A Univariate Attack against the Limited-Data Instance of Ciminion

Augustin Bariant

Inria, Paris, France
ANSSI, Paris, France

August 28, 2024

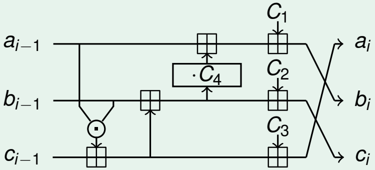
Advanced protocols

Advanced protocols

Zero-Knowledge, Multi-Party Computation or Fully Homomorphic Encryption protocols.

- Often operate on large finite fields $\mathbb{F}_q = \mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ or $\mathbb{F}_q = \mathbb{F}_{2^n}$ ($q \geq 2^{64}$).
- Allowed operations: $+$ and $\times$ in $\mathbb{F}_q$.
- All evaluated functions need to be converted into arithmetic circuits.

Example of an arithmetic circuit of a function: Ciminion round function



Cryptographic primitives in advanced protocols

Cryptographic primitives need to be combined with these protocols.

- ZK: hash functions for verification.
- MPC/FHE: symmetric ciphers for embedded encryption.
- These primitives are evaluated as arithmetic circuits.
- The arithmetic circuit representing AES is very heavy.

Use dedicated primitives: Arithmetization-Oriented (AO) primitives.

Arithmetization-Oriented (AO) primitives

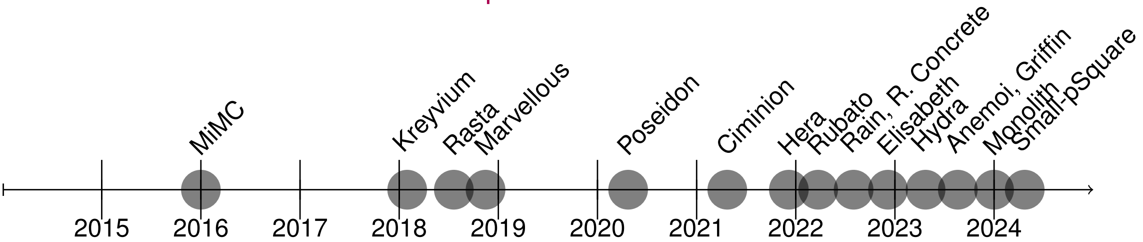
Traditional primitives

- Designed for bit-oriented platforms.
- Operate on bit sequences.
- Low resource consumption (time, etc.).
- S-boxes: small (4 to 8 bits lookups).
- Several decades of cryptanalysis.

Arithmetization-Oriented primitives

- Designed for advanced protocols.
- Operate on large finite fields $\mathbb{F}_q$.
- Low number of field multiplications.
- S-boxes: large (e.g. $x \mapsto x^\alpha$ on $\mathbb{F}_q$).
- $\leq$ 8 years of cryptanalysis.

Non-exhaustive timeline based on stap-zoo.com:



Statistical cryptanalysis of AO primitives: insights

- AO non-linear components are strong against statistical cryptanalysis.

Example: differential properties of AO S-boxes

For an S-box $x \mapsto x^3$, and $\delta_i \neq 0$:

- The equation $(x + \delta_i)^3 - x^3 = \delta_o$ is of degree 2 and has ≤ 2 solutions.
- The maximal differential transition probability is $\leq 2/q$ ($\leq 2^{-63}$ typically).

Example: differential properties of Toffoli gates

Toffoli gates: $(x, y, z) \mapsto (x, y, z + xy)$. Take $\delta_x \neq 0$.

- With an input difference $(\delta_x, 0, 0)$, the output difference is $(\delta_x, 0, \delta_x y)$
- q possible values for $\delta_x y$, each with proba $1/q$ ($\leq 2^{-64}$ typically).

AO primitives need to be designed to resist algebraic attacks.

