Polytechnic
UNIVERSITY

Optimizing the Energy Consumed by Secure Wireless Sessions-Wireless Transport Layer Security Care Study

Ramesh Karri, Piyush Mishra

April 2002

WICAT TR 02-004

Optimizing the Energy Consumed by Secure Wireless Sessions –Wireless Transport Layer Security Case Study

Ramesh Karri Piyush Mishra

ramesh@india.poly.edu pmishr01@utopia.poly.edu
718-260-3596 718-260-4011
Department of Electrical and Computer Engineering
Polytechnic University, Brooklyn
6 Metrotech Center, NY, US 11201

Abstract: ***In this paper we identified the various sources of energy consumption during the setup, operation and tear down of a secure wireless session by considering the wireless transport layer security protocol. Our analysis showed that data transfers during a secure wireless transaction, number and size of messages exchanged during secure session establishment and cryptographic computations used for data authentication and privacy during secure data transactions in that order are the main sources of energy consumption during a secure wireless session. We developed techniques based on information compression, session negotiation protocol optimization and hardware acceleration of crypto-mechanisms to reduce the energy consumed by a secure session. A mobile test bed was developed to verify our energy management schemes and to study the energy consumption vs. security trade-offs. Using our proposed schemes we were able to reduce the session establishment energy by more than 6.5× and the secure data transaction energy by more than 1.5× during data transmission and by more than 2.5× during data reception.***

Keywords: Mobile, Security, Wireless, Energy-efficient, WTLS, Secure session

1. Introduction

According to the industry projections, mobile computing and communication device market is poised to overtake the desktop computing market [1]. The widespread adoption of Internet combined with the anytime-anywhere access of mobile devices is driving a huge growth in mobile e-commerce applications such as on-line shopping, stock trading, web-based banking and electronic bill payment. Such confidential transactions over wireless public networks demand end-to-end secure connections to ensure data authentication, privacy and integrity [2].

Energy consumed by such secure wireless sessions on mobile devices is very significant. There are two main sources of energy consumption during a secure wireless transaction: (i) cryptographic computations used to establish the secure session and to support encryption and authentication during data transaction and (ii) message exchanges during secure session establishment and data transfers during secure data transactions. We considered a Symbol PPT2800™ Pocket PC device running Windows CE™ 3.0 operating system and equipped with an 11 Mbps

Spectrum24™ wireless LAN adapter card as the mobile test bed[1] to measure the energy consumed by a secure wireless session while transmitting 64 KB data over a 200 Kbps wireless channel. Session negotiation protocol used Diffie-Hellman key exchange and management [27] and secure data transaction used 3DES encryption [30] and SHA-256 message authentication code [31].

Figure 1 shows that message exchanges consume more than 90% of the system energy during session negotiation while during data transaction idle system consumes 44% of the system energy, followed by data transmission that consumes 35% and cryptographic computations that consume 21% of the system energy respectively.

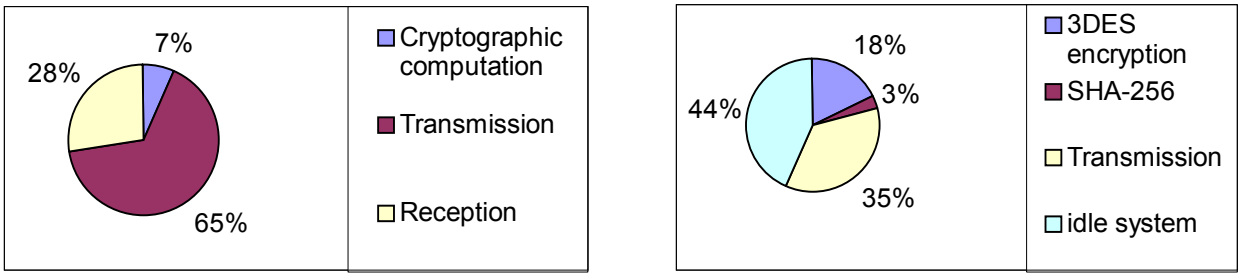

Figure 1: Energy consumed by a secure wireless session during (a) session negotiation and (b) secure data transaction while transmitting 64 KB data using WTLS protocol

In this paper we will accurately measure the energy consumed by various components of a security protocol used for establishing and managing a secure wireless session, present techniques to minimize their energy consumption and investigate the associated energy *vs.* security trade-offs. Related research has been done to reduce the energy consumed by wireless data transfers by optimizing network protocols [4, 5, 6, 7, 8, 9, 10, 11], developing adaptive network interface control [12, 13, 14], and developing adaptive error control techniques [15, 16, 17, 18]. Our study differs from these works in two aspects: (1) it will focus on the energy consumption characteristics of computation-intensive security protocols which, till now, have received little attention, and (2) unlike most of the other simulation based studies, we will use real-life mobile test bed to validate the techniques presented for reducing the energy consumed by a secure wireless session. The mobile test bed is described in detail in section 3.1. Further, these techniques were also verified using another mobile test bed, details of which can be found in [32].

