NaCl on 8-Bit AVR Microcontrollers*

Michael Hutter[1] and Peter Schwabe[2]

[1] Graz University of Technology
Institute for Applied Information Processing and Communications (IAIK)
Inffeldgasse 16a, 8010, Graz, Austria
michael.hutter@iaik.tugraz.at
[2] Radboud University Nijmegen
Digital Security Group
P.O. Box 9010, 6500GL Nijmegen, The Netherlands
peter@cryptojedi.org

Abstract. This paper presents first results of the Networking and Cryptography library (NaCl) on the 8-bit AVR family of microcontrollers. We show that NaCl, which has so far been optimized mainly for different desktop and server platforms, is feasible on resource-constrained devices while being very fast and memory efficient. Our implementation shows that encryption using Salsa20 requires 268 cycles/byte, authentication using Poly1305 needs 195 cycles/byte, a Curve25519 scalar multiplication needs 22 791 579 cycles, signing of data using Ed25519 needs 23 216 241 cycles, and verification can be done within 32 634 713 cycles. All implemented primitives provide at least 128-bit security, run in constant time, do not use secret-data-dependent branch conditions, and are open to the public domain (no usage restrictions).

Keywords: Elliptic-curve cryptography, Edwards curves, Curve25519, Ed25519, Salsa20, Poly1305, AVR, ATmega.

1 Introduction

This paper describes implementations of the Networking and Cryptography library (NaCl) [4] on 8-bit AVR microcontrollers. More specifically, we describe two different approaches, one aiming at higher speed, one aiming at smaller memory requirements, of porting NaCl to the AVR ATmega family of microcontrollers. The aim of the high-speed implementation is not to achieve the highest possible speed at all (memory-)costs for all primitives. Similarly, the aim of the low-memory implementation is not to obtain the smallest possible footprint without any performance considerations. The two implementations are rather

* This work was supported by the Austrian Science Found (FWF) under the grant number TRP251-N23. Part of this work was done while Peter Schwabe was employed by the Research Center for Information Technology Innovation, Academia Sinica, Taiwan. Permanent ID of this document: cd4aad485407c33ece17e509622eb554. Date: April 1, 2013

A. Youssef, A. Nitaj, A.E. Hassanien (Eds.): AFRICACRYPT 2013, LNCS 7918, pp. 156–172, 2013.

two example tradeoffs between speed and memory footprint that we consider reasonable and useful for various applications and different microcontrollers in the ATmega family.

Previous NaCl optimization focused on large general-purpose server and desktop CPUs; the "smallest" architecture targeted by previous NaCl optimization is ARMv7 CPUs with the NEON vector-instruction set [10]. Despite this focus on large processors, the NaCl designers claim in [4, Section 4] that

> "all of the cryptographic primitives in NaCl can fit onto much smaller CPUs: there are no requirements for large tables or complicated code"

This paper shows that this claim is actually correct.

The cryptographic primitives used by default in NaCl to provide public-key authenticated encryption are the Curve25519 elliptic-curve Diffie-Hellman key-exchange protocol [2], the Poly1305 authenticator [5], and the Salsa20 stream cipher [3]. The designers of NaCl announced, that the next release of NaCl will use the Ed25519 elliptic-curve signature scheme [7,8] to provide cryptographic signatures. This signature scheme—as described in the original paper and as implemented in this paper—uses the SHA-512 hash function [28].

We will put all software described in this paper into the public domain to maximize reusability of our results[1]. We will furthermore discuss possibilities for public benchmarking with the editors of eBACS [9] and XBX [35]. Currently eBACS does not support benchmarking on AVR microcontrollers; XBX only supports benchmarking of hash functions.

Main contribution. There exists an extensive literature describing implementations of cryptographic *primitives* on AVR microcontrollers and other embedded processors. Some of them have been integrated into libraries that offer a set of cryptographic functionalities, e.g., AVR-Crypto-Lib [15], TinyECC [24], NanoECC [32], or the AVR Cryptolibrary from Efton s.r.o. [13]. These libraries are specifically tailored to match the specific restricted environment of the AVR.

This paper is the first to describe implementations of the entire NaCl library on AVR microcontrollers. These include the cryptographic primitives Salsa20 [3], Poly1305 [5], Curve25519 [2], and Ed25519 [8]. All primitives are based—in contrast to existing AVR libraries—on at least 128-bit security and provide new speed records for that level of security. In addition, all functions run in constant time and do not contain secret-data-dependent branch conditions. This is important to provide a certain level of security against basic implementation attacks [22,25]. In particular the implementation is protected against *remote* side-channel attacks. Other cryptographic libraries for AVR do not address this issue. Moreover, the entire library is very small in size and requires only 17366 bytes of code, no static RAM, and less than 1350 bytes of stack memory; it therefore fits into very resource-constrained devices such as the very small ATmega family of microcontrollers, e.g., the ATmega32, ATmega328, and ATmega324A. Last but not least, we present new speed records for Salsa20 on AVRs and give first

[1] The software is available online at `http://cryptojedi.org/crypto/#avrnacl`