Automatic Generation of Cryptographic Code

Chapter Binary Elliptic Curves

Daniele Canavese, **Emanuele Cesena**
(ongoing work with Roberto M. Avanzi, Marco Pedicini)

Computer and Network Security Group
Dip. di Automatica e Informatica
Politecnico di Torino

CONCERTO (Torino) – 09-11/06/2010

Motivation (Cryptography)

"Who wants his (crypto) code to live forever" – *Queen* ;-)

Implementation is a crucial task in applied cryptography

- Good: rich mathematical structure, elegant description
- Bad: often difficult to achieve elegant code
  - E.g. optimizations (sometimes just for minimum functionality)

Motivation (Cryptography)

"Who wants his (crypto) code to live forever" – *Queen* ;-)

Implementation is a crucial task in applied cryptography

- Good: rich mathematical structure, elegant description
- Bad: often difficult to achieve elegant code
  - E.g. optimizations (sometimes just for minimum functionality)

Need for a new approach (a new programming language?)...

This talk stops here!

"After geometry and algebra there is logic, of course"

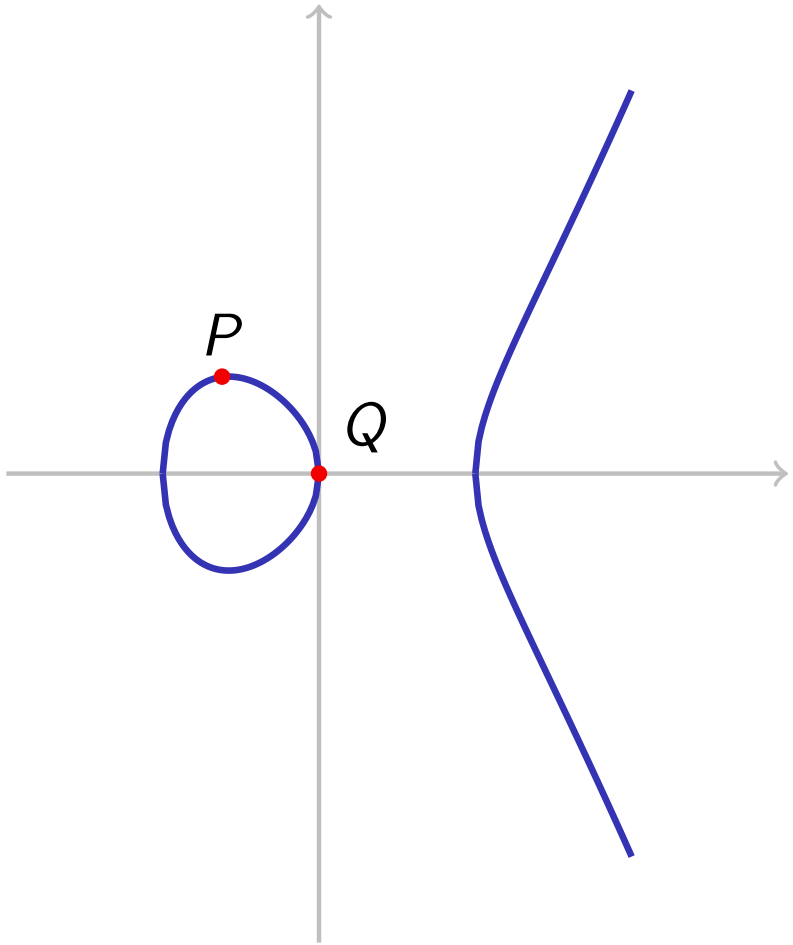
- ...can we embed properties directly into such a language?
- e.g. only poly-time (crypto) algorithms can be written?
- e.g. our optimizations do not "conflicts" with programs?

