6.857 Final Report: Low Power Cryptography

Sam Dorchuck: dorchuck, Vedaant Kukadia: vpk, Daniel Perry: perryd

May 12, 2020

1 Motivation/Applications

Perhaps the case study that most clearly illuminates the importance of cryptography in low power systems is the Sands Casino hack in 2014. In this case, a popular Las Vegas casino was breached through an Internet-connected fish tank in its lobby. Throughout the attack, nearly $40 million worth of data was exfiltrated through the same fish tank.

With growing importance of small communications devices and the internet of things, there has been an increasing need to perform many of the actions of computers with less power. One of these actions that is particularly important is cryptography. For example, one would want an implanted medical device to only be usable by the correct individual.

There are a variety of standard cryptographic implementations for less power constrained applications. However, customization is generally required to build algorithms that work in very low power environments. Using these algorithms necessarily entails trade offs, with lower power consumption tending to lead to lower security.

This paper will first review the general approaches towards low power cryptography. It will then examine the particular methods used for four major cryptographic use cases:

1. Symmetric key cryptography
2. Hashing
3. Public key cryptography

2 General Approaches

There are two main types of approaches to low power cryptography: Hardware approaches and software approaches. both of these methods are usually implemented to achieve maximum performance

2.1 Software

All cryptography procedures must, of course, be implemented in software. There are two primary ways to increase the performance of the software.

First, one can simply improve the implementation of an algorithm. However, as many cryptographic algorithms are already widely used, there is probably less scope to make improvements

based on this method.

Second, one can modify the algorithm itself. The simplest way to do this is to simply shrink the parameters in the algorithm to make it more efficient. For example, the standard symmetric key cipher, AES, has variants ranging from 10 rounds with 128 bit keys to 14 rounds and 256 bit keys.

Another way to improve the efficiency of the algorithm is to modify it to use less computationally expensive operations. For example, using more swap operations in the rounds of the efficient symmetric key ciphers Simon and Speck.

Finally, one can move to a completely different type of mathematics for the algorithm. For example, moving from discrete log to elliptic curve approaches for public key cryptography.

2.2 Hardware

This paper will mainly focus on software-based methods for low power cryptography but hardware also is a big major approach to implement low power cryptography. The main way to do this is by creating application specific integrated circuits that are optimized for the relevant cryptographic algorithm. FPGA's can also be used, but they are utilized much less frequently.

3 Cryptographic Use Cases

3.1 Symmetric Key Cryptography

Symmetric key encryption is the main way information is communicated securely. The current standard method to perform symmetric key encryption is AES. However, even the version of AES with the smallest key size often is two computationally intensive for very small devices.

There are two main algorithms for symmetric key cryptography: the Simon algorithm, and the Speck algorithm. Simon is optimized for hardware implementation, while Speck is optimized for software implementation.

These ciphers work on very limited hardware but do give up throughput, but this is probably acceptable given the smaller amount of communication low power devices will conduct.

They also give up security. The security margin for these algorithms is much tighter than AES. However, there are two reasons to be more comfortable with this reduced security. First, there is the fact that total amount of communication received by the adversary will most likely be substantially lower. Second, since many low power devices are not physically secured, physical attacks (such as side channel attacks) are likely to be a larger concern than advanced cryptoanalytic ones.

		hardware		software		
size	name	area (GE)	throughput (kbps)	flash (bytes)	SRAM (bytes)	throughput (kbps)
128/128	SIMON	1317	22.9	732	0	342
	SPECK	1396	12.1	396	0	768
	AES	2400	56.6	943	33	445

Table 1.1: Performance comparisons. Size is block size/key size; hardware refers to an ASIC implementation, and software to an implementation on an 8-bit microcontroller; clock speeds are 100 kHz (hardware) and 16 MHz (software). The best performance for a given size is indicated in red, the second best in blue. Numbers in brackets are our estimates; "–" means these values were unavailable at the time of writing.

3.2 Hashing

Undoubtedly, hashing is a major cryptographic primitive that is essential for low power cryptography. Over the course of our research, we encountered two major approaches to adapt hashing algorithms to low power environments. First, we explored approaches that optimize hardware for a family of algorithms currently in use on standard computing environments. Next, we examined hashing algorithms that are of an entirely different structure: a software approach of using cheap, computationally un-intensive operations to develop a light-weight hashing algorithm.

