2.

6. *LMP authentication and encryption*: Encryption keys are generated in this phase, which is the same as the final steps of pairing in Bluetooth versions up to 2.0+ EDR.

The contents of messages sent during the SSP are outlined in Figure 2, and used notations are explained in Table 2. Even though SSP improves the security of Bluetooth pairing, it has been shown that MITM attacks against Bluetooth 2.1+EDR devices are also possible. [16]

3. RULES FOR IDENTIFYING STRANGE COMMUNICATION BEHAVIOUR

Based on the strange communication behaviour of such Bluetooth devices that are performing various security attacks, we defined a set of rules to help identifying attacks in progress:

1. *Unusually many repeated failed authentication attempts*: This may indicate that an attacker is using On-Line PIN Cracking attack [17, 18] to discover the secret PIN code of a victim device.

2. *Unusually many repeated successful authentications and disconnections*: This may indicate that the attacker is performing a DoS attack (see Section 2).

Table 2: SSP protocol notations [1]

Term	Definition
PKx	Public key of device X
SKx	Private key of device X
DHKey	Diffie-Hellman key generated after key exchange
Nx	Nonce generated by device X
rx	Random number generated by device X; equals 0 in the Numeric Comparison association model
Cx	Commitment value from device X
f1	One-way function used to compute commitment values
f2	One-way function used to compute the link key
f3	One-way function used to compute check values
g	One-way function used to compute numeric check values
IOcapX	Input/Output capabilities of device X
BD_ADDR	48-bit Bluetooth device address

3. *Unusually many NAK transmissions*: This may indicate that the attacker is performing Big NAK (Negative Acknowledgement) attack [3] and thus putting the victim device on endless retransmission loop.

4. *Unusually long delays*: This may indicate that a MITM is between the communicating parties (see Section 2).

5. *Unusually many repeated POLL packets*: This may indicate that the attacker is keeping victim devices busy so that they will not go into sleep or low-power mode.

6. *Unusually high BER*: This may indicate that the attacker is disrupting the PHY.

7. *Unusually heavy traffic between two communicating parties*: This may indicate that the attacker is performing Battery Exhaustion attack [3].

8. *Sudden increase in transmit powers*: This may indicate that the attacker is using stronger RF signal in order to displace the active piconet device via Exploitation of a stronger RF signal attack [3].

9. *Two identical BD_ADDRs in the range of vulnerability*: This may indicate that the attacker is using BD_ADDR Duplication attack [3] in order to deny the access to the services from the legitimate piconet devices. Another possibility is that the attacker is performing some kind of impersonation attack in order to mislead the legitimate piconet devices.

10. *HV1 (High-quality Voice 1) SCO link established with the piconet master when other type of SCO or eSCO link could also have been used*: This may indicate that the attacker is performing SCO/eSCO attack [3] in order to reserve all piconet resources and thus the legitimate piconet devices are not getting the service within a reasonable time.

11. *L2CAP level request for the highest possible data rate or the smallest possible latency*: If such request is accepted, all throughput is reserved for the attacker and the legitimate piconet devices are not getting the service within a reasonable time.

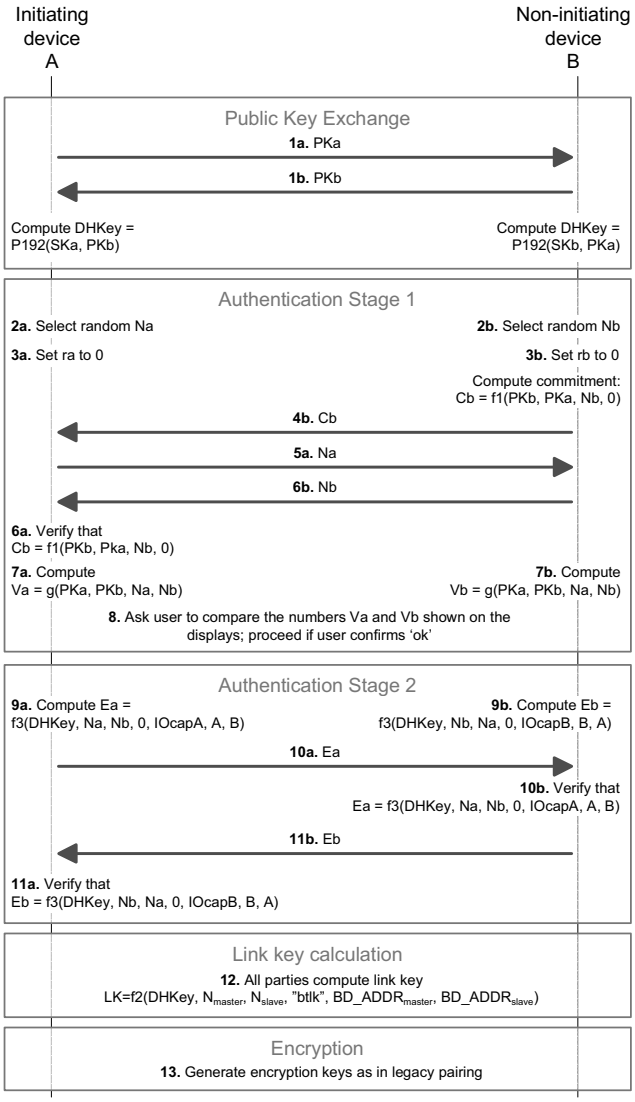

Figure 2: Numeric Comparison model of SSP [1]

