

Performance and Energy Efficiency of Block Ciphers in Personal Digital Assistants

Creighton T. R. Hager, Scott F. Midkiff, Jung-Min Park, Thomas L. Martin
Bradley Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061 USA
{chager, midkiff, jungmin, tlmartin} @ vt.edu

Abstract

Encryption algorithms can be used to help secure wireless communications, but securing data also consumes resources. The goal of this research is to provide users or system developers of personal digital assistants and applications with the associated time and energy costs of using specific encryption algorithms. Four block ciphers (RC2, Blowfish, XTEA, and AES) were considered. The experiments included encryption and decryption tasks with different cipher and file size combinations. The resource impact of the block ciphers were evaluated using the latency, throughput, energy-latency product, and throughput/energy ratio metrics.

We found that RC2 encrypts faster and uses less energy than XTEA, followed by AES. The Blowfish cipher is a fast encryption algorithm, but the size of the plaintext affects its encryption speed and energy consumption. Faster algorithms seem to be more energy efficient because of differences in speed rather than differences in power consumption levels while encrypting.

1. Introduction

A personal digital assistant (PDA) is a handheld device that was originally designed as a personal organizer, but that has evolved into a wireless communication device. Many commercial and military applications now require PDAs with secure wireless transmission capability. Moreover, information privacy has become a major concern for all users, even personal users. A variety of algorithms are available for a user to encrypt data and, thus, prevent eavesdroppers from obtaining critical or private information. However, the execution of encryption algorithms consumes both time and energy. While certain encryption algorithms may be less vulnerable to compromise than others, constantly using cryptographically strong algorithms may result in severely reduced lifetimes for battery-powered devices such as PDAs. In other words, utilizing stronger encryption

algorithms may consume more energy and drain the PDA battery faster than using less secure algorithms. Due to the processing requirements and the limited computing power in many PDAs, using strong cryptographic algorithms may also significantly increase the delay between data transmissions. Thus, users and, perhaps more importantly, software and system designers need to be aware of the benefits and costs of using various encryption algorithms.

This research answers questions regarding energy consumption and execution time for various encryption algorithms executing on a PDA platform with the goal of helping software and system developers design more effective applications and systems and of allowing end users to better utilize the capabilities of PDA devices. In particular, we experimentally measure and compare the energy consumption and computation time for four different block cipher encryption algorithms – RC2, Blowfish, the eXtended Tiny Encryption Algorithm (XTEA), and the Advanced Encryption Standard (AES) – executing on contemporary PDAs. We measure energy consumption for one device and latency and throughput for three different devices. The experiments consider different transfer sizes from one kilobyte to one megabyte. Based on the results, we develop observations that are intended to increase the awareness of and insight into the costs associated with using encryption algorithms.

This paper is divided into five sections. Section 2 discusses related work and briefly describes the encryption algorithms considered in this study. Section 3 describes the experimental setup and evaluation methods. Section 4 discusses the measurements and results. Finally, Section 5 summarizes the work, discusses contributions, and proposes future work.

2. Related Work and Algorithms Considered

This section discusses related work and introduces the four encryption algorithms investigated in this research.

2.1. Related Work

One area of related work is that of low power software. The consensus is that the potential for power savings in software is greater than the potential for savings in hardware, but that the software savings are more difficult to achieve [1]. Space limitations prevent a full overview of research in low-power software, but a common finding is that energy consumption is tied closely to execution time [2, 3]. The results in Section 4 indicate a similar conclusion. This paper studies the execution time and energy usage of encryption algorithms with various strengths of security. One application of such results is the possibility of adapting the level of security to available energy and processing resources. In a similar vein, Flinn and Satyanarayanan conducted a study of the energy savings of adapting the quality of a video player to available levels of network bandwidth [4].

Prior work from Ganesan and others [5] assess the feasibility of different encryption schemes for a range of embedded architectures using execution time overhead measurements. Dhawan [6] describes a study that compared the performance of several encryption algorithms in different network environments. First round AES candidates were compared in terms of efficiency by Bassham [7] using systems with processors in the 200 MHz to 500 MHz range. A study on Palm OS devices [8] also compared performance measurements of AES and other ciphers. Potlapally and others [9] investigated energy consumption of different ciphers with the Secure Sockets Layer [10]. Our study extends research on the PDA platform and focuses on both the computational performance and the energy consumption of block cipher encryption algorithms.

2.2. Encryption Algorithms Investigated

A block cipher is a type of symmetric-key encryption algorithm that encrypts data in blocks rather than encrypting one bit at a time in a stream, otherwise known as a stream cipher [11]. Stream ciphers are commonly used with streaming applications or for secure connections, such as Transport Layer Security [12], while block ciphers are typically used for storing data in a data base or encrypting files. Since streaming applications, such as streaming audio or video, require constant connectivity and resources that are not always available in PDAs, block ciphers were chosen for this work. Four block ciphers, described below, were used in this study: RC2, Blowfish, XTEA, and AES.

