Performance Evaluation of Bluetooth Security Mechanisms for Handheld Devices

Georgios Kambourakis*, Alexis Andreadis, Charalampos Paganos,
Angelos Rouskas and Stefanos Gritzalis

Department of Information and Communication Systems Engineering
University of the Aegean, Karlovassi, GR-83200 Samos, Greece
Tel: +30-22730-82247 Fax: +30-22730-82009
*Correspondent author email: {gkamb}@aegean.gr

***Abstract*— Bluetooth standard has been long criticized for various vulnerabilities and security inefficiencies, as its designers are trying to balance wisely between performance and complementary services including security. On the other hand, well respected security protocols like IP secure (IPsec) and Secure Shell (SSH) provide robust, low cost and easy to implement solutions for exchanging data over insecure communication links. Although, the deployment of these mechanisms is a well established and accustomed practice in the wireline world, more research effort is needed for wireless links, due to several limitations of the radio-based connections especially for handheld devices e.g. link unreliability, bandwidth, low processing power and battery consumption. This paper focuses on performance rather than on security, evaluating the efficiency of these de-facto security protocols over Bluetooth connections when low-end handheld devices are utilized. Several Personal Area Network (PAN) parameters, including absolute transfer times, link capacity and throughput, are evaluated. Our experiments employ both Bluetooth native security mechanisms as well as the two aforementioned protocols. Through a plethora of scenarios we offer a comprehensive in-depth comparative analysis of each of the aforementioned security mechanisms when deployed over Bluetooth links.**

***Index Terms*—** Bluetooth; Performance evaluation; Security; Security modes; IPSec; SSH.

I. Introduction

Bluetooth technology has already become the de-facto standard for replacing short range wired communications using radio technology [1]. According to estimations, devices incorporating Bluetooth are predicted to quadruple in number between now and 2008, from under 100m to about 440m. As a result, Bluetooth enabled devices are used in several different environments and cover a wide variety of applications. For instance, in mobile applications, a handheld device (e.g. smartphone, palmtop) periodically connects to the network to download music, to transfer files or to synchronize with one's laptop on calendar and other files [2]. Consequently, the security of these applications and the private information stored on the handheld devices becomes a prominent issue.

Thus, security features [3, 4] must be carefully considered and analyzed in order to decide whether Bluetooth technology indeed provides the right answer for any particular task or application. Until now, both Bluetooth Special Interest Group (SIG) [5] and several researchers have made a great contribution to Bluetooth security aspects, discovering numerous vulnerabilities and potential weaknesses and proposing solutions.

An obvious choice for any Bluetooth application would be to use Bluetooth encryption provided at link layer. Virtually all Bluetooth devices support this feature, and it is, in most cases, considered to be secure. However, this does not apply for all deployment scenarios. In order to establish a secure channel with another Bluetooth device, a pre-shared secret, called PIN is needed. A symmetric key is generated from this PIN. On customer devices this PIN usually consists of 4 or 5 digits. Supposing a whole piconet network would use this PIN to encrypt its communication, anyone knowing this PIN could theoretically decrypt all communication. On top of that, in applications like VoIP that mandate IP connectivity to Access Points (APs), the encryption would end at the AP, which means that the AP, or any host that can manipulate the communication between the Mobile Device and the other end, can expose the data (see Figure 1). Thus, it is obvious that Bluetooth encryption is not well suited for all applications which may exploit Bluetooth connections.

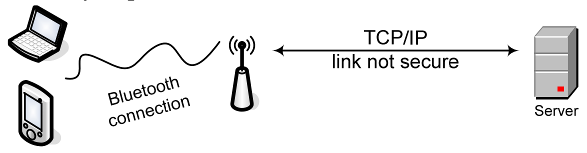

Figure 1. Scenario that requires upper layer security

Under these circumstances, the investigation of complementary and advanced security protocols apart from Bluetooth's native security mechanisms is an interesting research issue. At first, as Bluetooth wireless technology is targeting devices with particular needs and constraints (e.g. processing power and battery consumption) the trade-offs between security services and performance must be carefully considered. Moreover, as radio links generally suffer from limited bandwidth and are unreliable by nature, performance issues must be thoroughly investigated to make a decision whether certain security protocols and their mechanisms are advantageous over Bluetooth connections, delivering robust and agile security services within tolerable service response times.