Outline

1. Elliptic Curve Cryptography
2. Automatic Generation of Cryptographic Code
3. Experimental Results
4. Conclusion

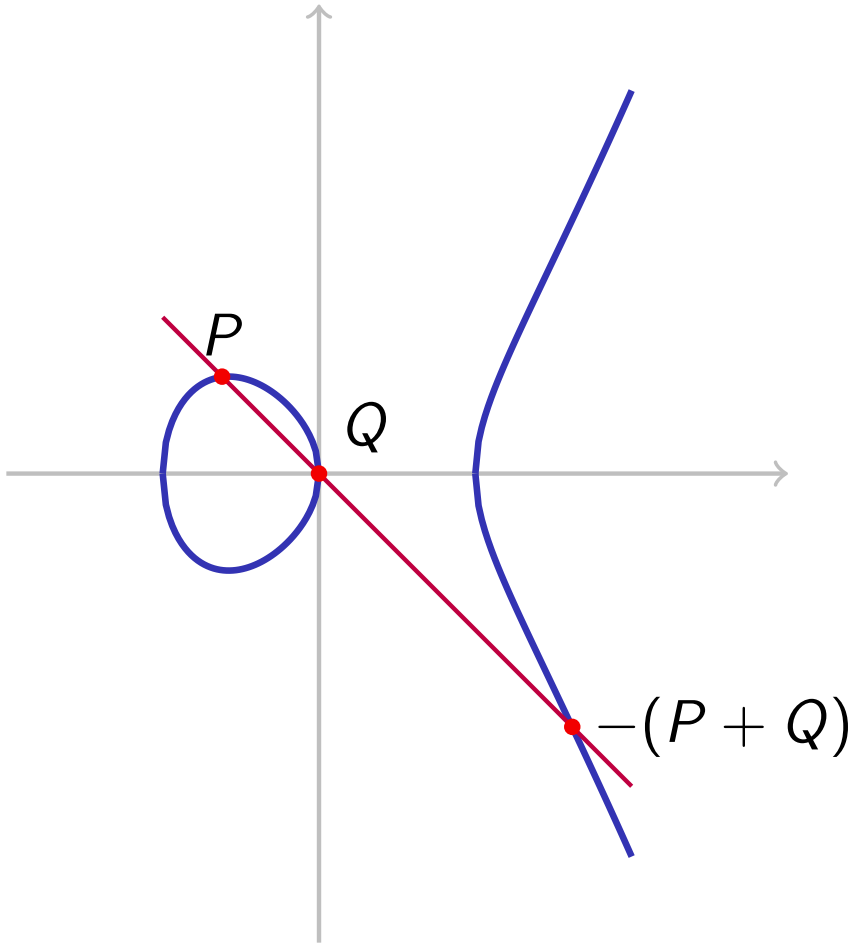
Elliptic Curve

$$y^2 = x^3 + ax + b \ , \qquad a, b \in \mathbb{F}$$



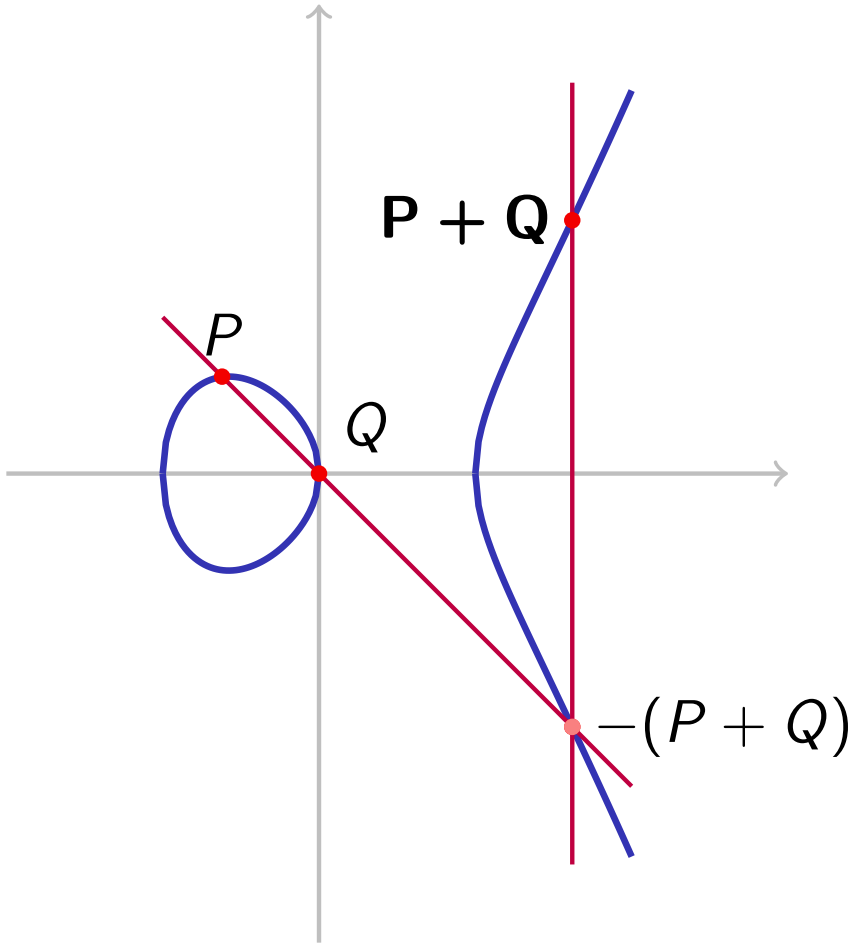
Elliptic Curve

$$y^2 = x^3 + ax + b \ , \qquad a, b \in \mathbb{F}$$



Elliptic Curve

$$y^2 = x^3 + ax + b \ , \qquad a, b \in \mathbb{F}$$



Elliptic Curve Cryptography

Discrete Logarithm Problem

Compute n, given points $P,\ nP \in E(\mathbb{F}_q)$

Elliptic curve in cryptography

- Miller and Koblitz (independently) in 1985
- Smaller key size w.r.t. finite fields for comparable security

Cryptographic Primitives based on DLP

- Diffie-Hellman Key Agreement
- ElGamal Encryption
- Digital Signature Algorithm
- ...

Issues implementing ECC

Choice of the best algorithm

- Coordinate system (Affine, Projective, López-Dahab)
- Scalar representation (NAF, w-NAF)
- Effect: many versions for the same function

Optimizations

- Routine specific
- Cross-layer
- Platform specific
- Effect: code unreadable

Evolution

- New hardware
- New ideas

Requirements

Main objective

Automatic generation of a greatly optimized crypto library

Requirements for the framework

- Focus on binary ECC
- Extensibility, code writing close to mathematical language

Requirements for the code generated

- Efficiency (time & space), Security, Portability...

Requirements

Main objective

Automatic generation of a greatly optimized crypto library

Requirements for the framework

- Focus on binary ECC
- Extensibility, code writing close to mathematical language