RC2 is a 64-bit block cipher with a variable size key. Like most block ciphers, RC2 uses a Feistel network [13] for diffusing the plaintext. A Feistel network is usually a combination of bit-shuffling through permutation boxes (P-boxes), simple non-linear operations using substitution

boxes (S-boxes), and linear mixing using the exclusive-or (XOR) operator. The block cipher iterates plaintext through its Feistel network to generate the ciphertext. Each iteration through the Feistel network is usually counted as one “round” of encryption. In RC2, its 18 rounds are arranged as a source-heavy Feistel network, which means the input to the round function is larger than the output of the round function. The Feistel network involves mixing and mashing rounds. The mixing rounds consist of sequentially interleaving an expanded encryption key with the plaintext, while the mashing rounds combine different pieces of the expanded key and the results of the mixing rounds. There are 16 rounds of mixing punctuated by two rounds of mashing [14].

Blowfish is a symmetric key (also known as a secret or private key) block cipher designed in 1993 by Schneier [15]. Blowfish has a 64-bit block size and a key length of anywhere from 32 bits to 448 bits. It is a 16-round Feistel cipher and uses large key-dependent S-boxes.

The Tiny Encryption Algorithm (TEA) is a block cipher noted for its simplicity of description and implementation (typically a few lines of code) [16]. However, TEA has a few vulnerabilities [17] and, consequently, XTEA was designed to correct those weaknesses [18]. XTEA is a 64-bit block Feistel network with a 128-bit key and a recommended 64 rounds.

AES, also known as Rijndael, is a block cipher adopted as an encryption standard by the U.S. government [19, 20]. AES is based on a substitution-permutation network and has a fixed block size of 128 bits and a key size of 128, 192, or 256 bits.

3. Methodology

The objective of this study is to determine the time and energy costs associated with specific encryption algorithms for PDAs. Several assumptions are necessary to limit the scope of this objective. The intent of these assumptions is to keep manageable the complexity of the implementation and analysis. Moreover, many case studies would be needed to consider all of the issues involved with different encryption algorithms and systems. We present a somewhat simple, but informative, set of case studies involving four encryption algorithms on three different PDA platforms, to gain insight into the cost of utilizing encryption algorithms in a PDA environment.

3.1. Assumptions and Experimental Setup

The selection of a particular evaluation technique can significantly impact the outcome of a performance evaluation study. Three techniques for performance evaluation are analytical, simulation, and measurement. These methods differ in terms of accuracy, cost, and required time. Based on these factors, experimental

measurement was the most appropriate technique for this case study. Analytical evaluation was ruled out due to the precision required in analytical models and discrepancies between different types of PDAs. Simulations typically offer less accuracy than actual measurements, but are also less costly and time consuming. In this case, the cost of implementing the experiments was relatively low because the required hardware and parts of the software base are readily available. Therefore, actual measurements with controlled workloads were used in this performance analysis.

Only block ciphers were considered in this research due to their frequent use, encryption speed, and ease of implementation. The methodology and, in a general sense, the results of this study are still applicable to other ciphers, algorithms, and security protocols with appropriate modification. More detailed consideration of techniques other than block ciphers is a possible area of future research. The study focuses on the resource implications of block ciphers. Cryptanalysis of the ciphers is not part of this research, but, of course, vulnerabilities and other cryptographic weaknesses must be considered by users and system designers.

Three commercially available PDAs running two major versions of Microsoft's Pocket PC (PPC) were used for the experiments, the Compaq (now Hewlett-Packard) iPAQ 3850, the Hewlett-Packard iPAQ 4150, and the Dell Axim X30. These devices were chosen because of their availability and differences in architecture and operating system (OS) versions. Relevant specifications of the Pocket PCs used in this study are listed in Table 1. The iPAQ 3850 is the oldest device and uses an Intel StrongArm processor. The more contemporary HP iPAQ 4150 and Dell Axim use Intel XScale processors. For its power source, the iPAQ 3850 uses a lithium polymer battery while the other two PDAs use lithium ion batteries. Latency and throughput were determined for all three devices. Energy consumption was examined only for the HP iPAQ 4150 because, with the equipment available for this research, power measurements are difficult for the other devices that do not have a removable battery or any other direct way to access the battery terminals. Note that the research methodology is not limited to these devices and may be extended to include other handheld devices, such as other PPC devices, PalmOS PDAs, Blackberry PDAs, smart phones, or even embedded devices for sensor networks. Note that, in general, power measurements are difficult for devices that do not have a removable battery or other easy way to access the battery terminals.

