Using Cryptol to explore NIST's Dual_EC_DRNG

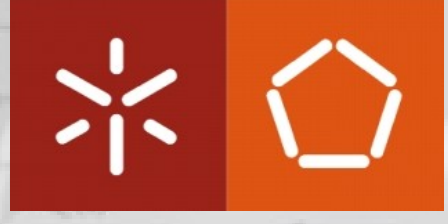
Cohesive Project - FMSE 13/14

Ana Carvalho
pg25335@alunos.uminho.pt

Catarina Correia
pg19643@alunos.uminho.pt

J.N. Oliveira

J. Hendrix



3 July 2014

Project Proposal

Main goals:

- Use **Cryptol** to explore **random number generators** with a backdoor based on **elliptic curves**.

Project Proposal

Main goals:

- Use **Cryptol** to explore **random number generators** with a backdoor based on **elliptic curves**.
- Show how the internal state can be recovered after observing a few bits of output.

We will get to the **interesting** stuff in just a minute.

But first...

theguardian

NSA FILES: *DECODED*

What the revelations mean for you.

N

NIST works to publish the strongest cryptographic standards possible, and uses a transparent, public process to rigorously vet its standards and guidelines. If vulnerabilities are found, NIST works with the cryptographic community to address them as quickly as possible.

In light of the concerns expressed regarding Dual_EC_DRBG, ITL is taking the following actions:

Recommending against the use of SP 800-90A Dual Elliptic Curve Deterministic Random Bit Generation: NIST strongly recommends that, pending the resolution of the security concerns and the re-issuance of SP 800-90A, the Dual_EC_DRBG, as specified in the January 2012 version of SP 800-90A, no longer be used.

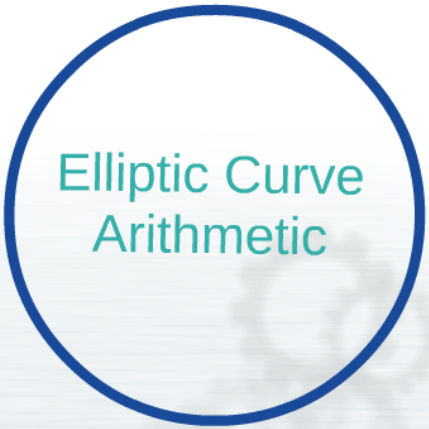
NIST works to publish the strongest cryptographic standards possible, and uses a transparent, public process to rigorously vet its standards and guidelines. If vulnerabilities are found, NIST works with the cryptographic community to address them as quickly as possible.

In light of the concerns expressed regarding Dual_EC_DRBG, ITL is taking the following actions:

Recommending against the use of SP 800-90A Dual Elliptic Curve Deterministic Random Bit Generation: NIST strongly recommends that, pending the resolution of the security concerns and the re-issuance of SP 800-90A, the Dual_EC_DRBG, as specified in the January 2012 version of SP 800-90A, no longer be used.

Back to the **technicalities**

Evolution Diagram

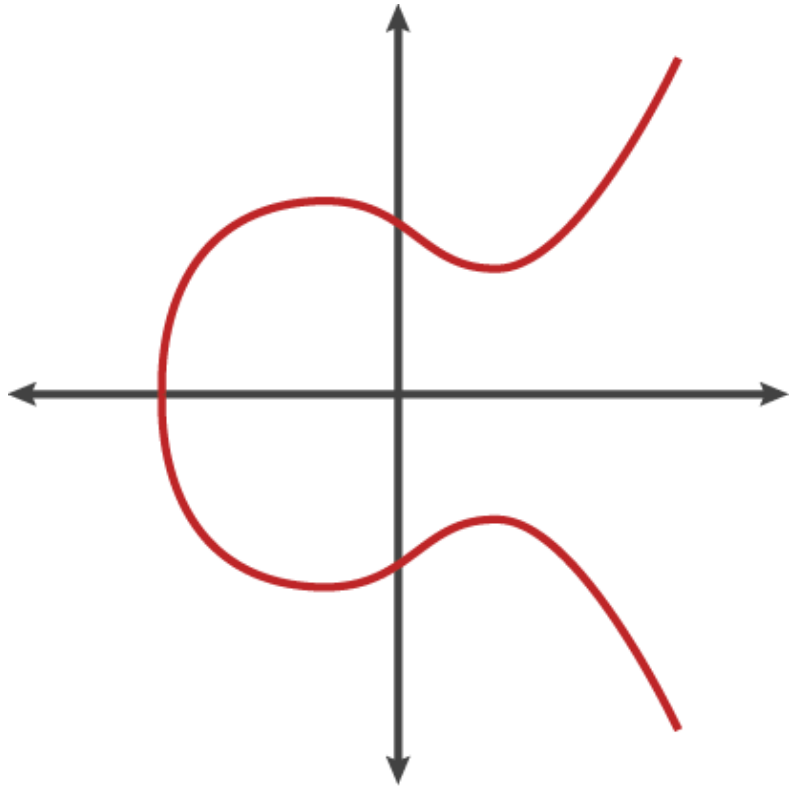


Elliptic Curves

An **elliptic curve** is given by the following equation:

$$\mathbf{y^2 = x^3 + ax + b} \tag{1}$$

where a, b satisfies $-16(4a^3 + 27b^2) \neq 0$.