[1] A detailed description of the mobile test bed and experimentation methodology is outlined in section 3.1

We will use Wireless Transport Layer Security (WTLS) [3] of the Wireless Application Protocol (WAP) suite as the example security protocol. As shown in Figure 2, WAP is a set of protocols in transport, security, transaction, session and application layer that enables the creation and execution of advanced mobile services.

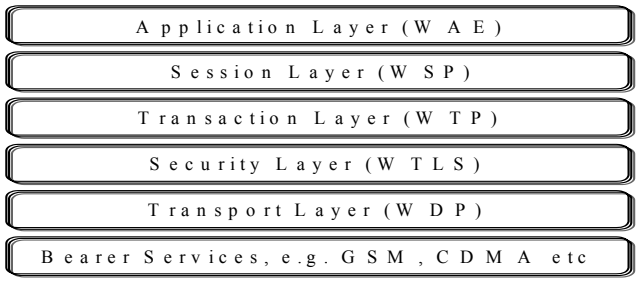

Figure 2: Wireless Application Protocol architecture

A secure wireless transaction between a web server and a mobile device is a two-step activity using WAP. Secure communication between the web server and WAP gateway over the wired channel uses Secure Session Layer (SSL) protocol while the wireless communication between the WAP gateway and the mobile device is secured using WTLS protocol. WTLS operates above the transport protocol layer and provides higher level layers with a secure transport service interface.

In section 2 we describe the setup, operation and tear down of a secure wireless session using WTLS protocol. In section 3 we describe the mobile test bed and experimentation methodology and summarize the energy consumed during the setup, operation and tear down of a secure wireless session. In section 4 we present techniques to reduce the energy consumed by a mobile device during a secure wireless session and the analysis of associated security *vs.* energy trade-offs. Finally, in section 5 we will summarize the results of this study.

2. Secure Wireless Session Management Using WTLS Protocol

WTLS uses a handshake protocol to establish a secure session between the client and the server before carrying out secure data transactions. Client initiates the handshake by sending a **client_hello** message to the server. This message contains session id, key refresh rate, private key encryption algorithms and their modes of operation, message authentication code (MAC) algorithms and a random number used to generate the encryption and MAC keys. The message also identifies the master_secret exchange protocol and its parameters. Master_secret is the secret shared exclusively between the communicating parties. Together with the exchanged random numbers it is used to generate the secret session keys for encryption and MAC. For example, if the Diffie-Hellman master_secret

exchange protocol is used [27], the client sends the generator (g) and the prime modulus (n) used to generate the master_secret[2].

The server responds with a **server_hello** message with the acceptable security association, another random number, a **server_certificate** for authenticating itself to the client, a **server_key_exchange** with its share of the master_secret (X, computed as $g^x \ mod \ n$ where x is a large random number), and a **certificate_request** from the client**.**

The client replies with a **client_certificate,** a **client_key_exchange** with its share of the master_secret (Y, computed as $g^y \ mod \ n$ where y is a large random number), and a **certificate_verify** message to explicitly verify its certificate.

Finally, the client and the server exchange **change_cipher_spec** messages to activate the session with the negotiated security association (encryption algorithm and its mode of operation, MAC algorithm, session id and key refresh rate) and **finished** messages to indicate successful session negotiation. Next, client and the server independently compute the master_secret $g^{xy} \ mod \ n$ as $Y^x \ mod \ n$ and $X^y \ mod \ n$ respectively and generate secret keys for encryption and MAC using shared master_secret, exchanged random numbers and current message sequence number. Figure 3 summarizes the handshake protocol used by WTLS for secure session negotiation.