During the last few years, various researchers have occupied themselves with studying various Bluetooth security parameters and some of them do explore performance parameters [6-17]. However, to the best of our knowledge, none of these works focuses on performance evaluation comparing Bluetooth's native security mechanisms with well-respected, strong security protocols like IPsec and SSH. Thus, this paper focuses on the performance issue of existing security protocols and mechanisms for handheld devices. We estimated the performance of both the built-in Bluetooth security mechanisms, namely security modes and two other standard security protocols operating at different layers of the TCP/IP protocol suite, namely SSH and IPsec. By utilizing the last two protocols, applications can communicate securely constructing a secure tunnel or Virtual Private Network (VPN).

To address the performance issue, we employed a properly designed test-bed and executed an adequate number of different scenarios testing Bluetooth security modes I and III, IPsec security mechanisms, considering Encapsulating Security Payload (ESP) and Authentication Header (AH), and finally SSH with distinct ciphering algorithms. Experiments, performed between palmtop and laptop machines, inspect several network parameters including absolute transfer times, link capacity and throughput. Although it may seem difficult to compare heterogeneous security mechanisms laying in different layers of the OSI model, our analysis concerns solely the performance in terms of service times when extra security is needed. The rest of the paper is structured as follows: the next section gives an overview of our experimental test-bed and related procedures, Section 3 presents the derived performance measurement results and Section 4 finally offers some concluding thoughts.

II. Experimental Framework Description

The experimental topology used consists of a laptop machine and a palmtop located at 10 meters apart and connected via Bluetooth adapters, thus forming a small Wireless PAN (WPAN) or piconet. The main components' characteristics, both software and hardware, are presented in Table 1. To estimate the performance of the network, the data were transmitted from one network node (server) to the other (client). Hence, in order to record the incoming and outcoming packets between the corresponding network entities and to calculate the network performance parameters we utilized on the server side the well known network analyzer "ethereal" [18], version 0.10.12, which in turn uses the "tcpdump" tool. In addition, for the Linux environment, we employed the BlueZ official Linux Bluetooth protocol stack [19], which provides support for the core Bluetooth layers and protocols.

Bluetooth supports three different security modes called security modes I, II and III, but in our tests we decided to use only security mode I, and III. Security mode I offers no real security as authentication and confidentiality services are disabled. On the other hand, security mode II provides security services after the connection between the two devices has been established and only if a given application has requested them. Thus, the security services in mode II depend on the application running. The last security mode is the most powerful among the three modes, because it mandates both authentication and confidentiality built-in mechanisms independently of the application running. These mechanisms are referred to as Bluetooth baseband security procedures, where the baseband layer deals with the SAFER+ algorithms [20]. As implied, one of the terminals was acting as a client and the other one as the server. Therefore, the server should require security and the client should respond accordingly.

Laptop Server	
Processor	Intel Celeron M. – 1.4 GHz
RAM	256 MB
Operating System	SUSE Linux Ver. 10.0
Bluetooth Adapter	Trust Bluetooth adapter Class 1
Palmtop Client	
Model	HP iPAQ h5400
Processor	400 MHz Intel XScale PXA250
RAM	64 MB
Operating System	Familiar PDA OS 0.8.4
Bluetooth Adapter	Bluetooth 1.1 compliant

Table 1. Hardware and software characteristics of the engaged machines

For IPsec, the engaged machines must have the same security policies in order to communicate securely. So, we configured Linux to use MD5 and SHA1 algorithms for data integrity and DES, 3DES algorithms for confidentiality in both machines by installing IPsec-tools (http://ipsec-tools.sourceforge.net/) and Openswan (www.openswan.org) as well. For SSH secured communication we used OpenSSH. In fact, many open-source projects exist. In addition to FreeS/WAN and openswan which both enable IPsec in the Linux kernel, openvpn (http://openvpn.net/) can be used to create TLS-encrypted point-to-point connections. For SSH confidentiality services we chose four algorithms to test namely, 3DES, AES, Arcfour and Blowfish. Finally, for both IPsec and SSH we employed only symmetric cryptography and manual keying procedures for the authentication of parties considering the fact that usually Bluetooth piconets are formed ad-hoc and their users do not hold public key certificates.

III. Performance Measures

As mentioned before, the experimental procedure consists of three main parts: evaluation of Bluetooth built-in security modes I (no security), and III (strong security), and estimation of the performance of IPsec and SSH mechanisms over Bluetooth links. In all scenarios we gathered measurements for the subsequent network performance parameters: absolute file Transfer Time (TT), Achieved Transfer Rate (ATR), and Throughput (THR). All measurements took place at the server node because of its processing power.

- ❑ The *transfer_time* represents the actual duration of transfers during a transaction.
- ❑ The *achieved_transfer_rate* represents the actual transfer rate achieved during a transaction.