Elliptic Curves

Properties of the addition law on an elliptic curve E:

- $P \oplus \mathcal{O} = \mathcal{O} \oplus P = P, \forall P \in E$ [Identity]
- $P \oplus (-P) = \mathcal{O}, \forall P \in E$ [Inverse]
- $(P \oplus Q) \oplus R = P \oplus (Q \oplus R), \forall P, Q, R \in E$ [Associative]

Elliptic Curves

Properties of the addition law on an elliptic curve E:

- $P \oplus \mathcal{O} = \mathcal{O} \oplus P = P, \forall P \in E$ [Identity]
- $P \oplus (-P) = \mathcal{O}, \forall P \in E$ [Inverse]
- $(P \oplus Q) \oplus R = P \oplus (Q \oplus R), \forall P, Q, R \in E$ [Associative]

It's a Group

Elliptic Curves

Properties of the addition law on an elliptic curve E:

- $P \oplus \mathcal{O} = \mathcal{O} \oplus P = P, \forall P \in E$ [Identity]
- $P \oplus (-P) = \mathcal{O}, \forall P \in E$ [Inverse]
- $(P \oplus Q) \oplus R = P \oplus (Q \oplus R), \forall P, Q, R \in E$ [Associative]

It's a Group

- $P \oplus Q = Q \oplus P, \forall P, Q \in E$ [Commutative]

Elliptic Curves

Properties of the addition law on an elliptic curve E:

- $P \oplus \mathcal{O} = \mathcal{O} \oplus P = P, \forall P \in E$ [Identity]
- $P \oplus (-P) = \mathcal{O}, \forall P \in E$ [Inverse]
- $(P \oplus Q) \oplus R = P \oplus (Q \oplus R), \forall P, Q, R \in E$ [Associative]

It's a Group

- $P \oplus Q = Q \oplus P, \forall P, Q \in E$ [Commutative]

It's an abelian Group

Note: in this group, multiplication, $n * P, \forall P \in E$, can be seen as adding P n times to itself.

Elliptic Curves in cryptography

In order to apply the theory of elliptic curves to cryptography the elliptic curve is defined over a finite field $\mathbb{F}_p$.

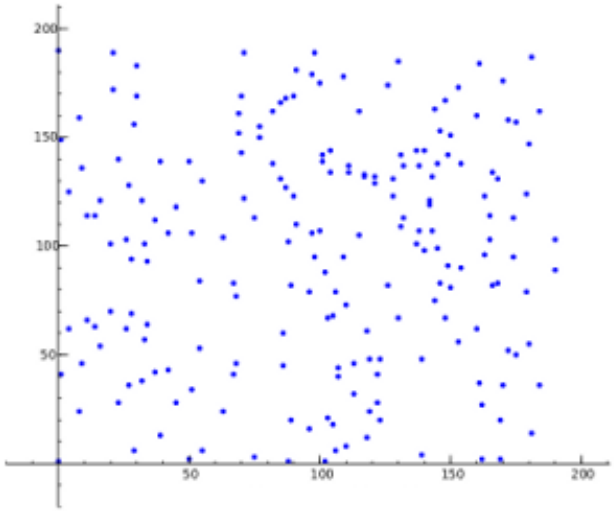
$$E : y^2 = x^3 - 3x + b \tag{2}$$

with $b \in \mathbb{F}_p$ satisfying $432 + 27b^2 \neq 0$.

$$E(\mathbb{F}_p) = \{(x, y) : x, y \in \mathbb{F}_p \wedge y^2 = x^3 - 3x + b\} \cup \mathcal{O}$$

Elliptic Curves in cryptography

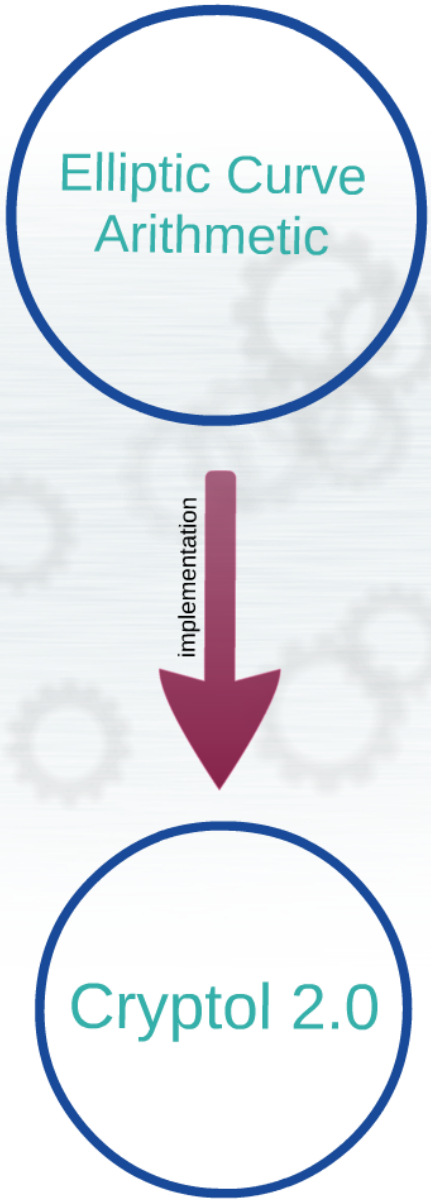
The plot of the elliptic curve over a finite field $\mathbb{F}_p$ turns out to be:



Observations

- The elliptic curve addition law is closed over $\mathbb{F}_p$.
- The addition law **satisfies** all properties listed before.

Evolution Diagram



CryptoI Implementation

Projective System - Jacobian Coordinates

$$Y^2 = X^3 - 3XZ^4 + bZ^6$$

- **ec_affinify(** (X_1, Y_1, Z_1)**)** $= (\frac{X_1}{Z_1^2}, \frac{Y_1}{Z_1^3})$ with $Z_1 \neq 0$
- The point at infinity $\mathcal{O} = (X, Y, 0)$
- The inverse of (X_1, Y_1, Z_1) is $(X_1, -Y_1, Z_1)$.

Projective system increases efficiency in computation!

