Refined Cryptanalysis of the GPRS Ciphers GEA-1 and GEA-2

Dor Amzaleg and Itai Dinur

Ben-Gurion University

EUROCRYPT 2022

GEA-1 and GEA-2

- GPRS is a **mobile data standard**, widely deployed in the **early 2000s**
  - Established by European Telecommunications Standards Institute (**ETSI**)
- **Encryption** protects against eavesdropping
- Initially two **proprietary** stream ciphers **GEA-1** and **GEA-2** were used
- At EC'21, [BDL+21] presented **first public analysis** of **GEA-1** and **GEA-2**
  - **First disclosure** of specification

[BDL+21] Beierle, C., Derbez, P., Leander, G., Leurent, G., Raddum, H., Rotella, Y., Rupprecht, D., Stennes, L.: Cryptanalysis of the GPRS Encryption Algorithms GEA-1 and GEA-2.

[BDL+21] (GEA-1)

- **GEA-1** and **GEA-2** have 64-bit session key
- For **GEA-1**, [BDL+21] described **weakness** allowing to recover the session key in time $T=2^{40}$, memory S = 44 GiB
  - Requires 65 bits of known **keystream** (e.g., from headers)
- **Weakness** believed to be **intentional**
  - Presumably due to **export regulations** on cryptography in 1998
- ETSI already **prohibited** support of **GEA-1** in **2013**

- [BDL+21] noticed that modern mobile phones still supported **GEA-1**
  - (Hopefully) fixed by now by adding test cases

[BDL+21] (GEA-2)

- **GEA-2** does not have significant weakness as **GEA-1**
- However [BDL+21] described attack with $T=2^{45}$, $S=2^{32}$
- Main practical obstacle: requires **all** 12800 keystream bits used to encrypt GPRS frame
  - Also presented data-time tradeoff: beats exhaustive search from about 1468 bits

Our Results

- Improve and refine results of [BDL+21]
- GEA-1:
- Reduce memory complexity by >8000 from 44 GiB to 4 MiB
  - Time remains $T=2^{40}$
- Attack runs on **laptop** in 2.5 hours
  - [BDL+21] ran it on cluster
- Motivation:
  - In early 2000's, high-memory attack **still not trivial** to run at scale
  - **Better understand** exact security of (once) **widely used** ciphers

- For GEA-2, present two attacks:
  - 1) Improved data-time tradeoff for data <7000 (consecutive keystream bits)
  - 2) Improved memory complexity in case data is fragmented
- Use **techniques** for solving k-XOR problems

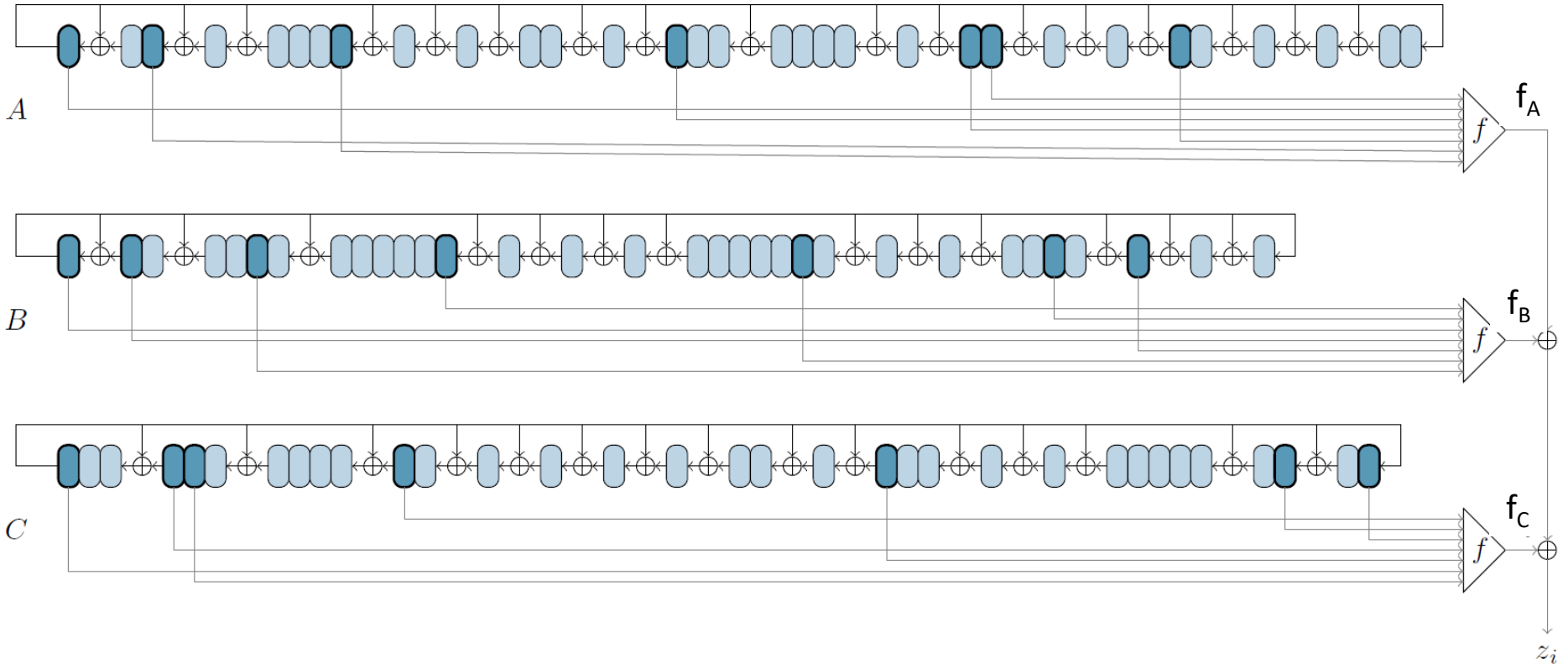
Outline

- **GEA-1 specification + Attack**
- GEA-2 specification + Attack 1
- Conclusions

GEA-1 Initialization

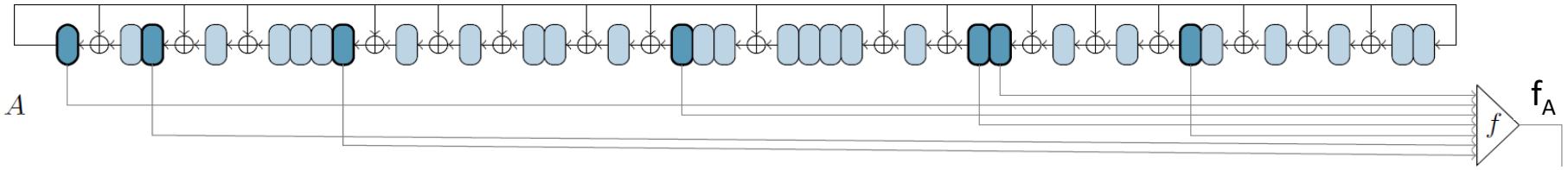
- Initialization: inputs - 64-bit **session key**, 32-bit (packet) **IV**
  - 1) 64-bit **key**, 32-bit **IV** used to compute 64-bit **seed**
  - 2) 64-bit **seed** used to initialize 96-bit **state** (31+32+33).

