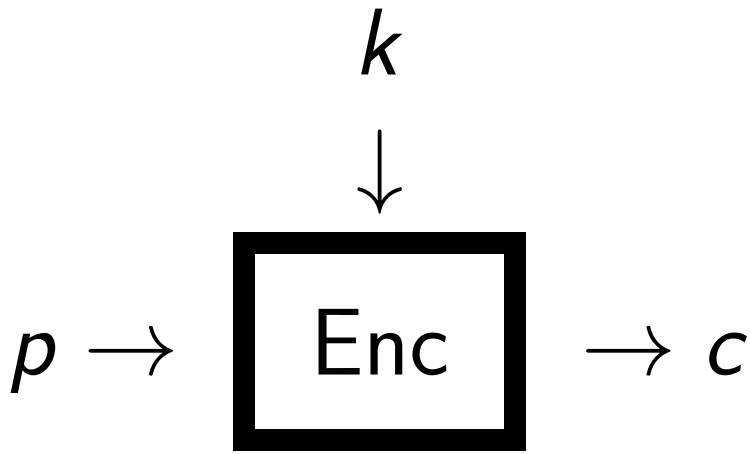
UNIVERSITÄT PASSAU

Automation of Algebraic Fault Attacks

Jan Horáček

Fakultät für Informatik und Mathematik
Universität Passau

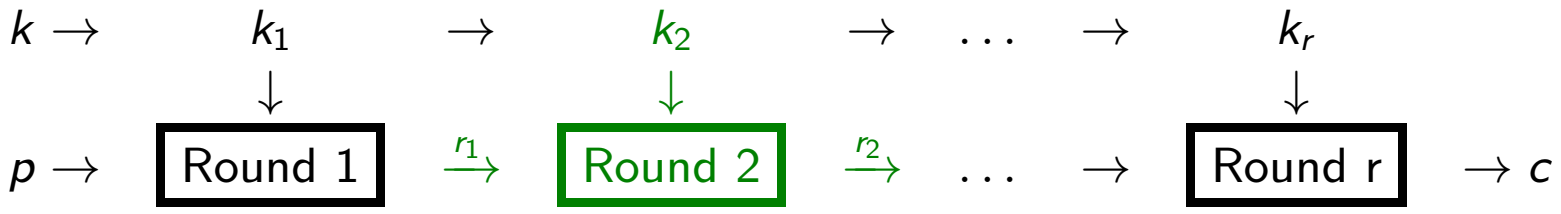
A Symmetric Block Cipher



A Symmetric Block Cipher

UNIVERSITÄT PASSAU

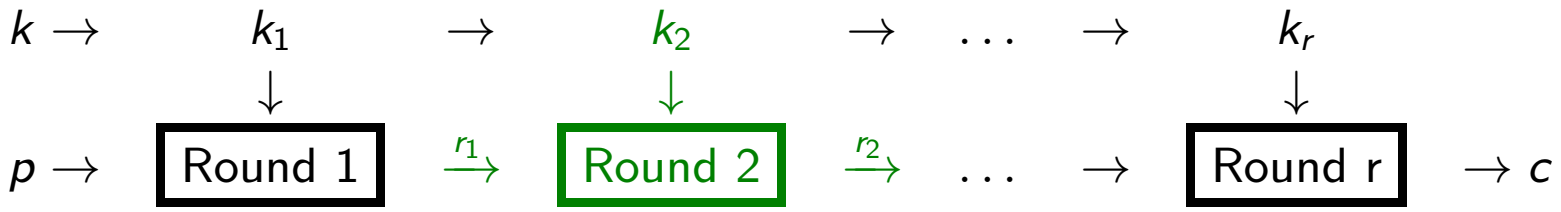
Zoom in on Enc



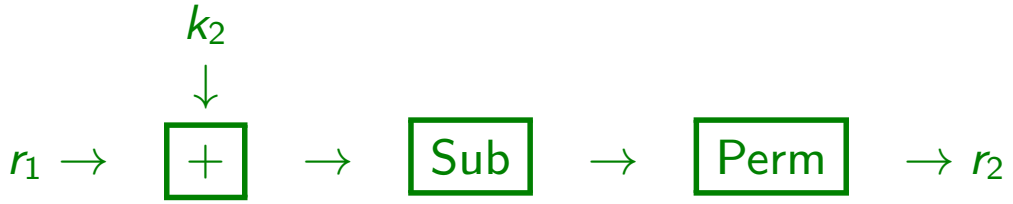
Zoom in on one Round

A Symmetric Block Cipher

Zoom in on Enc



Zoom in on one Round



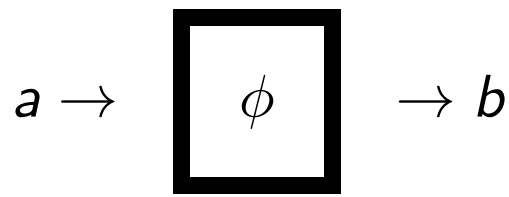
Algebraic Representation

UNIVERSITÄT PASSAU

Observation

- $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$
- Any map over a finite field is polynomial map.
- There exist polynomials $g_1, \ldots, g_m \in \mathbb{F}_2[x_1, \ldots, x_n]$ such that

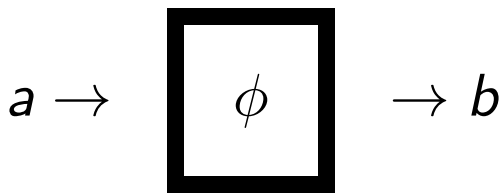
$$\phi(a) = (g_1(a), \ldots, g_m(a)), \text{ for all } a \in \mathbb{F}_2^n.$$



UNIVERSITÄT PASSAU

Observation

- $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$
- Any map over a finite field is polynomial map.
- There exist polynomials $g_1, \ldots, g_m \in \mathbb{F}_2[x_1, \ldots, x_n]$ such that



$$\phi(a) = (g_1(a), \ldots, g_m(a)), \text{ for all } a \in \mathbb{F}_2^n.$$

UNIVERSITÄT PASSAU

Attack framework

- Thus we may represent Enc as *a Boolean polynomial system*, where indeterminates represent p, c, k or inner states.
- An attacker has access to p and/or c, recovering k is nothing more than *solving* a polynomial system of equations over $\mathbb{F}_2$.
- This system is overdefined, sparse and has rich structure.

UNIVERSITÄT PASSAU

Attack framework

- Thus we may represent Enc as *a Boolean polynomial system*, where indeterminates represent p, c, k or inner states.