Algebraic attacks: examples on a block cipher

Consider a block cipher $E_K : \begin{cases} \mathbb{F}_q & \to \mathbb{F}_q \\ P & \mapsto C. \end{cases}$

- Integral attacks: exploit the low algebraic degree d_{alg} of E_K (over $\mathbb{F}_{2^n}$).
  - For any subspace S of $\mathbb{F}_{2^n}$ with $\dim(S) > d_{\text{alg}}$:

$$\sum_{x \in S} E_K(x) = 0.$$

  - Requires $2^{d_{\text{alg}}+1}$ data (typically, $d_{\text{alg}} \approx n$).
- Interpolation attacks: exploit the low univariate degree d of E_K.
  - Gather $E_K(x)$ for $d+1$ values x and perform a Fast Lagrange Interpolation.
  - Recover the coefficients of $E_K(x)$ and the entire mapping $x \mapsto E_K(x)$.
  - Requires $d+1$ data (typically, $d \approx q$).

These two attacks require a heavy amount of data.

Algebraic attacks: examples on a block cipher

Consider a block cipher $E_K : \begin{cases} \mathbb{F}_q & \to \mathbb{F}_q \\ P & \mapsto C. \end{cases}$

- Integral attacks: exploit the low algebraic degree d_{alg} of E_K (over $\mathbb{F}_{2^n}$).
  - For any subspace S of $\mathbb{F}_{2^n}$ with $\dim(S) > d_{\text{alg}}$:

$$\sum_{x \in S} E_K(x) = 0.$$

  - Requires $2^{d_{\text{alg}}+1}$ data (typically, $d_{\text{alg}} \approx n$).
- Interpolation attacks: exploit the low univariate degree d of E_K.
  - Gather $E_K(x)$ for $d+1$ values x and perform a Fast Lagrange Interpolation.
  - Recover the coefficients of $E_K(x)$ and the entire mapping $x \mapsto E_K(x)$.
  - Requires $d+1$ data (typically, $d \approx q$).

These two attacks require a heavy amount of data.

Algebraic attacks: examples on a block cipher

Consider a block cipher $E_K : \begin{cases} \mathbb{F}_q & \to \mathbb{F}_q \\ P & \mapsto C. \end{cases}$

- Integral attacks: exploit the low algebraic degree d_{alg} of E_K (over $\mathbb{F}_{2^n}$).
  - For any subspace S of $\mathbb{F}_{2^n}$ with $\dim(S) > d_{\mathrm{alg}}$:

$$\sum_{x \in S} E_K(x) = 0.$$

  - Requires $2^{d_{\mathrm{alg}}+1}$ data (typically, $d_{\mathrm{alg}} \approx n$).
- Interpolation attacks: exploit the low univariate degree d of E_K.
  - Gather $E_K(x)$ for $d+1$ values x and perform a Fast Lagrange Interpolation.
  - Recover the coefficients of $E_K(x)$ and the entire mapping $x \mapsto E_K(x)$.
  - Requires $d+1$ data (typically, $d \approx q$).

These two attacks require a heavy amount of data.

A low-data algebraic attack: the polynomial solving attack

The polynomial solving attack is an algebraic attack composed of two steps:

Modeling

Represent the primitive with a polynomial system $\mathcal{P}$.

- A solution to $\mathcal{P}$ leads to the key.
- Not trivial to find the best modeling.
- Usually requires a low amount of data.

$$\mathcal{P} = \begin{cases} P_1(X_1, \dots X_n) = 0 \\ \vdots \\ P_n(X_1, \dots X_n) = 0 \end{cases}$$

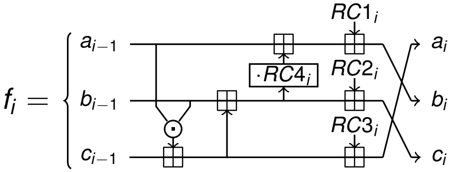
Solving

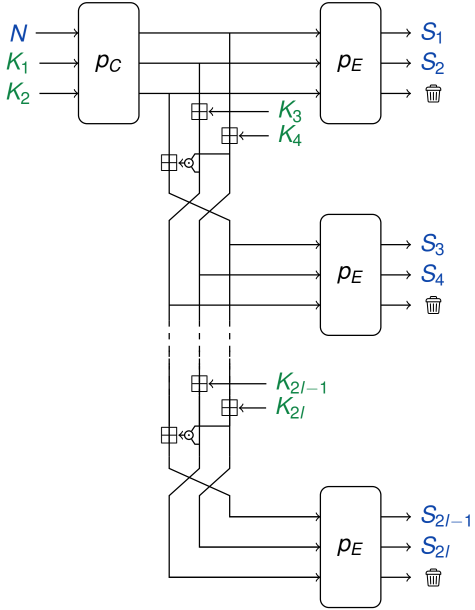
Find $(X_1, \dots X_n) \in \mathbb{F}_q^n$ which solves $\mathcal{P}$.