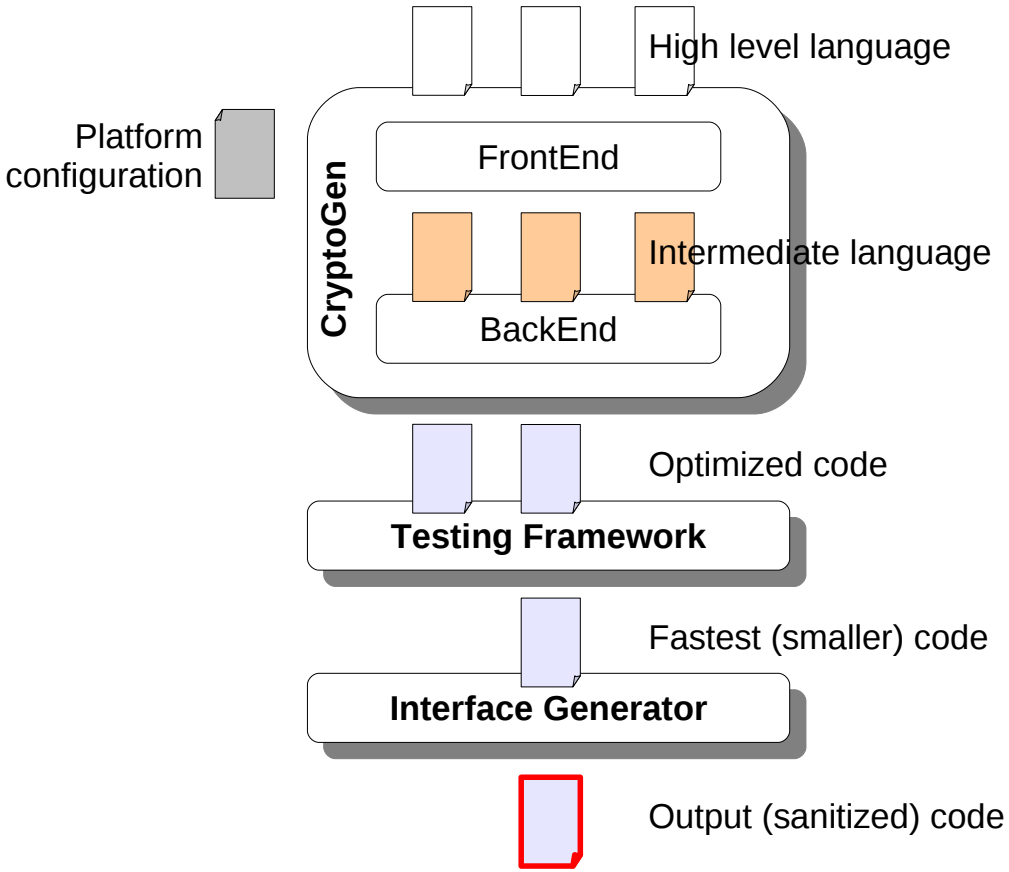
Requirements for the code generated

- Efficiency (time & space), Security, Portability...

Assumption

We TRUST the compiler

Architecture



CryptoGen (FrontEnd)

FrontEnd

Translates from high-level language to intermediate language

Plug-in mechanism to implement new algorithms

- Plug-ins are dynamic libraries (currently written in C++)
- Plug-in programmer may interact with FE/BE to improve results

Currently implemented algorithms

- Arithmetic in binary fields
  - Add, Mul (Comb, Karatsuba), Sqr, ...
- Arithmentic over binary elliptic curves
  - Add, Neg, Dbl (Affine, Projective, López-Dahab)
  - Scalar multiplication (Binary, NAF, w-NAF)

CryptoGen (BackEnd)

BackEnd

Performs optimizations and translates in C/C++

Optimizations

- Hardware-independent
- Hardware-dependent
  - Preliminary choice of better algorithms
  - Actual instantiation of BitVector (loop unrolling, scheduling...)

Current intermediate language

- Three-address code
- Inner/Outer types, RoutineCall, ControlStructure (if, while)
- Knowledge of BitVector, FieldElement, ECPoint types, operators and properties

Testing Framework and Interface Generator

Testing Framework

Times alternative versions of a function, selecting the fastest one

- Support for external code, i.e. not generated by CryptoGen
- Compatible with the ECRYPT Benchmarking of Cryptographic Systems