12. *Surprising connection attempts and data transfer requests from unknown Bluetooth devices*: This may indicate that Bluetooth virus or worm is trying to infect legitimate piconet devices.

13. *A Bluetooth device requests that the length of an encryption key must be shorter than 128 bits*: This may indicate that an attack against the Bluetooth encryption is in progress.

14. *RF signature mismatch*: This indicates that some kind of attack, such as impersonation attack, is in progress. Every transmitter has a unique RF signature [3, 19] which can be used to differentiate the legitimate devices from the devices that have alien RF signatures, i.e. a sample RF signature is needed from each legitimate device in order to detect alien RF signatures.

15. *SSP's Just Works association model activated between such devices that could use a more secure option (e.g. Numeric Comparison or OOB)*: This may indicate that a MITM is between the communicating parties (see Section 2). Only Bluetooth 2.1+EDR (or later) devices support SSP.

4. INTRUSION DETECTION AND PREVENTION SYSTEM

Our idea of the Intrusion Detection and Prevention System for Bluetooth networks is based on the set of rules described in Section 3. Our system consist of two parts: *Intrusion Detection System* and *Intrusion Prevention System*. In our system some commercially available Bluetooth protocol analyzer, such as LeCroy BTTracer/Trainer [20], equipped with signalling processing capabilities (some additional signalling processing hardware is required) takes care of the intrusion detection part.

When an intrusion is detected, the protocol analyzer immediately informs the network administrator (via Bluetooth) that Bluetooth network in under attack. This is so-called *manual administrative intrusion prevention* that can be used in all cases regardless of the capabilities of the legitimate Bluetooth devices. This system also requires LeCroy BTTracer/Trainer v2.2 software (or later) that provides CATC Scripting Language [21]. CATC Scripting Language provides an easy and efficient way for implementing various Bluetooth communicating issues in practice by allowing users to create C language scripts in order to automate the use of protocol analyzer.

The second part of our system, Intrusion Prevention System, is a small program that runs on all legitimate Bluetooth devices that allow programs to be installed, i.e. at least all PCs, laptops and mobile phones should be supported. This is so-called *automatic intrusion prevention*. It requires that all legitimate Bluetooth devices must run this special program in order to receive warning messages from our Intrusion Detection System. When a warning message is received, those devices that are under attack, perform automatic disconnection and refuse any further Bluetooth connections for predestined time. Intrusion Detection System also sends enough information (BD_ADDR, device capabilities information, user friendly name of the device, RF signature information, and so on) to the Intrusion Prevention System so that further connections from the same origin can be refused immediately by the Intrusion Prevention System.

Our Intrusion Detection System should work in the following way:

1. *A Bluetooth protocol analyzer monitors Bluetooth communication of the legitimate piconet devices non-stop*: Protocol analyzer have all legitimate BD_ADDR values that are allowed to communicate within the piconet and also other useful information about such devices (device capabilities information, user friendly name of the device, RF signature information, and so on).

2. *When an intrusion is detected, either manual administrative intrusion prevention or automatic intrusion prevention is applied*: We strongly recommend that the automatic intrusion prevention should be implemented. However, if automatic intrusion prevention is not implemented, at least Bluetooth network administrator is alerted immediately!

Our automatic Intrusion Prevention System should work in the following way:

1. *Intrusion Prevention System receives a warning message from the Intrusion Detection System*: When a warning message is received, automatic disconnection is performed and further Bluetooth connections are refused for predestined time.

2. *Intrusion Prevention System also receives enough information to prevent further attacks from the same origin*: We recommend that at least the following information about the attacking device should be received from the Intrusion Detection System and stored to the database: BD_ADDR value, device capabilities information, user friendly name of the device, and RF signature information.

5. CONCLUSIONS

It is difficult to create such an intrusion detection and prevention system that caters to all possible types of security attacks, as the security of Bluetooth is likely to be limited by the capabilities of the least powerful or the least secure device type. In fact, most Bluetooth security attacks are based exactly on this problem.

A set of rules that help to identify Bluetooth security attacks in progress were devised in the paper. Moreover, an idea of the new efficient Intrusion Detection and Prevention System for Bluetooth networks to prevent attacks in progress were proposed.