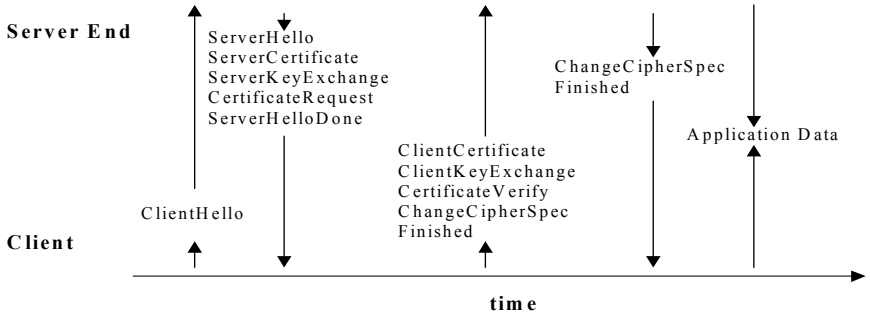

Figure 3: Messages exchanged during WTLS handshake protocol

After successfully establishing the secure session WTLS (either at the client or at the server) accepts plain text messages, computes the MAC, encrypts the data and transmits it. At the other end, received data is decrypted and verified. The security of a session is enhanced by periodically refreshing the encryption and MAC keys; for a key_refresh_rate of n, the client and the server generate new encryption and MAC secret keys after exchanging 2^n data blocks. Either side can terminate the session by sending a **closing** message.

[2] In this paper we use the Diffie-Hellman master_secret exchange protocol. RSA and EC protocols can be analyzed in a similar fashion.

3. Energy Consumption of a Secure Wireless Session

3.1 Test-bed for Energy Measurements

The mobile test bed shown in Figure 4 consists of a PPT2800™ Pocket PC device [29]equipped with an 11 Mbps Spectrum24™ wireless LAN Adapter card (IEEE 802.11b) [19], both from Symbol Technologies Inc., and a Wildcard™ FPGA board from Annapolis Micro Systems® [24]. The Pocket PC is running WindowsCE™ operating system on a 32-bit, 206 MHz StrongArm™ SA-1110 processor with 16 MB flash ROM, 16 MB RAM, 16 KB instruction cache and 32-way set associative 8 KB data write back data cache. The WLAN card is operating in polling mode P1 [19]. The Wildcard™ board supports a Xilinx® Virtex™ XCV300E field programmable gate array (FPGA) with 400,000 system gates and 130,000 block RAM bits and is used to accelerate the cryptographic computations.

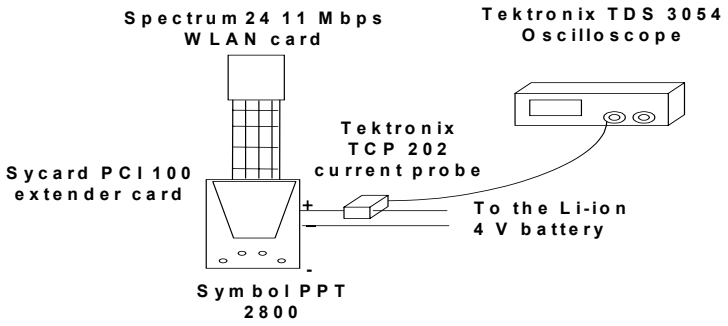

Figure 4: Mobile test bed for performance and energy measurements

We measure the current drawn by an application executing on the mobile test bed using a Tektronix TCP202 current probe (DC to 50 MHz, Min sensitivity: 10 mA/div, DC accuracy: ±1% with probe calibrator) and a Tektronix TDS 3054 oscilloscope (4 channel, 500 MHz, 5 GS/s) [28]. Voltage is held constant by repeatedly charging the battery. In order to ensure consistency and accuracy of our results, we averaged each of our results over several thousand iterations for each application with data residing in the main memory and data and instruction caches enabled. We assume that data transmission and reception rates are identical at 200 Kbps.

3.2 Sources of Energy Consumption

Energy consumed while establishing a secure session is the sum of the energy consumed during message exchanges, energy consumed to compute the master_secret, the encryption keys and the MAC keys, and the energy consumed to sign and verify the certificates. Energy consumed during a secure wireless data transaction is the sum of energy consumed by data authentication, data encryption, data transmission and the associated security

overheads, such as session and key refreshes. Rest of the section presents the results of our energy measurement experiments for these various sources of energy consumption.

3.2.1 The Wireless Transceiver Subsystem

The 11 Mbps Spectrum24® LA-4121 wireless LAN card operating at 5 V supports a continuous access mode (CAM) and five polling modes, P1 to P5 [19]. Current consumed by the WLAN card during active transmission and reception is approximately uniform across all polling modes. Table 1 shows that the current, and hence power, consumed by LA-4121 wireless LAN card depends upon its mode of operation. Transmission energy is the product of the power consumed in the transmit mode and the time to transmit the data.

	Continuous access mode		P1 polling mode	
	Current (mA)	Power (W)	Current (mA)	Power (W)
Sleep	-	-	10	0.05
Idle	170	0.85	30	0.15
Receive	190	0.95	190	0.95
Transmit	410	2.05	410	2.05

Table 1: Power consumed by 11 Mbps Spectrum24® LA-4121 WLAN card

3.2.2 Message Authentication Code

We used SHA-256, a variation of a 256-bit symmetric block encryption algorithm, as the message authentication code (MAC) [31]. SHA-256 encrypts the intermediate hash values using the message blocks as the keys. Figure 5 shows the energy consumed by optimized 'C' implementation of SHA-256 for different data sizes.

