SCIENCE
PASSION
TECHNOLOGY

# Gröbner Basis Attacks on Block Ciphers

Markus Schofnegger

31 Oct, 2019

## Introduction

- Core procedure:
  1. Represent the cipher (or components of it) as a set of equations
  2. Solve the resulting system for the unknown variables (e.g., key variables)
- Many attack strategies (as for other attacks: one has to be “creative”)
- Different solving techniques
- Complexities sometimes hard to estimate
- Strength of attacks greatly dependent on the structure of a cipher

# What is a Gröbner Basis? – Mathematical Background

- Given a set of equations $F = \{f_1, f_2, \ldots, f_n\}$, we convert it to a set of polynomials $P = \{p_1, p_2, \ldots, p_n\}$ (e.g., $x_1 + x_2 = x_3 \rightarrow x_1 + x_2 - x_3$)
- The set of solutions for $F$ is precisely the set of solutions for $P$ such that $p_1 = 0, p_2 = 0, \ldots, p_n = 0$ (this set of solutions is called an algebraic variety)
- Crucial point: the varieties of $P$ and Ideal$(P)$ are the same, which means they have the same solutions
  - ... but ideals are too large to use them efficiently

# What is a Gröbner Basis? – Mathematical Background cont.

**Definition (Gröbner Basis)**

A Gröbner basis of an ideal is a polynomial equation system with the same variety and which is easier to solve.

- Computing a Gröbner basis for an ideal can be computationally expensive
- Algorithms involve polynomial divisions
  - Use the leading terms of the polynomials
  - The term order describes how the terms in a polynomial are ordered and what the leading term is
  - Huge impact on the efficiency of the computation

## What is a Gröbner Basis? – Mathematical Background cont.

**Lemma (Triangular Shape)**

The reduced Gröbner basis $G = \{g_1, g_2, \ldots, g_n\}$ (in a specific term order) generating the zero-dimensional ideal $I$ is of the form

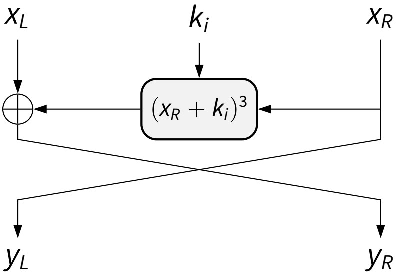
$$\begin{aligned} g_1 &= {x_1}^d + h_1(x_1), \\ g_2 &= x_2 + h_2(x_1), \\ &\vdots \\ g_n &= x_n + h_n(x_1), \end{aligned}$$

where $h_i$ is a polynomial in $x_1$ of degree at most $d - 1$.

- Note that $g_1$ is now a univariate equation and we can solve it by factorization!
- Use the result to solve for the other variables

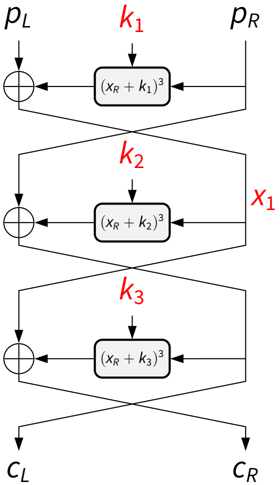
# First Target: The $\mathcal{PURE}$ Block Cipher

- Variant of the $\mathcal{KN}$ Feistel cipher proposed in 1995 [NK95] to be *provably resistant* against differential and linear attacks
- 64-bit blocks, 192-bit key $k = (k_i)_{i=1}^{6}$ with $k_i \in \mathbb{F}_{2^{32}}$
- Simplified round function (6 rounds in total):




- Computation of $x^3$ in $\mathbb{F}_{2^{32}}$

# Gröbner Basis Attack on the 3-Round $\mathcal{PURE}$ Cipher




- Our key variables are $k_1, k_2, k_3$
- We introduce an additional intermediate variable $x_1$ for our equations
- The system of equations describing the cipher is then

$$
\begin{aligned}
x_1 + (p_R + k_1)^3 + p_L &= 0,\\
c_L + (x_1 + k_2)^3 + p_R &= 0,\\
c_R + (c_L + k_3)^3 + x_1 &= 0
\end{aligned}
$$

($p_L, p_R, c_L, c_R$ are known)

- But there is a problem...