In an ideal scenario, a constant data rate should be maintained between the two communication end-points. However, due to various reasons, mainly related to the wireless medium nature, this parameter is changing over time. We should underline the fact that bytes_sent and bytes_received could also contain retransmitted bytes.
achieved_transfer_rate(Kbps) = ((bytes_sent + bytes_received) * 8) / TT

- ❑ *Throughput* represents the percentage of *achieved_transfer_rate* over the practical *maximum_transfer_rate* of the link, which in our case is 723 Kbps:
throughput(%) = *achieved_transfer_rate* / *max_transfer_rate* * 100.

Finally, *achieved_transfer_rate_improvement* is a comparison metric that indicates the improvement of the achieved_transfer_rate with respect to the Bluetooth mode I achieved transfer rate *achieved_transfer_rate_B_I* and is calculated as:

achieved_transfer_rate_improvement(%)=
(*achieved_transfer_rate* - *achieved_transfer_rate_B_I*)
/ *achieved_transfer_rate_B_I* * 100

A positive value implies that the performance (or channel throughput) has increased compared to the Bluetooth mode I achieved transfer rate, while a negative one means that the performance has decreased. Measurements were gathered during repeated FTP file transfers, between the laptop server and the PDA client. Each file was transferred twelve times and only average values were recorded. In all scenarios, the ping response times between client and server were varying among 19.7 and 21.8 msecs.

A. *Bluetooth Security modes I and III evaluation*

Measurements for testing Bluetooth modes I and III were gathered by transferring four different files between the client and the server. The files' sizes were 5.26, 7.0, 10.5 and 15 Mbytes respectively. Figure 2 provides a graphical representation of these values. As we can easily notice the results are generally as expected, but there are some interesting points which need further analysis. At first, the TT metric is slightly higher for mode III, as well as the ATR is higher for mode I. This happens because mode III mandates authentication (handshake) at the beginning of each transaction. Keep in mind that the handshake time is included in TT too.

Moreover, encryption algorithms are applied during the transaction for mode III and as a result the overall transfer time is increased. We can also perceive that the larger the file size is, the longer the TT difference between mode I and mode III is expected to be. This situation is also depicted in the respective plot of Figure 2. In general, these measurements advocate that mode I utilizes the network better than mode III. Because of the volatile nature of the wireless link, we also report standard deviation (SD) for the measured values in Table 2.

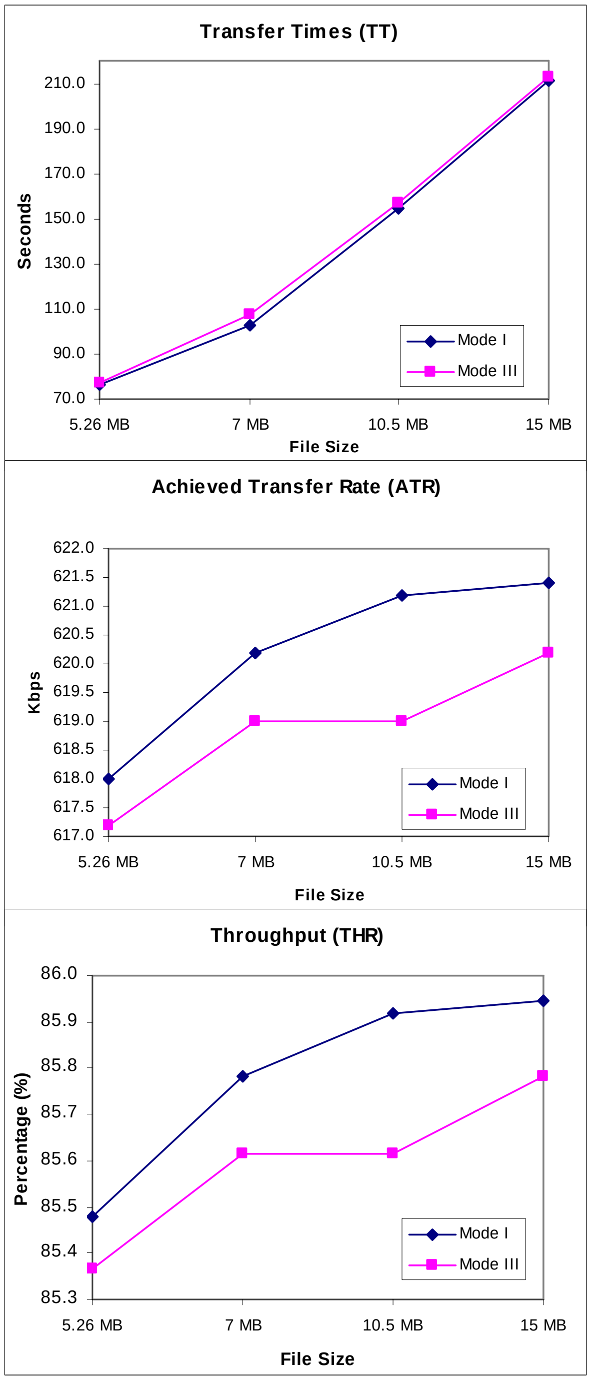

Figure 2. Average metric values for network parameters measured / Bluetooth Modes I & III

