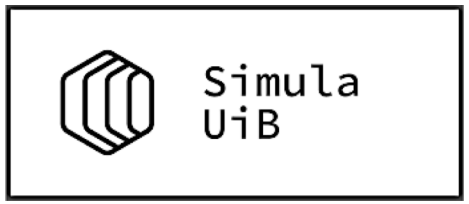
domain specific ciphers

Carlos Cid
Royal Holloway University of London
Simula UiB



block cipher research

- **before AES**: a handful of block ciphers available, eg DES, IDEA, Blowfish
- **after AES**: there are **100+** block ciphers to choose from.
  - more new designs appear every year (particularly *lightweight*), but the vast majority will never be used in applications.
  - AES works well in most non-constrained applications, is widely supported in crypto-libraries, and has special hardware instructions on many modern processors.
- **this talk**: look at ciphers for applications for which AES is not *that* good..

Alex Biryukov @alexcryptan · 10 Oct
Replying to @veorq
AES is Not great for IoT/Lightweight,

Not great if you need AEAD

and I would second - not great for Whitebox.

JP Aumasson @veorq · 9 Oct
Replying to @veorq
I'll start: in pay-TV there are unusual requirements, here's 2 cases:
- needed a cipher fast in hardware but super slow in software (led to DVB-CSA3 and custom things)
- needed a block cipher incorporating IP, to sue organizations cloning/emulating our code

2 1 13

JP Aumasson @veorq · 9 Oct
another one: 10 years ago in some custom RFID product, using a stream cipher was simpler and more efficient plus the customer wanted a custom design

1 3

JP Aumasson @veorq · 9 Oct
encountered the customer-wants-a-custom-design several times, most of the time for bad reasons; last time was a couple months ago and customer offered to pay us to design a custom cipher but I convinced them AES was just fine

3 7

JP Aumasson @veorq · 9 Oct
another case: some software protection frameworks rely on "white-box" and obfuscated versions of custom cipher to make it harder to reverse engineer than if it were AES; some ciphers better lend themselves to these techniques :)

1 5

JP Aumasson
@veorq

F#CK AES
what's your examples of use cases and scenarios where AES couldnt be used? (for too big/slow/notstandard/etc.)

2:38 pm · 9 Oct 2019 · Twitter Web Client

9 Retweets **36** Likes

Orr Dunkelman @CryptoOrrDun · 9 Oct
Replying to @veorq
For MPC protocols.

When you must use table-based implementation (lack of native support, need for speed) thus "enjoying" cache timing attacks.

When you need a block cipher to also be a base for hashing.

But most of the time, AES is a great solution.

2 4

JP Aumasson @veorq · 9 Oct
right I forgot the MPC/FHE/ZK circuits case (with things like LowMC)
/cc @zooko

hashing: what I've seen is "we only have AES how can we hash with it?", rather than "we have a blockcipher-based hashing mode, which cipher to pick?"

2 4

domain specific ciphers

- past few years have seen a number of new applications for symmetric-key cryptography – beyond traditional confidentiality and authentication in two-party communication.
  - in many cases requiring dedicated symmetric-key designs, to support and/or enable these new applications.
  - security goal may be defined based on the constraints of the application.
- Format Preserving Encryption (legacy systems)
- White-Box cryptography (obfuscation for deployment on untrusted systems)
- Algebraic ciphers for advanced applications (MPC, FHE, ZKP)

Format Preserving Encryption (FPE)

- **Format Preserving Encryption:** symmetric-key ciphers that encrypt plaintext in some particular format into the same format.
  - example: 16-digit credit card numbers, social security number, image files, or even ANSI C programs
  - important application: deployment in legacy systems, as drop-in replacement of plaintext values with their ciphertexts (eg retail systems handling CC numbers)
  - requirement: deterministic, tweakable, flexibility
  - off-the-shelf ciphers (eg AES) generally not suitable for non-binary formats