# Gröbner Basis Attack on the 3-Round $\mathcal{PURE}$ Cipher cont.

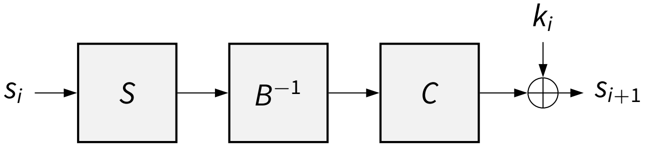
- We have 3 equations in 4 variables (our system is *underdetermined*)
- Simple solution: Use a second (plaintext, ciphertext) pair
  - Introduce a new variable $x_2$ for the second pair ($k_1$, $k_2$, $k_3$ stay the same)
- Add equations:

$$\begin{aligned} x_2 + ({p_R}^{(2)} + k_1)^3 + {p_L}^{(2)} &= 0, \\ {c_L}^{(2)} + (x_2 + k_2)^3 + {p_R}^{(2)} &= 0, \\ {c_R}^{(2)} + ({c_L}^{(2)} + k_3)^3 + x_2 &= 0 \end{aligned}$$

- Now we have 6 equations in 5 variables and we can solve it!
- Result: 96-bit key $k = (k_1, k_2, k_3)$ found in under 1 second on a normal laptop

# Second Target: The JARVIS Block Cipher

- Block cipher proposed in 2018 for "algebraic" use cases [AD18]
- $n$-bit blocks and keys
- Simple round function:




- $S$ computes the inverse, i.e., $S(x) = x^{-1}$
- $B$ and $C$ are low-degree affine polynomials

# Rewriting the Inverse Function

## Example (3-bit S-Box)

| $x$ | 0x0 | 0x1 | 0x2 | 0x3 | 0x4 | 0x5 | 0x6 | 0x7 |
|---|---|---|---|---|---|---|---|---|
| $S(x)$ | 0x0 | 0x1 | 0x5 | 0x6 | 0x7 | 0x2 | 0x3 | 0x4 |

- Over $\mathbb{F}_{2^3}$, this S-box computes:

$$S(x) = x^{2^n-2} = x^6 = \begin{cases} 0 & x = 0 \\ x^{-1} & \text{otherwise} \end{cases}$$

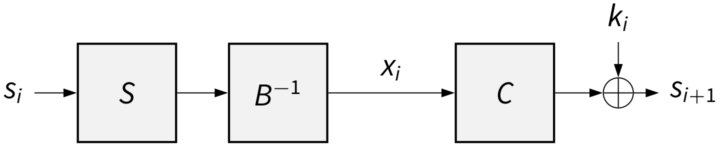
- Since $S$ computes the inverse of $x$ in $\mathbb{F}_{2^3}$ for all $x \neq 0$, we can also write $\forall x \neq 0 : x \cdot y = 1$ (now a degree-2 equation instead of a degree-6 one!)
  - For sufficiently large block sizes, we can assume that $x \neq 0$ with high probability

# Attack Idea

- Rewrite the inverse function as a low-degree function
- $B$ and $C$ have only low degree
- Introduce intermediate variables
  - Avoid forward computation of the inverse
  - Avoid forward computation of (high-degree) $B^{-1}$

# Introducing the Variables

- New variables $x_i$:




- New equation for two consecutive rounds:

$$(C(x_i) + k_i) \cdot B(x_{i+1}) = 1$$

  for $1 \leqslant i \leqslant r - 1$ (recall that $S$ computes the inverse)

- Two more equations for plaintext and ciphertext, and equations for round keys
- At the end: $2r + 1$ equations in $2r + 1$ variables

# Complexity of the Attack

- There exist complexity estimations for the case in which the number of equations equals the number of variables
- Unfortunately, complexities are too high when using this approach
  - For example, 6 of 12 rounds of 128-bit JARVIS already need around $2^{120}$ computations
- So... what can we do?
  - Reduce the number of variables!
  - Describe every round key in terms of the master key
  - Skip every second intermediate variables