    Map $M:\{0,1\}^{64} \rightarrow \{0,1\}^{96}$ is **linear**



GEA-1 Notation

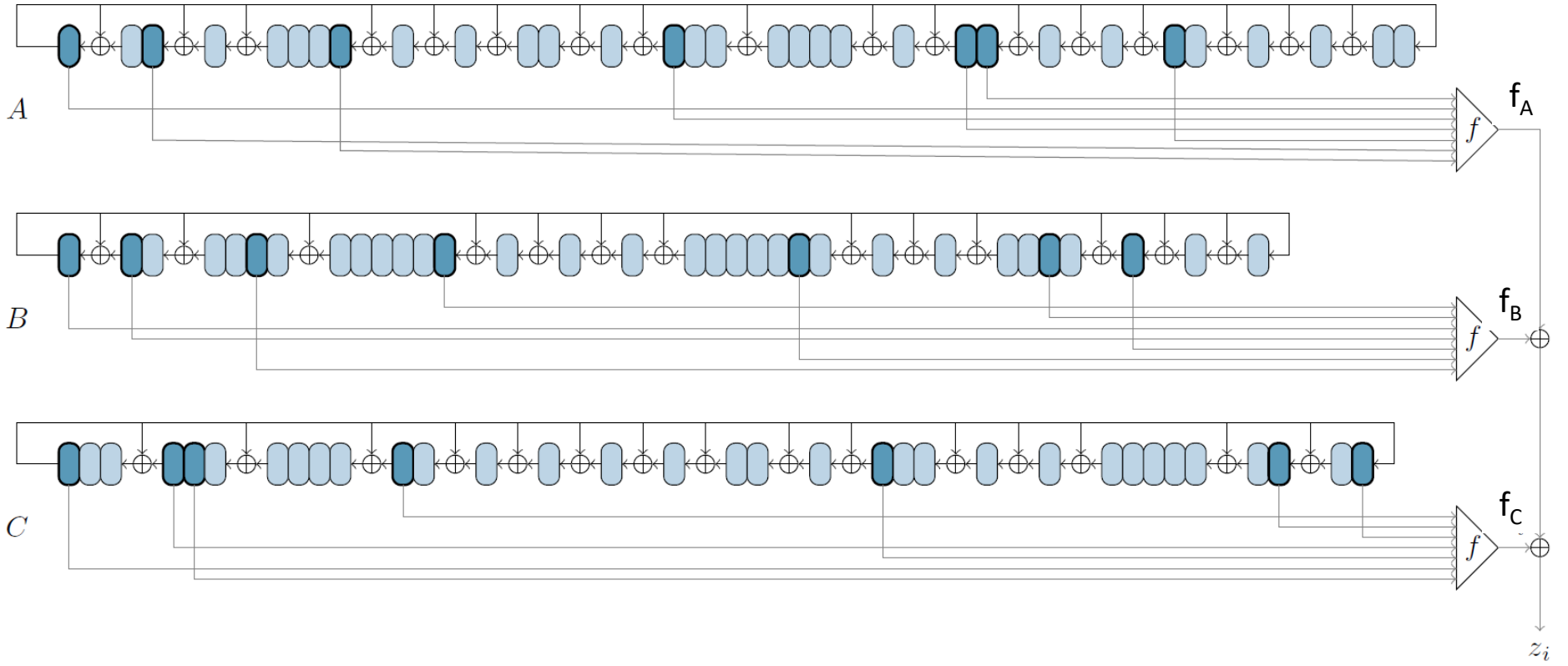
- For internal $\underline{A} \in \{0,1\}^{31}$ of A, $f_A(\underline{A})_{[\tau]}$ is function that computes next τ output bits from state $\underline{A}$:
- For $i=1,\ldots,\tau$:
  - 1) Apply $f:\{0,1\}^7 \rightarrow \{0,1\}$ to (bits of) $\underline{A}$ to obtain $f_A(\underline{A})_i$
  - 2) Clock **LFSR** to **update** internal state



GEA-1 Keystream Generation

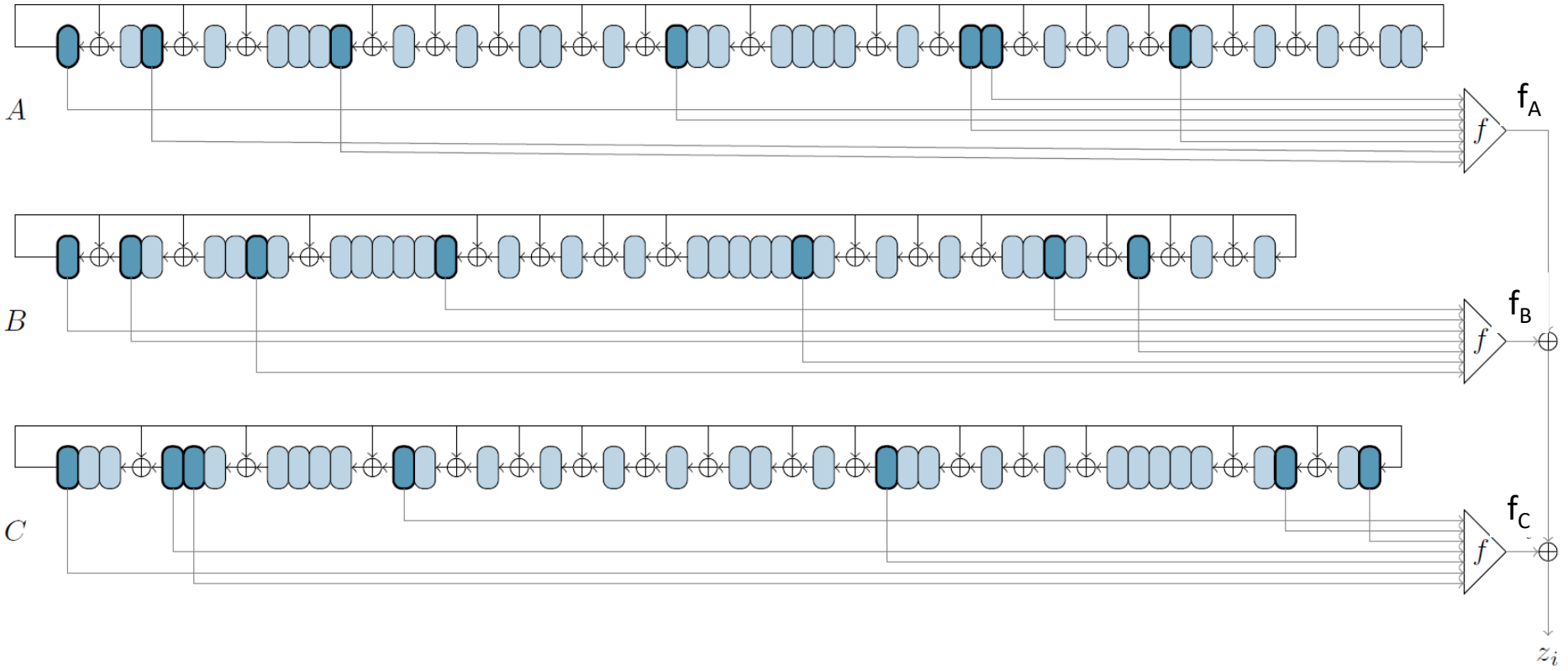
- **Keystream generation**
  - **Inputs** – 96-bit initial state (A,B,C), 12800-bit **packet** p

- - Compute **keystream** $z = (f_A(A) \oplus f_B(B) \oplus f_C(C))_{[12800]}$
- - Output ciphertext $c = p \oplus z$