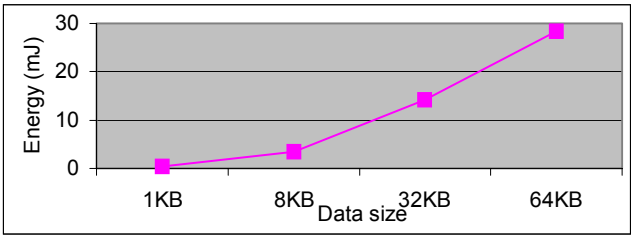

Figure 5: Energy consumed by SHA-256 MAC

3.2.3 Data Encryption

Data Encryption Standard (DES) and triple-DES (3DES) are private key encryption algorithms that have been widely used for more than 20 years now to ensure data privacy [30]. Recently National Institute of Standards and Technologies (NIST) selected Rijndael as the Advanced Encryption Standard (AES) to replace DES and 3DES in systems with higher security and performance requirements [20]. AES supports multiple user key lengths (128, 192, or 256 bits) and multiple data block sizes (128, 192 or 256 bits) [21]. Security level and the energy

consumption of a private key encryption algorithm increase with the number of encryption rounds and the length of the user key. Table 2 summarizes the energy consumed by round key generation and encryption of optimized 'C' implementations of AES. Energy consumed by AES key generation increases with the key size at a rate greater than the energy consumed by its encryption due to an increase in the complexity of key generation and an increase in the number of round keys to be generated.

AES key generation			
	128-bit key	192-bit key	256-bit key
Energy (μJ)	10.44	13.70	17.44
Time (μS)	7.48	9.82	12.47

(a)

AES encryption			
	128-bit key	192-bit key	256-bit key
Energy/bit (μJ)	0.067	0.07	0.075
Throughput (Mbps)	25.96	24.58	24.1

(b)

Table 2: Energy consumed by AES (a) key-generation and (b) encryption

3.3 Energy Consumed by a Secure Wireless Session

3.3.1 Negotiating a Security Association

Energy consumed by Diffie-Hellman based handshake protocol message exchanges and associated cryptographic computations are summarized in Table 3. The energy values in bold corresponds to the client. Energy consumed by the handshake protocol depends on the master_secret exchange protocol, the level of security (size of certificates, size of the master_secret, size of the encryption and MAC keys) and the number and size of messages exchanged.

Messages exchanged		Energy consumed (milli Joule)		
		Cryptographic computations	Message exchanges	
			Transmit	Receive
M1	Client_hello	**0.01**	**15.7**	6.8
M2	Server_hello	0.01	2.6	**1.3**
	Server_certificate	-	337.2	**145**
	Server_key_exchange	60.21	337.2	**145**
	Request_certificate	-	5.3	**2.3**
M3	Client_certificate	-	**337.2**	145
	Client_key_exchange	**60.74**	**337.2**	145
	Change_cipher_spec,finished	-	**2.6**	1.3
M4	Change_cipher_spec,finished	-	2.6	**1.3**
-	Encryption and MAC secret key computation at the client	**12.33**	-	-
-	Certificate verify at client	**1.21**	-	-
-	Certificate verify at server	1.21	-	-
-	Encryption and MAC secret key computation at the server	12.33	-	-
	CLIENT	**74.3**	**692.7**	**295**
	SERVER	73.8	685	298

Table 3: Energy consumed by WTLS handshake protocol using Diffie-Hellman key exchange and management

A client establishing a secure session consumes approximately 1062 milli Joules, 7% of which is consumed by cryptographic computations and 93% is consumed by message exchanges. When encryption and MAC keys are periodically refreshed only 12.33 milli Joules are expended (at the client and at the server), $\frac{1}{85}$ of the 1062 milli Joules required for establishing a new secure session.

3.3.2 Secure Data Transaction

Table 4 summarizes the energy consumed during secure wireless data transmission assuming the following security association: 128-bit key AES encryption, SHA-256 MAC authentication, key refresh rate that entails re-computing the encryption and MAC keys every 128 KB of data, and session refresh every 2 MB of data.