	MODE I		
File Size (MB)	TT (sec)	ATR (Kbps)	THR (%)
5.26	0.5	2.6	0.4
7	0.1	0.9	0.1
10.5	0.4	1.6	0.2
15	0.2	0.5	0.1
	MODE III		
5.26	0.1	1.3	0.2
7	0.5	3.2	0.4
10.5	0.1	0.5	0.1
15	0.6	2.2	0.3

Table 2. Standard deviation for all Bluetooth scenarios

B. *Secure Shell (SSH) Evaluation*

Experimental procedures for the SSH mechanism [21, 22] consider the transfer of the same four files, as before, between the client and the server. Table 3 displays the average times of all metrics used, while Table 4 presents the corresponding standard deviation values.

	5.26 MB		
	TT (sec)	ATR (Kbps)	THR (%)
3DES	90.1	526.4	72.8
AES128	90.2	525.6	72.7
Arcfour	90.5	523.8	72.5
Blowfish	90.5	523.6	72.4
	7 MB		
3DES	116.9	555.6	76.9
AES128	116.9	556.2	76.9
Arcfour	117.3	554.2	76.6
Blowfish	117.6	552.8	76.4
	10.5 MB		
3DES	163.0	581.8	80.5
AES128	162.9	582.4	80.5
Arcfour	163.1	581.6	80.5
Blowfish	162.8	582.6	80.6
	15 MB		
3DES	221.3	603.2	83.4
AES128	221.3	603.6	83.5
Arcfour	221.6	602.4	83.3
Blowfish	222.1	601.2	83.1

Table 3. Average values for network parameters measured (SSH)

As we can notice, SSH gives highly increased transfer times when compared to Bluetooth security modes. For instance, we can spot a difference of +12.6 seconds to +13.4 seconds for the smallest file depending on the cipher used. Moreover, it is more than obvious that all the ciphers used are more or less of the same performance. This is easily proven if we examine for example the achieved transfer rates in each case, which shown very slight differences.

Another interesting assumption that we can make is that as the size of the file increases, the achieved transfer rate and the throughput becomes bigger. This happens because of the procedure of the authentication which takes place during the initial SSH handshake. In any case it should be noted that the improvement in the achieved transfer rates always compared to Bluetooth Security Mode I, induced by SSH are negative for any scenario. This means that Bluetooth's native mechanisms offer better bandwidth and network utilization at almost all cases examined. This remark is confirmed by the values given in Table 5.

	5.26 MB		
	TT	ATR	THR
3DES	0.4	2.1	0.3
AES128	0.9	5.5	0.7
Arcfour	0.1	0.4	0.1
Blowfish	0.6	3.8	0.5
	7 MB		
3DES	0.4	2.1	0.3
AES128	0.4	1.9	0.2
Arcfour	0.2	1.1	0.2
Blowfish	1.0	4.9	0.7
	10.5 MB		
3DES	1.0	3.3	0.5
AES128	1.0	3.9	0.5
Arcfour	0.5	1.9	0.3
Blowfish	0.6	1.9	0.3
	15 MB		
3DES	0.8	2.5	0.3
AES128	0.9	2.3	0.3
Arcfour	0.6	1.7	0.2
Blowfish	0.7	1.9	0.3

Table 4. Standard deviation for all SSH scenarios

Size	Bluetooth Mode I	3DES	AES128	RC4	Blowfish
5.26	618.0	-14.8	-15.0	-15.2	-15.3
7	620.2	-10.4	-10.4	-10.6	-10.9
10.5	621.2	-6.3	-6.2	-6.4	-11.0
15	621.4	-2.9	-2.9	-3.3	-3.3

Table 5. % ATR deterioration for SSH

C. *IPsec evaluation*

The procedure for the IPsec protocol [23-25] considers once again the transfer of the same four files between the client and the server. IPsec uses two mechanisms (protocols) that may be used independently or jointly to secure the outcoming traffic, namely Authentication Header (AH) offering data origin, connectionless data integrity and optionally replay protection, and Encapsulating Security Payload (ESP) offering confidentiality and protection against traffic analysis. In our scenarios we utilized both mechanisms, using the MD5 and SHA1 algorithms for integrity and DES, 3DES to support confidentiality services. Note however that MD5 is not considered secure anymore and is reported here for the shake of completeness. In total, we deployed 6 scenarios as shown in Table 6.