FPE – constructions

- generic FPE construction: ranking + cycle walking
  - ranking: bijection between D and Z_N
  - cycle walking: embed Z_N into GF(2^n), inducing permutation on Z_N from n-bit cipher
  - generally not efficient
- preferred design strategy
  - Feistel network over Z_N x Z_M , round function based on block cipher
    - Feistel is good if *enough* rounds are used
    - FPE as (AES) mode of operation
    - NIST standards (SP 800-38G): FF1 & FF3
    - problems with security, exploiting flexibility (small domain), size of tweak space, (small) number of rounds
    - efficiency: a few AES calls per round
- **research challenge**: non-Feistel, secure, efficient FPE designs

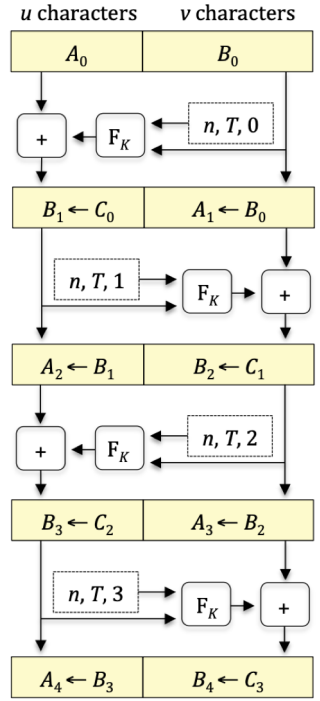
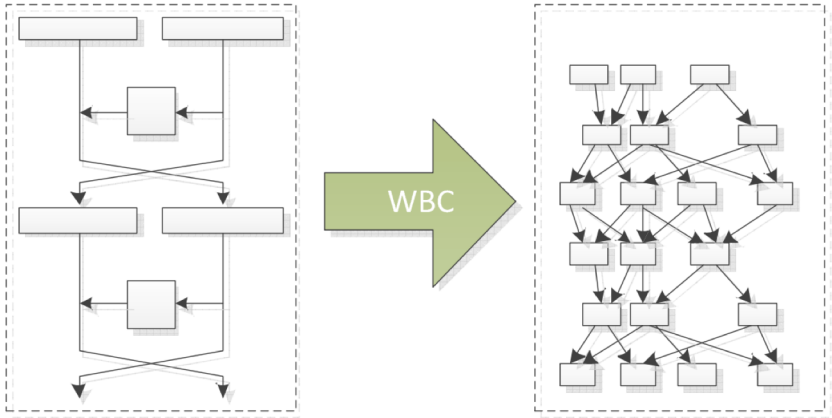


figure source: Draft NIST Special Publication 800-38G, Revision 1

white-box cryptography

- cryptographic systems for deployment on *untrusted systems.*
  - solution: embed key into the cipher implementation, and obfuscate it.
  - applications: card emulation into mobile phone; DRM systems
- proposed approach: transform implementation into collection of TLUs
- WB-ing traditional ciphers (eg AES) is hard!
  - strong diffusion means number of TLUs soon explodes
  - WhibOx challenges: 100+ white-boxed AES-128 implementations... all broken



- call for dedicated, WB-friendly designs (maybe for specific threat model)
  - **research challenge**: security, acceptance