Cryptol Implementation

Elliptic Curve and Finite Field operations

- Scalar multiplication in $E(\mathbb{F}_p)$ using Montgomery Ladder;
- Exponentiation in $\mathbb{F}_p$;
- Square-root in $\mathbb{F}_p$ (for NIST's p_{256});

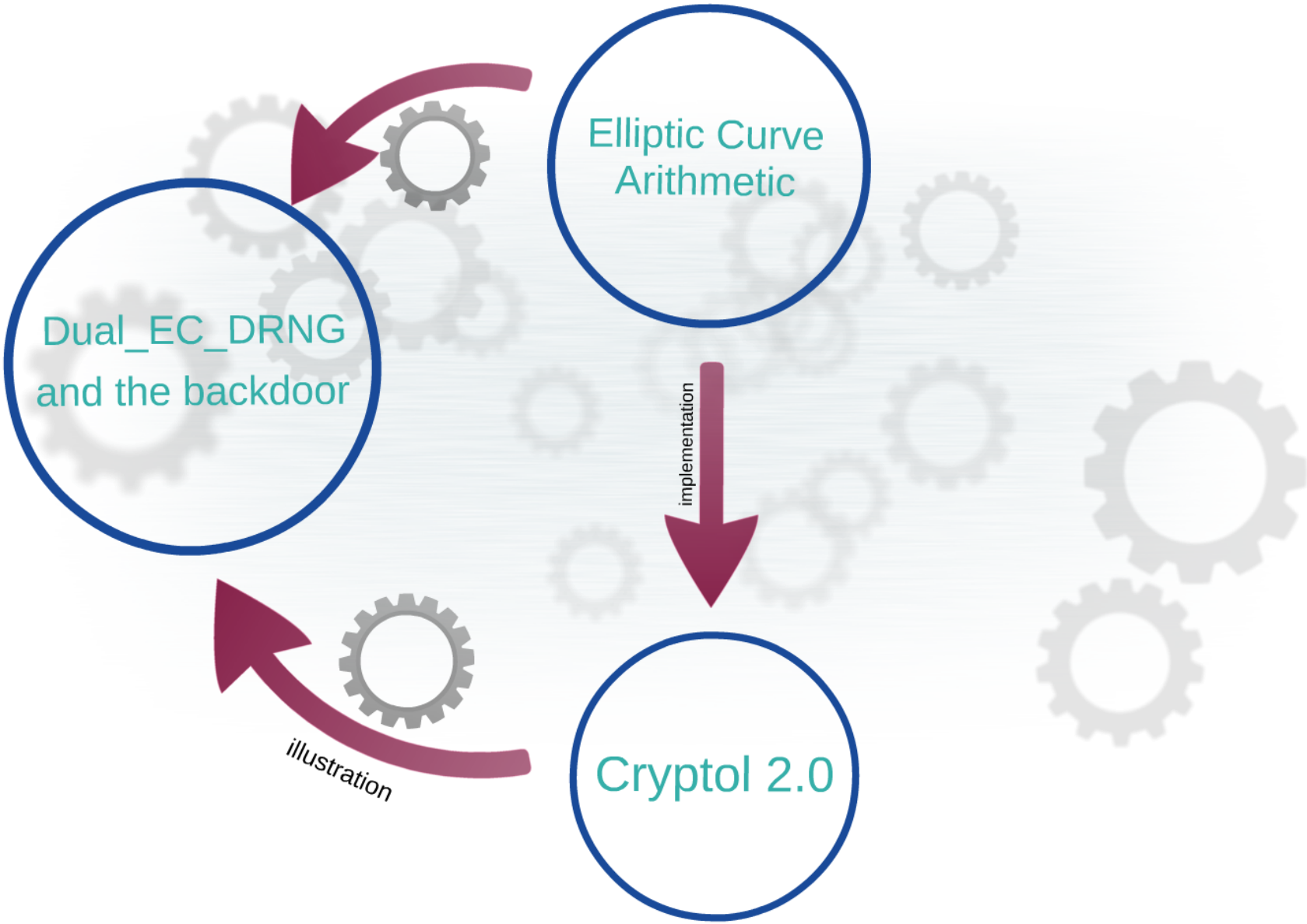
CryptoI Implementation

Elliptic Curve and Finite Field operations

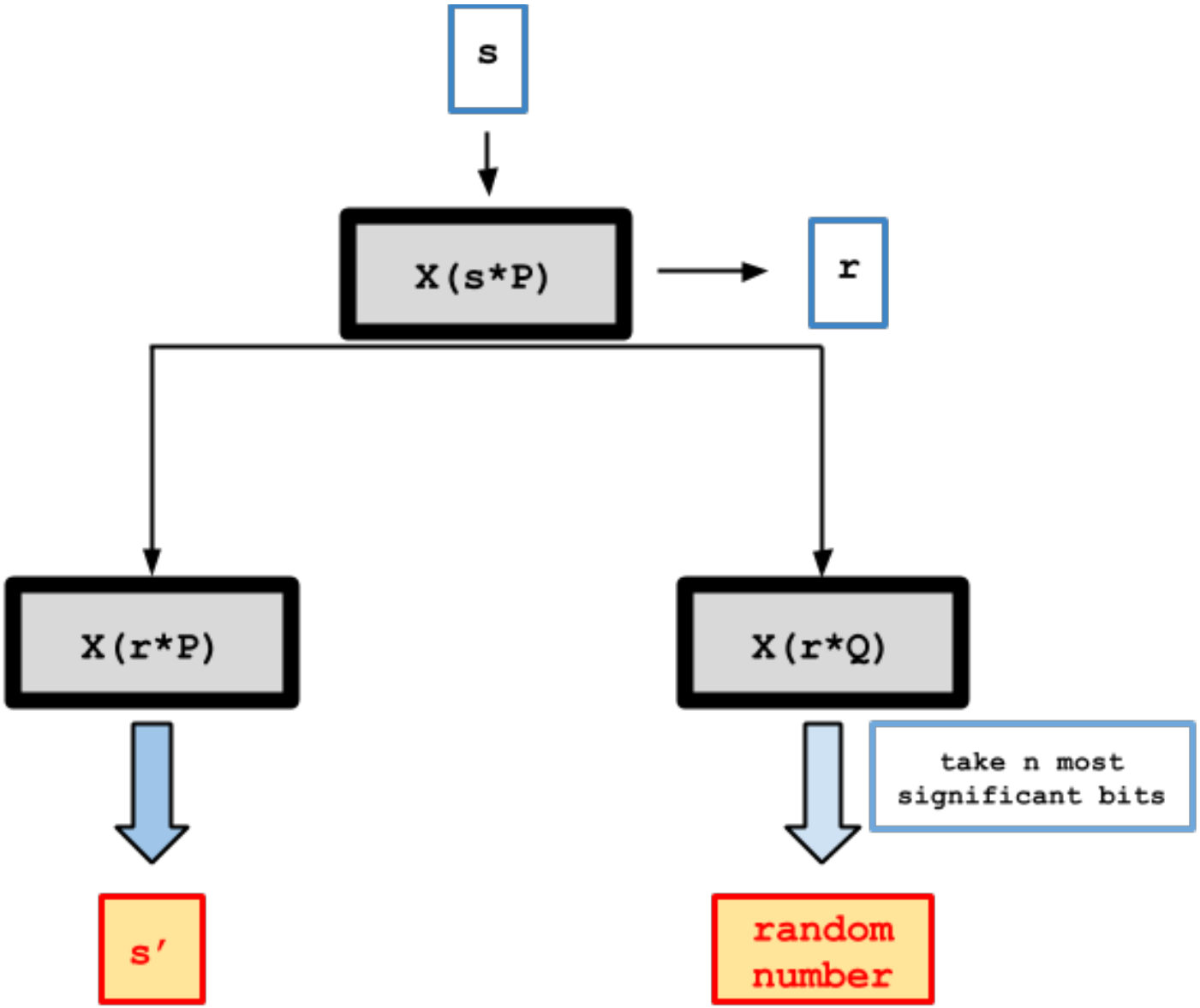
- Scalar multiplication in $E(\mathbb{F}_p)$ using Montgomery Ladder;
- Exponentiation in $\mathbb{F}_p$;
- Square-root in $\mathbb{F}_p$ (for NIST's p_{256});

