DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Implementation of Attribute-Based Encryption in Rust on ARM Cortex M Processors

Daniel Bücheler

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Implementation of Attribute-Based Encryption in Rust on ARM Cortex M Processors

Implementierung von Attributbasierter Verschlüsselung in Rust auf ARM Cortex M Prozessoren

Author:	Daniel Bücheler
Supervisor:	Prof. Dr. Claudia Eckert
Advisor:	Stefan Hristozov
Submission Date:	15.04.2021

I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 15.04.2021 Daniel Bücheler

Acknowledgments

Abstract

Contents

1 Introduction

1.1 Section

Citation test [**latex**].

1.1.1 Subsection

See Table 1.1, Figure 1.1, Figure 1.2, Figure 1.3.

Table 1.1: An example for a simple table.

A	B	C	D
1	2	1	2
2	3	2	3

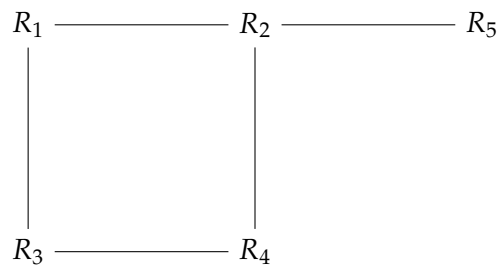


Figure 1.1: An example for a simple drawing.

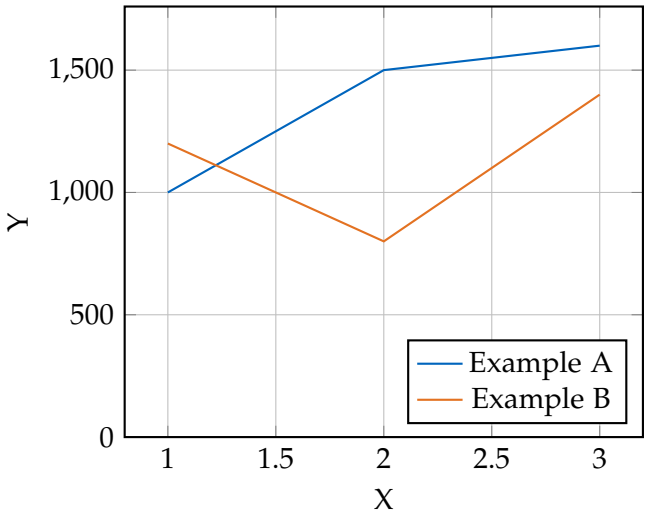


Figure 1.2: An example for a simple plot.

```
SELECT * FROM tbl WHERE tbl.str = "str"
```

Figure 1.3: An example for a source code listing.

2 Introduction to Mathematical Foundations

This chapter shall provide a high-level introduction to the cryptographical and mathematical tools used to implement attribute-based encryption in this thesis. For further reference, please refer to

2.1 Classic Symmetric and Asymmetric Cryptography

Today's conventional cryptography knows two main classes of cryptosystems: *Symmetric* or *Private-Key systems* and *Asymmetric* or *Public-Key systems*. The main difference lies in their use of encryption and decryption keys:

In *symmetric* systems, the key used for encryption and decryption is identical. That is, a user *Alice* encrypting a message to send to another user *Bob* will encrypt the message using a key k that she had agreed on with Bob beforehand. When he receives the encrypted message, Bob will use the same key k to decrypt it.

In *asymmetric* systems, on the other hand, the keys used for encryption and decryption differ. When Alice encrypts a message with key k_{enc}, she will not be able to encrypt it again. Instead, when Bob receives the encrypted message, he will use a different key k_{dec} to decrypt it. Thus, in symmetric systems, keys always come in *pairs* of an *encryption key* k_{enc} and a *decryption key* k_{dec}. Because k_{enc} can not be used to decrypt messages meant for Bob, there is no harm to making it publicly available. For example, he might put it up on his website for anyone wishing to send him an encrypted message to download. This is why k_{enc} is also often called the *public key* and k_{dec} the *private key*.

Asymmetric cryptosystems make secure communication among a large group of participants much easier: Consider n participants wanting to communicate securely using a symmetric system. Each participant would need to share a unique secret key with each of the other participants, requiring a total of $\frac{n(n-1)}{2}$ keys. In the asymmetric setting, one key per participant is sufficient: The same public key may be shared with the whole group, as the private key remains private anyway. This reduces the total number of keys to n.

Another problem remains, however: Encrypting a single message to a large number of participants remains requires encrypting it with everyone's public key separately.

For a large number of recipients, this gets very expensive in terms of computational effort and storage space. So, for example, to encrypt a message for all students of a certain university, we'd need to obtain each student's private key and encrypt the message with each key separately.

Even worse, what if we want to encrypt data for any participant that has a certain characteristic, even if they haven't joined the system yet (i.e. there is no public key for them yet)?

2.2 Elliptic Curves

The mathematics of modern cryptosystems (including, but not limited to ABE) work on a great variety of mathematical structures, and elliptic curves are just one of them. They have become very popular since their discovery by *citation needed* in *year needed* because they provide an equivalent security level at shorter key length and smaller computational cost than other systems (e.g. based on RSA) **katz_introduction_2015**.

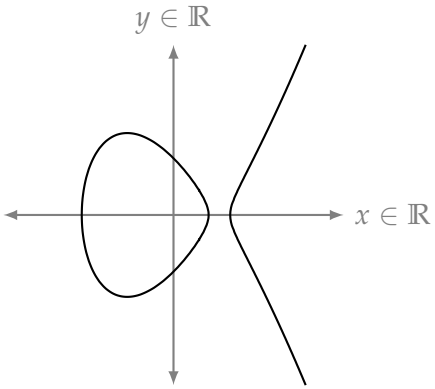
Elliptic curves (over finite fields) are defined by equations of the following form, where $p \leq 5$ prime: **katz_introduction_2015**

$$y^2 = x^3 + ax + b \pmod{p} \tag{2.1}$$

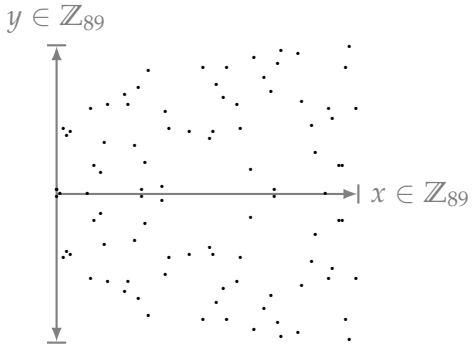
a and b are the curve parameters, and it is required that $4a^2 + 27b^2 \neq 0 mod p$.**katz_introduction_2015**

Then, the elliptic curve $E(\mathbb{Z}_p)$ is the set of coordinates $(x, y) \in \mathbb{Z}_p^2$ that satisfies the equation above. A special value $\mathcal{O}$ to denote the special *point at infinity*. **katz_introduction_2015**

$$E(\mathbb{Z}_p) := \{(x, y) | x, y \in \mathbb{Z}_p \text{ and } y^2 = x^3 + ax + b \bmod b\} \cup \{\mathcal{O}\} \tag{2.2}$$



$y^2 = x^3 - 2x + 1$ over $\mathbb{R}$



$y^2 = x^3 - 2x + 1$ over $\mathbb{Z}_{89}$

List of Figures

List of Tables