We used the Microsoft's Visual Studio .NET and Compact Framework (CF) as the software development environment. Code was written in C# to create the applications and block ciphers for the experiments. Rather than creating implementations "from scratch,"

existing block cipher implementations [21-24] written in C, C++, and C# were ported for use in this study. Validation was needed to determine if the block ciphers function correctly. To verify proper implementation, the block ciphers were applied to their corresponding standard test vectors.

Table 1. Specifications of devices used

Device	CPU	Battery	OS
HP iPAQ 3850	206 MHz Intel StrongARM	1400 mAh	MS PPC 2002
HP iPAQ 4150	400 MHz Intel PXA255	1000 mAh	MS PPC 2003
Dell Axim X30	624 MHz Intel PXA270	950 mAh	MS PPC 2003 SE

Table 2. Block cipher operating parameters

Block Cipher	Key Size (bits)	Block Size (bits)	Rounds
RC2	40	64	18
Blowfish	448	64	16
XTEA	128	64	64
AES	256	128	14

The operating parameters for the block ciphers in our experiments are presented in Table 2. Most of the parameters, such as key size and number of rounds, were maximized when using the ciphers for our experiments. Since longer keys and more rounds require more time and effort to attack the cipher, this could enable us to measure the performance and energy consumption when the algorithm offers the maximum security. The block ciphers operated in electronic code book (ECB) mode [11]. ECB mode was mainly chosen due to its straightforward implementation. The ECB mode does not require any additional operations for each algorithm to encrypt or decrypt a file. Cipher Block Chaining (CBC) overcomes the problems of repetition and order interdependence in ECB mode, but requires extra memory and operations involving an Initial Vector (IV). Cipher FeedBack (CFB) and Output FeedBack (OFB) modes treat plaintext as a stream of bits, however, we do not use stream oriented data for our experiments. Moreover, the objective is to compare the inherent algorithms of the block ciphers not the different cipher operating modes.

For the plaintext data to be encrypted (or ciphertext data to be decrypted), five different files of different sizes were generated with random content. File sizes in bytes were 2^{10} (1,024 bytes or 1 KB), 2^{12} (4,096 bytes or 4 KB),

2^{15} (32,768 bytes or 32 KB), 2^{17} (131,072 bytes or 128 KB), and 2^{20} (1,048,576 bytes or 1 MB). Each plaintext file was encrypted 100 times with each block cipher using the operating parameters specified in Table 2. Likewise, the ciphertext versions of the files were decrypted 100 times each. There are twenty combinations of different file sizes and different block ciphers. For example, one combination uses AES with a 256-bit key, 128-bit block, and 14 rounds to encrypt 2^{12} bytes (4 KB) of plaintext data 100 times.

For each combination, the selected file was loaded into the PDA's memory. The unencrypted data blocks were kept statically in memory for the duration of the experiments. Static keys were used for encrypting the files. In addition, the encrypted memory contents were flushed after each encryption process to eliminate the effects of caching. Similarly, the decryption processes used encrypted data blocks that were statically stored in memory and generated from the encryption procedures. To minimize the current drawn from the battery by other causes, each Pocket PC device used its dimmest backlight setting, any wireless devices were disabled, and all other active background programs were terminated before running each test.

3.2. Metrics and Measurement Methods

Evaluating the impact of encryption algorithms on processor and battery resources was the focus of this study. The following sections explain the metrics and methods used in the study.

3.2.1. Latency and Throughput. Metrics for block cipher performance indicate the speed and efficiency of encrypting plaintext and decrypting encrypted text. To measure the encryption or decryption latency, the query performance frequency and query performance counter functions [25] were invoked from the "CoreDll.dll" library in the operating system. The .NET platform invoke facility (P/Invoke) can expose functions in any dynamic link library (DLL). These functions measure the frequency and current clock value of the central processing unit (CPU). The query performance frequency function was used to determine the CPU frequency of the device. The query performance counter function was called before and after each encryption or decryption process to measure elapsed clock cycles. The difference between the counter values was divided by the CPU frequency to obtain the encryption or decryption time in seconds. Encryption and decryption latencies represent the encryption and decryption times per byte or file. The encryption and decryption throughput values were calculated by dividing the number of bytes encrypted or decrypted by the time required to encrypt or decrypt that number of bytes.

3.2.2. Energy Consumption. Other metrics indicate how much power and energy is consumed by a block cipher for encryption or decryption. Two different ciphers can be compared (at least roughly) to determine which is more power or energy efficient. To measure the current power level, the battery of a device is first removed. A resistor is placed in series with the battery and the device. A multimeter is used to measure the voltage drop across the resistor.

We conducted power measurements using an HP iPAQ 4150, a very low resistance (0.025Ω) precision resistor in series between a benchtop power supply and the device itself, and an Agilent 3458A $8\frac{1}{2}$ Digit Multimeter, as shown in Figure 1. The benchtop power supply was an Agilent 66319B Mobile Communications DC Source. The multimeter measured the voltage across the resistor 10,000 times per second. The resulting voltage measurements were then multiplied by the 4.1 V input voltage and divided by the 0.025Ω resistance to calculate the power values. This procedure is indicated by Equation 1.

$$P(t) = V(t) \cdot V_{input} / R \quad (1)$$



Figure 1. Setup to measure energy consumption.