- An attacker has access to p and/or c, recovering k is nothing more than *solving* a polynomial system of equations over $\mathbb{F}_2$.
- This system is overdefined, sparse and has rich structure.

UNIVERSITÄT PASSAU

Attack framework

- Thus we may represent Enc as *a Boolean polynomial system*, where indeterminates represent p, c, k or inner states.
- An attacker has access to p and/or c, recovering k is nothing more than *solving* a polynomial system of equations over $\mathbb{F}_2$.
- This system is overdefined, sparse and has rich structure.

UNIVERSITÄT PASSAU

End of the story?

Nope!

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Solvers Review

UNIVERSITÄT PASSAU

- Brute search with clever evaluation mechanism: libFES.
- Finding good generators: Gröbner basis, Border Basis.
- Rewriting to linear system: XL, XSL.
- Substitute linear polynomials & eliminate: ElimLin.
- Combinatorial approach: Semaev algorithm.
- Transform polynomials to CNF clauses: SAT solving.
- Transformation to inequalities: Integer Programming.
- Combinations of them.

Fault Attacks

- ... is a type of *active side-channel* attacks against cryptographic implementations.
- The attacker is able to induce *faults* during the computation of $\mathrm{Enc}(p, k)$.
- The attacker observes the faulty c' and the faulty-free ciphertext c.

Figure: Laser Station 2 by riscure

Fault Attacks

- ... is a type of *active side-channel* attacks against cryptographic implementations.
- The attacker is able to induce *faults* during the computation of $\mathrm{Enc}(p, k)$.
- The attacker observes the faulty c' and the faulty-free ciphertext c.