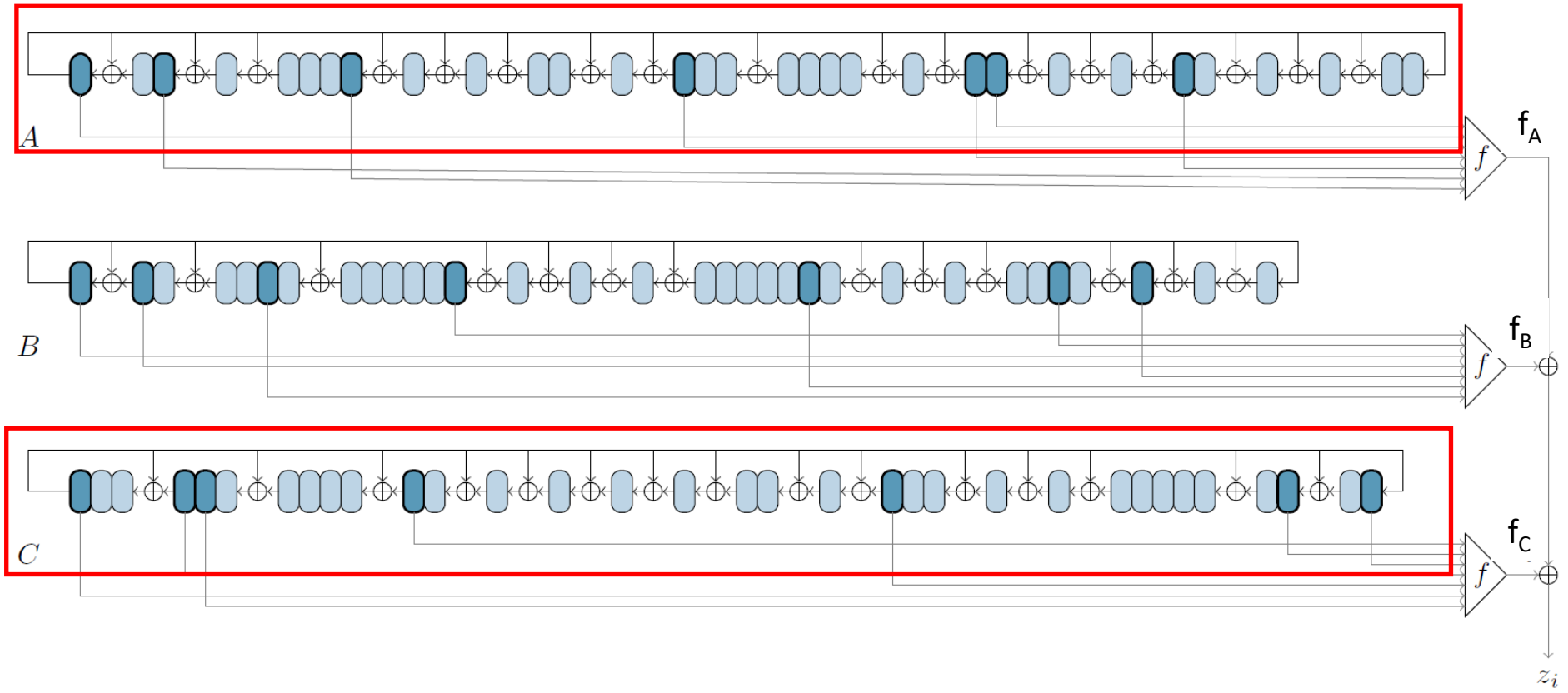
GEA-1 Attack Outline

- Attacker obtains 65-bit keystream z for **some packet**:
  - Eavesdrops to get c, computes $z = p \oplus c$ for known p
- Given z, runs (some) attack to recover initial state (A,B,C)
- **Inverts initialization** to obtain **seed** and then **session key**
- Can **decrypt** full **GPRS session**



GEA-1 Initialization Weakness [BDL+21]

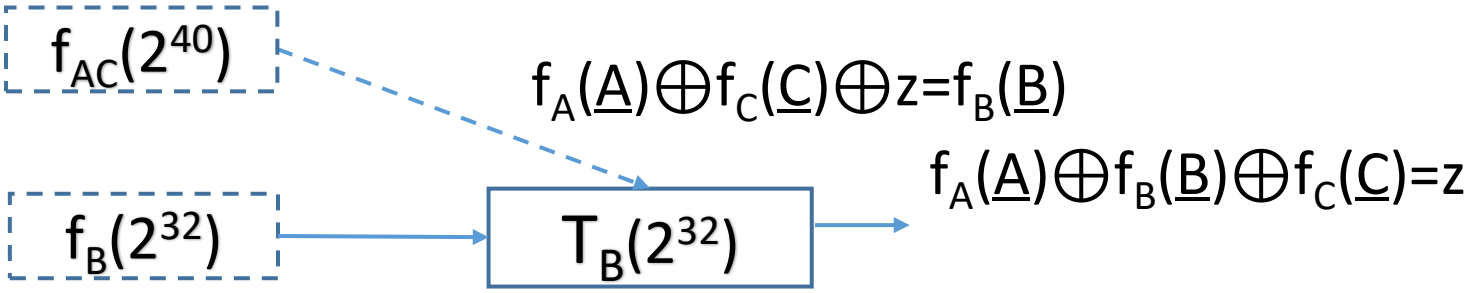
- Initialization: inputs - 64-bit session **key**, 32-bit (packet) **IV**
  - 1) 64-bit **key**, 32-bit **IV** used to compute 64-bit **seed**
  - **2) 64-bit seed initializes 96-bit state by map M:$\{0,1\}^{64} \rightarrow \{0,1\}^{96}$**
- Joint (A,C) 64-bit state **can obtain** 2^{40} values out of 2^{64} **(!)**
  - $\mathrm{Dim}(\mathrm{Im}(M_{AC})) = 40$



GEA-1 Attack [BDL+21] (Simplified)

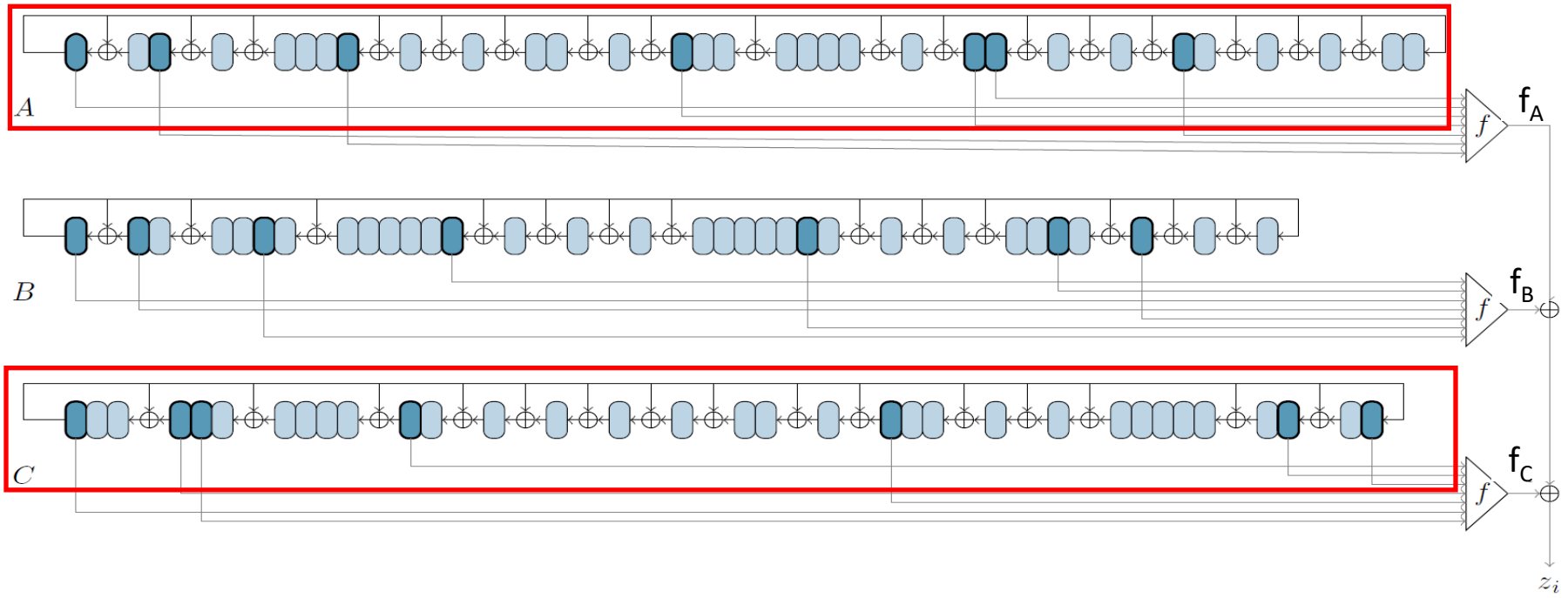
- Search for $(\underline{A},\underline{B},\underline{C})$ s.t $f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus f_C(\underline{C}) = z$ or $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z = f_B(\underline{B})$
  - Remark: register outputs of length 65-bits
- 1) Build table T_B for f_B: For each $\underline{B} \in \{0,1\}^{32}$, store $(\underline{B}, f_B(\underline{B}))$ in T_B, sorted by $f_B(\underline{B})$
- 2) For each $(\underline{A},\underline{C}) \in Im(M_{AC})$, search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B