Energy consumption for each encryption and decryption task, E_{task} , is computed using Equation 2.

$$E_{task} = \sum_{i=0}^n [P(t_i) - P_{idle}] \cdot T \quad (2)$$

An encryption or decryption task begins at time t_0 , which is the time when the measured power level significantly increases as a cipher begins encrypting or decrypting a file. The task ends at time t_n , which is the time when the measured power level significantly decreases as the task completes encrypting or decrypting the file. As indicated in Equation 2, E_{task} is based on the $n+1$ multimeter measurements from t_0 to t_n . Each measured power value during the encryption or decryption task, $P(t_i)$, $i = 0, \dots, n$, is reduced by the average power level measured when the device is idle,

P_{idle} . The sum of the adjusted power values from t_0 to t_n is multiplied by the sampling interval, $T = 100 \mu s$ (since there are 10,000 samples per second), to determine E_{task} , the energy consumed by the task. A MATLAB script was written to process the data collected from the multimeter.

Note that the duration of the interval $t_n - t_0$ is an alternate indicator of encryption or decryption time. This method was not used here, however, since it is not as accurate as the procedure using clock cycle measurements as described in Section 3.2.1. Results computed using the two methods do generally agree as described in Section 4.

3.2.3. Energy-Latency Product and Throughput/Energy Ratio. The power measurements and encryption and decryption performance results can be combined to determine the energy-latency product and the throughput/energy ratio. These metrics can be used to compare the block ciphers in terms of energy efficiency at the byte or packet level. The energy-latency product is the encryption or decryption latency multiplied by power and represents the amount of energy required for the encryption or decryption task. It can be divided by the number of bytes encrypted or decrypted to determine the Joules of energy consumed per byte of data. This can be thought of as the energy cost per unit of data encrypted or decrypted.

The throughput/energy ratio is the encryption or decryption throughput result divided by power and indicates the number of bytes encrypted or decrypted per joule of energy expended. This can be thought of as the benefit or utility for encryption or decryption that can be obtained from a joule of energy.

4. Results and Discussion

This section presents and discusses the experimental results for latency, throughput, and energy consumption.

4.1. Latency and Throughput

The experiments to determine latency and throughput consisted of twenty different combinations of cipher algorithm and file size, as described in Section 3.1. For each combination, the specific file encryption and decryption tasks were replicated 100 times. Using the methods described in Section 3.2.1, the encryption and decryption times were calculated for each task and then averaged for the 100 replications. Encryption and decryption times were also calculated using the hardware measurements from the multimeter on the HP iPAQ 4150 PDA for general verification. Even though the hardware measurements generate relatively similar encryption and decryption times (differences are within 7%), the software measurements from the clock cycle function calls are considered to be more accurate for computing encryption

and decryption times and are used for the rest of the analysis. Average encryption and decryption times for each combination of cipher scheme and file size are used to compute the average performance metrics for each of the three devices. These results for latency and throughput for encryption (Enc) and decryption (Dec) for the three different PDAs used in this study are shown in Tables 3 through 5. Variance in the measurements typically fell under 0.1% of the average values. The algorithm with the largest variance in measurements was XTEA, which had variances that were approximately 0.5% of the average values. These were caused by a few outliers in the raw data.

Table 3. Performance results for the iPAQ 3850

Cipher	File Size (bytes)	Enc/Dec Latency (μs /byte)		Enc/Dec Throughput (KBps)	
		Enc	Dec	Enc	Dec
RC2	2^{10}	3.24	3.98	301	246
	2^{12}	2.60	2.69	376	363
	2^{15}	2.45	2.49	398	392
	2^{17}	2.36	2.45	415	398
	2^{20}	2.32	2.43	421	402
Blowfish	2^{10}	10.08	10.34	97	94
	2^{12}	3.68	3.39	266	288
	2^{15}	1.52	1.46	644	668
	2^{17}	1.25	1.23	780	797
	2^{20}	1.10	1.09	891	900
XTEA	2^{10}	11.57	10.80	84	90
	2^{12}	11.40	11.33	86	86
	2^{15}	11.88	11.77	82	83
	2^{17}	11.79	11.43	83	85
	2^{20}	12.36	11.39	79	86
AES	2^{10}	19.82	20.98	49	47
	2^{12}	19.88	18.90	49	52
	2^{15}	21.63	19.25	45	51
	2^{17}	21.78	18.98	45	51
	2^{20}	20.81	18.75	47	52