Mobile-to-server transaction energy (milli Joule)		
	2560 KB data	8 KB data
SHA-256 MAC	1130	3.53
AES-128 encryption	1372	4.29
Transmission	13480	42.13
Key-refresh	245	-
Idle system	16604	51.87
Total	32831	101.82

Table 4: Energy consumed by secure wireless data transmission

Energy consumed by the idle system is inversely proportional to the sustained throughput of the system, where sustained throughput of a system is determined by a variety of factors, including the network conditions, efficiency of the wireless protocols and the requirements of the application. Energy consumed by data transmission and reception increases linearly with the input data size and the energy consumed by the cryptographic computations increases linearly with both data size and security level. For example, while transferring an 8 KB of data does not entail any key refresh overhead, transferring 2560 KB of data entails 19 key refreshes, consuming 245 milli Joules at the client and at the server. Refreshing the secret keys entails generating new encryption and MAC keys and generating round keys from the new encryption key.

Similarly, large encryption key and higher number of encryption rounds also improves the level of security at the cost of extra energy consumption and performance degradation. Cryptographic computations (key refresh, data encryption and MAC authentication) consume 7.7% of the total energy for transferring 8 KB data, and this increases to 8.4% of the total energy for transferring 2560 KB data.

4. Reducing the System Energy Consumption

Securely transferring 2560 KB of data consumes 33893 milli Joules. Handshake protocol used to negotiate the secure session consumes 3% of this energy, data transfers consume 40%, data encryption, authentication and key refreshes consume 8%, and the idle system consumes 49%. On the other hand, securely transferring 8 KB data consumes 1164 milli Joules, of which 91.2% is consumed by the handshake protocol, 3.6% by data transfers, 0.7% by data encryption and authentication and 4.5% by the idle system. Therefore, for small data transactions energy consumed by the handshake protocol is most dominant, but as the data size increases energy consumed by data transmission, encryption and authentication become more significant. In this section we will investigate compression, handshake protocol optimization and hardware acceleration of cryptographic mechanisms to target these sources of energy consumption.

4.1 Impact of Compression

Compressing the data before encryption and transmission and decompressing it after reception and decryption reduces the energy consumed by a secure wireless transaction if the energy savings due to the reduced data size are more than the extra energy consumed by compression and decompression. Similarly, compressing the messages exchanged during secure session negotiation reduces the energy consumed by the handshake protocol.

To study the impact of compression we used optimized 'C' implementation of DEFLATE loss-less data compression algorithm [22]. Related research has been done to evaluate of the impact of data compression on the performance of virtual private networks [34] and on the performance of applications using WML byte codes in WAP environment [35]. Compression level, history window size and memory-level are the three important parameters that affect the energy consumed by DEFLATE compression. Table 5 summarizes the energy consumed by DEFLATE on the mobile test bed while compressing 1KB, 8 KB and 64 KB size benchmarks from Calgary corpus [23].

Data size		Compression level =9 Memory level =9	Compression level =1 Memory level =9	Compression level =9 Memory level =1	Compression level =5 Memory level =5
64 KB	Energy (milli Joule)	1004.15	132.59	2785.15	395.79
	Compression ratio	4.482	3.5884	4.0042	4.3256
8 KB	Energy (milli Joule)	55.71	46.11	65.52	31.69
	Compression ratio	3.5085	3.1059	3.1035	3.4782
1 KB	Energy (milli Joule)	26.62	34.57	14.24	14.76
	Compression ratio	2.8679	2.7861	2.4805	2.8254

Table 5: Energy consumed by DEFLATE compression

From Table 5 it can be seen that matching the compression block size to the data cache size (8 KB for our mobile test bed) saves significant energy. Further, increasing either the compression level or the memory level results in a proportional increase in the energy consumed without a corresponding increase in the compression ratio. On the other hand, increasing the size of the history window yields a proportional increase in the compression ratio without a corresponding increase in the energy consumed. We found that medium compression level (level 5), a medium memory level (level 5) and a maximum history window size (32 KB) combination achieves a compression ratio close to the best, while consuming significantly less energy. In this paper we will use DEFLATE with these parameters. Further our experiments showed that energy consumed by decompression is approximately one-tenth of the energy consumed by compression since decoding is very simple and fast.

Therefore, while transmitting data the compression block size should be matched to the data cache size of the device and while receiving data large compression block size (larger the better) should be used to reduce the client energy. Such an asymmetric compression arrangement can be agreed upon during the secure session negotiation.

4.1.1 Compressing Handshake Messages

Generation and exchange of the client and server certificates (server_certificate, client_certificate) and the master_secret (client_key_exchange, server_key_ exchange) consumes more than 90% of the handshake protocol energy.

	Energy consumed by handshake protocol (milli Joule)	
	Uncompressed	Compressed
Transmit	692.7	199.1
Receive	295	84.8
Cryptographic computations	74.3	74.3
Compress/Decompress	-	568.7
Total	1062	926.9
Energy saving factor		1.15 ×

Table 6: Energy consumed by WTLS handshake protocol

Since decompression energy is very small compared to energy for compression, it is always energy-efficient for a mobile device to receive compressed data. Hence, if the WAP gateway compresses all messages following the **server_hello** message, energy consumed by the client during handshake protocol can be reduced. Table 6 shows a 1.15× reduction in the energy consumed by client during the handshake by compressing the 64-Kbit certificates and the 64-Kbit master_secret messages exchanged by the client and the server.

