Performance Analysis of Cryptographic Protocols on Handheld Devices

Patroklos G. Argyroudis, Raja Verma, Hitesh Tewari and Donal O'Mahony
Networks and Telecommunications Research Group
Department of Computer Science
University of Dublin, Trinity College
{argp, vermar, htewari, omahony}@cs.tcd.ie

Abstract

The past few years have witnessed an explosive growth in the use of wireless mobile handheld devices as the enabling technology for accessing Internet-based services, as well as for personal communication needs in ad hoc networking environments. Most studies indicate that it is impossible to utilize strong cryptographic functions for implementing security protocols on handheld devices. Our work refutes this. Specifically, we present a performance analysis focused on three of the most commonly used security protocols for networking applications, namely SSL, S/MIME and IPsec. Our results show that the time taken to perform cryptographic functions is small enough not to significantly impact real-time mobile transactions and that there is no obstacle to the use of quite sophisticated cryptographic protocols on handheld mobile devices.

Keywords: Mobile communications, wireless security, handheld devices, cryptographic protocols, performance analysis.

1 Introduction

The use of mobile computing devices (e.g. handhelds, palmtops, mobile phones) has increased over the years, particularly during the last decade. Personal Digital Assistants (PDAs) started initially as devices to store personal information. As they have grown more compact with more powerful CPUs, they have evolved to support more advanced communications applications that have traditionally been the domain of workstations. At the same time there have been significant changes in the way business is done with the introduction of electronic commerce endeavors through the Internet. Electronic commerce involves the use of strong cryptographic functions and protocols in order to provide adequate security services for payment transactions. These functions can be easily afforded by fixed workstations, but the literature [1, 2] would suggest that on mobile devices are slow and expensive due to constrained processors, limited memory and battery life. The latest generations of mobile devices are equipped with much faster CPUs, which facilitate the use of strong cryptographic functions for the construction of security-related protocols. In this paper we present a thorough performance assessment of the three most commonly used security protocols for Internet transactions on wireless mobile devices. Specifically, we benchmark the Secure Sockets Layer (SSL) [3] as the standard security protocol for protecting a wide range of interactive network applications such as Web commerce, S/MIME [4] as the industry standard for providing message-oriented security services and IP-level security (IPsec) [5] as the primary technology for creating virtual private networks and offering protection at the network-layer. The operational scenarios we examine

describe the most common applications in mobile communications and wireless ad hoc environments.

In the remainder of the article we start by presenting the parameters that are common for all the tests we have performed. Next we briefly present each of the investigated security protocols followed by the specific parameters of the utilized scenarios and the observed performance results. In turn we analyze the SSL, S/MIME, IPsec protocols and we also present the timing measurements of the low-level cryptographic primitives such as symmetric and asymmetric operations, as well as message digests. We conclude with a discussion on the possibilities that are opened with the use of strong cryptography on wireless mobile devices and describe potential directions for future work.

2 Methodology

We begin by describing in detail the parameters of the experiments we have performed. The hardware platform we use is the HP (Compaq) iPAQ H3630 [6] with a 206 MHz StrongARM processor and 32MB RAM (16MB ROM), running the Windows CE Pocket PC 2002 [7] operating system. For the implementation of the investigated protocols we have employed the Windows CE port of the OpenSSL [8] cryptographic toolkit, version 0.9.7b. We have also performed the same experiments by utilizing the Microsoft Cryptography API [9] and the results of the timing measurements were approximately the same. When the investigated scenarios required a communication link between two peers, as in the case of SSL transactions, we have used IEEE 802.11b wireless LAN [10] cards for the handheld devices.

All the experiments were performed with RSA keys of 1,024 and 2,048 bits size, with small public exponents (*e* was given the value 65,537) making the public key operations significantly faster than the private key operations. We feel that 512 bits keys are too short for sensitive data and therefore cannot be used in experiments that try to capture the realistic requirements of secure transactions. Moreover, we have created a certification authority (CA) that directly issued certificates for the public keys of the peers involved in the tests making the certificate chains one certificate long, thus requiring a single verification operation.

