Generation of good bijective S-BOXes using a Reversed Genetic Algorithm

Georgi Ivanov, IMI-BAS, Bulgaria
Nikolay Nikolov, IMI-BAS, Bulgaria
Svetla Nikova, KU Leuven, Belgium

International Workshop on Boolean Functions and their Applications,
Rosendal, Norway, September 2–7, 2014

KU LEUVEN

S-boxes – often the only non-linear part of an symmetric crypto algorithm

To ensure resistance against linear/differential cryptanalysis:

- Increase number of active S-boxes (by stronger linear layer)
- **Large S-boxes**

Outline

- Construction Techniques
- Motivation
- Our algorithms
- Results

KU LEUVEN

Construction Techniques

- Algebraic constructions
  - Finite Field Inversion
  - Power Mappings
- Constructions from small to large s-boxes
  - Gerard et. al at CHES 2013, Gross et.al at FSE 2014.
  - Previously used in Whirlpool, Noekeon, Misty, Khazard, etc.
- Pseudo-random Generation
- Heuristic Approaches

Known Results (n = 8)

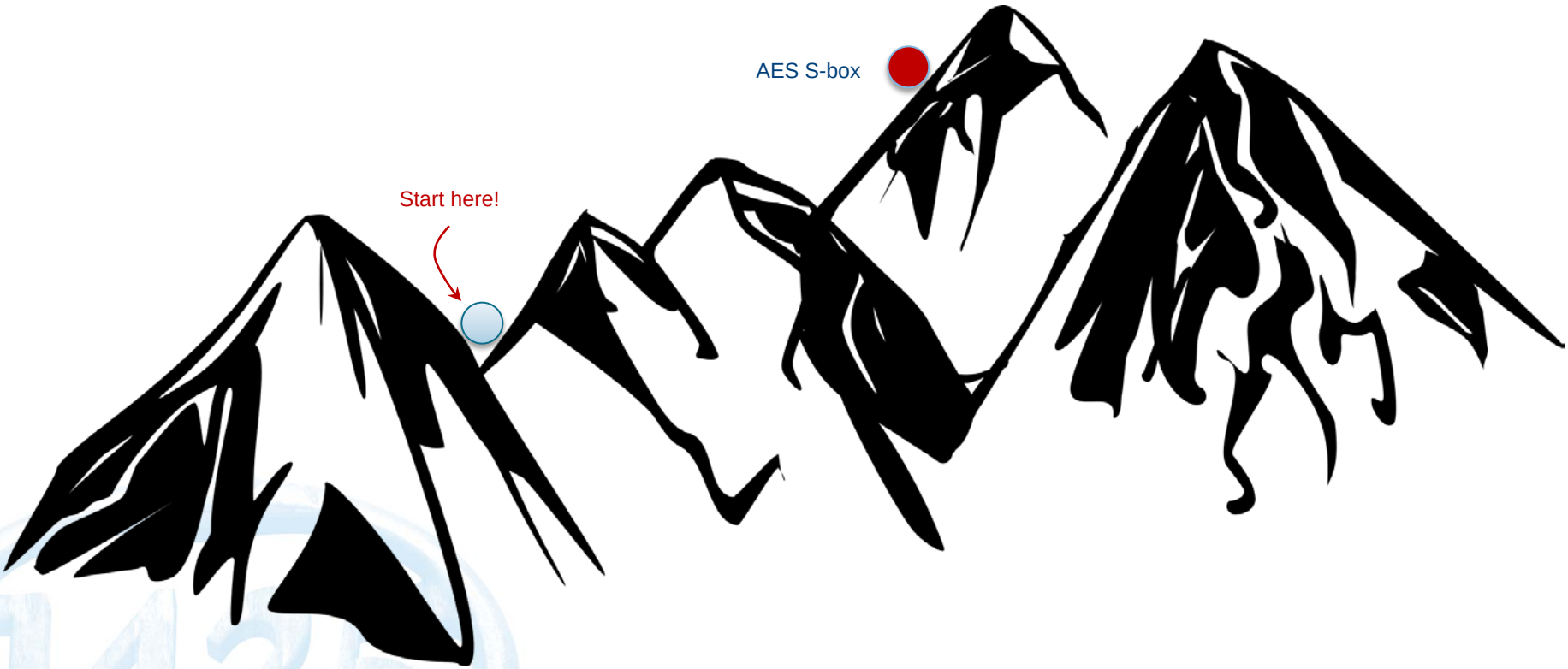
	Non-linearity	degree	$\|AC\|_{max}$	δ	Fixed points	Linear redundancy
Finite Field Inv., Power mappings	112	7	32	4	0-2	complete
4-bit to 8-bit constructions	64-96	6-7	-	16-32	-	-
PRND Search	94-100	6-7	96-106	8	0	zero
Heuristics	98-104	6-7	56-80	6	0-2	zero