- Use state-of-the-art Gröbner basis or univariate solving algorithms.
- Different complexity formulas depending on the method used.

Ciminion [Dobraunig, Grassi, Guinet & Kuijsters, EC'21]

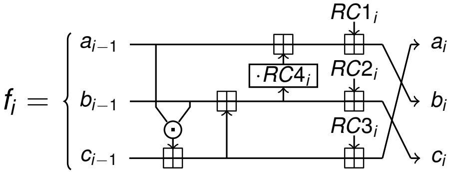
- Nonce-based stream cipher on $\mathbb{F}_q$.
  - N different every query.
  - For each N, generate a sequence S_i.
  - $\log(q)$-bit of security.
- Secret subkeys $K_i \in \mathbb{F}_q$.
- Security based on truncated outputs.
- p_C and p_E permutations of $\mathbb{F}_q^3$.
- $p_C = f_{r_C} \circ \cdots \circ f_1$.
- $p_E = f_{r_E + r_C} \circ \cdots \circ f_{r_C+1}$.
- f_i: quadratic round function.

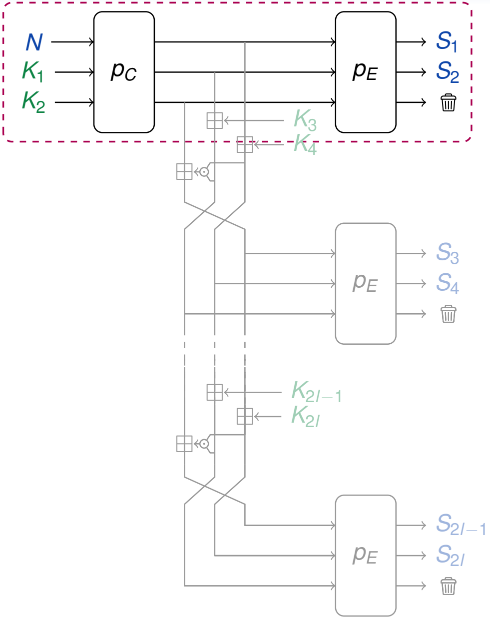




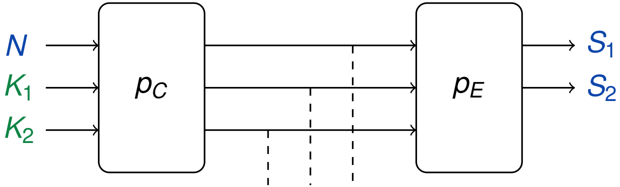
Ciminion [Dobraunig, Grassi, Guinet & Kuijsters, EC'21]

- Nonce-based stream cipher on $\mathbb{F}_q$.
  - N different every query.
  - For each N, generate a sequence S_i.
  - $\log(q)$-bit of security.
- Secret subkeys $K_i \in \mathbb{F}_q$.
- Security based on truncated outputs.
- p_C and p_E permutations of $\mathbb{F}_q^3$.
- $p_C = f_{r_C} \circ \cdots \circ f_1$.
- $p_E = f_{r_E + r_C} \circ \cdots \circ f_{r_C+1}$.
- f_i: quadratic round function.





Security analysis of the designers



- Quadratic round function. $p_E \circ p_C$ of degree $2^{r_C + r_E}$.