Simple Dual_EC_DRNG and computation of the backdoor!

Evolution Diagram



Dual_EC_DRNG



Dual_EC_DRNG

Implementation

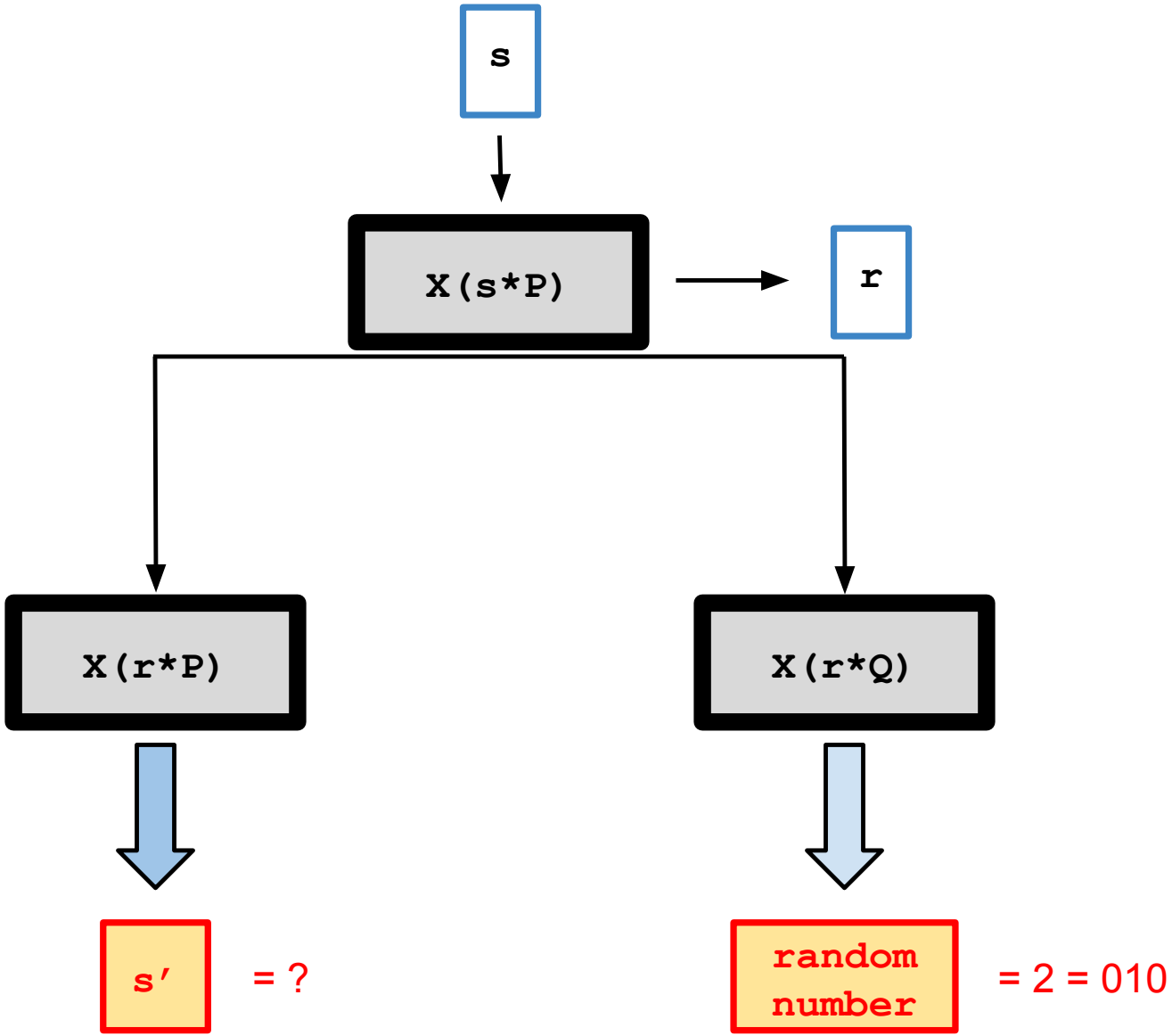
– NIST curve $p - 256$
– small curve $p - 5$