# Relate Round Keys to the Master Key

- Two consecutive round keys are related by

$$k_{i+1} = \frac{1}{k_i} + c_i$$

- Therefore, each round key is a rational function of the master key $k_0$ in degree 1:

$$k_{i+1} = \frac{\alpha_i \cdot k_0 + \beta_i}{\gamma_i \cdot k_0 + \delta_i}.$$

- $\alpha_i$, $\beta_i$, $\gamma_i$, and $\delta_i$ are constants, and can be precomputed

## Skipping Intermediate Variables

- For each intermediate variable $x_i$, note that:

$$B(x_i) = \frac{1}{C(x_{i-1}) + k_{i-1}}, \qquad C(x_i) = \frac{1}{B(x_{i+1})} + k_i$$

- We find low-degree affine polynomials $D$ and $E$ such that

$$D(B) = E(C)$$

- Applying these yields

$$D\left(\frac{1}{C(x_{i-1}) + k_{i-1}}\right) = E\left(\frac{1}{B(x_{i+1})} + k_i\right)$$

- Now we can remove every second variable!

# Complexity of Improved Attack

- Equations for the plaintext and ciphertext have to be added
  - In total, we have $\frac{r}{2} + 1$ equations and the same number of variables
  - New equations have slightly higher degrees (applications of $D$ and $E$)
- Complexity estimates for Jarvis instances:

| $r$ | $n_v$ | Complexity in bits |
|---|---|---|
| 10 (Jarvis-128) | 6 | 100 |
| 12 (Jarvis-192) | 7 | 119 |
| 14 (Jarvis-256) | 8 | 138 |
| 16 | 9 | 156 |
| 18 | 10 | 175 |
| 20 | 11 | 194 |

## There's more to it...

- Same strategy works for FRIDAY, a hash function based on JARVIS
    - By exploiting the internals of the hash function, the attacks becomes even better
- Full details given in the paper [ACG+19]
- Maybe the strategies are applicable to other similar designs as well?
- Different perspectives
    - Designer: Make one step of the attack sufficiently expensive
    - Attacker: Evaluate complexities of *all necessary steps*
    - ... both are not trivial (active research, see e.g. [ST19])

# Gröbner Bases – Complexity

- Reminder: Computing a Gröbner basis only one of the steps in the attack
  - In most cases, we expect it to be the most expensive one
  - Complexity difficult to estimate (depends on number of variables, number of equations, degrees, ...)
  - Last step (factorization) might also be a bottleneck
- Most theoretic results apply to “random” systems
  - However, cryptographic schemes tend to be well-structured
- Advantage: The attack does not need many (plaintext, ciphertext) pairs (sometimes, even one pair is enough!)
- Protection (simplified): Force attacker to use many variables, increase degrees of equations

# Gröbner Basis Attacks – Summary

- In short: simplify an equation system and solve it
- Recently, they gain importance due to new ciphers which exhibit a “nice” algebraic structure
    - Design of such algorithms is motivated by new use cases
    - Gröbner bases can provide strong attacks against such ciphers
- In general: difficult to apply Gröbner bases to bit-based schemes (i.e., working in $\mathbb{F}_2$)
    - Many variables
    - Approaches based on SAT solvers also efficient
    - See e.g. MQ challenge (https://www.mqchallenge.org/)

# References I


[ACG+19] Martin R. Albrecht, Carlos Cid, Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, and Markus Schofnegger. **Algebraic Cryptanalysis of STARK-Friendly Designs: Application to MARVELlous and MiMC.** ASIACRYPT (1). Lecture Notes in Computer Science. 2019.

[AD18] Tomer Ashur and Siemen Dhooghe. **MARVELlous: a STARK-Friendly Family of Cryptographic Primitives.** IACR Cryptology ePrint Archive 2018 (2018), p. 1098.

[NK95] Kaisa Nyberg and Lars R. Knudsen. **Provable Security Against a Differential Attack.** J. Cryptology 8.1 (1995), pp. 27–37.

[ST19] Igor Semaev and Andrea Tenti. **Probabilistic analysis on Macaulay matrices over finite fields and complexity of constructing Gröbner bases.** IACR Cryptology ePrint Archive 2019 (2019), p. 903.