Figure: Laser Station 2 by riscure

Fault Attacks

- ... is a type of *active side-channel* attacks against cryptographic implementations.
- The attacker is able to induce *faults* during the computation of $\mathrm{Enc}(p, k)$.
- The attacker observes the faulty c' and the faulty-free ciphertext c.

Figure: Laser Station 2 by riscure

“Ad-hoc” Fault Attacks

- For a concrete cipher, the attacker finds a good *spot* to place faults.
- He analyses the *fault propagation*.
- He derives *fault equations* and substitutes c and c'.

Figure: Laser Station 2 by riscure

“Ad-hoc” Fault Attacks

UNIVERSITÄT PASSAU

- For a concrete cipher, the attacker finds a good *spot* to place faults.
- He analyses the *fault propagation*.
- He derives *fault equations* and substitutes c and c'.

Figure: Laser Station 2 by riscure

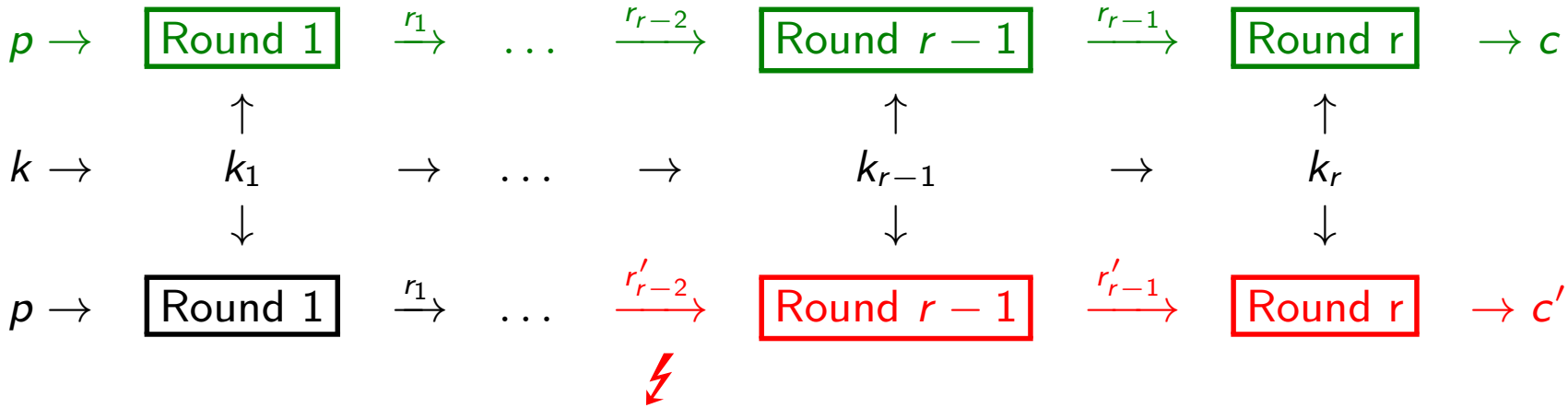
“Ad-hoc” Fault Attacks

UNIVERSITÄT PASSAU

- For a concrete cipher, the attacker finds a good *spot* to place faults.
- He analyses the *fault propagation*.
- He derives *fault equations* and substitutes c and c'.