First and foremost, all network metrics for IPsec are remarkably concentrated. Standard deviation values rendered in Table 7 confirm this remark. Surprisingly, IPsec gives better transfer times for all file sizes when compared to Bluetooth and SSH. This is also confirmed by %ATR improvement for IPsec shown in Table 8. In particular, all IPsec times are very close to those of Bluetooth's mode I, while at the same time are considerably better than SSH's. Note, that IPsec renders 210,5 secs as the highest time duration for transferring the biggest file, while correspondingly SSH gives 222.1 secs, mode III produces 213.2 secs and mode I 211.6 secs. This is partially due to substantially increased (and highly stabilized) bandwidth that IPsec generates. The aforementioned observations are also confirmed by the fact that during IPsec measurements we had a very low rate of packet loss reported by the Ethereal utility.

It is important to note that the throughput was better when using ESP. On the contrary, when using AH, the throughput for transferring the files was lower. This can be explained by the fact that authentication is applied in AH.

	5.26 MB		
	TT (sec)	ATR (Kbps)	THR (%)
AH_MD5	72.8	683.4	94.5
AH_SHA1	72.8	683.2	94.5
ESP_DES_MD5	74.4	681.0	95.0
ESP_3DES_MD5	73.8	681.0	95.7

ESP_DES_SHA1	74.2	680.0	95.2
ESP_3DES_SHA1	74.2	681.0	95.2
	7 MB		
AH_MD5	100.0	682.8	94.4
AH_SHA1	99.9	683.0	94.5
ESP_DES_MD5	102.0	686.6	95.0
ESP_3DES_MD5	102.2	685.2	94.8
ESP_DES_SHA1	102.0	686.6	95.0
ESP_3DES_SHA1	101.8	688.2	95.2
	10.5 MB		
AH_MD5	145.9	682.6	94.4
AH_SHA1	145.7	683.4	94.5
ESP_DES_MD5	148.6	688.2	95.2
ESP_3DES_MD5	148.6	687.8	95.1
ESP_DES_SHA1	148.5	688.4	95.2
ESP_3DES_SHA1	148.6	688.0	95.2
	15 MB		
AH_MD5	205.2	683.4	94.5
AH_SHA1	205.1	683.8	94.6
ESP_DES_MD5	208.9	688.8	95.3
ESP_3DES_MD5	209.1	688.0	95.2
ESP_DES_SHA1	209.2	688.0	95.2
ESP_3DES_SHA1	210.5	683.6	94.6

Table 6. Average values for network parameters measured (IPsec)

	5.26 MB		
	TT	ATR	THR
AH_MD5	0.0	0.5	0.05
AH_SHA1	0.1	0.4	0.1
ESP_DES_MD5	0.1	0.4	0.1
ESP_3DES_MD5	0.5	4.5	0.6
ESP_DES_SHA1	0.0	0.4	0.06
ESP_3DES_SHA1	0.0	0.4	0.06
	7 MB		
AH_MD5	0.1	0.8	0.12
AH_SHA1	0.1	0.0	0.05
ESP_DES_MD5	0.3	2.1	0.28
ESP_3DES_MD5	1.3	8.6	1.19
ESP_DES_SHA1	0.6	3.7	0.53
ESP_3DES_SHA1	0.1	0.4	0.1
	10.5 MB		
AH_MD5	0.1	0.5	0.06
AH_SHA1	0.2	0.9	0.1
ESP_DES_MD5	0.1	0.8	0.09
ESP_3DES_MD5	0.1	0.8	0.08
ESP_DES_SHA1	0.0	0.5	0.02
ESP_3DES_SHA1	0.1	0.7	0.06
	15 MB		
AH_MD5	0.2	0.5	0.08
AH_SHA1	0.1	0.4	0.03
ESP_DES_MD5	0.1	0.4	0.04
ESP_3DES_MD5	0.1	0.0	0.03
ESP_DES_SHA1	0.3	1.0	0.13
ESP_3DES_SHA1	2.4	7.6	1.05

Table 7. Standard deviation of measurements of all IPsec scenarios

		AH_		ESP_DES_		ESP_3DES_	
File Size	**Bluetooth Mode_I**	**MD5**	**SHA1**	**MD5**	**SHA1**	**MD5**	**SHA1**
5.26	618.0	10.6	10.6	11.1	11.4	11.9	11.4
7	620.2	10.1	10.1	10.7	10.7	11.5	11.0
10.5	621.2	9.9	10.0	10.8	10.8	10.7	10.8
15	621.4	10.0	10.0	10.8	10.7	10.7	10.0