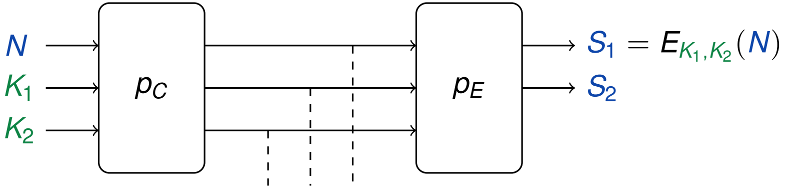
Security against interpolation attacks

- $E_{K_1,K_2}(N)$ of degree $d = 2^{r_C + r_E - 1}$.
- Possible to interpolate with $d + 1$ data.
- Not applicable if the attacker can query $< d$ data.

The limited-data variant of Ciminion

Maximum $\sqrt{q}$ data queries for the attacker. r_C chosen such that $d = 2^{r_C + r_E - 1} \approx q^{\frac{3}{4}}$.

Security analysis of the designers



- Quadratic round function. $p_E \circ p_C$ of degree $2^{r_C + r_E}$.

Security against interpolation attacks

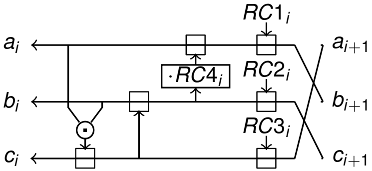
- $E_{K_1,K_2}(N)$ of degree $d = 2^{r_C + r_E - 1}$.
- Possible to interpolate with $d + 1$ data.
- Not applicable if the attacker can query $< d$ data.

The limited-data variant of Ciminion

Maximum $\sqrt{q}$ data queries for the attacker. r_C chosen such that $d = 2^{r_C + r_E - 1} \approx q^{\frac{3}{4}}$.

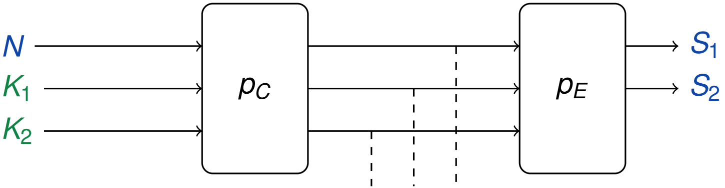
A new attack based on univariate polynomial solving

- Observation: the inverse round function also quadratic.



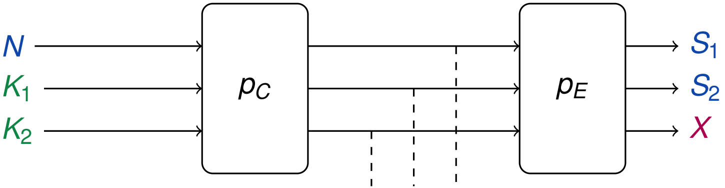
Our attack builds a polynomial the other way around.

A new attack based on univariate polynomial solving



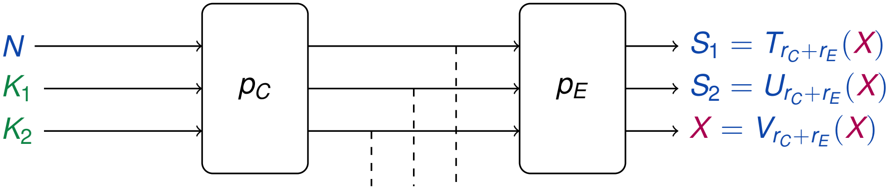
1. The attacker queries S_1 and S_2 under the nonce N.
2. Set the truncated value to an unkown variable X and interpret outputs as polynomials.
3. The attacker computes $T_0(X), U_0(X), V_0(X) = p_C^{-1} \circ p_E^{-1}(S_1, S_2, X)$.
   - Evaluate the inverse round function on polynomials of $\mathbb{F}_q[X]$.
4. The attacker solves $T_0(X) - N = 0$ (degree $\approx q^{\frac{3}{4}}$).
5. The attacker recovers X and computes $K_1 = U_0(X)$ and $K_2 = V_0(X)$.