- Complexity: $T = 2^{40}$, $S = 2^{32}$ words (44 GiB)
- Goal: obtain $T = 2^{40}$, $S = 2^{19}$ words (4 MiB)



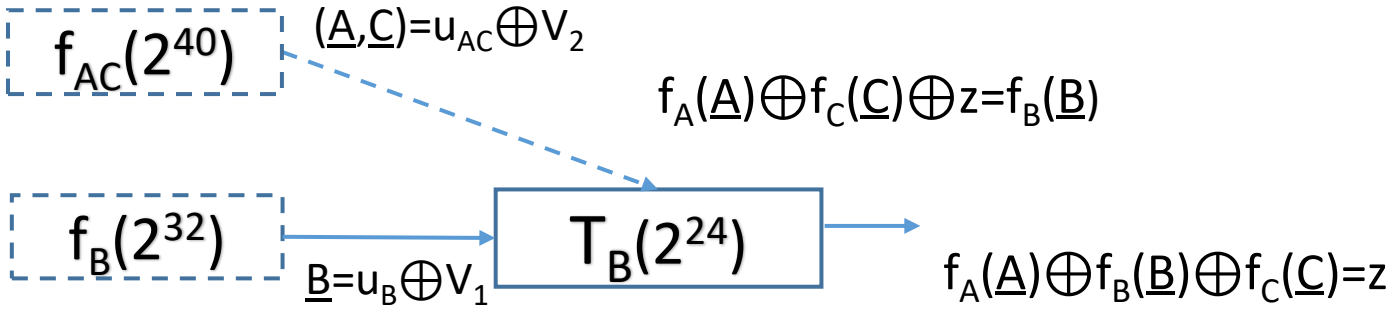
GEA-1 Attack [BDL+21]

- Exploit **linear relations** between B and (A,C)
  - Recall: $\dim(\mathrm{Im}(M)) = 64$, $\dim(\mathrm{Im}(M_{AC})) + \dim(\mathrm{Im}(M_B)) = 40+32=72$
- There is **"shared" subspace** $U \subseteq \mathrm{Im}(M)$ of dim $72-64=8$ s.t:
  - $\mathrm{Im}(M_B) = U_B \boxplus V_1$, $\dim(V_1)=24$
  - $\mathrm{Im}(M_{AC}) = U_{AC} \boxplus V_2$, $\dim(V_2)=32$
- "Shared" subspace U used in [BDL+21] to obtain 8 more filtering bits



Improved GEA-1 Attack

- There is **"shared" subspace** $U \subseteq Im(M)$ of dim $72-64=8$ s.t:
  - $Im(M_B) = U_B \boxplus V_1$, $dim(V_1)=24$
  - $Im(M_{AC}) = U_{AC} \boxplus V_2$, $dim(V_2)=32$
- **"Shared" subspace** can be used to **reduce memory**
  - Iterate over all $u \in U$ in outer loop, build **smaller table** T_B per u (simple rearrangement of [BDL+21])
- a) For each $u \in U$
  - 1) For each $v_1 \in V_1$, let $\underline{B}=u_B \oplus v_1$. Store $(\underline{B}, f_B(\underline{B}))$ in T_B
  - 2) For each $v_2 \in V_2$, let $(\underline{A},\underline{C})=u_{AC} \oplus v_2$. Search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B
- Complexity: $T = 2^{40}$, $S = 2^{24}$ words (256 MiB)



Improved GEA-1 Attack

- a) For each $u \in U$
  - 1) For each $v_1 \in V_1$, let $\underline{B} = u_B \oplus v_1$. Store $(\underline{B}, f_B(\underline{B}))$ in T_B
  - 2) For each $v_2 \in V_2$, let $(\underline{A}, \underline{C}) = u_{AC} \oplus v_2$. Search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B
- Complexity: $T = 2^{40}$, $S = 2^{24}$ words (256 MiB)

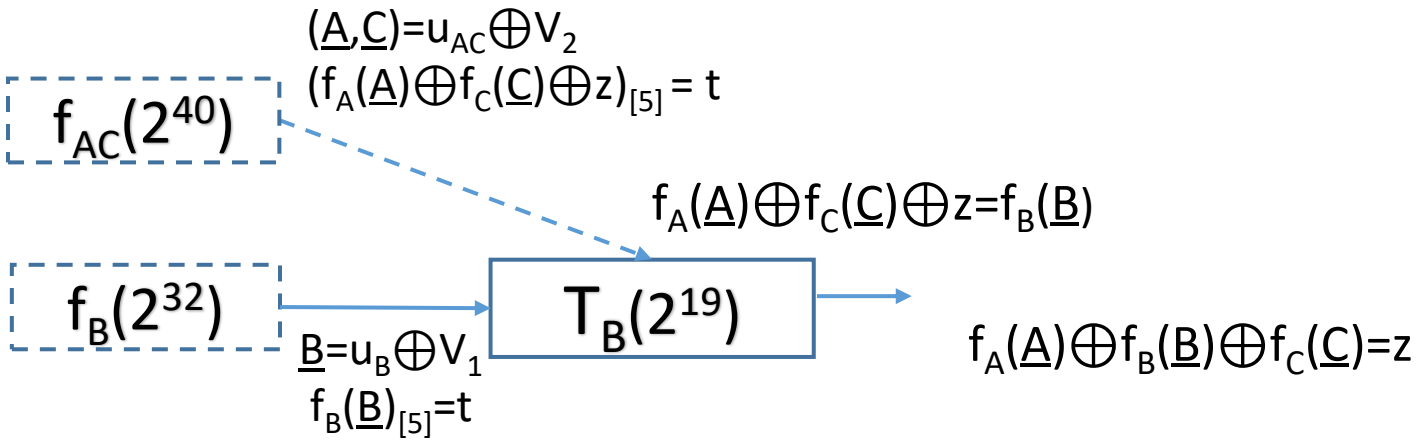
- Can memory be further reduced?
- Note: steps **not balanced**
  - Complexities: **1)** $T = 2^{24}$, $S = 2^{24}$,**2)** $T = 2^{32}$, $S = negl$