figure source: http://www.whiteboxcrypto.com/

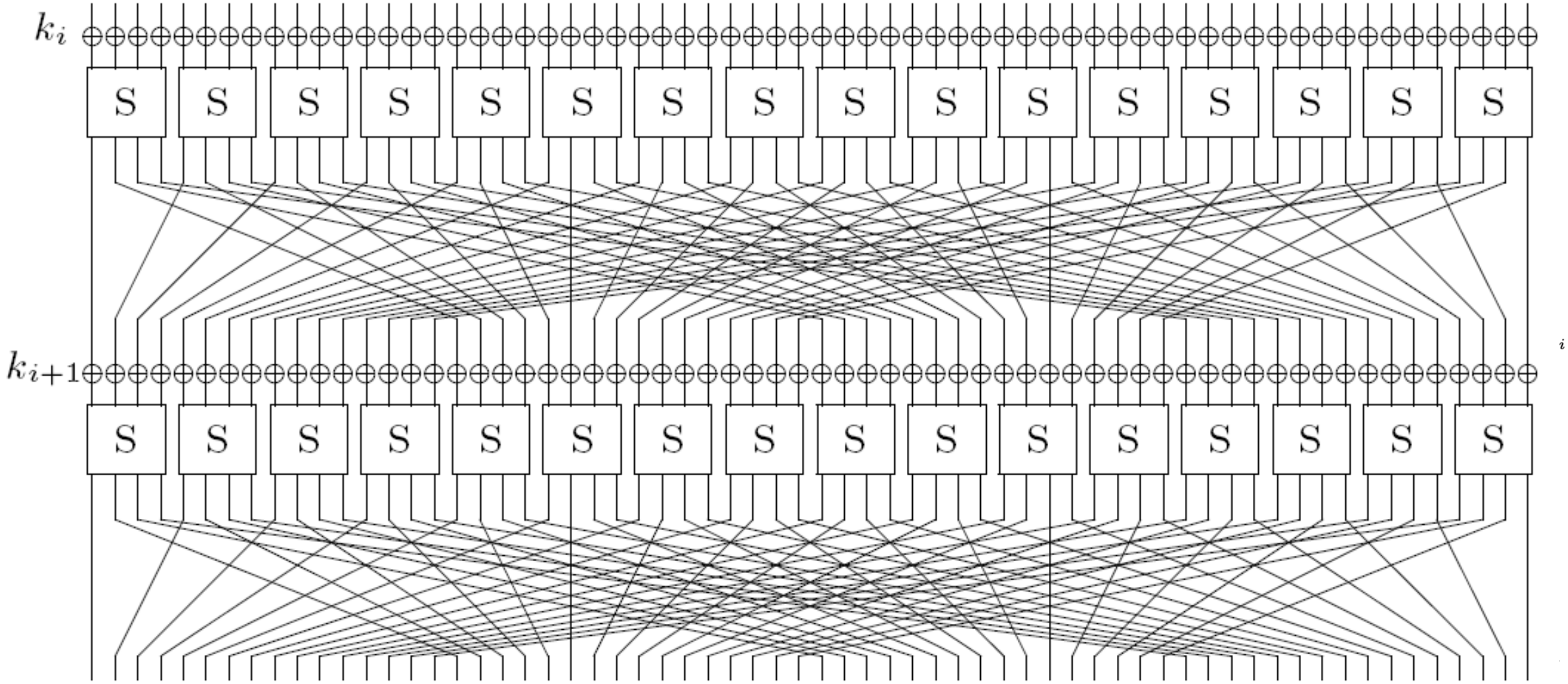
algebraic ciphers for advanced applications

- specialised designs for **emerging new applications** of symmetric cryptography: MPC, FHE, ZKPs
- ciphers typically aim to **minimize** some metric of relevance to the efficiency of these applications:
  - low multiplicative complexity and depth of (binary) circuit
  - simple algebraic structure, natively defined over a large finite field
- often the goal is not confidentiality, eg we may be interested in constructing collision-resistance hash functions.

ciphers for MPC and FHE

- symmetric-key designs (binary) **minimizing multiplicative complexity** (MC) and/or **multiplicative depth** (total and per-bit)
- in MPC and FHE applications, number of multiplications and the multiplicative depth of circuit strongly affect complexity (communication/computation), while linear operations (XOR) are *essentially* free!
  - applications: secure computation of encryption operation; hybrid FHE.
- AES is not a particularly suitable construction in these environments.
  - modern ciphers balance linear/non-linear components.
  - MPC/FHE ciphers call for a more unbalanced design approach.