A new attack based on univariate polynomial solving



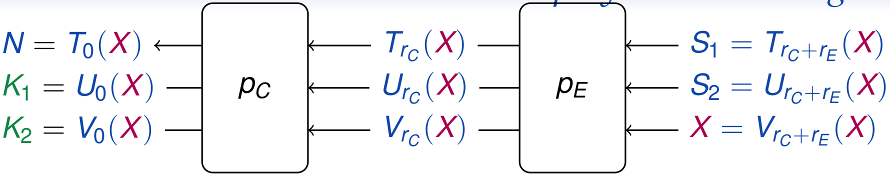
1. The attacker queries S_1 and S_2 under the nonce N.
2. Set the truncated value to an unkown variable X and interpret outputs as polynomials.
3. The attacker computes $T_0(X), U_0(X), V_0(X) = p_C^{-1} \circ p_E^{-1}(S_1, S_2, X)$.
   - Evaluate the inverse round function on polynomials of $\mathbb{F}_q[X]$.
4. The attacker solves $T_0(X) - N = 0$ (degree $\approx q^{\frac{3}{4}}$).
5. The attacker recovers X and computes $K_1 = U_0(X)$ and $K_2 = V_0(X)$.

A new attack based on univariate polynomial solving

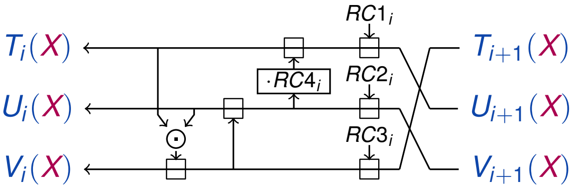


1. The attacker queries S_1 and S_2 under the nonce N.
2. Set the truncated value to an unkown variable X and interpret outputs as polynomials.
3. The attacker computes $T_0(X), U_0(X), V_0(X) = p_C^{-1} \circ p_E^{-1}(S_1, S_2, X)$.
   - Evaluate the inverse round function on polynomials of $\mathbb{F}_q[X]$.
4. The attacker solves $T_0(X) - N = 0$ (degree $\approx q^{\frac{3}{4}}$).
5. The attacker recovers X and computes $K_1 = U_0(X)$ and $K_2 = V_0(X)$.

A new attack based on univariate polynomial solving

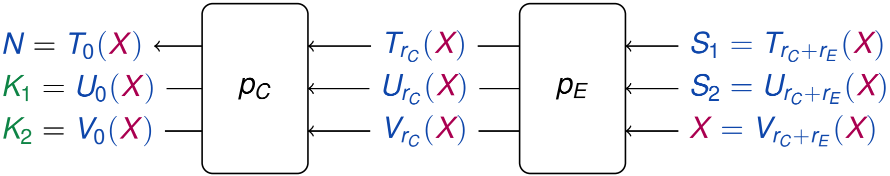


1. The attacker queries S_1 and S_2 under the nonce N.
2. Set the truncated value to an unkown variable X and interpret outputs as polynomials.
3. The attacker computes $T_0(X), U_0(X), V_0(X) = p_C^{-1} \circ p_E^{-1}(S_1, S_2, X)$.
   - Evaluate the inverse round function on polynomials of $\mathbb{F}_q[X]$.



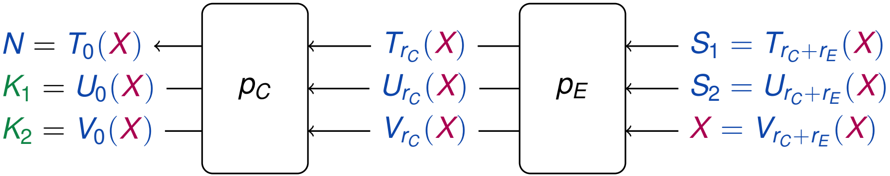
4. The attacker solves $T_0(X) - N = 0$ (degree $\approx q^{\frac{3}{4}}$).

A new attack based on univariate polynomial solving



1. The attacker queries S_1 and S_2 under the nonce N.
2. Set the truncated value to an unkown variable X and interpret outputs as polynomials.
3. The attacker computes $T_0(X), U_0(X), V_0(X) = p_C^{-1} \circ p_E^{-1}(S_1, S_2, X)$.
   - Evaluate the inverse round function on polynomials of $\mathbb{F}_q[X]$.
4. The attacker solves $T_0(X) - N = 0$ (degree $\approx q^{\frac{3}{4}}$).
5. The attacker recovers X and computes $K_1 = U_0(X)$ and $K_2 = V_0(X)$.