The average latency and throughput values remain relatively stable when encrypting and decrypting different file sizes, except for the Blowfish cipher which seems to work more efficiently with larger file sizes. This could be due to the way Blowfish uses one function to process the entire buffer of plaintext and to encrypt the blocks with only a few operations in each round. The XOR operation

is used frequently within Blowfish, compared to the other ciphers that mostly use addition. For file sizes larger than 4 KB, the Blowfish cipher is the fastest, followed by RC2, XTEA, and AES, in that order. This ordering appears to be the same across all three Pocket PC devices. However, there are some discrepancies with smaller file sizes. When encrypting or decrypting the two smallest file sizes (1 KB and 4 KB), the ordering of ciphers from fastest to slowest is RC2, Blowfish, XTEA, and AES for the iPAQ 3850. However, XTEA encrypts faster than Blowfish on the other two Pocket PCs and the ordering from fastest to slowest is RC2, XTEA, Blowfish, and AES.

The RC2 cipher is fastest for small files, most likely because it has only one P-box for key expansion loaded into memory, versus one P-box and four S-boxes in Blowfish. The delay of loading the P- and S-boxes into memory becomes more significant when the size of the plaintext approaches the size of the boxes. RC2 also uses fewer rounds than XTEA.

Table 4. Performance results for the HP iPAQ 4150

Cipher	File Size (bytes)	Enc/Dec Latency (µs/byte)		Enc/Dec Throughput (KBps)	
RC2	2 ¹⁰	1.95	1.92	501	509
	2 ¹²	1.63	1.66	598	588
	2 ¹⁵	1.51	1.59	646	615
	2 ¹⁷	1.51	1.57	645	620
	2 ²⁰	1.51	1.57	648	622
Blowfish	2 ¹⁰	6.40	6.49	153	151
	2 ¹²	2.20	2.13	445	458
	2 ¹⁵	0.94	0.93	1043	1045
	2 ¹⁷	0.85	0.84	1146	1169
	2 ²⁰	0.76	0.76	1284	1290
XTEA	2 ¹⁰	3.32	3.55	294	275
	2 ¹²	3.39	3.12	288	313
	2 ¹⁵	3.29	3.41	297	286
	2 ¹⁷	3.30	3.28	296	298
	2 ²⁰	3.30	3.41	296	287
AES	2 ¹⁰	10.78	10.97	91	89
	2 ¹²	10.24	9.98	95	98
	2 ¹⁵	10.02	9.94	97	98
	2 ¹⁷	10.01	9.92	98	98
	2 ²⁰	9.99	9.94	98	98

Table 5. Performance results for the Dell Axim X30

Cipher	File Size (bytes)	Enc/Dec Latency (µs/byte)		Enc/Dec Throughput (KBps)	
RC2	2 ¹⁰	1.45	2.24	675	436
	2 ¹²	1.13	1.16	861	842
	2 ¹⁵	0.99	1.06	984	917
	2 ¹⁷	1.01	1.05	968	934
	2 ²⁰	1.01	1.04	970	935
Blowfish	2 ¹⁰	4.65	4.63	210	211
	2 ¹²	1.46	1.42	669	687
	2 ¹⁵	0.64	0.66	1526	1490
	2 ¹⁷	0.59	0.57	1657	1702
	2 ²⁰	0.51	0.51	1914	1925
XTEA	2 ¹⁰	2.31	2.72	423	359
	2 ¹²	2.29	2.13	426	459
	2 ¹⁵	2.23	2.28	438	429
	2 ¹⁷	2.24	2.18	436	448
	2 ²⁰	2.24	2.18	437	448
AES	2 ¹⁰	7.19	7.60	136	128
	2 ¹²	6.92	6.56	141	149
	2 ¹⁵	6.54	6.51	149	150
	2 ¹⁷	6.54	6.49	149	151
	2 ²⁰	6.57	6.52	149	150

The AES cipher was always the slowest of the four block ciphers in our experiments. The series of linked mathematical operations with P-boxes and S-boxes gives AES its strong cryptographic properties, but at the cost of more memory and processing requirements compared to the other three block ciphers.

The XScale processor of the HP iPAQ 4150 and Dell Axim X30 appears to favor the XTEA cipher. RC2 is about twice as fast as XTEA for the HP iPAQ 4150 and Dell Axim X30, while RC2 is about 4.5 times faster than XTEA for the iPAQ 3850. XTEA is about as fast to 1.75 times faster compared to AES on the iPAQ 3850, while XTEA is about as fast to three times faster than AES on the HP iPAQ 4150 and Dell Axim X30.

The performance differences between encryption and decryption are small (typically less than an 8% difference). However, decryption is, overall, slightly faster than encryption when using the Blowfish and AES ciphers. There is some variability in the results for the XTEA cipher. This is most likely due to the large number of rounds XTEA requires for encryption and decryption.

Furthermore, RC2 decrypts slower than encrypts because there are four extra AND operations per mixing round in the decryption algorithm of RC2. The differences are apparent in the smallest file size.