ciphers for MPC and FHE – constructions



PRESENT: lightweight cipher, 64-bit blocks, 80/128-bit keys, 31 Rounds, 1075 GE.ISO standard for lightweight block cipher

ciphers for zk proof systems

- ZK (zero-knowledge) proof systems: schemes that allow *prover* to "convince" a *verifier* of a particular statement, eg
  "*I know an input x that produces y = F(x)*"
  such that the verifier learns nothing that it did not know before (apart of the validity of the statement)
  - properties: completeness, soundness and zero-knowledge
- proofs are typically done by producing an **encoded transcript** of the execution of F on x, which the verifier "queries" to become convinced the statement is true.

ciphers for zk proof systems

- modern popular application: deploying zk proof systems in blockchains, to provide **anonymity** (to transaction parties) and **confidentiality** (to transaction amounts)
  - proofs are produced and stored in the blockchain, which users can verify to get convinced of integrity of blockchain data
    - examples: Zcash, Monero
  - we want proof systems that produce compact proofs, can be efficiently verified, and scale well.
- typical statement to be proved:

"I know a leaf of a Merkle tree with root y"

ie transcripts will correspond to repeated invocations of cryptographic hash functions when running through an authentication path on a MT .

ciphers for zk proof systems

- modern proof systems transform the *computational execution* into an *algebraic circuit:* represent execution as a set of **algebraic constraints**, ie equations over a finite field, satisfied by a valid transcript.
  - these will be then represented by a large univariate polynomial that is used to convince the verifier.
  - the size of the transcript and the number and degree of these equations directly affect the efficiency of the zk proof systems – **you want to minimize them!**
- therefore, when computation is Merkle tree traversing, we would like **ZKP-friendly** hash functions:
  - natively defined via algebraic operations of low degree on a small state of variables over a finite field, executed a small number of times.

zk-SNARK

- *zero-knowledge Succinct Non-interactive Argument of Knowledge.*
  Main features:
  - succinctness (fast verification and small proofs) and non-interactive (does not require interaction between prover and verifier).
  - requires CRS shared between prover and verifier (trusted set-up)
  - security assumption: knowledge of exponent assumption.
- zk-SNARK converts computation into arithmetic circuit, with bilinear gates over finite (prime) field
  - then construct **Rank 1 Constraint System** (R1CS), which can be used to verify the assignments into the circuit satisfy the constraints of the gates (ie correct computation)
  - these are bundled together into very large univariate polynomials (**QAP form**), and verification is done by checking t(x).h(x) = r(x).u(x)
    - succinctness means that verifier only needs to check equality for a random secret value x = s.

zk-SNARK complexity

- complexity of zk-SNARK (size of proofs and verification time) are directly affected by the size of the polynomials t(x), h(x), r(x) and u(x).
  - which follow from the size of the R1CS system

- for notable application: in Zcash, shielded transactions don't use digital signatures, but rather employ zk-SNARK to prove transactions are valid and in the Merkle tree that stores all coins.
  - so we want to use hash functions that minimize number of multiplications in GF(p), reducing the number of constraints and the degree of the polynomials.

zk-STARK (Ben-Sasson et al. in 2018)

- *zk Scalable and Transparent Argument of Knowledge.* Main features:
  - scalability: verification time is poly-logarithm in the size of the circuit; proving time is quasi-linear.
  - transparency, post-quantum security.
  - prover 10x faster than SNARKS; verifier 2x faster; but proof size 100x larger!
- given computation over T cycles operating on state of w elements of $GF(2^n)$, the arithmetization phase consists of:
  - algebraic execution trace (AET): array with T . w elements representing execution state
  - algebraic intermediate representation (AIR): generalization of R1CS from SNARKS, degree d polynomials describing transaction relations.
  - low degree extension (LED): convert AET/AIR into single univariate polynomial