Today, NIST recommends the use of SHA-3 for hasing on standard computing environments. SHA-3 is notable for its "sponge construction", where data is absorbed into the sponge and then squeezed out to make the hash. In our research, we discovered a proposed Sponge Hash Algorithm (SHAT) for low-power environments, using a similar construction to SHA-3. SHAT repeatedly applies a Perm algorithm, which in turn applies 48 rounds of a STEP function. To optimize STEP for different hardware designs, factoring in power availability and on-chip space, the STEP hardware circuit can be parallelized and unfolded.

In parallelizing, the circuit's operations are not done sequentially, yielding a larger circuit with more throughput. Similarly, unfolding a STEP round circuit means transforming a 1-STEP round circuit into a 2-STEP round circuit. The new 2-STEP round circuit requires slightly less than twice the on-chip space, consumes significantly more power, but has much higher throughput. The below figure illustrates the tradeoffs between power consumption and throughput for various unfolded circuit sizes:

TABLE 7: Performance results of unrolling steps constructions.

Number of iteration rounds	Area (GE)	Delay (ns)	Power (μW)	Throughput at 10 MHz (Mbps)
48	965	0.94	27.27	6.67
24	1930	1.92	78.34	13.33
16	2895	2.91	131.78	20.00
12	3860	3.91	185.06	26.67
8	5790	5.90	291.77	40.00
6	7720	7.89	398.42	53.33
4	11580	11.87	611.78	80.00
3	15440	15.84	825.06	106.67
2	23160	23.80	1251.70	160.00
1	46320	47.71	2509.70	320.00

With such a design, engineers of low power systems can choose the circuit that works best for their power and space constraints, and optimize hashing throughput. Analysis of the SHAT algorithm reveals that it has significantly higher throughput per unit area than any other major hashing algorithm, illustrating its effectiveness in power and space-constrained systems.

In our research, we encountered a Light-Weight One-Way Cryptographic Hash Algorithm (LOCHA). Interestingly, LOCHA presented an entirely new way of hashing information, relying on computationally cheap operations such as mod and swap. The result is a hashing algorithm requiring significantly fewer clock cycles of computation. Additionally, LOCHA is designed to have a smaller digest so that less information must be communicated by the power-constrained device. As such, LOCHA has fewer bits of security than traditional hash algorithms, but its speed and efficiency make it well suited for low-power environments. The below figure compares LOCHA performance against popular, traditional hashing algorithms:

Table 1. Comparative Performance.

Scheme	Communication Overhead (Hash Digest in bits)	Computation Overhead (clock cycles)	Storage Overhead	
			RAM/ROM	Number of Registers
MD5	128	36360	32 RAM of 32-bit block data and ROM of 2368 bits/296 byte	12 registers of 32 bits
SHA-1	160	84272	32 RAM of 32-bit block data	12 Registers of 32 bits
LOCHA	96	2952	1 ROM of 804 bits and 1 ROM of 970 bits	4 registers of 16 bits, 18 registers of 8 bits

3.3 Public-key Cryptography

Public-key encryption generally requires significant amounts of computing power, and thus has largely been considered too slow for general purpose encryption. As a result, public-key cryptography is mostly used for secure key sharing, for other cryptographic algorithms such as symmetric key encryption. However, as perfect forward secrecy becomes standard in more and more security protocols, and the need to more quickly and efficiently generate keys, as well as the increase in use of low-power decivces including mobile phones and wireless sensor networks, the need for low-power cryptography is growing quickly.

Public-key encryption algorithms are largely based on modular multiplication. This includes RSA, Diffie-Helman, and even DSS. In order to speed up processing, these algorithms often use the Chinese Remainder Theorem, but even then computation may take a while and consume large amounts of power, especially as key sizes grow. Recent advancements in the field of more secure cryptography, and low-power solutions has shed light on the use of elliptic curve cryptography (ECC). Two commonly used curves include curves over $GF(p)$, and curves over $GF(2^n)$, a GF over plynomials of size n, with the latter having the benefit of computations without carries.

Recent advances in ECC have shed light on comparable levels of security to other widely used public-key algorithms, while using smaller key sizes. This is especially important for devices with limited size and computational power, as smaller key sizes mean savings in memory, bandwidth, and computational savings. It has been shown that RSA with a 1024-bit key size provides the same level of security as ECC with a 160-bit key size. Following current recommendations for data encryption past 2010, this results in RSA with 2048-bit key size, or equivalently ECC with a 224-bit key size.

