Exploring Password Authenticated Key Exchange Algorithms

Final Year Project Screencast

Sam Leonard

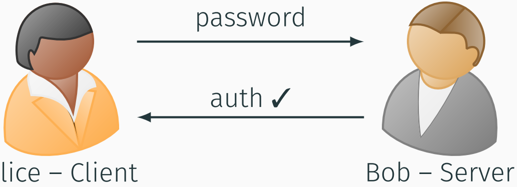
Supervisor: Bernardo Magri

Table of contents

1. Introduction
2. Context
3. Demo

Intro

Motivation



Traditional Authentication

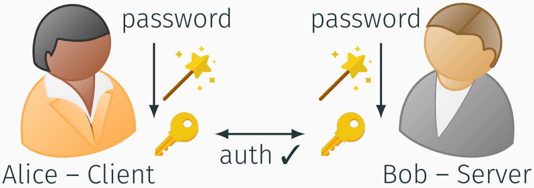
Motivation

PAKEs are a radically different solution to this problem.

- the password never leaves a user's device
- an eavesdropper cannot learn enough information to attack the protocol
- both the server and client are authenticated with each other

Context

What are PAKEs?

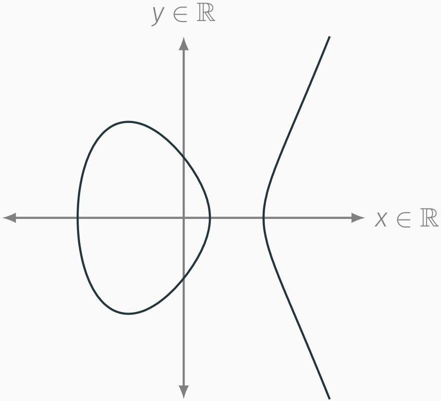


Example Balanced PAKE

SPAKE2

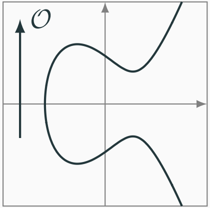
Alice		Bob
$x \overset{\$}{\leftarrow} \mathbb{Z}_p$		$y \overset{\$}{\leftarrow} \mathbb{Z}_p$
$X \leftarrow g^x$		$Y \leftarrow g^y$
$X^* \leftarrow X \cdot M^{pw}$		$Y^* \leftarrow X \cdot N^{pw}$
	$\xrightarrow{\quad X^* \quad}$	
	$\xleftarrow{\quad Y^* \quad}$	
$K_A \leftarrow (Y^*/N^{pw})^x$		$K_B \leftarrow (X^*/M^{pw})^y$
$SK_A \leftarrow H(A, B, X^*, Y^*, Ka)$		$SK_B \leftarrow H(A, B, X^*, Y^*, Kb)$

Elliptic Curves

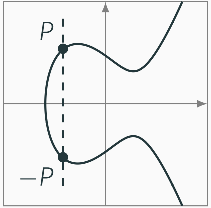


$y^2 = x^3 - 2x - 1$ over $\mathbb{R}$

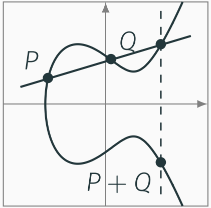
Point addition



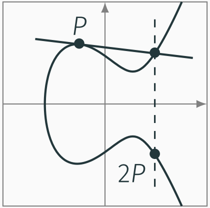
Neutral element $\mathcal{O}$



Inverse element $-P$



Addition $P+Q$
“Chord rule”



Doubling $P+P$
“Tangent rule”

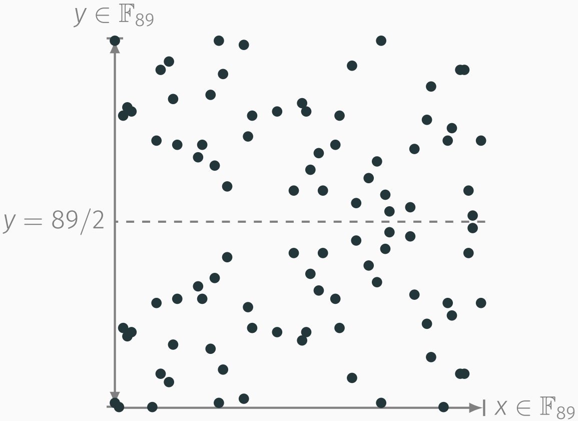
Finite Fields

Computers cannot represent the real numbers.