zk-STARK: example I

Example. how to construct the AIR for a simple example: let $\mathcal{C}$ be the computation of the Fibonacci sequence (recall, starting with $s_0 = 1$ and $s_1 = 1$, the Fibonacci sequence is defined by having each element of the sequence as the sum of the two previous ones, i.e. $s_i = s_{i-1} + s_{i-2}$). Assume we will run the computation for T steps. We use a sufficiently large finite field in our representation so that the sequence does not "wrap around" when executing the computation.
We may describe the Algebraic Execution Trace (AET) as a $(T+1) \times w$ array, with two algebraic registers in each cycle ($w = 2$), and the first row representing the initial state:

$$\begin{array}{cc} 1 & 1 \\ 1 & 2 \\ 2 & 3 \\ 3 & 5 \\ 5 & 8 \\ 8 & \vdots \end{array}$$

For each cycle, we have X_0, X_1 representing the state, and Y_0, Y_1 the next state. We can define the Algebraic Intermediate Representation (AIR), which describes the transition relation constraints:

$$\{X_0 + X_1 - Y_1, X_1 - Y_0\}.$$

In this representation, the AET has size $(T+1) \cdot w$, with entries in $\mathbb{F}_p$, and the AIR consists of two linear polynomials involving two consecutive states.

example adapted from: Abdelrahaman et al. Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. https://eprint.iacr.org/2019/426

zk-STARK: example II

We may loosen up our definition of AIR to have polynomials the transition relation involving more than two consecutive states. In this case, we could describe the AET of the computation of the Fibonacci sequence as a $(T+2) \times w$ array, by having one algebraic register in each cycle ($w = 1$), with the first two rows representing the initial states:

$$\begin{matrix} 1 \\ 1 \\ 2 \\ 3 \\ 5 \\ \vdots \end{matrix}$$

Now for each cycle, we have X representing the current state, and Y and Z representing the next, and second next states, respectively. Then we can define the AIR describing the transition relation constraint by a single polynomial:

$$\{X + Y - Z\}.$$

example adapted from: Abdelrahaman et al. Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. https://eprint.iacr.org/2019/426

zk-STARK complexity

STARK efficiency metric:

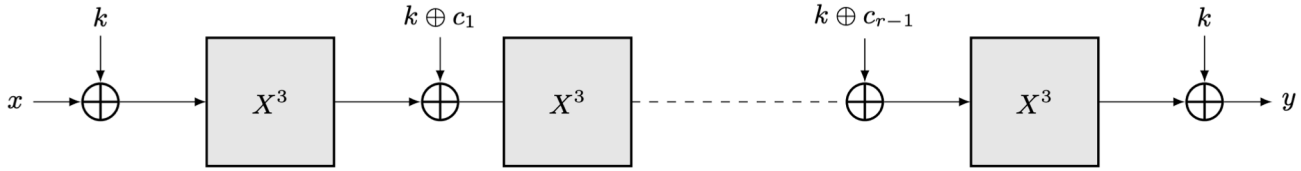
$$\text{STARK-Complexity} = T \cdot w \cdot D$$

thus for typical application – Merkle tree traversing – we are looking for hash functions that minimize complexity above:

- small state of elements of $GF(2^n)$
- defined via algebraic operations over $GF(2^n)$ of low degree
- a small number of rounds

MiMC (Albrecht et al. 2016)

- algorithm operating natively over a large field GF(q), aiming to Minimize Multiplicative Complexity.
  - based on the power function $f(x) = x^3$ over GF(q), for $q = 2^n$ or prime
  - block cipher uses an iterated design, with round function $F(x) = (x + k + c_i)^3$, for r rounds, where k is the secret key, and ci are round constants



  - permutation: fix key as zero (in GF(q))
  - hash function (MiMChash): use permutation in the sponge framework
- ps:
  - field has to be selected such that cubic map is invertible.
  - MiMC-n/n: cipher defined over $GF(2^n)$, ie n-bit block and key sizes
  - MiMC-2n/n: Feistel version (2n-bit block and n-bit key size): cubic as round function