4.2. Energy

All energy-related metrics are based on experiments with only the HP iPAQ 4150. The graphs in Figures 2 through 5 each show power consumption for a single execution of each block cipher encrypting a 2^{15} -byte file. The voltage across the resistor in the experimental setup described in Section 3.2.2 was measured for 0.5 seconds in each case. The surges of power at the beginning and end of each encryption task are likely due to the allocation and deallocation of memory. Activity before the encryption task in Figures 3 and 4 is a result of loading the plaintext file into memory.

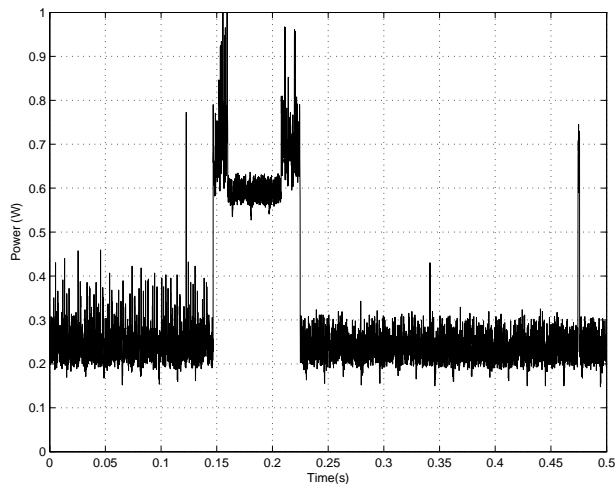


Figure 2. Power for RC2 encrypting a 2^{15} -byte file.

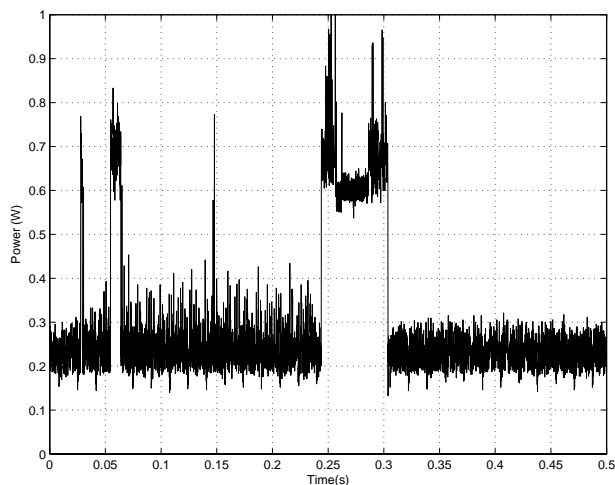


Figure 3. Power for Blowfish encrypting a 2^{15} -byte file.

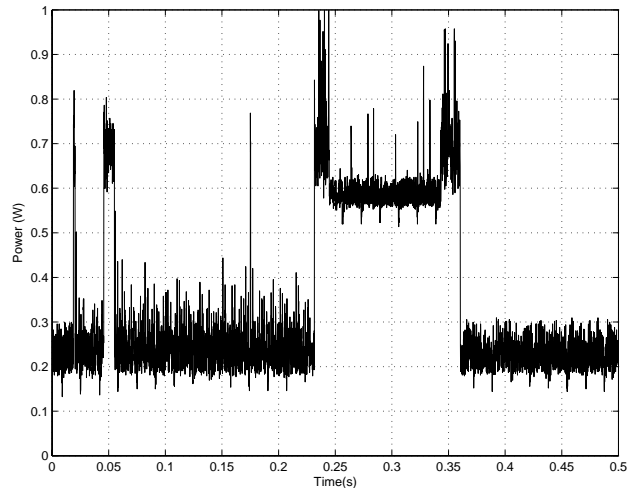


Figure 4. Power for XTEA encrypting a 2^{15} -byte file.

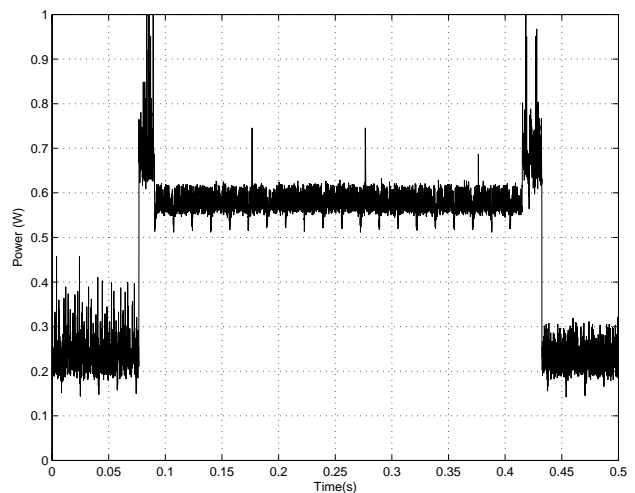


Figure 5. Power for AES encrypting a 2^{15} -byte file.