- Use **clamping through precomputation**, similarly to k-XOR algorithms
- Add (5-bit) condition to loop: for $t \in \{0,1\}^5$, store $(\underline{B}, f_B(\underline{B}))$ in T_B only if $f_B(\underline{B})_{[5]} = t$
  - Reduces size of T_B by 2^5

Improved GEA-1 Attack

- a) For each $u \in U$
  - 1) For each $v_1 \in V_1$, let $\underline{B}=u_B \oplus v_1$. Store $(\underline{B}, f_B(\underline{B}))$ in T_B
  - 2) For each $v_2 \in V_2$, let $(\underline{A},\underline{C})=u_{AC} \oplus v_2$. Search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B

- a) For each $u \in U$, $\mathbf{t \in \{0,1\}^5}$
  - 1) For each $v_1 \in V_1$, let $\underline{B}=u_B \oplus v_1$. If $\mathbf{f_B(\underline{B})_{[5]} = t}$, store $(\underline{B}, f_B(\underline{B}))$ in T_B
  - 2) For each $v_2 \in V_2$, let $(\underline{A},\underline{C})=u_{AC} \oplus v_2$. **If** $\mathbf{(f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z)_{[5]} = t}$, search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B



Improved GEA-1 Attack

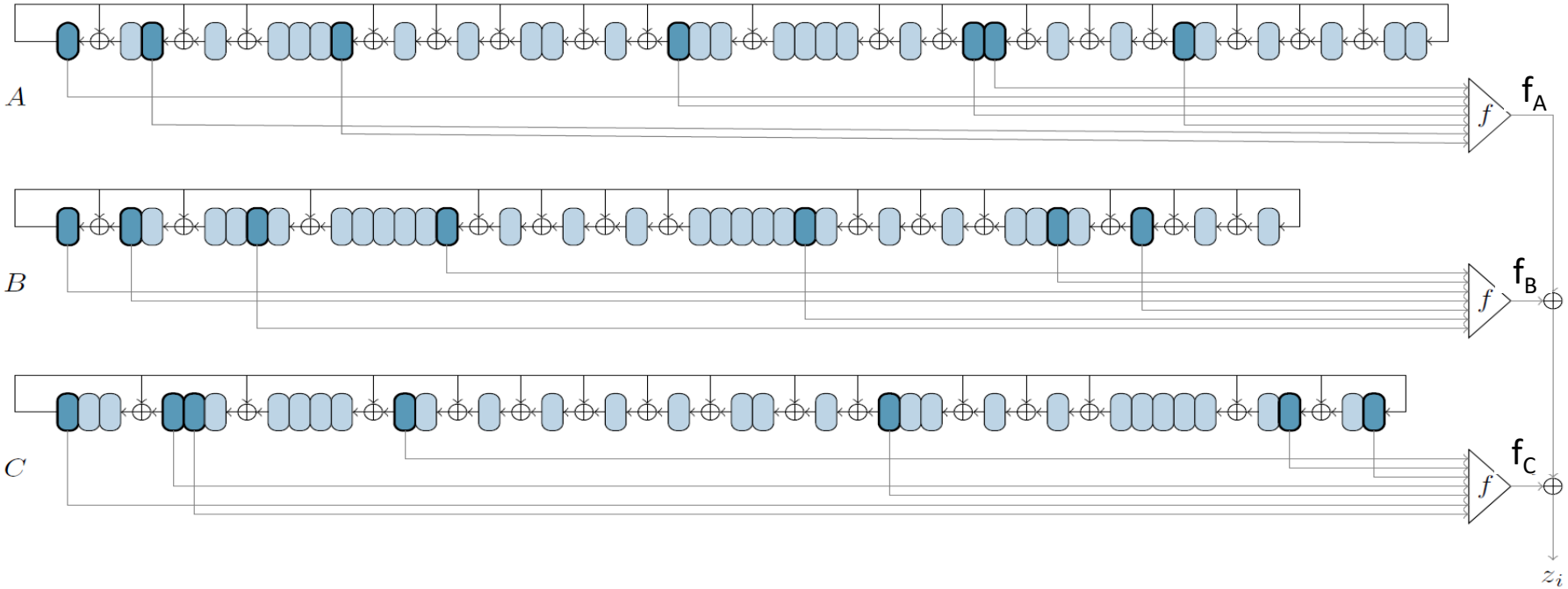
- a) For each $u \in U$
  - 1) For each $v_1 \in V_1$, let $\underline{B} = u_B \oplus v_1$. Store $(\underline{B}, f_B(\underline{B}))$ in T_B
  - 2) For each $v_2 \in V_2$, let $(\underline{A}, \underline{C}) = u_{AC} \oplus v_2$. Search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B

- a) For each $u \in U$, $\mathbf{t \in \{0,1\}^5}$
  - 1) For each $v_1 \in V_1$, let $\underline{B} = u_B \oplus v_1$. If $\mathbf{f_B(\underline{B})_{[5]} = t}$, store $(\underline{B}, f_B(\underline{B}))$ in T_B
  - 2) For each $v_2 \in V_2$, let $(\underline{A}, \underline{C}) = u_{AC} \oplus v_2$. **If** $\mathbf{(f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z)_{[5]} = t}$, search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B

- $S = 2^{24-5} = 2^{19}$ (4 MiB)
- **Problem**: step 2 not efficient ($T = 2^{32}$)
  - Overall: $2^5 \cdot 2^8 \cdot 2^{32} = 2^{45}$

Improved GEA-1 Attack

- 2) For each $v_2 \in V_2$, let $(\underline{A},\underline{C}) = u_{AC} \oplus v_2$. If $(f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z)_{[5]} = t$, search $f_A(\underline{A}) \oplus f_C(\underline{C}) \oplus z$ in T_B
- Implement **2)** in $T = 2^{27}$, S=negl (details in paper)
- Main observation
  - $f_A(\underline{A})_{[5]}$ depends on 26 bits of $\underline{A}$, $f_C(\underline{C})_{[5]}$ depends on 22 bits of $\underline{C}$
- Gives $T=2^{40}$, $S=2^{19}$ (4 MiB)



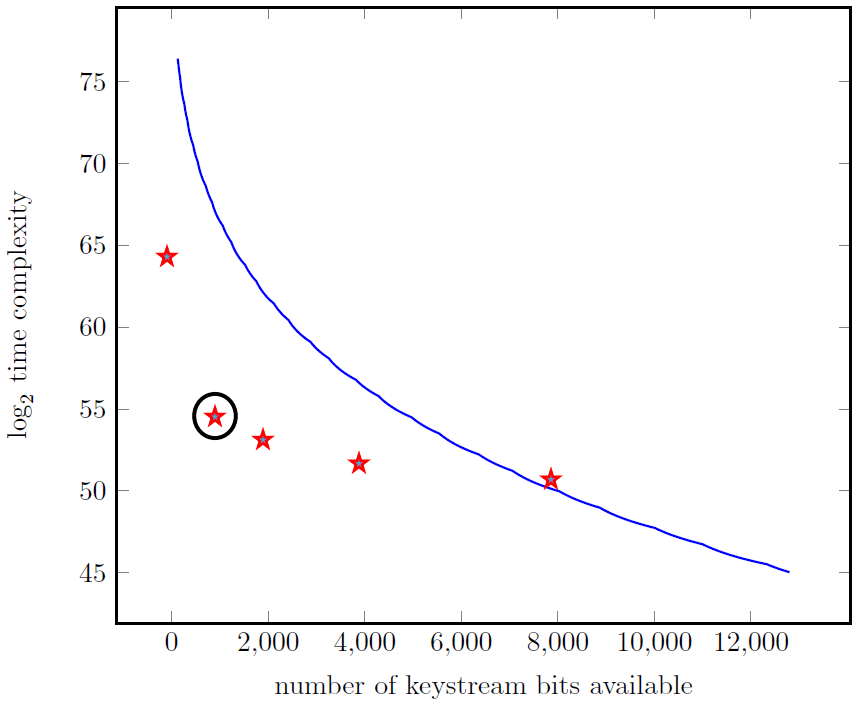
Outline

- GEA-1 specification + attack
- **GEA-2 specification + attack 1**
- Conclusions