MiMC – security

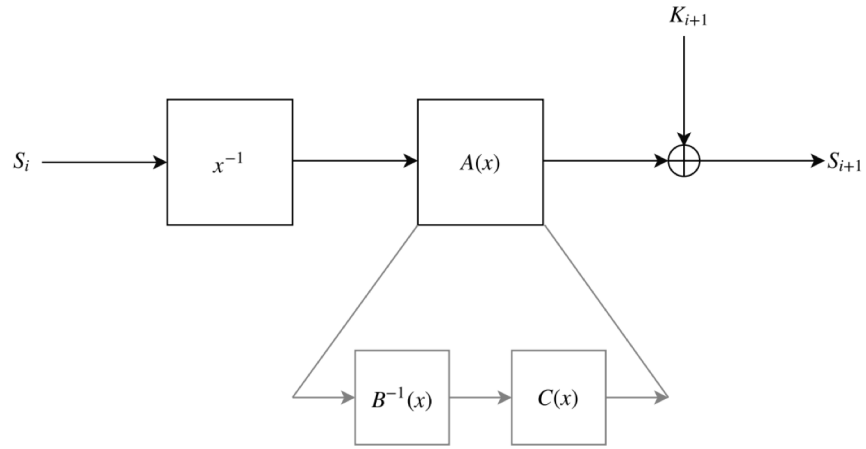
- conventional (statistical) cryptanalysis does not apply: differential, linear, etc, cryptanalysis are not effective after a few rounds r.
- the only (foreseen) threats are algebraic cryptanalytic techniques, attempting to explore the simple algebraic structure.
  - interpolation attack
  - algebraic polynomial attack
- complexity of these two attacks are used to derive the number of rounds r in MiMC

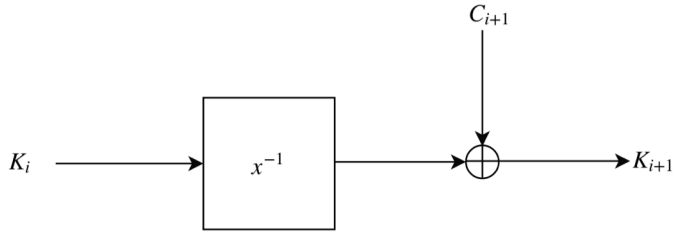
MARVELlous STARK-friendly ciphers

- family of cryptographic algorithms specifically designed for STARK efficiency proposed by Ashur and Dhooghe in the autumn 2018
  - first members of the family: **JARVIS** (block cipher) and **FRIDAY** (hash function)
    - announced at Ethereum DevCon 4 (Nov 2018)
    - considered for deployment in blockchain systems (eg Zcash, Ethereum)

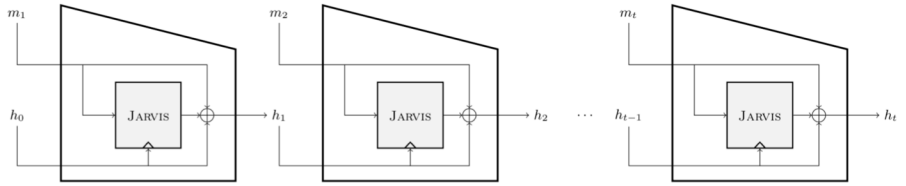
- similar design to MiMC, but much lower number of rounds:
  - ciphers use inversion in a large binary field for its non-linear operation
  - thus regarded (and *promoted*) as based/related to AES