3 Secure Sockets Layer (SSL)

The Secure Sockets Layer (SSL), the latest version of which is also known as Transport Layer Security (TLS), is by far the most widely deployed security protocol in the world [11]. Almost all Web traffic related to electronic commerce is being actively protected by it. Although the SSL protocol has been thoroughly analyzed on the wired Internet and found to be especially satisfactory, its use on mobile handheld devices has not been equally extensive mainly due to performance limitations. Therefore, SSL-based security solutions need to be examined more thoroughly in the context of handheld devices.

In order to investigate the overhead of SSL in both the handshake procedure and in bulk data transfer we employed a scenario of a simple file transmission of 1 MB (1,048,576 bytes) between two handheld devices. As we are also interested in an ad hoc communication environment where the participating entities function as peers, we have enabled both client-side and server-side authentication. Although SSL session resumption was not used, we have not measured the time required for the SSL context initialization at each iteration. The utilized SSL context was initialized with RSA for

authentication, Diffie-Hellman for key exchange, SHA-1 for message digesting and Rijndael (with a 256 bits key) for bulk encryption. Furthermore, it should be noted that we have used full SSL handshakes with no abbreviations and no certificate caching. The average time required for a full handshake with both peers having keys of 1,024 bits is 1.14 seconds (1,145 milliseconds) and 2.06 seconds (2,062.97 milliseconds) with keys of size 2,048 bits (see Figure 1). The whole transaction including the SSL handshake and the encrypted file transfer required 7.76 seconds (7,759.14 milliseconds) in the first case and 8.73 seconds (8,732.33 milliseconds) in the second one. In order to have a clear understanding of the overhead introduced by SSL in this scenario we run the same file transfer without any transport-layer protection. The average time taken was 4.25 seconds (4,256.78 milliseconds). Although the observed overhead is significant, it does not prohibit the use of SSL on handheld devices since the required 2 seconds in the case of 2,048 bits key pairs realistically allows even casual Web browsing.

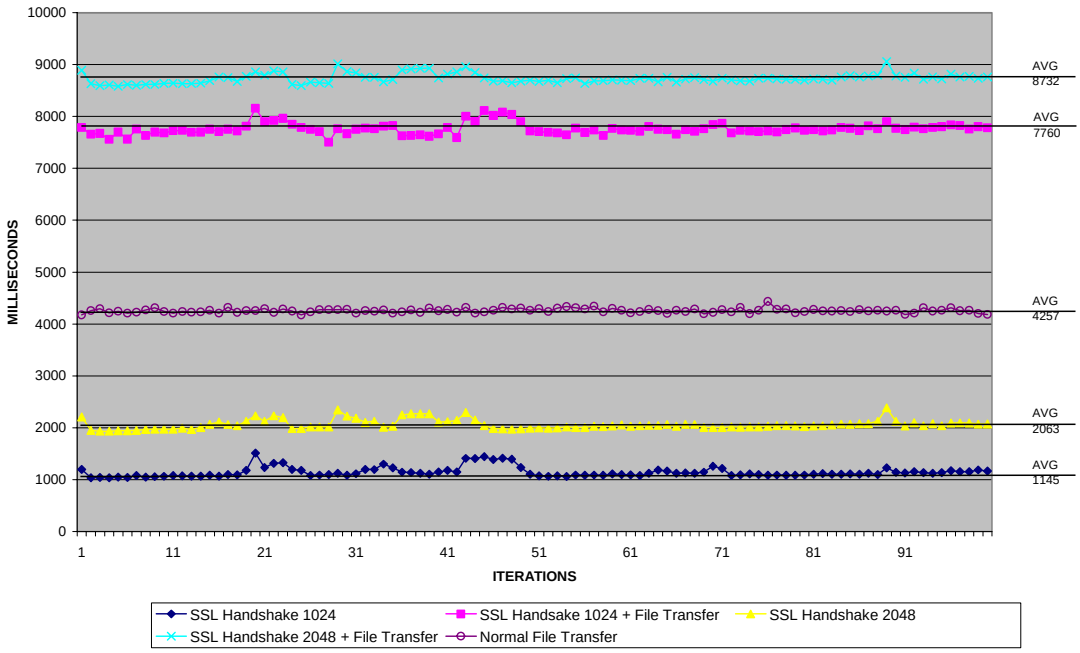

Figure 1. File transfer timing measurements with and without SSL protection.