Table 8. % ATR improvement for IPsec

D. Comments on the Results

This section provides a comparative view of the conducted results. Also, we attempt to provide a better explanation of the experiment results. But before that we must shortly discuss important characteristics of Bluetooth connections that may affect the performance of the connection. Bluetooth employs FHSS (Frequency Hopping Spread Spectrum) to avoid interference. There are 79 - 23 in some countries - hopping frequencies, each having a bandwidth of 1MHz. Frequency hopping is assisted with fast ARQ (Automatic Repeat Request), CRC (Cyclic Redundancy Check), and FEC (Forward Error Correction) to achieve high reliability on the wireless links. All the data/control packet transmissions are synchronized by the master. Slave units can only send in the slave-to-master slot after being addressed in the preceding master-to-slave slot, with each slot lasting 625 microseconds.

For real-time data such as video, Synchronous Connection Oriented (SCO) links are used, while for data transmission, Asynchronous Connectionless Link (ACL) links are employed. There are several ACL packet types, differing in packet length and whether they are FEC coded or not. The FEC coding scheme used in ACL DM mode is a shortened Hamming code, where each block of 10 information bits is encoded into a 15 bit codeword, and is capable of correcting single bit error in each block. Table 9 shows the different ACL packet types and their properties. The values in the table are theoretical without packet overhead. For example, over an ACL link using DH5, one can send about 300 to 320 kbit/s of UDP user data, while the theoretical limit is 433.0 kbit/s.

This means that in order to overcome the effect of low and varying link quality on throughput, the selection of the optimal link layer packet size, under estimated channel conditions is crucial. Indeed some research work [26] point this out by evaluating the 'optimal' link layer packet size based on the current bit error rate of the channel. Moreover, in regions that Wi-Fi networks coexist with Bluetooth and because Wi-Fi and Bluetooth utilise spectrum in different ways, they can cause considerable interference between each other (depending on the relative location of the 802.11b and Bluetooth devices) [27]. By transmitting at the highest power level, Bluetooth class 1 devices would create more interference than Bluetooth's class 2 and class 3 devices, which transmit at lower power levels. Furthermore, because each Bluetooth PAN will occupy the entire ISM band, two or more coexisting Bluetooth PANs will occasionally collide, possibly causing loss of data packets. Of course, apart from implementation issues (e.g. protocol stacks), the aforementioned parameters are closely related and can affect real Bluetooth connections and the results gathered in this paper. For instance, all experiments were conducted inside the coverage area of the University's hot-spot.

In the following we present comparative graphs only for two of the three network parameters, transfer times and throughput. Consequently, Figure 3 illustrates a comparison of the transfer times for 6 selected scenarios in total. We easily spot that all times, especially for file sizes smaller than 10.5 MB, seem to be highly concentrated. This means that (excluding SSH ones) we have marginal differences between the performances' of the conducted scenarios. But, the bigger the size gets, the difference tends to slightly decrease. Apart from the fact that all tests have the Bluetooth link parameter in common, this can be explained by the fact that Bluetooth modes and IPsec utilize the network better.

Mode	FEC	Packet (bytes)	Size (kbps)	Symmetric (kbps)	Asymmetric (kbps)
DM1	2/3	0-17	108.8	108.8	108.8
DM3	2/3	0-121	258.1	387.2	54.4
DM5	2/3	0-227	286.7	477.8	36.3
DH1	no	0-27	172.8	172.8	172.8
DH3	no	0-183	390.4	585.6	86.4
DH5	no	0-339	433.9	723.2	57.6

Table 9. Packet types for Bluetooth ACL Connections

On the downside, SSH does not always provide peak network performance because it traditionally has been more focused on providing security. In a nutshell, SSHv2 introduced an additional form of flow control that requires the receiver to ACK each packet before more packets can be sent. Most implementations seem to use packet sizes of 16K or occasionally 32K, with some going as low as 4K. This means that no matter how fast the link, every e.g. 16K the transmission stops for 1 round trip time awaiting the other side to sent its ACK (referred to as a window adjust in SSHv2 terminology). In addition to the protocol-level handbrake, the SSH File Transfer Protocol (SFTP) protocol that runs on top of SSH contains its own handbrake. This protocol recommends that reading and writing is limited to less than 32K of data, even though it is running over the reliable SSH transport which is in turn runs over the reliable TCP/IP transport. One common implementation limits SFTP packets to 4K bytes, resulting in a mere 4% link utilization in the previously-presented scenario.