JARVIS and FRIDAY





- JARVIS:
  - iterated block cipher with one round consisting of:
    - inversion of GF(2^n)
    - composition of two F_2-affine operators
    - subkey addition (key schedule uses inversion only)
  - defined for n=128, 160, 192 and 256, with **r = 10, 11, 12 and 14 rounds,** resp. (same as AES!)
- FRIDAY:
  - hash function using using FRIDAY in the Miyaguchi-Preneel mode of operation as a compression function in the MD scheme



figures source: MARVELlous paper -- https://eprint.iacr.org/2018/1098.pdf

JARVIS: design rationale & security

- inversion over GF(2^n) results on algebraic constraint of degree two: $xy + 1 = 0$
  - in fact, to account to the zero input, we must have $x^2y + x = 0$
  - low degree + small number of rounds = STARK-friendliness
- however a MiMC-like cipher using inversion rather than cubic map is insecure – interpolation attack requires 4 p/c pairs (regardless of number of rounds!!)
  - B and C are F_2-affine operations, can be expressed by (linearized) polynomials over F_2^n
  - choice of B, C of degree four (=low-degree algebraic constraints)…
  - …but JARVIS uses $B^{-1} \cdot C$ to reach high algebraic degree over F_2^n
  - traditional cryptanalysis doesn't apply (eg differential/linear cryptanalysis), only route of analysis: algebraic techniques

digression: algebraic cryptanalysis

- in the context of symmetric-key cryptography, algebraic cryptanalysis is typically referred as Algebraic Attacks
  - set up and solve a system of equations arising from a stream cipher or block cipher, to recover the encryption key (or other secret information, eg stream cipher secret state).
  - more generally however, algebraic cryptanalysis: study algebraic systems to obtain some non-trivial insight into the algorithm.
- two well-defined tasks/challenges for the cryptanalyst:
  1. how to construct the system of equations.
  2. how to solve the resulting system (or obtain some insight into the cipher).

algebraic cryptanalysis of JARVIS and FRIDAY

joint work with M. Albrecht, L. Grassi, D. Khovratovich, R. Lueftenegger, C. Rechberger and M. Schofnegger, ASIACRYPT 2019

- goal: mounting a direct algebraic attack against JARVIS
- recall the two steps in mounting an algebraic attack:
  1. Describe the cipher as a system of polynomial equations
  2. Solve the system using a computer algebra method
- for step 2, the best known method is to compute the associated Gröbner basis, using the F4/F5 GB algorithms
  - these algorithms work by constructing sparse matrices of increasing size (corresponding to increasing degree polynomials)
  - degree of regularity: largest degree reached
  - complexity:

$$\mathcal{C}_{\mathrm{GB}} \in \mathcal{O}\left(\binom{n_v + D_{\mathrm{reg}}}{D_{\mathrm{reg}}}^{\omega}\right)$$

algebraic cryptanalysis of JARVIS and FRIDAY

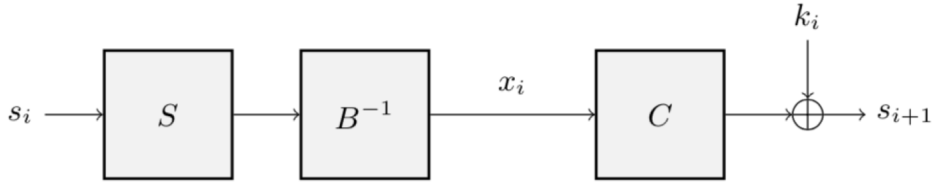
- **first attempt**, with natural system (one new variable per operation): prohibitively expensive!
- **second attempt**: one variable per round
  - system: 2r + 1 equations on 2r + 1 variables of degree 2-8

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

$$B(x_1) \cdot (p + k_0) = 1,$$

$$C(x_r) = c + k_r$$

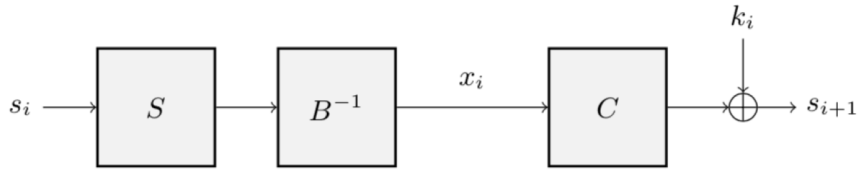
$$(k_{i+1} + c_i) \cdot k_i = 1$$



  - D_{reg} = 8r + 1
  - we can break up to 6 rounds of JARVIS-128 (complexity ~ 2^{120})

can we do better?