4.1.2 Compression of Secure Wireless Transactions

Energy consumed during transmission and reception of uncompressed and compressed, AES encrypted 2560 KB and 8 KB data are summarized in Table 7. Energy consumed by data transmission is 10× the energy consumed by AES encryption which in turn is 0.15× the energy consumed by DEFLATE data compression for 8 KB compression block size[3].

Besides reducing the size of the data to be encrypted and transmitted, compression also reduces the energy consumed by key refreshes. For example, if encryption and MAC keys are refreshed after exchanging 128 KB of data, an uncompressed 2560 KB data transfer requires 19 key refreshes and consumes 245 milli Joules, while the compressed 736 KB data (=2560 KB÷compression ratio of 3.4782) requires only 5 key refreshes and consumes 64 milli Joules. Therefore, data compression reduces (a) the transmission, reception, encryption and decryption energy during a secure data transaction (b) the number of key refreshes required and the corresponding energy, and (c) the energy consumed by the idle system.

	Energy consumed (milli Joule)	
	2560 KB data	
	Uncompressed	Compressed
Compress	-	10141
AES-128 encrypt	1372	394.6
AES-128 decrypt		
SHA-256 sign	1130	324.9
SHA-256 verify		
Transmit	13480	3876
Receive	5803	1668
Decompress	-	1014
Key-refresh	245	64.45
Idle system	16604	4773
Total transmit energy	32831	19574
Transmit energy saving factor	-	1.68 ×
Total receive energy	25154	8239
Receive energy saving factor	-	3.05 ×

Table 7: Energy consumed during a secure wireless transaction

4.2 Optimizing the Handshake Protocol

The handshake protocol discussed until now consumes significant energy, with the generation and exchange of certificates and master_secret material consuming more than 90% of the total client energy during handshake.

As shown in section 4.1.1, compressing the client and the server certificates reduces the client energy consumption by 1.15×. Client energy can also be reduced by 1.5× by modifying the handshake protocol such that

[3] We used the energy consumed by wireless transmission from Section 3.2.1, energy consumed by keyed-MAC from Section 3.2.2, energy consumed by 128-bit key Rijndael encryption from Section 3.2.3 and energy consumed by DEFLATE compression from Section 4.1

the server looks up the client's certificate from its own source (**handshake variant 1**). Embedding the client's share of master_secret in its certificate can further reduce its energy by another 1.7×. When establishing a new session, a client-server pair can exchange new client and server random numbers and combine these with previously negotiated security association (**handshake variant 2**). Finally, implanting the master secret in the server and mobile device eliminates the energy consumed by master_secret exchange messages (**handshake variant 3**) and reduces the energy consumed by the client by 33×. Table 8 shows the energy consumed by these various handshake protocols.

	Handshake protocol energy consumption (milli Joule)			
	Basic	variant 1	variant 2	variant 3
Transmit	692.7	101	18.3	18.3
Receive	295	84	2.6	2.6
Master_secret computation + authentication	62	62	62	-
Key generation	12.3	12.3	12.3	12.3
Total	1062	259.3	75.2	33.2
Energy saving factor		4.1 ×	14.12 ×	32 ×

Table 8: Energy consumed by various handshake protocols

We propose an **adaptive handshake** that uses **handshake variant 1** to establish a new secure session and **handshake variant 2** to refresh the security association by exchanging new client and server random numbers if the session lasts beyond a certain number of messages determined by the security requirements of the session or if the session is disrupted abruptly due to bad channel conditions or temporary network outage. Allowing the client and the server to use compression immediately following the **server_hello** message minimizes the energy consumed during the **handshake variant 1** protocol. Since **handshake variant 3** is inflexible (master_secret is implanted permanently into the server and the client) and vulnerable to physical and side-channel attacks to recover the implanted master_secret, we did not include it in the adaptive handshake protocol.

4.3 Impact of FPGA Implementation of AES (Rijndael)

Implementing cryptographic computations in hardware improves the throughput and reduces the energy consumption. Results of our AES implementation on Wildcard™ FPGA board are summarized in Table 9 (a). Area is computed as the number of Virtex slices used by the design where a Virtex slice contains two look-up tables and one look-up table can implement a four input-one output logic function. Throughput of encryption, defined as the number of bits encrypted per second, is calculated as

$$\frac{\#\text{ of bits encrypted}}{\#\text{ of clock cycles for encryption} \times \text{clock period}}$$

We used Xilinx® FPGA power estimation tool Xpower™ [33] to estimate the energy consumed by our FPGA implementation of AES. Table 9 shows that the FPGA implementation improves the throughput and reduces the energy consumed by AES encryption.