GEA-2 Attack

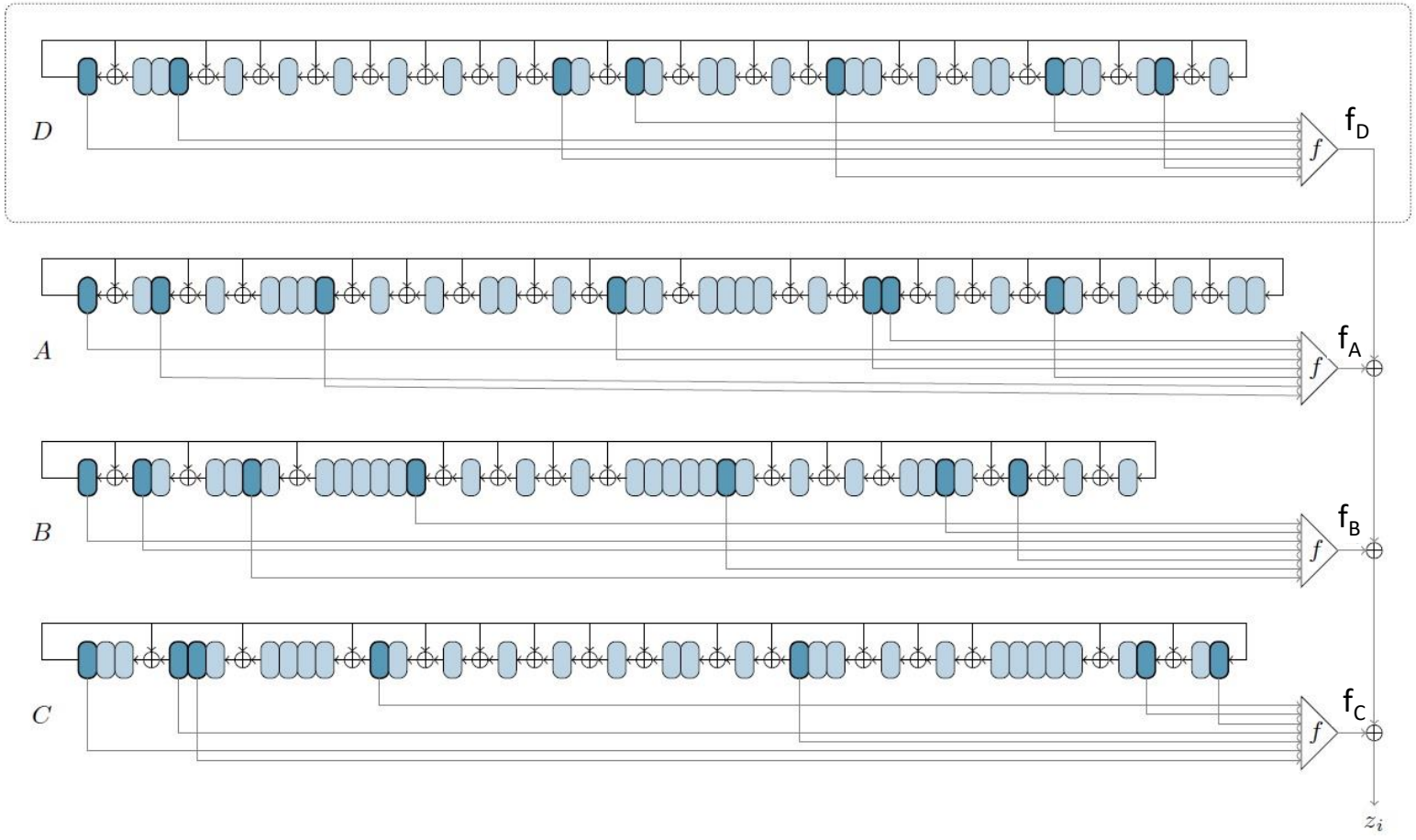
- **GEA-2** does not have significant weakness as **GEA-1**
- However [BDL+21] describe (algebraic) data-time tradeoff attack
  - Beats exhaustive search from about 1468 keystream bits

- Describe different attack
  - Improves [BDL+21] for <7000 available keystream bits
  - E.g., given 1100 bits, $T=2^{54}$
  - Attack is **generic** on XOR stream cipher combiners



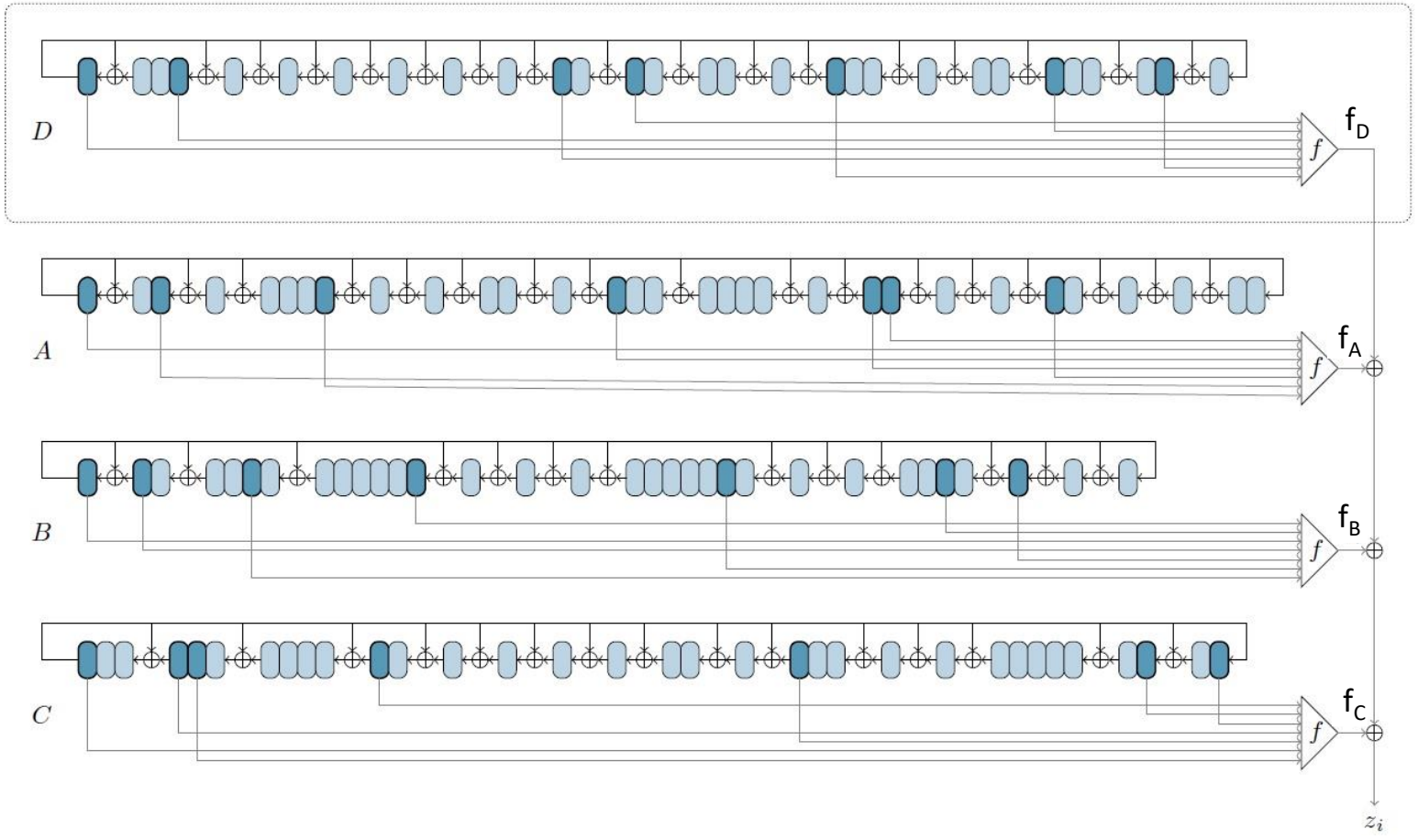
GEA-2 Initialization

- Initialization: inputs - 64-bit session **key**, 32-bit **IV**
  - Initialize 125-bit state (A,B,C,D) (31+32+33+29)