In general, security attacks are hard to prevent in wireless networks, especially when intrusion detection and prevention system is not used. This paper is intended to help Bluetooth network administrators and Bluetooth device manufacturers to implement efficient Bluetooth intrusion detection and prevention systems!

6. REFERENCES

[1] Bluetooth SIG. *Bluetooth specifications 1.0, 1.1, 1.2, 2.0+EDR and 2.1+EDR*. Technical specifications, https://www.bluetooth.org, 1999–2007.

[2] Bluetooth SIG. *Bluetooth Technology in Hands of One Billion*. Press release, http://www.bluetooth.com/Bluetooth/SIG/Billion.htm, November 14, 2006.

[3] R. Morrow. *Bluetooth – Operation and use*. New York, McGraw-Hill, 2002.

[4] H. Cheung. *How To: Building a BlueSniper Rifle – Part 1*. SmallNetBuilder, Pudai LLC, homepage, http://www.smallnetbuilder.com/content/view/24256/98, 2005.

[5] H. Cheung. *How To: Building a BlueSniper Rifle – Part 2*. SmallNetBuilder, Pudai LLC, homepage, http://www.smallnetbuilder.com/content/view/24228/98, 2005.

[6] Bluetooth security & Bluetooth hackers community blog. *Bluetooth Sniffing For Less*. Homepage, http://bluetoothsecurity.wordpress.com/2007/05/12/bluetooth-sniffing-for-less, 2007.

[7] M. Moser. *Busting The Bluetooth Myth – Getting RAW Access*. Remote-exploit.org, Research Report, http://www.remote-exploit.org/research/busting_bluetooth_myth.pdf, 2007.

[8] Darkircop. *CSR Sniffer – Firmware assembler and disassembler*. Homepage, http://darkircop.org/bt/bt.tgz, 2007.

[9] D. Spill and A. Bittau. BlueSniff – Eve meets Alice and Bluetooth. *Proceedings of the First USENIX Workshop on Offensive Technologies (WOOT'2007)*, Boston, MA, August 6, 2007.

[10] D. Spill and A. Bittau. *BlueSniff*. University College London, http://www.cs.ucl.ac.uk/staff/a.bittau/gr-bluetooth.tar.gz, 2007.

[11] J. Massey, G. Khachatrian, and M. Kuregian. SAFER+. *Proceedings of the 1st Advanced Encryption Standard Candidate Conference*, National Institute of Standards and Technology, August 20–22, 1998.

[12] Y. Shaked and A. Wool. Cracking the Bluetooth PIN. *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys'2005)*, Seattle, WA, June 6–8, pp. 39–50, 2005.

[13] M. Jakobsson and S. Wetzel. Security weaknesses in Bluetooth. *Lecture Notes in Computer Science*, vol. 2020, Springer-Verlag, pp. 176-191, 2001.

[14] D. Kügler. Man in the middle attacks on Bluetooth. *Lecture Notes in Computer Science, Financial Cryptography*, vol. 2742, Springer-Verlag, pp. 149–161, 2003.

[15] A. Levi, E. Cetintas, M. Aydos, C. Koc, and M. Caglayan. Relay Attacks on Bluetooth Authentication and Solutions. *Proceedings of the 19th International Symposium on Computer and Information Sciences (ISCIS'2004)*, Kemer-Antalya, Turkey, 2004.

[16] K. Hyppönen and K. Haataja. "Niño" Man-In-The-Middle Attack on Bluetooth Secure Simple Pairing. *Proceedings of the IEEE Third International Conference in Central Asia on Internet, The Next Generation of Mobile, Wireless and Optical Communications Networks (ICI2007)*, Tashkent, Uzbekistan, September 26-28, 2007.

[17] O. Whitehouse. *@Stake – Where Security & Business Intersect*. Research report, CanSecWest/core04, Vancouver, http://cansecwest.com/csw04archive.html, 2004.

[18] K. Haataja. Two Practical Attacks Against Bluetooth Security Using New Enhanced Implementations of Security Analysis Tools. *Proc. 2nd IASTED International Conference on Communication, Network and Information Security (CNIS'2005)*, Phoenix, Arizona, USA, Nov. 14–16, pp. 13–18, 2005.

[19] J. Shandle. *University research aims at more secure Wi-Fi*. EE Times Online, newscopy, http://www.eetimes.com/news/latest/showArticle.jhtml?articleID=192501255, September 1, 2006.

[20] LeCroy – Protocol Solutions Group. *LeCroy BTTracer/Trainer and Merlin II*. Homepage, http://www.lecroy.com/tm/products/ProtocolAnalyzers/bluetooth.asp?menuid=60, 2007.

[21] LeCroy – Protocol Solutions Group. *CATC Scripting Language Reference Manual for LeCroy Bluetooth Analyzers – Manual Version 1.21*. LeCroy – Protocol Solutions Group, http://www.lecroy.com/tm/library/manuals/ProtocolAnalyzers/PDF/BTCSL_d121.pdf, Feb. 10, 2004.