Since hardware resources are limited in portable mobile devices, all cryptographic operations cannot be implemented in hardware and a trade-off should be made between the area overhead, performance and energy benefits. For example, the mix-column operation in AES is the performance bottleneck in its software implementation, consuming approximately 10× more time and energy than any other operation. Therefore, implementing the mix-column operation in FPGA and the rest of AES in software improves performance and reduces energy while consuming only 250 Virtex slices (< 7% of the AES implementation).

	AES-128 encryption	
	Software	Hardware
Area (# of slices)	-	3973
Throughput (Mbps)	25.963	124.8
Energy/bit (micro Joule)	0.067	0.0012

(a)

	Energy consumed by secure wireless data transaction of 2560 KB data (milli Joule)	
	Uncompressed	Compressed
Total transmit (software encryption/decryption)	32831	19574
Total transmit (hardware encryption/decryption)	31484	19186
Transmit energy saving factor	1.04 ×	1.02 ×
Total receive (software encryption/decryption)	25154	8239
Total receive (hardware encryption/decryption)	23807	7851
Receive energy saving factor	1.06 ×	1.05 ×

(b)

Table 9: (a) Software vs. Hardware implementation of AES encryption and (b) the impact of hardware implementation of encryption on the energy consumed by a secure wireless session

5. Summary

WTLS offers three levels of security: periodically refreshing the secure session, refreshing the encryption and MAC keys after 2^n data transactions and varying the key length and the number of rounds of encryption. Presented techniques affect WTLS secure sessions at all three security levels, as shown in Table 10.

Security level	Scheme
Refreshing the secure session	Protocol optimization, Compression
Refreshing the secret keys	Compression
Increasing the length of keys and number of encryption rounds	Hardware acceleration of cryptographic computations

Table 10: Improving the secure session energy consumption at various security levels

Let us examine the impact of the presented techniques on the energy consumption characteristics of a secure wireless session while transmitting 20 MB data over a 200 Kbps wireless channel with key refresh every 128 KB of data and session refreshes every 2 MB of data.

Table 11 compares the energy consumed by an un-optimized scheme using **basic handshake**, supporting AES-128 encryption and SHA-256 MAC in software and no compression scheme with the energy consumed by an optimized scheme using **adaptive handshake**, supporting AES-128 encryption and SHA-256 MAC in hardware and using DEFLATE data compression. We assume a compression ratio of 3.48 corresponding to the data block size of 8 KB. Figures in '()' refer to the frequency of the corresponding operations. For example, Un-optimized scheme requires 9 **basic handshakes** and 150 key refreshes for the entire data transaction. Optimized scheme reduces the energy consumed during data transmission by more than 1.5× and the energy consumed during data reception by more than 3×. Since energy saved by encrypting the compressed data in hardware is not significant, adopting a performance, energy consumption and area tradeoff based software-hardware co-design approach, as discussed in section 4.3, is more beneficial.

	Un-optimized secure session		Optimized secure session	
	Session parameters	Energy (milli Joule)	Session parameters	Energy (milli Joule)
Handshake	Basic (9)	9558	Version 1 (1), Version 2 (1)	335
Compress	-	-	DEFLATE	79225
SHA-256 sign	Software	8828	Hardware	2538
SHA-256 verify				
AES-128 encrypt	Software	10719	Hardware	3082
AES-128 decrypt				
Transmit	-	105313	-	30278
Receive	-	45336	-	13034
Decompress	-	-	DEFLATE	7988
Key refresh	(150)	1934	(43)	554
Idle system	-	129719		37295
Total transmit	-	266071	-	153307
Transmit energy saving factor				1.74 ×
Total receive	-	206094	-	64826
Receive energy saving factor				3.18 ×

Table 11: Energy savings for the client due to secure session optimization

6. References

1. J. A. Senn, "The emergence of M-Commerce", IEEE Computer, December 2000, pp. 148-150.
2. D. Clark, "Encryption advances to meet Internet challenges", IEEE Computer online magazine, December 2000.

