

Faster Binary Curve Software: A Case Study

Billy Bob Brumley^(✉)

Department of Pervasive Computing,
Tampere University of Technology, Tampere, Finland
`billy.brumley@tut.fi`

Abstract. For decades, elliptic curves over binary fields appear in numerous standards including those mandated by NIST, SECG, and ANSI X9.62. Many popular security protocols such as TLS explicitly support these named curves, along with implementations of those protocols such as OpenSSL and NSS. Over the past few years, research in improving the performance and/or security of these named curve implementations has pushed forward the state-of-the-art: e.g. projective lambda coordinates (Oliveira et al.) and commodity microprocessors featuring carryless multiplication instructions for native polynomial arithmetic (Intel, ARM, Qualcomm). This work aggregates some of these new techniques as well as classical ones to bring an existing library closer to the state-of-the-art. Using OpenSSL as a case study to establish the practical impact of these techniques on real systems, results show significant performance improvements while at the same time adhering to the existing software architecture.

Keywords: Applied cryptography · Public key cryptography · Elliptic Curve Cryptography · OpenSSL

1 Introduction

Of the many types of public key cryptography available, elliptic curve cryptography (ECC) offers many attractive advantages – the main being the small size of private and public keys. Furthermore, since the introduction in the 1990s ECC has undergone extensive standardization – including NIST [20, D.1.3], SECG, and ANSI X9.62. Generally, these standardized curves come in two flavors:

- Elliptic curves over prime fields \mathbb{F}_p ;
- Elliptic curves over binary fields \mathbb{F}_{2^m} .

For elliptic curves in software, elliptic curves over \mathbb{F}_p are a more popular choice for performance reasons because the finite field arithmetic is easier to implement efficiently since most microprocessors feature native integer multiplication instructions as a building block for multi-precision arithmetic. Recognizing this

Supported in part by TEKES grant 3772/31/2014 Cyber Trust and COST Action IC1306.

practical limitation, academic efforts for high speed ECC placed more research efforts on elliptic curves over \mathbb{F}_p .

For elliptic curves over \mathbb{F}_{2^m} , historically software needed to revert to primitive table lookup methods for finite field arithmetic since it was uncommon (to say the least) to feature a native polynomial multiplication instruction. However, this trend shifted roughly five years ago when chip makers such as Intel, ARM, and Qualcomm started introducing such instructions into the Instruction Set Architecture (ISA) for commodity microprocessors. This left a gap in research for high speed ECC software for elliptic curves over \mathbb{F}_{2^m} – for example, there were really no major innovations in projective coordinate systems since López and Dahab in 1999 [17]. Oliveira et al. changed that recently in 2014 with λ -projective coordinates [21].

This gap in academic research also left a gap in practical elliptic curve software libraries. The best example of this is OpenSSL, which – since introducing ECC support in 2005 – has seen no new optimizations for elliptic curves over \mathbb{F}_{2^m} , despite heavy optimizations for elliptic curves over \mathbb{F}_p .

The goal of this paper is to fill that gap and measure the real world impact of these new optimizations for elliptic curves over \mathbb{F}_{2^m} . The resulting OpenSSL source code patches yield performance improvements that remarkably approach 6-fold in some cases. Section 2 gives background on binary elliptic curves and discusses various coordinate systems. Section 3 gives an overview of the ECC software architecture within OpenSSL. Section 4 discusses the optimizations implemented in this paper, and gives benchmarking results. Section 5 draws conclusions.

2 Binary Elliptic Curves

For a finite field \mathbb{F}_{2^m} , fix curve coefficients $a_2, a_6 \in \mathbb{F}_{2^m}$ and all of the (x, y) solutions to the equation

$$E : y^2 + xy = x^3 + a_2x^2 + a_6$$

over \mathbb{F}_{2^m} for $x, y \in \mathbb{F}_{2^m}$ along with the identity element (∞ , point at infinity) form a finite Abelian group relevant to applied cryptography. The majority of standardized curves of this form further restrict the values that curve coefficients a_2 and a_6 can take for efficiency reasons. Generally, there are two major types.

Pseudo-Random Curves. These curves fix $a_2 = 1$ and $a_6 \in \mathbb{F}_{2^m}$ derived pseudo-randomly. Curves of this type include, but are not limited to: B-163, B-233, B-283, B-409, and B-571.

Koblitz Curves. These curves [15] fix $a_2 \in \mathbb{F}_2$ and $a_6 = 1$. Curves of this type include, but are not limited to: K-163, K-233, K-283, K-409, K-571, and sect239k1.