A new attack based on univariate polynomial solving



1. The attacker queries S_1 and S_2 under the nonce N.
2. Set the truncated value to an unkown variable X and interpret outputs as polynomials.
3. The attacker computes $T_0(X), U_0(X), V_0(X) = p_C^{-1} \circ p_E^{-1}(S_1, S_2, X)$.
   - Evaluate the inverse round function on polynomials of $\mathbb{F}_q[X]$.
4. The attacker solves $T_0(X) - N = 0$ (degree $\approx q^{\frac{3}{4}}$).
5. The attacker recovers X and computes $K_1 = U_0(X)$ and $K_2 = V_0(X)$.

Solving polynomial systems: the univariate case

One univariate equation of degree d in $\mathbb{F}_q$:

$$\mathcal{P} = \left\{ P(X) = 0. \right.$$

General idea

[BBLP, ToSC'22]

- The field equation $X^q - X$ cancels all elements in $\mathbb{F}_q$:

$$X^q - X = \prod_{\omega \in \mathbb{F}_q} (X - \omega).$$

- Compute $R(X) = \gcd(P(X), X^q - X)$ efficiently with fast polynomial operations.
- $R(X)$ is of low degree and has the same roots in $\mathbb{F}_q$ as $P(X)$. Recover the roots.

Univariate solving: more details

Operation cost on polynomials of degree d [CK, AI'91; Moenck, ACMSTC'73; Strassen, TCS'75]

- Multiplication, euclidian division: $\mathcal{O}(d \log(d) \log(\log(d)))$.
- GCD: $\mathcal{O}(d \log(d)^2 \log(\log(d)))$

Algorithm for univariate solving ($P(X) = 0$)

- Compute $Q(X) = X^q \bmod P(X)$ using fast exponentiation ($\log(q)$ steps).
- Compute $R(X) = \gcd(Q(X) - X, P(X))$.
- $R(X) = \gcd(X^q - X, P(X))$ is of small degree. Recover its roots (e.g. with factoring).

- Solving complexity quasi-linear in d: $\mathcal{O}(d \log(q) \log(d) \log(\log(d)))$ operations.
- Cheaper than factoring which costs $\mathcal{O}(d^{1.815} \log(q))$.

Univariate solving: more details

Operation cost on polynomials of degree d [CK, AI'91; Moenck, ACMSTC'73; Strassen, TCS'75]

- Multiplication, euclidian division: $\mathcal{O}(d\log(d)\log(\log(d)))$.
- GCD: $\mathcal{O}(d\log(d)^2\log(\log(d)))$

Algorithm for univariate solving ($P(X)=0$)

- Compute $Q(X)=X^q \bmod P(X)$ using fast exponentiation ($\log(q)$ steps).
- Compute $R(X)=\gcd(Q(X)-X, P(X))$.
- $R(X)=\gcd(X^q-X, P(X))$ is of small degree. Recover its roots (e.g. with factoring).

- Solving complexity quasi-linear in d: $\mathcal{O}(d\log(q)\log(d)\log(\log(d)))$ operations.
- Cheaper than factoring which costs $\mathcal{O}(d^{1.815}\log(q))$.

Univariate solving: more details

Operation cost on polynomials of degree d [CK, AI'91; Moenck, ACMSTC'73; Strassen, TCS'75]

- Multiplication, euclidian division: $\mathcal{O}(d \log(d) \log(\log(d)))$.
- GCD: $\mathcal{O}(d \log(d)^2 \log(\log(d)))$

Algorithm for univariate solving ($P(X) = 0$)

- Compute $Q(X) = X^q \bmod P(X)$ using fast exponentiation ($\log(q)$ steps).
- Compute $R(X) = \gcd(Q(X) - X, P(X))$.
- $R(X) = \gcd(X^q - X, P(X))$ is of small degree. Recover its roots (e.g. with factoring).