Finally, Figure 4 depicts a comparison of the achieved throughput for the specific 6 scenarios. This plot gives a clearer idea about the achieved network performance. In short, IPsec scenarios visibly have the best performance by far followed closely by the two Bluetooth's security modes. Moreover, we can make a very important observation about the SSH's performance. It is obvious that SSH's throughput increases as the file's size increases. This happens because of the handshaking phase which takes place during the initialization of each transaction. So, as the size of the transferred file increases, the impact of handshaking decreases and thus we notice an increase in the throughput. We should also report that the throughput of the other two scenarios remains more or less stable for all the file sizes we utilized. Another important issue is that during the experiments we observed a significant rate of packet loss for both Bluetooth Security Modes and SSH scenarios affecting their overall performance. Certainly, the main reason for this is the volatile nature of the wireless connection itself.

Additionally, it is well known that the addition of an IPSec header may cause IP fragmentation. However, the main concern in IPsec overhead is in the encryption, decryption and authentication of the actual IPsec (ESP and/or AH) packets. Tunnel setup and rekeying occur much less frequently than packet processing and, except in highly unusual circumstances, their overheads are not worth worrying about. According to some other works [28], utilizing low-end machines, a 60 MHz Pentium running a host-to-host tunnel to another machine shows an FTP throughput of slightly over 5Mbit/s either way. Thereafter, we can conclude that in our case the IPsec mechanisms running on "relatively" low-end processors is not really a bottleneck. The overall performance is rather affected most by the quality of the Bluetooth link itself meaning that due to better utilization of the link, possibly due to optimal ACL scheme and lower packet drop rate, IPsec performs slightly better than native Bluetooth modes do.

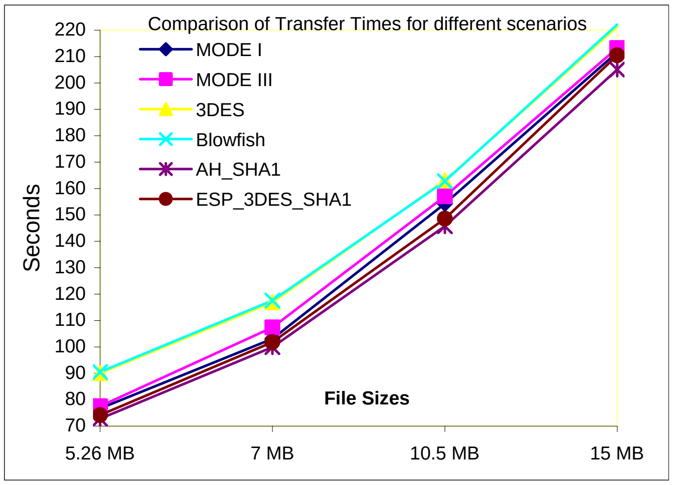

Figure 3. Comparison of network transfer times for 6 different scenarios

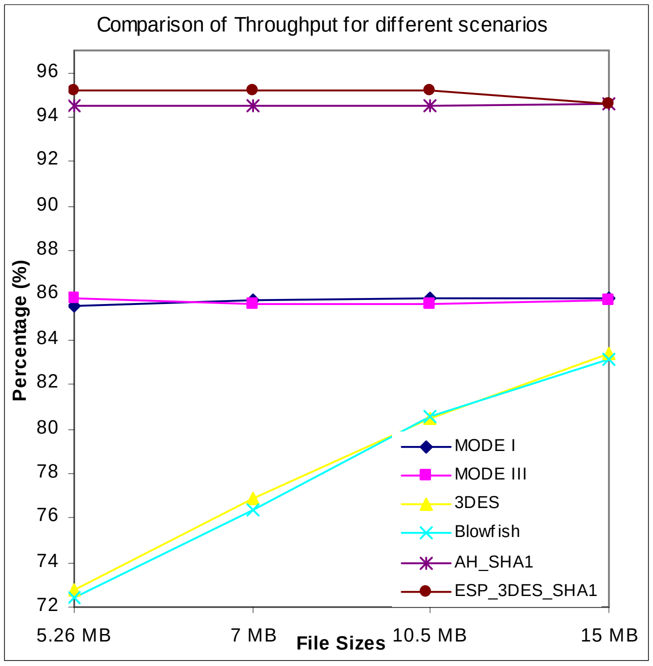

Figure 4. Comparison of network throughput for 6 different scenarios

IV. Conclusions and Future Work

This paper addresses performance issues for Bluetooth host-to-host connections. Three distinct categories of scenarios were used to test whether well respected security mechanisms of Internet and application layers of the TCP/IP suite are advantageous when deployed over Bluetooth PANs compared to Bluetooth native security modes. The results disclose that IPsec utilizes better the wireless link and thus provides radically improved transfer times when compared with SSH. Native Bluetooth modes service times are close to those of IPsec's thus significantly better from SSH ones. On the other hand, there is an important disadvantage which is the high amount of the memory resources IPsec consumes.