Curve data

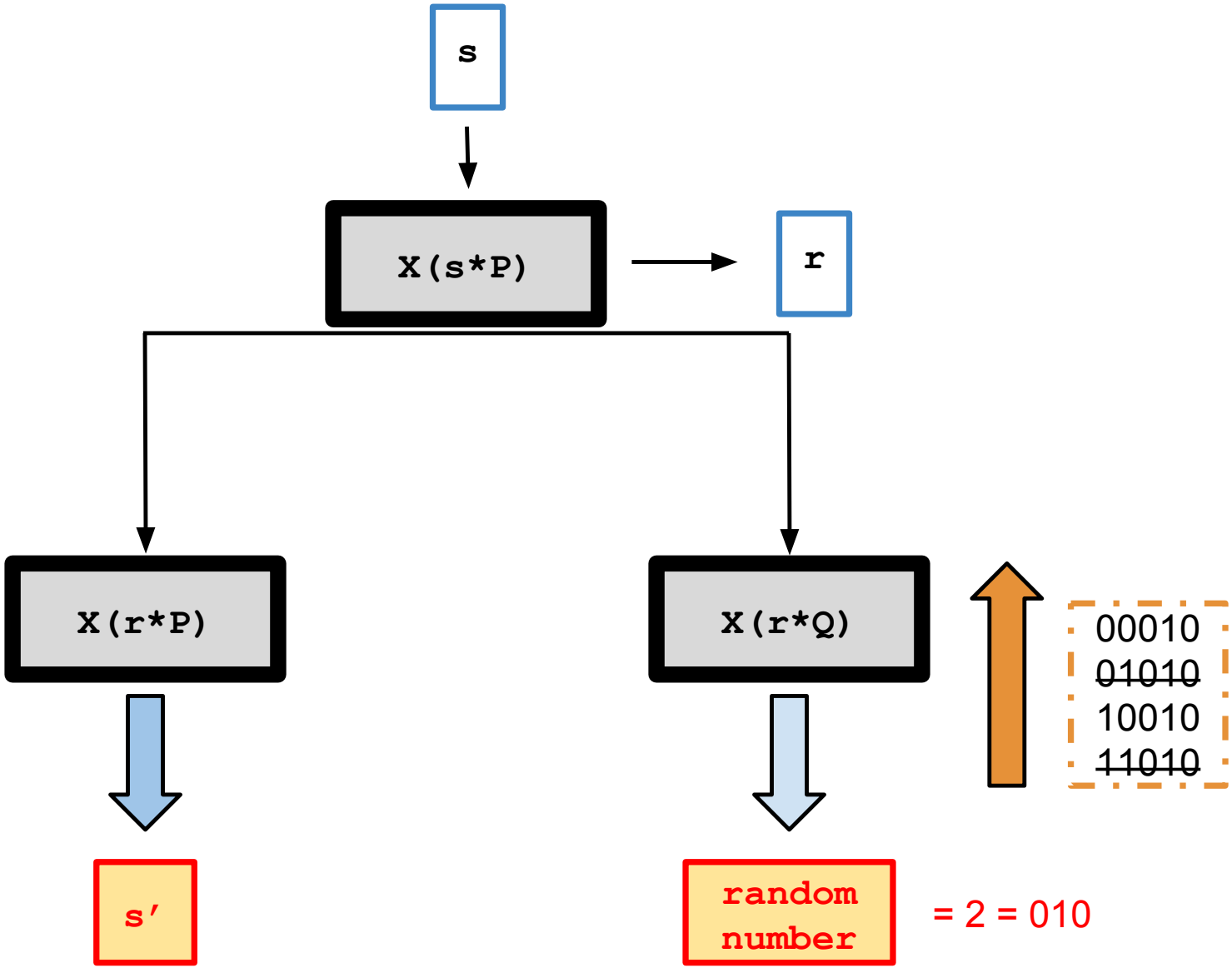
- $y^2 = x^3 - 3x + 7$ *over* $\mathbb{F}_{23}^*$
- **Input points:** $P = (22, 3)$ $Q = (14, 15)$
- **seed:** 7 (hidden)

P = 5*Q! (secret)