algebraic cryptanalysis of JARVIS and FRIDAY



- third attempt:
  - look for F2-affine operators, low degree linearized affine polynomials, D,E, such that D(B) = E(C), and so

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

  - we were able to find D,E, and thus have one variable for each two rounds
    - we also express all subkeys as a rational function of the master degree of degree 1.
  - resulting on a system describing r-round JARVIS with
    - r/2 – 1 equations of deg=40, one of deg=24, and one of deg=5, over r/2+1 variables
    - D_{reg} = 39 (r/2) – 11
      (**higher** than 8r + 1, but with fewer variables -- r/2+1 compared to 2r + 1)

algebraic cryptanalysis of JARVIS and FRIDAY

- we mount successful key recovery attacks on JARVIS for over 20 rounds!

ps1: we assume w=2.8, but also give complexity for w=2 in brackets.

r	n_v	D_{reg}	Complexity in bits
6	4	106	63 (45)
8	5	145	82 (58)
10 (JARVIS-128)	6	184	100 (72)
12 (JARVIS-192)	7	223	119 (85)
14 (JARVIS-256)	8	262	138 (98)
16	9	301	156 (112)
18	10	340	175 (125)
20	11	379	194 (138)

- the attack can be extended to pre-image recovery against FRIDAY

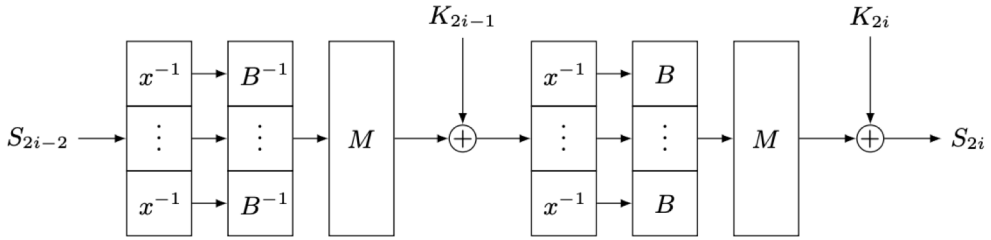
r	n_v	D_{reg}	Complexity in bits
6	3	94	48 (34)
8	4	125	65 (47)
10 (JARVIS-128)	5	156	83 (59)
12 (JARVIS-192)	6	187	101 (72)
14 (JARVIS-256)	7	218	118 (85)
16	8	249	136 (97)
18	9	280	154 (110)
20	10	311	172 (123)

ps2: we run several experiments on reduced round-version, and the attacks works better in practice than estimated in theory.

ZKP-friendly ciphers

- impact: designers abandoned the MiMC-like AES-based design.
  - however the problem of designing secure STARK/SNARK-friendly ciphers remains an area of a lot of contemporary interest (and potentially large rewards!)

- MARVELlous family has new members...
  - **Vision** and **Rescue**
  - new designs are moving remarkably close to AES...for example, one round of Vision



- other STARK-friendly designs from other teams:
  - HADES-MiMC
  - gMiMC

Abdelrahaman et al. Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. https://eprint.iacr.org/2019/426

conclusions

- a number of new (advanced) cryptographic applications call for new symmetric-key designs
  - new designs' goals often go beyond traditional confidentiality and authentication in two-party communication.
- particularly exciting area are algebraic ciphers for ZKP schemes
  - security of these ciphers are not well understood – more research required
  - after years being left on the backbenches of cryptanalysis, could algebraic attacks against block ciphers become relevant?

Thank you!