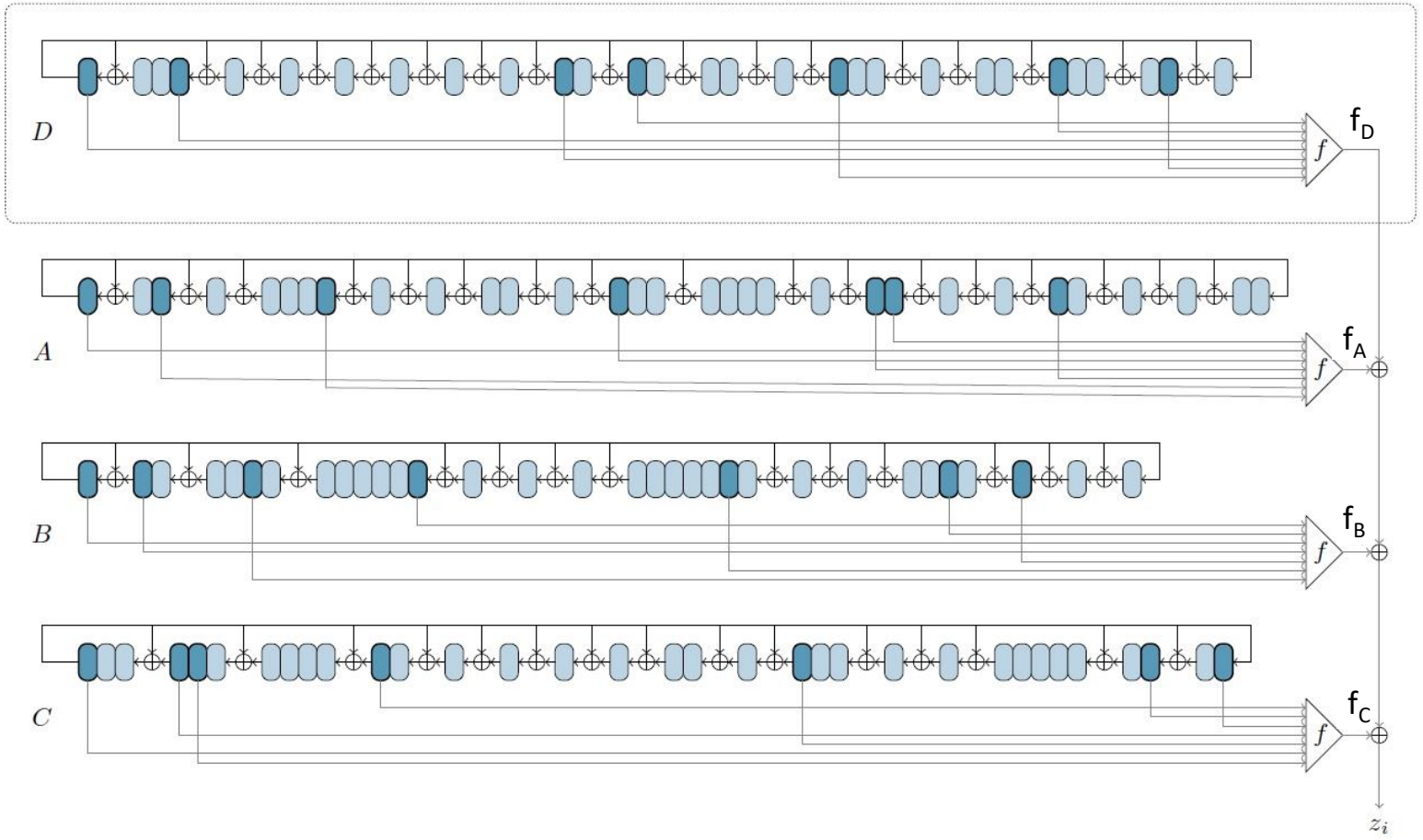
GEA-2 Keystream Generation

- Keystream generation
  - Inputs – 125-bit initial state (A,B,C,D), 12800-bit packet p
- - Compute keystream $z = (f_A(A) \oplus f_B(B) \oplus f_C(C) \oplus f_D(D))_{[12800]}$
- - Output ciphertext $c = p \oplus z$