The graphs show the relationship between the block ciphers in terms of the durations required to complete the encryption task. Blowfish exhibits slightly higher speed (shorter duration) compared to RC2, followed by XTEA, and, then, AES by a wide margin. The encryption latency in the graphs is consistent with the performance of the ciphers shown in Tables 3 through 5. The power levels during the encryption task itself are relatively stable and are about the same for all four ciphers, implying that the time spent performing encryption affects energy efficiency more than differences in power drain while performing encryption.

Average energy consumption for the encryption tasks was calculated using the procedure described in Section 3.2.2. The energy-latency product and throughput/energy ratio were determined from the energy consumption values for different ciphers and file sizes. Results are presented in Tables 6 and 7. The performance and energy

metrics are highly correlated with one another since, as stated above, energy consumption depends mostly on the time required for the encryption or decryption task.

Note that the energy consumption values used do not include energy consumption associated with an idle device. The energy consumption values consider only the additional energy required for encryption. In other words, the total energy consumed from the battery during an encryption task will be the energy consumed for encryption plus the energy consumed during idle mode.

Table 6. Energy-latency products for the HP iPAQ 4150

Cipher	File Size (bytes)	Encryption ($\mu\text{J/B}$)	Decryption ($\mu\text{J/B}$)
RC2	2^{10}	0.633	0.623
	2^{12}	0.497	0.506
	2^{15}	0.401	0.420
	2^{17}	0.387	0.403
	2^{20}	0.385	0.401
Blowfish	2^{10}	1.858	1.883
	2^{12}	0.643	0.624
	2^{15}	0.276	0.276
	2^{17}	0.244	0.239
	2^{20}	0.237	0.236
XTEA	2^{10}	1.003	1.071
	2^{12}	0.857	0.789
	2^{15}	0.912	0.946
	2^{17}	0.887	0.881
	2^{20}	0.902	0.931
AES	2^{10}	2.829	2.879
	2^{12}	2.599	2.531
	2^{15}	2.524	2.505
	2^{17}	2.506	2.483
	2^{20}	2.594	2.582

In the iPAQ 4150, AES has the highest average values for the energy-latency product of the four block ciphers, which means that it consumes the most energy per byte when encrypting or decrypting plaintext. These high values are due to the large number of sequential operations required for encrypting each block of data in AES. The high energy-latency product values of AES are consistent with its high latency results shown in Table 4. XTEA consumes less energy per byte followed by RC2.

The Blowfish cipher has the lowest average energy-latency product when encrypting and decrypting large files, but this value increases as the plaintext file size decreases. A similar trend can be found in the average encryption latency values for Blowfish. The throughput/energy ratios of all the ciphers are also correlated to the throughput values in Table 4 and yield similar trends and relative comparisons.

Table 7. Throughput/energy ratios for the HP iPAQ 4150

Cipher	File Size (bytes)	Encryption (MB/J)	Decryption (MB/J)
RC2	2^{10}	1.507	1.531
	2^{12}	1.919	1.886
	2^{15}	2.381	2.269
	2^{17}	2.463	2.366
	2^{20}	2.480	2.378
Blowfish	2^{10}	0.513	0.506
	2^{12}	1.484	1.529
	2^{15}	3.451	3.457
	2^{17}	3.909	3.986
	2^{20}	4.027	4.046
XTEA	2^{10}	0.951	0.890
	2^{12}	1.113	1.208
	2^{15}	1.046	1.009
	2^{17}	1.075	1.083
	2^{20}	1.057	1.024
AES	2^{10}	0.337	0.331
	2^{12}	0.367	0.377
	2^{15}	0.378	0.381
	2^{17}	0.381	0.384
	2^{20}	0.368	0.369

5. Conclusions

This paper described experiments and analysis intended to increase awareness of and insight into the processing and energy costs associated with utilizing certain block ciphers on PDAs and other resource-limited devices. Results for both performance, including latency and throughput, and energy consumption were presented for different ciphers applied to plaintext files of different sizes. For the handheld devices used in our experiments,

we found that RC2 is faster than XTEA, which in turn is faster than AES for all files sizes. The relative performance of Blowfish depends on the size of the plaintext file, but, overall, Blowfish is a fast encryption algorithm. Our results also indicate that all ciphers consume a similar amount of power while executing, so faster algorithms consume less energy because they operate at an elevated level of power for less time. The methodology of these experiments can be applied to characterize other encryption algorithms, including stream ciphers, and other devices, for example with other operating systems and processors. Such extensions are left as future work.

In addition to the results of this research, users, software developers, and system designers need to consider a broad range of factors when employing encryption. The experiments reported here consider the performance and energy consumption of encryption algorithms in isolation. Other operations, possibly using other resources, such as sending and receiving data over a wireless interface, playing audio over speakers, displaying video and graphics on a display, and running background applications may have more of an affect on battery life or system performance than encryption.