As future work we would like to expand this study investigating the performance of asymmetric cryptography mechanisms e.g. public key certificates and to support authentication services in the context of such protocols that promote automatic keying. Another direction is to detect how much energy is required for this sort of secure connections, as mobile devices can not afford batteries with unlimited capacity.

V. References

[1] Hewlett Packard Smart Handheld Group, Bluetooth technology overview.

[2] The official Bluetooth wireless info site, http://www.bluetooth.com

[3] Christian Gehramann, Joakim Persson and Ben Smeets, Bluetooth Security

[4] Nikhil Anand, An overview of Bluetooth Security.

[5] The official Bluetooth membership site, https://www.bluetooth.org.

[6] Francia, G., Kilaru, A., Le Phuong and Vashi, M. "An empirical study of Bluetooth performance", In proceedings of the 2nd annual conference on Mid-south college computing, ACM International Conference Proceeding Series; Vol. 61, pp. 81 – 93, 2004.

[7] Howitt, I. "Bluetooth performance in the presence of 802.11b WLAN", IEEE Transactions on Vehicular Technology, Volume: 51 Issue: 6, pp. 1640-1651, 2002.

[8] Wang Feng, Arumugam, N. and Krishna, G.H., "Performance of a Bluetooth piconet in the presence of IEEE 802.11 WLANs", in proc of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pp. 1742 - 1746 vol. 4, 2002.

[9] Yujin Lim, Jesung Kim, Sang Lyul Min and Joong Soo Ma, "Performance evaluation of the Bluetooth-based public Internet access point", in proceedings of the 15th International Conference on Information Networking, pp. 643 – 648, 2001.

[10] Karnik, A. Kumar, A., "Performance analysis of the Bluetooth physical layer", in proc. of IEEE International Conference on Personal Wireless Communications, pp. 70 – 74, 2000.

[11] Guillermo A. Francia III, Aditya Kilaru, Le Phuong and Mehul Vashi, An empirical study of Bluetooth performance

[12] N. Golmie, O. Rebala, Techniques to improve the performance of TCP in a mixed Bluetooth and WLAN Environement.

[13] Martin Connoly and Cormac J. Sreenan, Analysis of UDP performance over Bluetooth.

[14] S. Zeadally, A. Banda, A.Kumar, Improving Bluetooth performance in 802.11 interfere environments.

[16] Daniele Miorandi, Carlo Caimi, Andrea Zanella, Performance Characterization of a Bluetooth Piconet with Multi-Slot Packets.

[17] Carlos de Morais Cordeiro, Djamel Sadok, Dharma P. Agrawal, Modeling and evaluation of Bluetooth MAC protocol.

[18] Ethereal, http://www.ethereal.com

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

[20] Massey, J., Khachatrian, G. & Kuregian, M., "Nomination of SAFER+ as Candidate Algorithm for the Advanced Encryption Standard (AES)", In Proc. 1st Advanced Encryption Standard CandidateConf., www.ee.princeton.edu/~rblee/safer+, 1998.

[21] IETF Secure Shell (secsh) Working Group, http://www.ietf.org/html.charters/secsh- charter.html

[22] OpenSSH, http://www.openssh.org

[23] Kent, S. & Atkinson, R., "Security Architecture for the Internet Protocol", IETF RFC 2401, 1998.

[24] Kent, S. & Atkinson, R., "IP Authentication Header (AH)", IETF RFC 2402, 1998.

[25] Kent, S. & Atkinson, R., "IP Encapsulating Security Payload (ESP)", IETF RFC 2406, 1998.

[26] Ling-Jyh Chen, Rohit Kapoor, M. Y. Sanadidi, Mario Gerla, "Enhancing Bluetooth TCP Throughput via Link Layer Packet Adaptation", IEEE ICC '04, Vol.7, pp. 4012- 4016.

[27] Hoi Kit Yip and Yu-Kwong Kwok, "A Performance Study of Packet Scheduling Algorithms for Coordinating Colocated Bluetooth and IEEE 802.11b in a Linux Machine", Proceedings of the 7th International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'04).

[28] FreeSwan, Performance of FreeSwan, http://www. freeswan. org/freeswan_trees/freeswan-1.95/doc/performance.html.