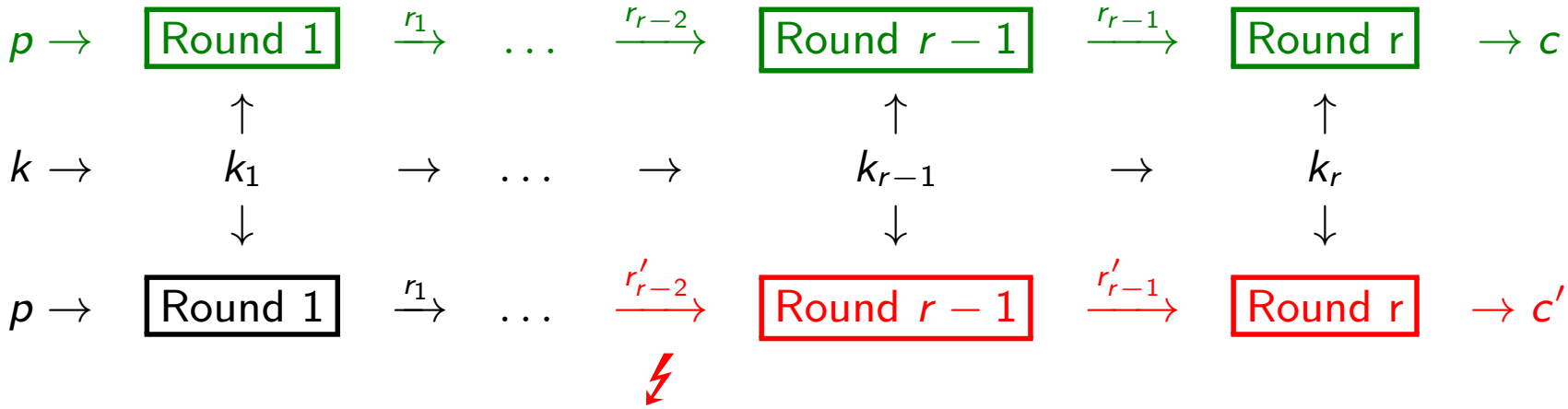
Figure: Laser Station 2 by riscure

Algebraic Fault Attacks



Scenario I

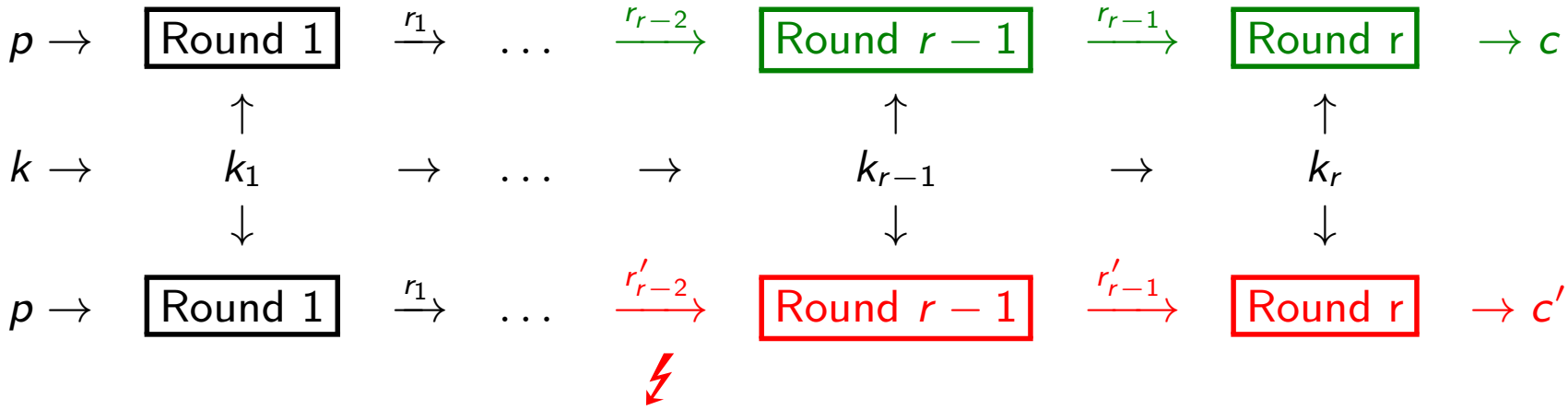
- The red and green part with $r_{r-2} + r'_{r-2} = \delta$ is algebraically represented.
- The solution give us unique k_{r-1} and k_r.