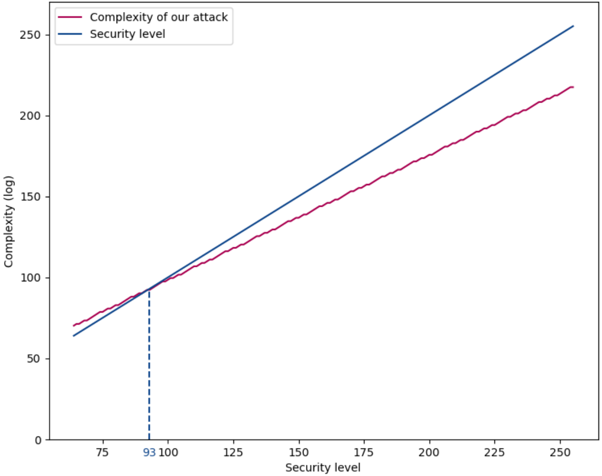
- Solving complexity quasi-linear in d: $\mathcal{O}(d \log(q) \log(d) \log(\log(d)))$ operations.
- Cheaper than factoring which costs $\mathcal{O}(d^{1.815} \log(q))$.

Our new univariate attack: complexity

- Asymptotic complexity of the attack:

$$\tilde{\mathcal{O}}(2^{r_C+r_E-1}) = \tilde{\mathcal{O}}(q^{3/4}+7).$$

- Security level claimed: q.
- This attack breaks the security claims for $q \geq 93$.
- Overwhelming constant & logarithmic terms for small q.

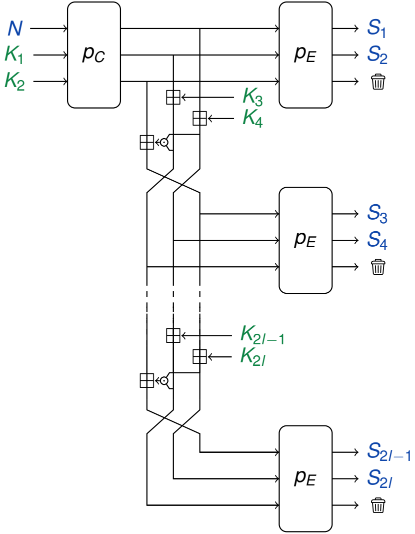


Comparision with other attacks

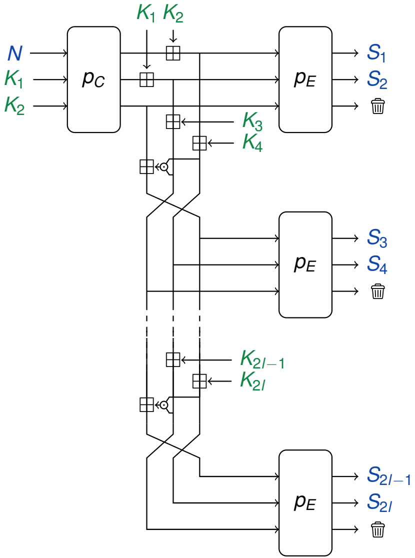
Attack type	Generic r_C, r_E		Full-instance attacks		Reference
	Data	Time	Standard	Limited-data	
Gröbner basis (SKR)	8	$\mathcal{O}(2^{4\omega r_E})$	$q \geq 587$	$q \geq 587$	[BBLP, ToSC'22]
Integral (dist.)	$\mathcal{O}(2^{r_C+r_E})$	$\mathcal{O}(2^{r_C+r_E})$	-	-	[ZLLL, ISC'23']
Univariate (SKR)	2	$\tilde{\mathcal{O}}(2^{r_C+r_E})$	-	$q \geq 93$	[This work]

- $2.41 \leq \omega \leq 3$ is the linear algebra exponent.
- SKR denotes subkey recovery.

Mitigation of the attack: a costless example



↦



Conclusion & takeaways

- Attack against the full limited-data variant of Ciminion.
- Polynomial solving attacks often only require a few data samples.
- Finding the roots in $\mathbb{F}_q$ of a polynomial is quasi-linear in its degree.

Thank you for your attention.

Conclusion & takeaways

- Attack against the full limited-data variant of Ciminion.
- Polynomial solving attacks often only require a few data samples.
- Finding the roots in $\mathbb{F}_q$ of a polynomial is quasi-linear in its degree.

Thank you for your attention.