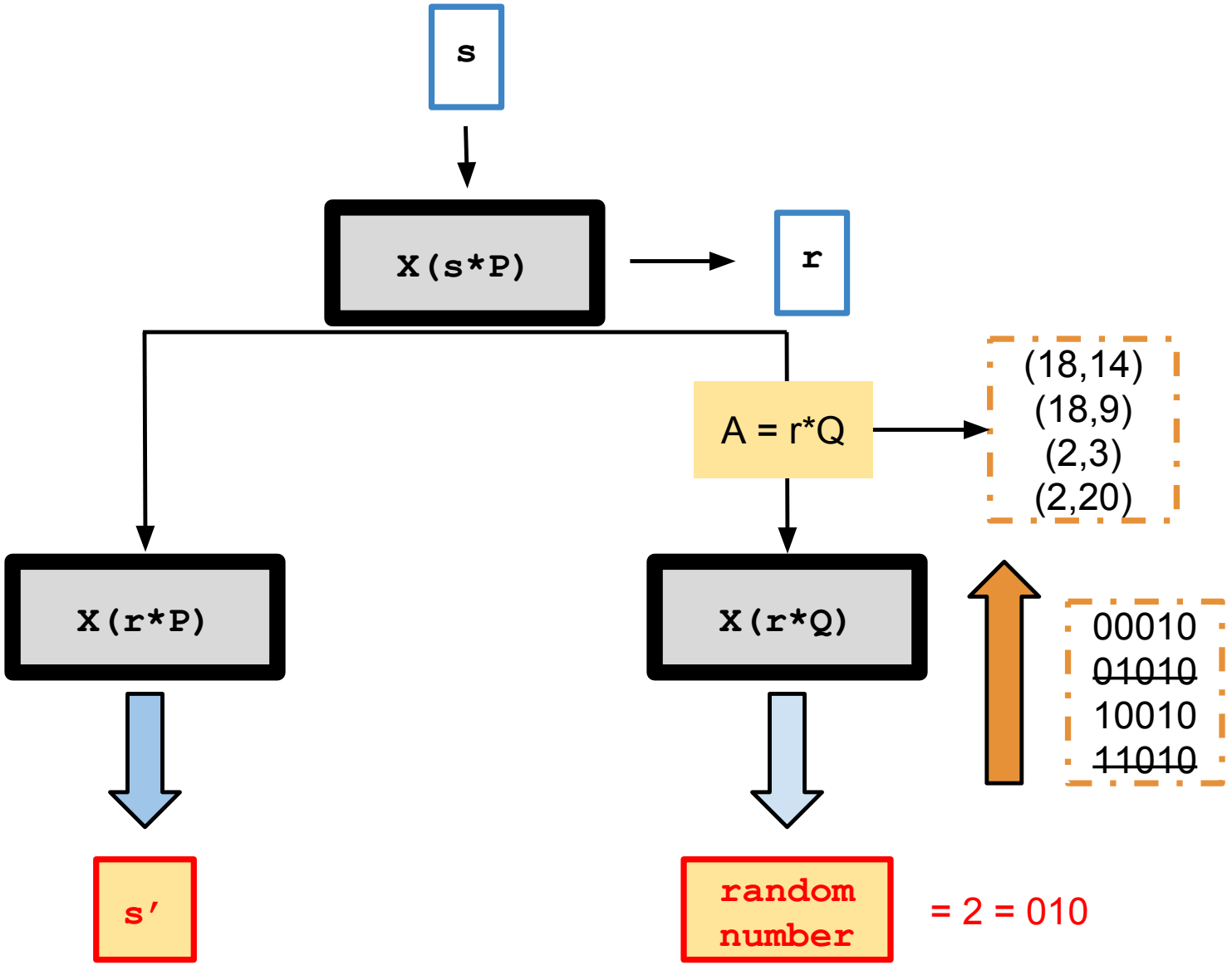
Dual_EC_DRNG backdoor



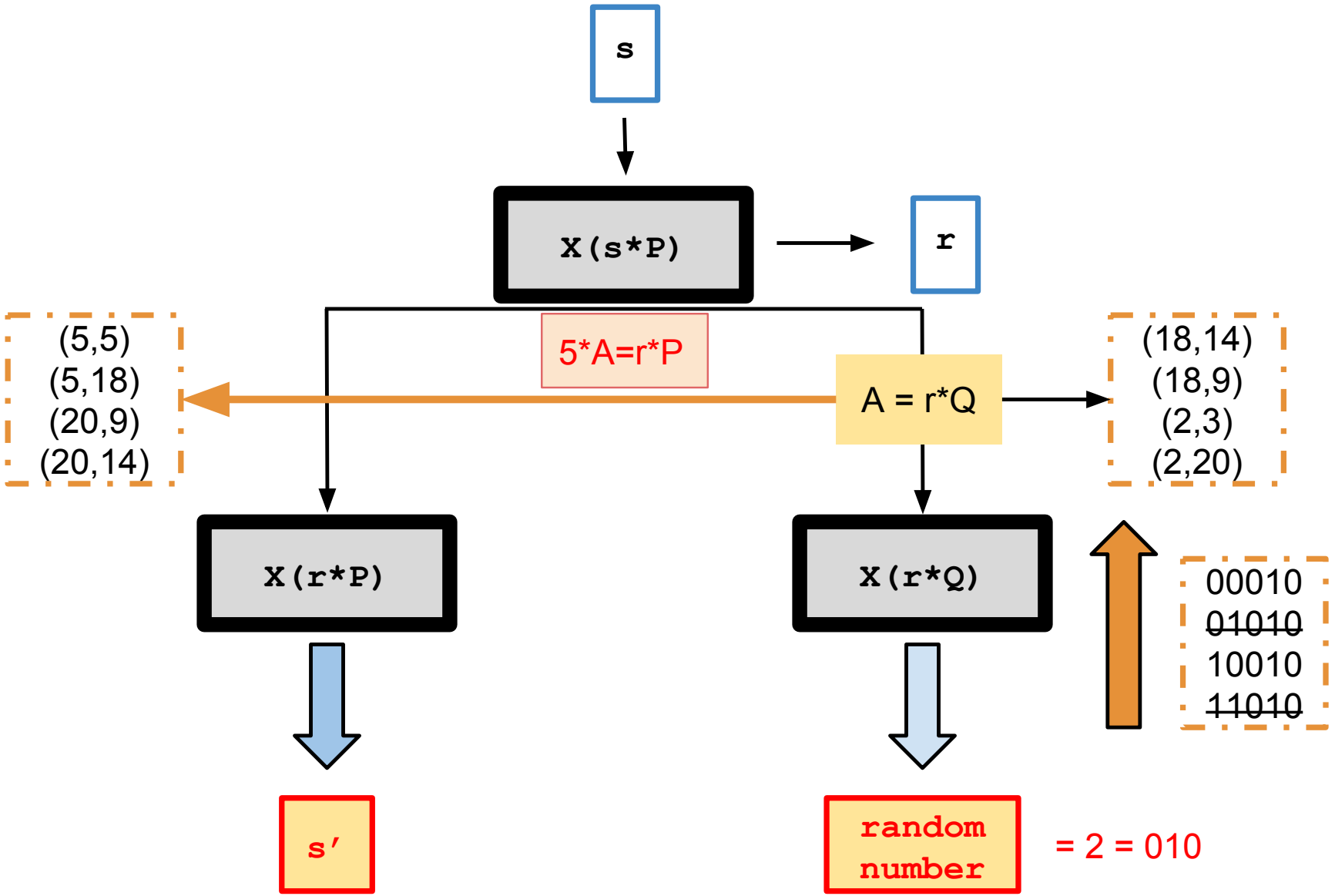
Dual_EC_DRNG backdoor



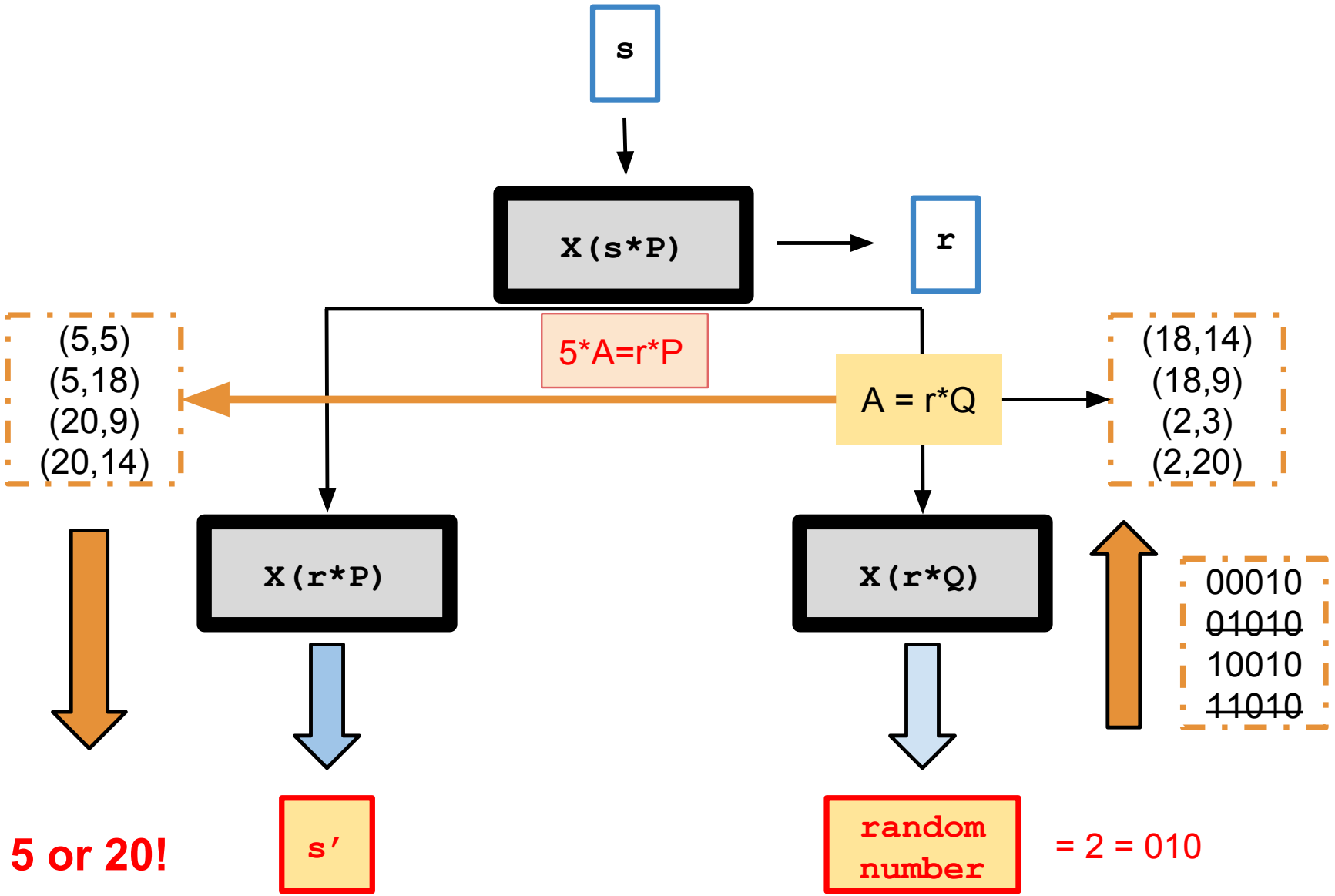
Dual_EC_DRNG backdoor



Dual_EC_DRNG backdoor



Dual_EC_DRNG backdoor

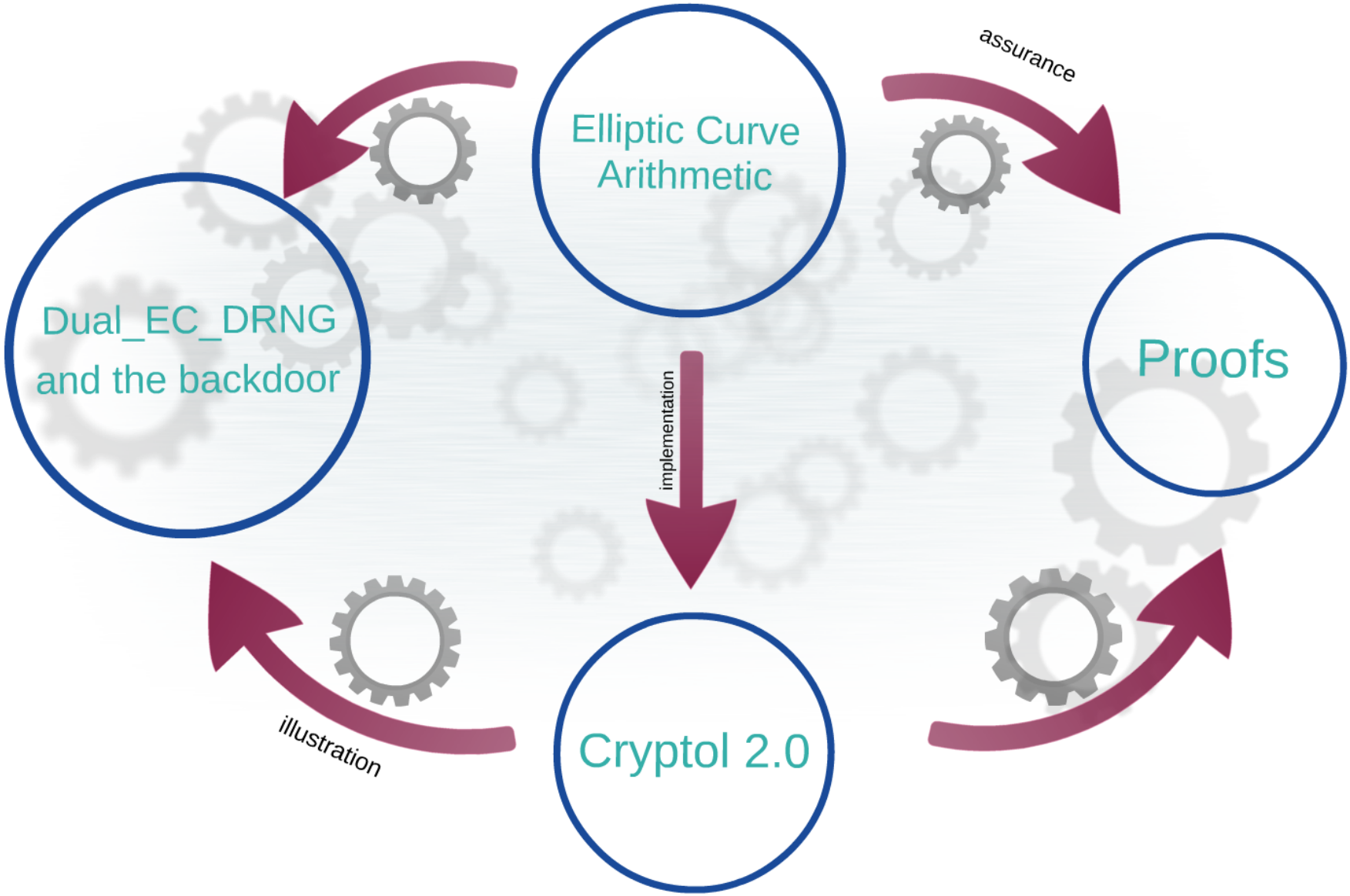


Dual_EC_DRNG backdoor

- There are 2^n alternatives for the random number.
- Some of the alternatives can be excluded.
- In the example, it only took 2 blocks to be **100% sure**

We can make an educated guess if we know how P and Q are related!

Evolution Diagram



Proofs in Cryptol

Correctness properties can be explicitly checked by the Cryptol toolset.

Proofs in Cryptol

Correctness properties can be explicitly checked by the Cryptol toolset.

```
:prove property
```

- Proves properties automatically.

Proofs in Cryptol

Correctness properties can be explicitly checked by the Cryptol toolset.

```
:prove property
```

- Proves properties automatically.

```
:check property
```

- Tests the property at random values to give a quick feedback. The check command will alert for bugs.

Proofs in Cryptol

Correctness properties can be explicitly checked by the Cryptol toolset.

```
:prove property
```

- Proves properties automatically.

```
:check property
```

- Tests the property at random values to give a quick feedback. The check command will alert for bugs.

```
:sat property
```

- Find arguments to a bit-valued function such that the property will be satisfied.

Proving the standards

Identity

- $P \oplus \mathcal{O} = \mathcal{O} \oplus P = P, \forall P \in E$

```
Main> :check point_add_ident curve23
Using exhaustive testing.
passed 1024 tests.
QED

Main> :check point_add_ident_ll curve23