In our experiments we have also investigated the overhead of SSL in respect to battery power. Handheld devices are totally depended on the available battery energy and therefore expensive operations should be identified. In order to analyze energy consumption we have employed a file transfer of 20,480 bytes, which was run continuously between two handheld devices until the batteries were completely exhausted, with and without SSL protection. Both handheld devices shutdown after the same length of time (approximately 2 hours and 45 minutes), but the SSL version protected with 1,024 bits key pairs achieved 4,906 transfers while the non-encrypted version achieved almost 80,000. The SSL operations are as expected more time consuming and require a greater amount of CPU time to execute. However, the investigated scenario was trying to capture the demands of cryptographically expensive applications, like multiparty conferencing. In most common less CPU demanding applications the impact on execution time is naturally lower. We must note at this point that the energy experiments we conducted are only indicative since

several factors that have an impact on battery life, such as ambient temperature and humidity, were not taken into account.

The overhead that is introduced by using SSL as the method of providing transport-layer security for network transactions on handheld devices is considerable. However, at just over 1 second for a handshake with 1,024 bits key pairs, it will not inhibit mobile transactions.

4 Secure/Multipurpose Internet Mail Extensions (S/MIME)

Secure/Multipurpose Internet Mail Extensions (S/MIME) is the industry standard for providing message-oriented security services for Internet electronic mail. The design approach followed by S/MIME is to treat a message as a single object and to provide the required security services for that object using symmetric and asymmetric cryptography. Therefore, S/MIME can be utilized as the security solution for any communication protocol that uses the store-and-forward delivery architecture of electronic mail. One could argue that transport-layer security protocols, like SSL, can also be employed for this purpose, however they do so by violating the end-to-end security principle and do not offer non-repudiation, among other shortcomings [11].

S/MIME provides the capability of securing normal electronic mail messages of arbitrary content formatted according to the MIME standard. For our tests we used a normal electronic mail formatted according to MIME 1.0, with content type plain text. The size of this message was 2,092 bytes, a typical size of a normal message exchanged during everyday transactions. We have investigated two different application scenarios. In the first one the sender signs the message according to the S/MIME standard and sends it to the receiver who verifies the signature, providing only authentication. The average time required for a sender to sign a message is 110 milliseconds with a 1,024 bits key, and 545 milliseconds with a 2,048 bits key, roughly five times greater. The verification operation performed by the receiver takes an average time of 42 milliseconds using a 1,024 bits key, and 176 milliseconds when a 2,048 bits key is used (see Figure 2).

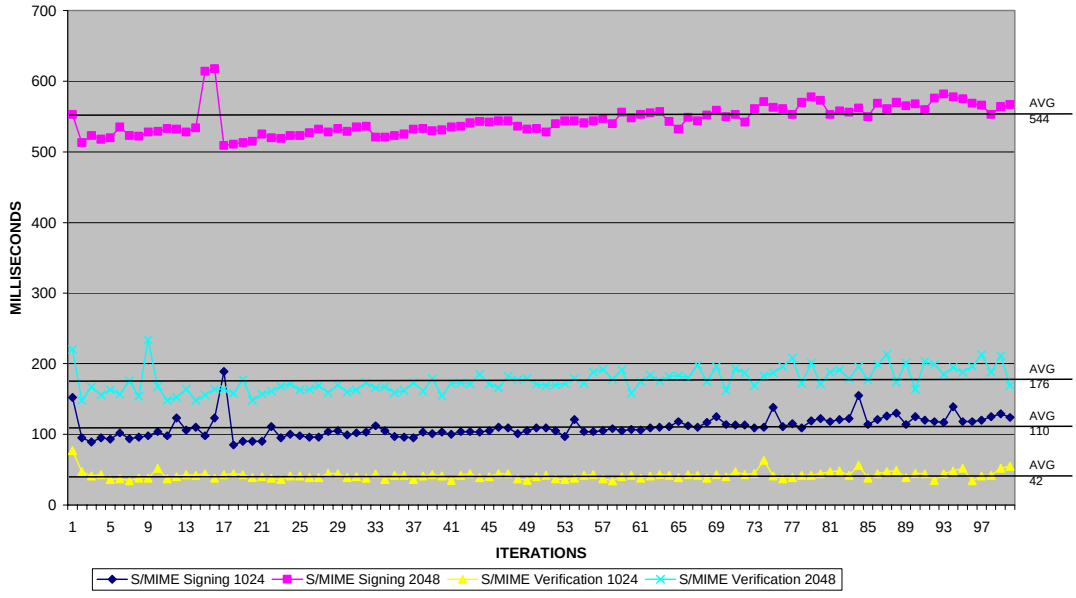

Figure 2. S/MIME signing and verification timing measurements.