Instead a finite set must be chosen instead.
The Finite Field of integers mod some prime p is used instead of the reals.

This is notated $\mathbb{F}_p$, GF(p).

Elliptic Curves Over Finite Fields



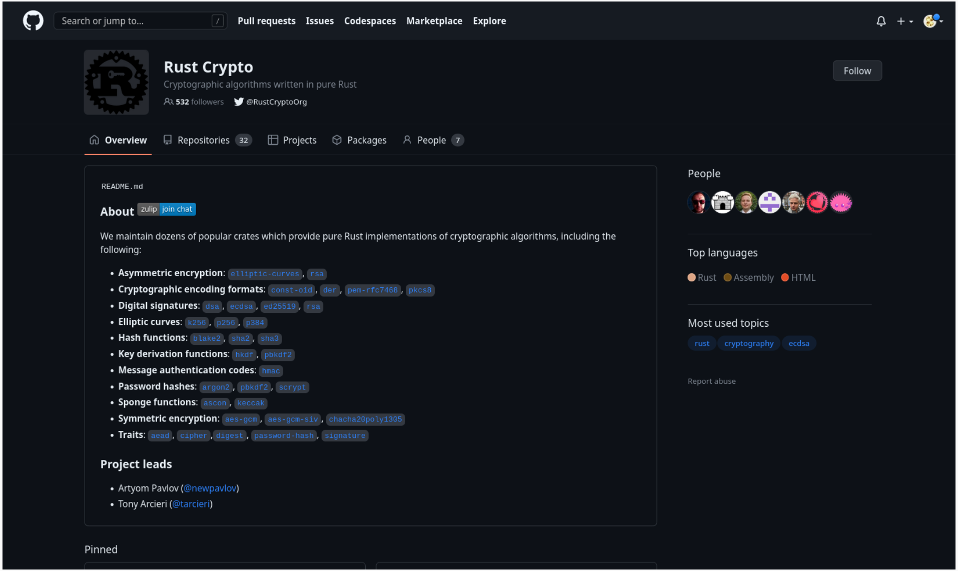
$y^2 = x^3 - 2x + 1$ over $\mathbb{F}_{89}$

AuCPace

An Augmented PAKE designed for the Industrial Internet of Things (IIOT).

- Proved secure in the Universal Composability framework
- Optimised to run efficiently on small microcontrollers
- Three variants to allow users to adapt the protocol to their setting:
    - Strong AuCPace – provides pre-computation resistance by blinding the salt value
    - Partial Augmentation – server stores a long term keypair for each user
    - Implicit Mutual Authentication – removes a round of messages

RustCrypto



Rust Crypto

Cryptographic algorithms written in pure Rust

532 followers @RustCryptoOrg

README.md

About zulip join chat

We maintain dozens of popular crates which provide pure Rust implementations of cryptographic algorithms, including the following:

- **Asymmetric encryption:** `elliptic-curves`, `rsa`
- **Cryptographic encoding formats:** `const-oid`, `der`, `pem-rfc7468`, `pkcs8`
- **Digital signatures:** `dsa`, `ecdsa`, `ed25519`, `rsa`
- **Elliptic curves:** `k256`, `p256`, `p384`
- **Hash functions:** `blake2`, `sha2`, `sha3`
- **Key derivation functions:** `hkdf`, `pbkdf2`
- **Message authentication codes:** `hmac`
- **Password hashes:** `argon2`, `pbkdf2`, `scrypt`
- **Sponge functions:** `ascon`, `keccak`
- **Symmetric encryption:** `aes-gcm`, `aes-gcm-siv`, `chacha20poly1305`
- **Traits:** `aead`, `cipher`, `digest`, `password-hash`, `signature`

Project leads

- Artyom Pavlov (@newpavlov)
- Tony Arcieri (@tarcieri)

Pinned

People

Top languages

● Rust ● Assembly ● HTML

Most used topics

rust cryptography ecdsa

Report abuse

Demo

Conclusion

I have implemented the AuCPace protocol and all it’s variants in an ergonomic Rust libary. The library has been open-sourced through the RustCrypto.

Thank you for watching!