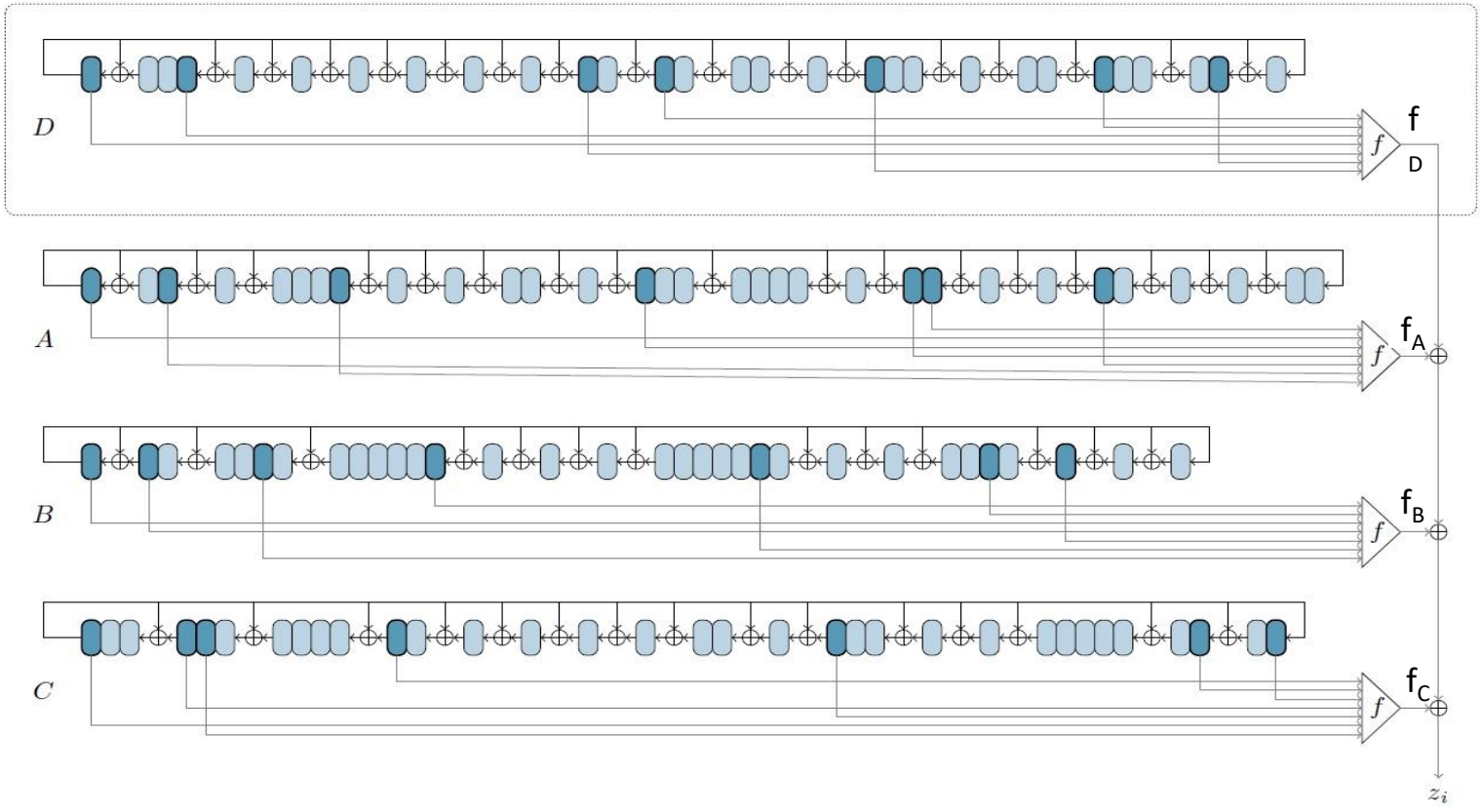
GEA-2 Attack Outline

- Assume for simplicity all registers of length 32 bits
- Attack **recovers initial state** given ℓ known keystream bits
  - From initial state can easily **compute session key**



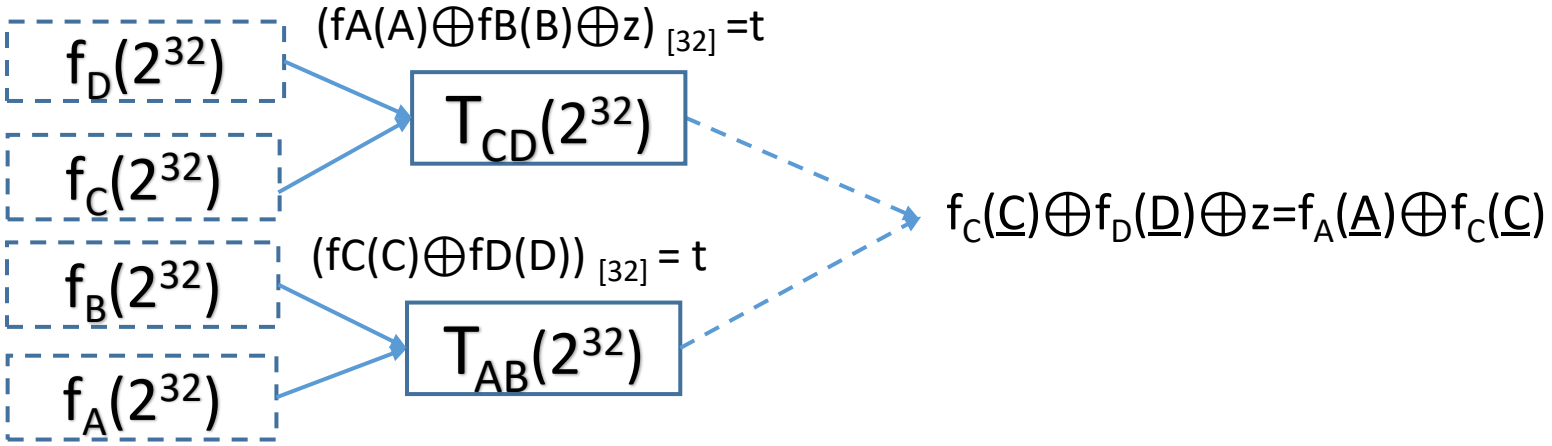
State Recovery GEA-2 Attack for ℓ=128

- Assume z of length ℓ=128
- Idea: look for $(\underline{A},\underline{B},\underline{C},\underline{D})$ s.t $f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus f_C(\underline{C}) \oplus f_D(\underline{D}) = z$ or $f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus z = f_C(\underline{C}) \oplus f_D(\underline{D})$
- Since each $(\underline{A},\underline{B})$ and $(\underline{C},\underline{D})$ has 2^{64} states, standard MITM attack gives $T = 2^{64}$, $S = 2^{64}$



GEA-2 Attack for ℓ=128 - Improved Space

- Based on Schroeppel-Shamir algorithm for subset-sum
- Observation: if $f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus f_C(\underline{C}) \oplus f_D(\underline{D}) = z$ and $(f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus z)_{[32]} = t$, then $(f_C(\underline{C}) \oplus f_D(\underline{D}))_{[32]} = t$
- a) For each $t \in \{0,1\}^{32}$
  - 1) By MITM, compute all $(\underline{A},\underline{B})$ s.t. $(f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus z)_{[32]} = t$. Store in T_{AB}
  - 2) By MITM, compute all $(\underline{C},\underline{D})$ s.t. $(f_C(\underline{C}) \oplus f_D(\underline{D}))_{[32]} = t$. Store in T_{CD}
  - 3) Merge T_{AB} and T_{CD} on $f_C(\underline{C}) \oplus f_D(\underline{D}) \oplus z = f_A(\underline{A}) \oplus f_C(\underline{C})$
- Size of T_{AB} ,T_{CD} is 2^{32}
- Complexity: $T = 2^{32} \cdot 2^{32} = 2^{64}$, $S = 2^{32}$



GEA-2 Attack for ℓ>128

- Attack with $\ell=128$ has $T = 2^{64}$
- Can we optimize when $\ell>128$?
- Idea: **artificially** create **multiple solutions**
  - **Solution** = internal $(\underline{A},\underline{B},\underline{C},\underline{D})$ **GEA-2** state at some clock C
- Find **one of many** solutions more **efficiently**
  - Similar to Wagner's algorithm for k-XOR problem

- a) For each $t \in \{0,1\}^{32}$
  - 1) By MITM, compute all $(\underline{A},\underline{B})$ s.t. $(f_A(\underline{A}) \oplus f_B(\underline{B}) \oplus z)_{[32]}=t$. Store in T_{AB}
  - 2) By MITM, compute all $(\underline{C},\underline{D})$ s.t. $(f_C(\underline{C}) \oplus f_D(\underline{D}))_{[32]}=t$. Store in T_{CD}
  - 3) Merge T_{AB} and T_{CD} on $f_C(\underline{C}) \oplus f_D(\underline{D}) \oplus z = f_A(\underline{A}) \oplus f_C(\underline{C})$

- Assuming r solutions -> need to iterate over $2^{32}/r$ values of t
- Complexity: $T = 2^{64}/r$

GEA-2 Attack for ℓ>128

- How to produce r **solutions** (target states) given ℓ>128 keystream bits?
- Look at ≈ℓ shifted keystreams produced by ≈ℓ internal states (details in paper)
  - Similar idea used in [BDL+21] in a different attack
- Attack is **generic** on all (4) XOR stream cipher combiners

Outline

- GEA-1 specification + attack
- GEA-2 specification + attack 1
- **Conclusions**

Conclusions and Open Problems

- Described improved and refined attacks on GEA-1 and GEA-2
- Techniques based on **new applications** of k-XOR algorithms to stream cipher cryptanalysis
  - GEA-2 attack is **generic** on all (4) XOR stream cipher combiners

- Open problem: **further improve** GEA-2 attacks for low data

Thanks for your attention!