The second scenario provides both confidentiality and authentication by signing and encrypting the message. The sender initially signs the message and then randomly generates a key known as the Content Encryption Key (CEK) that uses it to encrypt the message using Triple-DES. The final step is to encrypt the CEK with the recipient's public key. The recipient decrypts the CEK using her private key and then the message using the CEK. Finally, the sender's signature is verified to provide authentication. This second scenario captures in greater detail the requirements of a real-world store-and-forward system. According to the observed results illustrated in Figure 3 the average time required for a sender to construct such a message is 0.7 seconds (711.17 milliseconds) with a 1,024 bits key, and 1.3 seconds (1,267.84 milliseconds) with a 2,048 bits key. The operations performed by the recipient add an overhead of 0.15 seconds (150.62 milliseconds) in the first case, and 0.64 seconds (645.11 milliseconds) in the second.

We believe that the timing results of both key sizes are realistic for a message-oriented system providing both confidentiality and authentication. Specifically, the observed overhead of approximately 1 second that is introduced at the sender side and half a second at the receiver side when both confidentiality and authentication with 2,048 bits keys pairs is required is not prohibitive for even real-time store-and-forward systems employed on handheld devices.

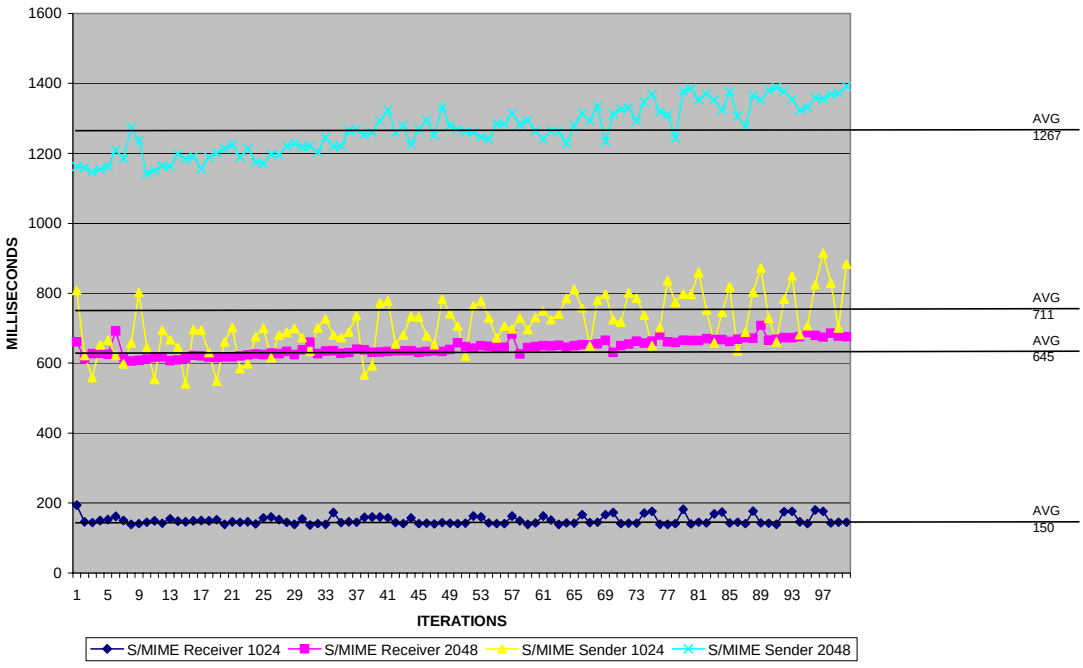

Figure 3. S/MIME sender and receiver timing measurements.