Wireless sensor networks (WSN) are increasingly being used in a variety of different applications, including health monitoring, industrial control, user authentication, etc. Because of the limited power availability and constrained size of the devices, public key cryptography is often deemed too computationally heavy for encryption tasks and so these devices primarily rely on symmetric key encryption. Much of the current landscape on low power public key cryptography explores using new advancements in low power PKC within WSN. To further explore this, Wander et al. compared the energy consumption from RSA and ECC. The table below summarizes the results.

Algorithm	Signature		Key Exchange	
	Sign	Verify	Client	Server
RSA-1024	304	11.9	15.4	304
ECDSA-160	22.82	45.09[3]	22.3	22.3
RSA-2048	2302.7	53.7	57.2	2302.7
ECDSA-224	61.54	121.98[3]	60.4	60.4

Table 2: Energy cost of digital signature and key exchange computations [mJ].

While we see that ECDSA signatures are far cheaper than RSA signatures, RSA verifications slightly outperform ECDSA verifications. The important thing to note here, however, is the rate at which energy consumption increases when moving between the two levels of security (RSA-1024 and ECDSA-160) to (RSA-2048 and ECDSA-224). Another thing to note is is that these ECDSA calculations assume two point multiplications, however, there exist optimizations to reduce this but

with the tradeoff of more memory.

Overall, it has been shown that PKC approaches to WSN are a promising alternative, and the use of ECC can lead to significant power savings. In addition to power savings, ECC has also been shown to decrease public-key communication costs and thus should be a continued area of research.

4 Conclusion

In conclusion this paper has examined the different approaches taken towards low power cryptography. Moreover, this paper has examined the specific low power algorithms used for 3 key applications. In terms of applications in real world cryptography, each of these applications are interconnected. As the use of mobile devices and wireless sensor networks increases, the need to transmit secure information and preserve privacy will greatly increase. Additionally, as security standards increase, whether it be by increasing key sizes, or by requiring perfect forward secrecy to be standard in all security protocols, the need for low power cryptography is becomes highly relevant.

5 Bibliography

References

[1] Alvarez, Rafael, Caballero Cándido, Juan Santonja, and Antonio Zamora. *"Algorithms for Lightweight Key Exchange."* MDPI. Multidisciplinary Digital Publishing Institute, June 27, 2017.

https://www.mdpi.com/1424-8220/17/7/1517/htm.

[2] Beaulieu, Ray, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. *"The Simon and Speck Families of Lightweight Block Ciphers."* National Security Agency, June 13, 2013.

https://eprint.iacr.org/2013/404.pdf.

[3] Chowdhury, Amrita Roy, Tanusree Chatterjee, and Sipra DasBit. *"LOCHA: A Light-Weight One-Way Cryptographic Hash Algorithm for Wireless Sensor Network."* Procedia Computer Science. Elsevier, June 5, 2014.

https://www.sciencedirect.com/science/article/pii/S187705091400653X.

[4] Gaubatz, Gunnar, Jens-Peter Kaps, Erdinc Ozturk, and Berk Sunar. *"State of the Art in Ultra-Low Power Public Key Cryptography for Wireless Sensor Networks."* Cryptography and Information Security Lab, Worcester Polytechnic Institute, March 8, 2005.

https://ieeexplore.ieee.org/document/1392819.

[5] Jones, Scott. *"What You Need to Know About SHA-3 for Embedded System Security."* StackPath, May 20, 2019.

https://www.electronicdesign.com/technologies/embedded-revolution/article/21808025/what-you-need-to-know-about-sha3-for-embedded-system-security.

[6] Kitsos, Oufopavlou, Selimis, and Sklavos. *"Low Power Cryptography."* Journal of Physics: Conference Series 10, 2005.

https://iopscience.iop.org/article/10.1088/1742-6596/10/1/084/pdf.

[7] Wander, A S, N Gura, H Eberle, V Gupta, and S C Shantz. *"Energy Analysis of Public-Key Cryptography for Wireless Sensor Networks."* Third IEEE International Conference on Pervasive Computing and Communications, March 8, 2005.

https://ieeexplore.ieee.org/document/1392772.

[8] Zhang, Yunlong, Joohee Kim, Ken Choi, and Taeshik Shon. *"High Performance and Low Power Hardware Implementation for Cryptographic Hash Functions."* SAGE Journals, March 1, 2014.

https://journals.sagepub.com/doi/full/10.1155/2014/736312.