Scenario I

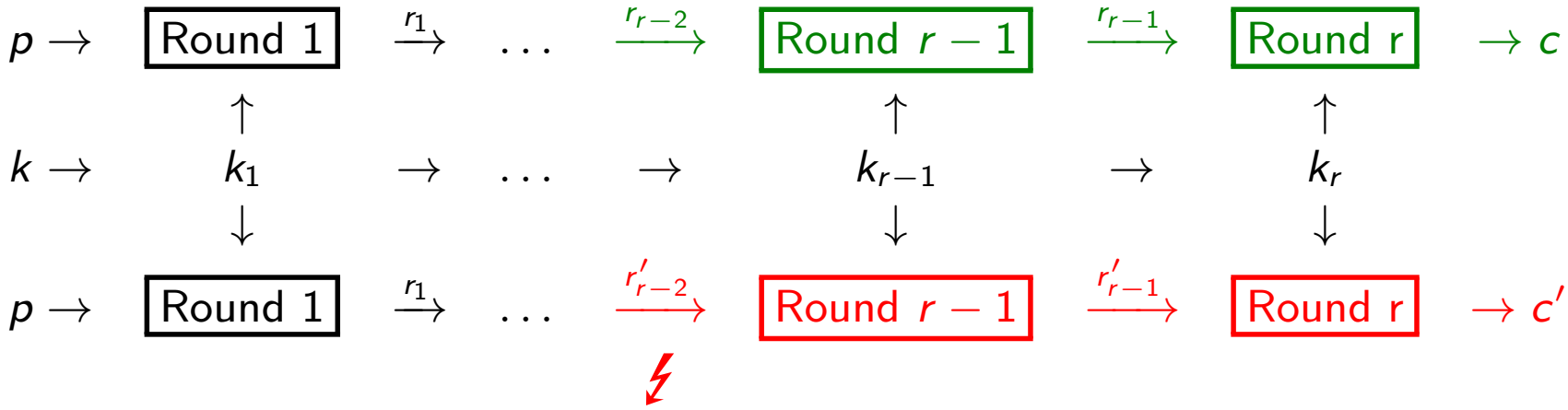
- The red and green part with $r_{r-2} + r'_{r-2} = \delta$ is algebraically represented.
- The solution give us unique k_{r-1} and k_r.

Algebraic Fault Attacks



Scenario II

- The red and green part with $r_{r-2} + r'_{r-2} = \delta$ is algebraically represented.
- The solution give us the set of possible candidates for k_{r-1} and k_r.



Scenario II

- The red and green part with $r_{r-2} + r'_{r-2} = \delta$ is algebraically represented.
- The solution give us the set of possible candidates for k_{r-1} and k_r.

Final Remarks

UNIVERSITÄT PASSAU

- There are many possibilities how to form polynomials (e.g. meet-in-the-middle)
- There are many possibilities how to perform the attack (e.g. faults in the key schedule).
- The attack can be generalized to more fault injections.
- The fault equations are derived **automatically** from the algebraic description of the cipher.
- This concept can be easily generalized to other cryptographic primitives.

Final Remarks

UNIVERSITÄT PASSAU

- There are many possibilities how to form polynomials (e.g. meet-in-the-middle)
- There are many possibilities how to perform the attack (e.g. faults in the key schedule).
- The attack can be generalized to more fault injections.
- The fault equations are derived **automatically** from the algebraic description of the cipher.
- This concept can be easily generalized to other cryptographic primitives.

Final Remarks

UNIVERSITÄT PASSAU

- There are many possibilities how to form polynomials (e.g. meet-in-the-middle)
- There are many possibilities how to perform the attack (e.g. faults in the key schedule).
- The attack can be generalized to more fault injections.
- The fault equations are derived **automatically** from the algebraic description of the cipher.
- This concept can be easily generalized to other cryptographic primitives.

Final Remarks

UNIVERSITÄT PASSAU

- There are many possibilities how to form polynomials (e.g. meet-in-the-middle)
- There are many possibilities how to perform the attack (e.g. faults in the key schedule).
- The attack can be generalized to more fault injections.
- The fault equations are derived **automatically** from the algebraic description of the cipher.
- This concept can be easily generalized to other cryptographic primitives.

Final Remarks

UNIVERSITÄT PASSAU

- There are many possibilities how to form polynomials (e.g. meet-in-the-middle)
- There are many possibilities how to perform the attack (e.g. faults in the key schedule).
- The attack can be generalized to more fault injections.
- The fault equations are derived **automatically** from the algebraic description of the cipher.
- This concept can be easily generalized to other cryptographic primitives.

UNIVERSITÄT PASSAU

Thank you!