5 IP-level Security (IPsec)

IPsec consists of a set of protocols that provide security services for any application that uses the Internet Protocol (IP). These protocols guarantee the secure transmission of data between two systems anywhere in a networked environment. The goal of

IPsec is to provide integrity, confidentiality and authenticity. Moreover, it should be as resistant as possible to traffic analysis, replay and man-in-the-middle attacks. The IPsec protocol suite is consisting of three different protocols [5]. First of all, the Encapsulating Security Payload (ESP) which is added to an IP datagram and provides confidentiality, integrity, and authenticity of the transferred data. The Authentication Header (AH) is also added to an IP datagram and provides integrity and authenticity of the transmitted packets. AH does not provide confidentiality for the data of network packets since this is the service explicitly provided by ESP. The third protocol is the Internet Key Exchange (IKE), which is based on the Diffie-Hellman exchange and is used to negotiate the security association between the two endpoints that need to communicate. A *security association* (SA) consists of the cryptographic keys and the negotiated algorithms supported by the peers needed to exchange data securely. IPsec has been criticized for being exceptionally complex and this fact hinders in depth security evaluations [12]. In order to analyze the performance of IPsec on handheld devices we chose not to implement it, as this would be error prone and the compatibility with the specification questionable. Instead we have calculated the time requirements of an ordinary IPsec negotiation between two peers that do not share a pre-established security association, based on the observed performance of the low-level cryptographic operations that take place (see Table 1).

Operation	Time	Iterations
DES	7.354 seconds (7,354 ms)	100,000 encryptions and 100,000 decryptions
SHA	19.111 seconds (19,111 ms)	100,000
1,024 bits RSA signing	782.593 seconds (782,593 ms)	10,000
1,024 bits RSA verification	50.125 seconds (50,125 ms)	10,000
2,048 bits RSA signing	4,972.798 seconds (4,972,798 ms)	10,000
2,048 bits RSA verification	156.006 seconds (156,006 ms)	10,000

Table 1. Timing measurements of low-level cryptographic primitives on an iPAQ H3630.

When a host wishes to communicate with another host with whom it does not share a security association it has to negotiate one using IKE. The entire procedure has two phases. The purpose of the first phase is to construct a secure and authenticated channel to exchange further IKE traffic and this can be accomplished in two different modes, the *main mode* and the *aggressive mode*. We chose to base our calculations on the main mode with signature authentication since it provides identity protection by utilizing an anonymous Diffie-Hellman exchange and therefore it is applicable to ad hoc environments. Each peer needs to perform one RSA signing and one RSA verification operation, as well as one SHA message hashing operation. Therefore a successful completion of the first phase requires approximately 167 milliseconds with 1,024 bits RSA key pairs and 1 second (1,026 milliseconds) with 2,048 bits key pairs. The second phase handles the establishment of security associations, between two hosts that have completed the first phase, for a specific type of traffic. This is accomplished with the *quick mode*, in which all the payloads of the exchanged messages are encrypted with the keying material specified by the previously negotiated security association. The successful completion of the quick mode requires three DES symmetric encryption operations as well as three SHA hashing operations. Hence the calculated time needed for the completion of the quick mode procedure is approximately 0.68 milliseconds. We have to stress at this point that we have not taken into account the time needed for possible certificate parsing,

key derivation, network latency and encoding of the signatures in the PKCS #1 format. Leaving out these times, we can see that an IPsec handshake should take approximately 0.16 seconds for a 1,024 bits key and just over a second for a 2,048 bits key.

The calculations we have performed regarding the overhead of IPsec in key exchanges illustrate the feasibility of using it on mobile constrained devices. Even when the two peers that wish form a secure channel do not share a pre-existing key the time required for negotiating one is negligible and therefore has a minimal impact on the applications employed at a higher layer.