Testing Framework and Interface Generator

Testing Framework

Times alternative versions of a function, selecting the fastest one

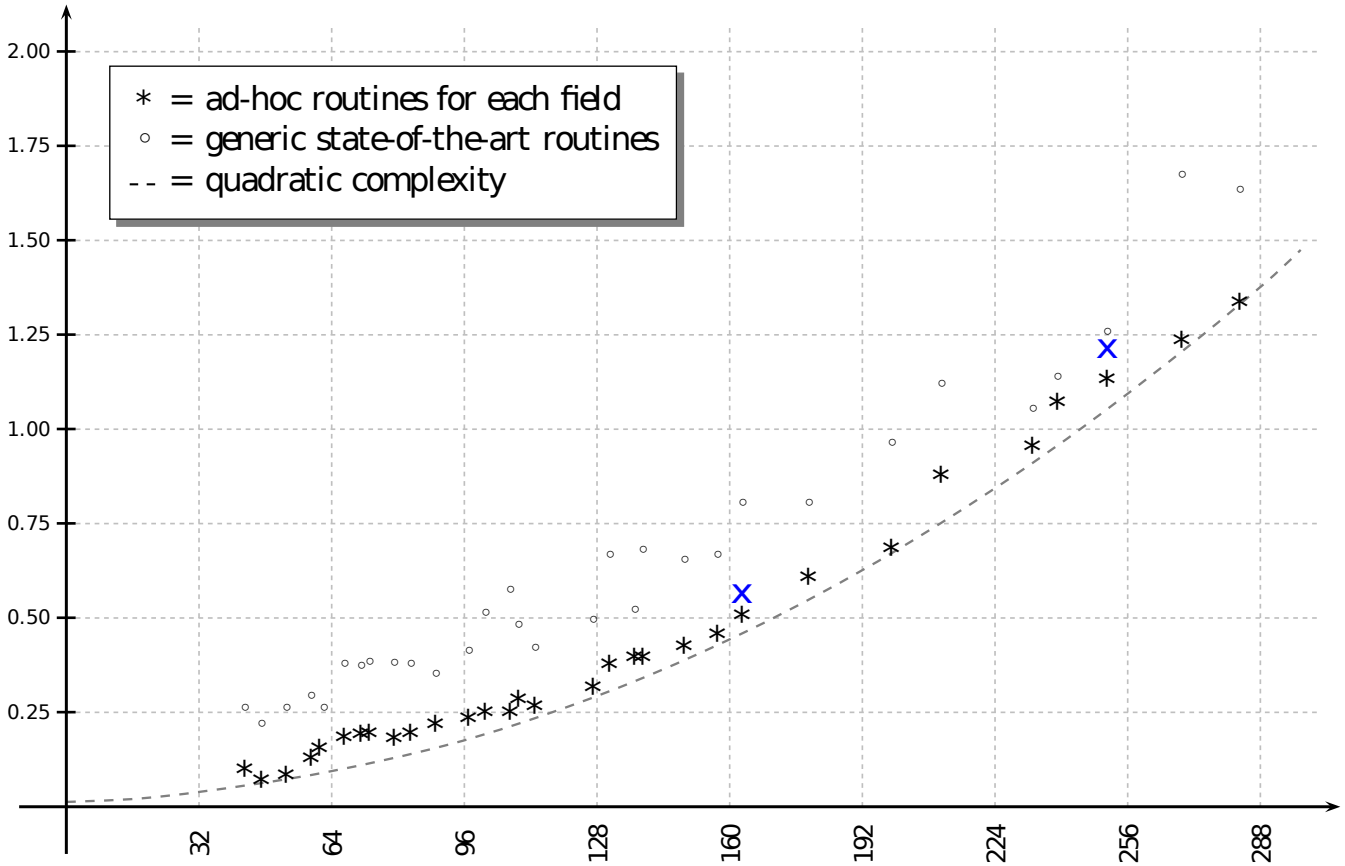
- Support for external code, i.e. not generated by CryptoGen
- Compatible with the ECRYPT Benchmarking of Cryptographic Systems

Interface Generator

Enrich the interface of a function, enhancing security

- Sanitization
- Information hiding

Finite Fields



Data from R. M. Avanzi, N. Thériault, Effects of Optimizations for Software Implementations of Small Binary Field Arithmetic, WAIFI 2007

Future Works

Enhancement to CryptoGen

- Design of a specific language for ECC (DLP-based? asymmetric? cryptography)
- Support for multiple high-level languages
- Provide code annotation
- Improve optimization techniques

Add new cryptographic primitives

- Pairing-based cryptography
- ECC over prime fields
- Symmetric cryptography

Conclusion

Automatic generation of a greatly optimized crypto library

- CryptoGen
- Testing Framework
- Interface Generator

Implementation of binary ECC

- Binary fields
- Elliptic curves
- Scalar multiplication

Performance close to hand-written hand-optimized code

- Higher maintainability
- Some "tricks" still to be added

Thank you for your attention! Any questions?