Using exhaustive testing.
passed 1024 tests.
QED
```

Checks for the identity properties. Each one of them took less than a minute.

Proving the standards

Commutativity

- $P \oplus Q = Q \oplus P, \forall P, Q \in E$

```
Main> :check point_add_is_comm curve23
Using exhaustive testing.
passed 1048576 tests.
QED
```

Checks for the commutativity property. The full execution took less than five minutes.

Proving the standards

Associativity

- $(P \oplus Q) \oplus R = P \oplus (Q \oplus R), \forall P, Q, R \in E$

```
Main> :check point_add_is_comm curve23
Using random testing.
passed 10800000 tests.
Coverage: 1.01% (4000000 of 2^^30 values)
```

Comments on Cryptol

Pros

- Cryptol is polymorphic over word sizes, great for cryptographic applications;

Comments on Cryptol

Pros

- Cryptol is polymorphic over word sizes, great for cryptographic applications;
- Cryptol reuses the specifications to check and prove properties.

Comments on Cryptol

Pros

- Cryptol is polymorphic over word sizes, great for cryptographic applications;
- Cryptol reuses the specifications to check and prove properties.

BUT ...

Comments on Cryptol

Pros

- Cryptol is polymorphic over word sizes, great for cryptographic applications;
- Cryptol reuses the specifications to check and prove properties.

BUT ...

Cons

- Inability to use the :prove command;

Comments on Cryptol

Pros

- Cryptol is polymorphic over word sizes, great for cryptographic applications;
- Cryptol reuses the specifications to check and prove properties.

BUT ...

Cons

- Inability to use the :prove command;
- Could not load a reasonable size files;

Comments on Cryptol

Pros

- Cryptol is polymorphic over word sizes, great for cryptographic applications;