6 Related Work

Our work complements previous attempts to implement and use cryptographic protocols on mobile handheld devices. Although a comprehensive comparison between our work and previous similar attempts cannot be accomplished due to different hardware and software parameters, there are some useful comments that can be noted regarding advances in handheld computing technology. In [1] the authors examine the performance of Kilobyte SSL (KSSL), a small footprint SSL client for the Java 2 Micro-Edition platform, on a 20 MHz Palm CPU with RSA keys of sizes 768 and 1,024 bits. Their results indicate that a full SSL handshake between a handheld client and a desktop server with only server-side authentication requires 10-13 seconds, which can be reduced to 7-8 seconds with certificate caching. RSA operations on the same platform require 0.5-1.5 seconds. RSA operations were also investigated in the context of electronic commerce through the use of handheld devices [2]. The platform in this case was a PalmPilot Professional with a Motorola DragonBall chip at 16 MHz, running the PalmPilot port of the SSLeay cryptographic library. The observed results for RSA operations with 512 bits key pairs were 3.4 minutes for key generation, 7 seconds (7,028 milliseconds) for signing and 1.4 seconds (1,376 milliseconds) for verification.

7 Discussion and Conclusion

This paper demonstrates the feasibility of using strong cryptographic protocols on mobile handheld devices. We have presented a thorough performance analysis of the three most common security protocols used for a wide variety of applications in the wired Internet. Our investigation covered the SSL protocol that provides transport-layer security by protecting all traffic that utilizes TCP, S/MIME that can be used to secure systems that follow the store-and-forward architecture of the Internet electronic mail and IPsec as a generic solution for protecting IP-based network traffic. In the case of SSL we observed that a full handshake with mutual authentication and 2,048 bits key pairs requires approximately 2 seconds. Although this is a significant overhead, it is small enough to allow even frequent short-lived secure HTTP transactions. We must note at this point that the exact overhead of Web browsing largely depends on whether persistent HTTP connections are used (by using *Connection: Keep-Alive* headers) or a new connection is opened per downloadable component. However, most modern Web browsers and servers support this functionality by allowing the reuse of secure socket objects. Message-oriented applications protected by the S/MIME protocol are also feasible on handheld devices since the maximum observed measurement in the investigated scenarios was around 1 second. The comparison between our work and previous related work revealed

interesting results regarding the advances of constrained devices. We have observed that full SSL handshakes with mutual authentication have become faster by approximately 10 seconds and the order of time required for RSA operations has been reduced from seconds to milliseconds. However, such a comparison is only useful to demonstrate advances in handheld computing technology since the hardware platform we have used is more fitting to perform CPU intensive cryptographic operations. Our plans for future work on the subject involve the investigation of other handheld devices, like the Microsoft Smartphone, as well as other operating systems. Furthermore, we plan to analyze the overall performance of different IPsec implementations and determine the exact introduced overhead. We believe that currently available handheld devices can form the foundation of secure ubiquitous computing environments since they can facilitate the use of strong cryptographic functions.

Acknowledgements

We would like to thank Steven Reddie for the Windows CE port of the OpenSSL cryptographic toolkit and the anonymous reviewers for their helpful comments.

References

[1] V. Gupta and S. Gupta, "Securing the Wireless Internet", *IEEE Communications*, vol. 39, no. 12, December 2001, pp. 68-74.

[2] N. Daswani and D. Boneh, "Experimenting with Electronic Commerce on the PalmPilot", *Proc. Eurocrypt'99*, LNCS 1648, Springer Verlag, February 1999, pp. 1-16.

[3] T. Dierks and C. Allen, "The TLS Protocol Version 1.0", IETF RFC 2246, January 1999.

[4] B. Ramsdell, "S/MIME Version 3 Message Specification", IETF RFC 2633, June 1999.

[5] S. Kent and R. Atkinson, "Security Architecture for the Internet Protocol", IETF RFC 2401, November 1998.

[6] iPAQ, see http://www.compaq.com/support/handhelds/iPAQ_H3600.html.

[7] Windows CE, see http://www.microsoft.com/windowsce/.

[8] The OpenSSL Project, see http://www.openssl.org/.

[9] D. Esposito, "Supporting CryptoAPI in Real-World Applications", *Microsoft Interactive Developer*, http://www.microsoft.com/mind/0697/crypto.asp, June 1997.

[10] IEEE Computer Society LAN/MAN Standards Committee, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Std. 802.11-1997. IEEE, New York, NY 1997.

[11] E. Rescorla, *SSL and TLS – Designing and Building Secure Systems*, Addison-Wesley, 2000.

[12] N. Ferguson and B. Schneier, "A Cryptographic Evaluation of IPsec", http://www.counterpane.com/ipsec.html, February 1999.