Acknowledgments

This research was supported in part by a grant from Microsoft Research and, for Creighton Hager, a National Science Foundation Integrative Graduate Education and Research Training (IGERT) grant (award DGE-9987586).

References

- [1] S. Kiaei and S. Devadas, "Which Has Greater Potential Power Impact: High-level Design and Algorithms or Innovative Low Power Technology," in *Proc. 1996 International Symposium on Low Power Electronics and Design*, 1996, pp. 175.
- [2] V. Tiwari, S. Malik, and A. Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software Power Minimization," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 4, no. 2, pp. 437-445, 1994.
- [3] T. Simunic, L. Benini, and G. D. Micheli, "Energy-efficient Design of Battery-powered Embedded Systems," in *Proc. 1999 International Symposium on Low Power Electronics and Design*, 1999, pp. 212-217.
- [4] J. Flinn and M. Satyanarayanan, "PowerScope: A Tool for Profiling the Energy Usage of Mobile Applications," in *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999, pp. 2-10.
- [5] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu, "Analyzing and Modeling Encryption Overhead for Sensor Network Nodes," in *Proc. 2nd ACM International Conference on Wireless Sensor Networks and Applications*, 2003, pp. 151-159.
- [6] P. Dhawan, "Performance Comparison: Security Design Choices - Building Distributed Applications with .NET," Microsoft Developer Network, 2002. Available at <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/bdadotnetarch15.asp>.
- [7] L. Bassham, National Institute of Standards and Technology, "Efficiency Testing of ANSI C Implementations of Round1 Candidate Algorithms for the Advanced Encryption Standard," 1999. Available at <http://csrc.nist.gov/encryption/aes/round1/r1-ansic.pdf>.
- [8] D. S. Wong, H. H. Fuentes, and A. Chan, "The Performance Measurement of Cryptographic Primitives on Palm Devices," in *Proc. 17th Annual Computer Security Applications Conference*, 2001, pp. 92-101.
- [9] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, "Analyzing the Energy Consumption of Security Protocols," in *Proc. International Symposium on Low Power Electronics and Design*, 2003, pp. 30-35.
- [10] "SSL 3.0 Specification," Available at <http://wp.netscape.com/eng/ssl3/>.
- [11] B. Schneier, *Applied Cryptography*, 2nd ed., New York, NY, John Wiley & Sons, 1996.
- [12] T. Dierks and C. Allen, "The TLS Protocol Version 1.0," IETF RFC 2246, 1999.
- [13] A. J. Menezes, P. C. v. Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, 5th ed., Boca Raton, FL, CRC Press, 2001.
- [14] R. Rivest, "A Description of the RC2(r) Encryption Algorithm," IETF RFC 2268, 1998.
- [15] B. Schneier, "Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)," in *Proc. Fast Software Encryption, Cambridge Security Workshop*, 1993, pp. 191-204.
- [16] R. Needham and D. Wheeler, "TEA, a Tiny Encryption Algorithm," in *Proc. Fast Software Encryption, Cambridge Security Workshop*, 1994, pp. 363-366.
- [17] J. Kelsey, B. Schneier, and D. Wagner, "Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X NewDES, RC2, and TEA," *Lecture Notes in Computer Science*, 1334, 1997, pp. 233-246.

[18] R. Needham and D. Wheeler, "Tea Extensions," Technical Report, Computer Laboratory, University of Cambridge, 1997. Available at <http://www.cl.cam.ac.uk/ftp/users/djw3/xtea.ps>.

[19] National Institute of Standards and Technology, U.S. Department of Commerce, "Advanced Encryption Standard (AES)," FIPS PUB 197, 2001. Available at <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

[20] J. Daemen and V. Rijmen, "AES Submission Document on Rijndael," 1998. Available at <http://csrc.nist.gov/CryptoToolkit/aes/rijndael/Rijndael-ammended.pdf>.

[21] G. Anescu, "A C++ Implementation of the Rijndael Encryption/Decryption method," The Code Project, 2002. Available at <http://www.codeproject.com/cpp/aes.asp>.

[22] M. Hahn, "Blowfish.NET 1.01," 2004. Available at <http://www.hotpixel.net/bfnet101.zip>.

[23] pbrooks, "Tiny Encryption Algorithm (TEA) for the Compact Framework," The Code Project, 2004. Available at <http://www.codeproject.com/netcf/teaencryption.asp>.

[24] W. Dai, "Crypto++® Library 5.2.1," 2004. Available at <http://www.eskimo.com/~weidai/cryptlib.html>.

[25] J. D. Meier, S. Vasireddy, A. Babbar, and A. Mackman, "How To: Time Managed Code Using QueryPerformanceCounter and QueryPerformanceFrequency," Microsoft Developer Network, 2004. Available at <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag/html/scalenethowto09.asp>.