   http://www.computer.org/computer/articles/August/technews800.htm

3. Wireless Application Protocol: Wireless Transport Layer Security Specifications, Feb 2000. http://www.wapforum.org
4. R. Kravets, P. Krishnan, “Power management techniques for mobile communication”, Proceedings, ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM), August, 1999
5. P. Agrawal, “Energy efficient protocols for wireless systems”, IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Boston, USA, September, 1998, Vol. 2, pp 564-569
6. C. Rohl, H. Woesner, A. Wolisz, “A short look on power saving mechanisms in the wireless LAN standard draft IEEE 802.11, Advances in Wireless Communications, Kluwer Academic Publishers, 1998
7. A. Chockalingam, M. Zorzi, “Energy consumption performance of a class of access protocols for mobile data networks”, Proceedings, IEEE Vehicular Technology Conference (VTC), Ottawa, May 1998
8. R. Kravets, K. Calvert, K. Schwan, “Payoff adaptation of communication for distributed interactive applications”, Journal on High Speed Networking: Special Issue on Multimedia Communications, 1998
9. S. Singh, C. S. Raghavendra, “PAMAS-Power Aware Multi-Access Protocol with signaling for ad-hoc networks”, Computer Communications Review, July 1998
10. S. Singh, C. S. Raghavendra, “Power efficient MAC protocol for multihop radio networks”, Proceedings, IEEE International Symposium on Personal, Indoor, Mobile Radio Communication (PIMRC), Boston, USA, September 1998, Vol. 1, pp. 153-157
11. J. M. Rulnick, N. Bambos, “Mobile power management for maximum battery life in wireless communications network”, IEEE International Conference on Computer Communications (INFOCOM), CA, USA, March, 1996, Vol. 2, pp. 443-450
12. D. Duchamp, N. F. Reynolds, “Measured performance of a wireless LAN”, Conference on Local Computer Networks, September 1992, pp. 494-499
13. J. Ebert, B. Stremmel, E. Wiederhold, A. Wolisz, “An energy-efficient power control approach for WLANs”, Journal of Communications and Networks, September, 2000, Vol. 2, pp. 197-206
14. A. Kamerman, L. Monteban, “WaveLAN-II: A high performance wireless LAN for the unlicensed band”, Bell Labs Technical Journal, 1997
15. M. Zorzi, R. R. Rao, “Error control and energy consumption in communications for nomadic computing”, IEEE Transactions on Information Theory, March, 1997, Vol. 46, pp. 279-289

16. M. Zorzi, R. R. Rao, "Energy constrained error control for wireless channels", IEEE Personal Communications, December 1997, Vol. 4, pp. 27-33
17. P. Lettieri, C. Fragouli, M. B. Srivastava, "Low power error control for wireless links", Proceedings, ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM), Budapest, Hungary, August1997, pp. 139-150
18. V. Tsaoussidis, H. Badr, X. Ge, K. Penikousis, "Energy/throughput tradeoffs of TCP error control strategies", Proceedings, IEEE Symposium on Computers and Communications, (ISCC), July, 2000, Antibes, France
19. Spectrum24® High Rate LA 41X1 PC Card, http://www.symbol.com/products/wireless/la41x1.html
20. http://csrc.nist.gov/encryption/aes
21. J. Daemen, V. Rijmen, "AES proposal: Rijndael",

http://www.esat.kuleuven.ac.be/~rijmen/rijndael/rijndaeldocV2.zip

22. DEFLATE Compressed Data Format Specification version 1.3.

http://www.kblabs.com/lab/lib/rfcs/1900/rfc1951.txt.html

23. T.C. Bell, "Text compression", Prentice Hall, Englewood Cliffs, NJ, 1990
24. Annapolis Micro Systems Wildcard FPGA board, http://www.annapmicro.com/products.html
25. ftp://dspftp.ece.ubc.ca/pub/tmn/qcif_source
26. http://www.ee.ubc.ca/image/
27. W. Diffie, M. E. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, v. IT-22, n. 6, November 1976, pp. 644-654
28. http://www.tektronix.com
29. Symbol PPT2800 series portable pen terminal,

http://www.symbol.com/products/mobile_computers/mobile_ppc_ppt2800.html

30. http://csrc.nist.gov/encryption
31. http://csrc.nist.gov/encryption/tkhash.html
32. R. Karri, P. Mishra, "Technical Report, Fall 2001", Polytechnic University, Brooklyn, NY

http://emme.poly.edu/Goodies/TEMP/TechnicalReportSpring2001.pdf

33. "XILINX XPower: Power estimation tool for programmable logic"

http://www.xilinx.com/xlnx/xil_prodcat_product.jsp?title=xpower

34. J. P. McGregor, R. B. lee, “Performance impact of data compression on virtual private network transactions”, Proceedings, IEEE Conference on Local Computer Networks, 2000
35. E. Ojanen, J. Veijalainen, “Compressibility of WML and WMLScript byte cod: Initial results”, Proceedings, IEEE Workshop on Research Issues in Data Engineering (RIDE), February 2000, CA USA