- Cryptol reuses the specifications to check and prove properties.

BUT ...

Cons

- Inability to use the :prove command;
- Could not load a reasonable size files;
- Dual_EC is not polymorphic because of type variables.

Contributions and Conclusions

Achievements

- Implementation of Dual_EC_DRNG for multiple curves;

Contributions and Conclusions

Achievements

- Implementation of Dual_EC_DRNG for multiple curves;
- Testing of elliptic curve operations with NIST examples;

Contributions and Conclusions

Achievements

- Implementation of Dual_EC_DRNG for multiple curves;
- Testing of elliptic curve operations with NIST examples;
- Check (and proving) of group laws;

Contributions and Conclusions

Achievements

- Implementation of Dual_EC_DRNG for multiple curves;
- Testing of elliptic curve operations with NIST examples;
- Check (and proving) of group laws;
- Report of running bugs in Cryptol.

Future Work

- Extend the proofs to bigger curves;

Future Work

- Extend the proofs to bigger curves;
- Develop an elliptic curve DiffieHellman key-agreement (ECDH);

Future Work

- Extend the proofs to bigger curves;
- Develop an elliptic curve DiffieHellman key-agreement (ECDH);
- Compare Crytpol proof system with other theorem provers.

Acknowledgements

Acknowledgements

- **Mr. Joe Hendrix (supervisor)**

Acknowledgements

- **Mr. Joe Hendrix (supervisor)**
- **Prof. José Nuno Oliveira (tutor)**

Acknowledgements

- **Mr. Joe Hendrix (supervisor)**
- **Prof. José Nuno Oliveira (tutor)**
- **Formal Methods community!**

Acknowledgements

- **Mr. Joe Hendrix (supervisor)**
- **Prof. José Nuno Oliveira (tutor)**
- **Formal Methods community!**

Thank you!

Questions?