Motivation and goals

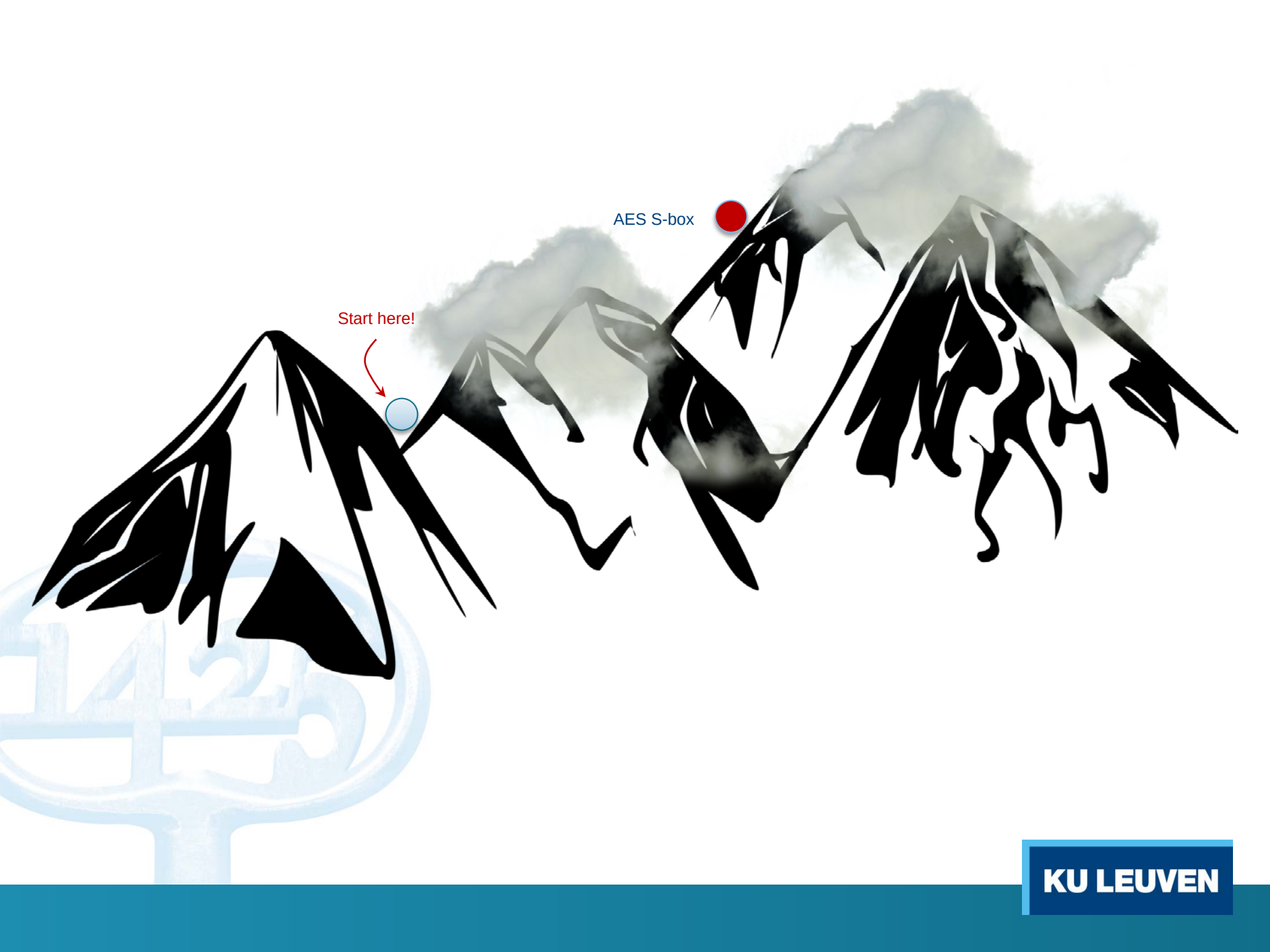
- Obtaining large sets of bijective S-boxes, from (8 x 8) to (16 x 16), with properties close to the best ones known

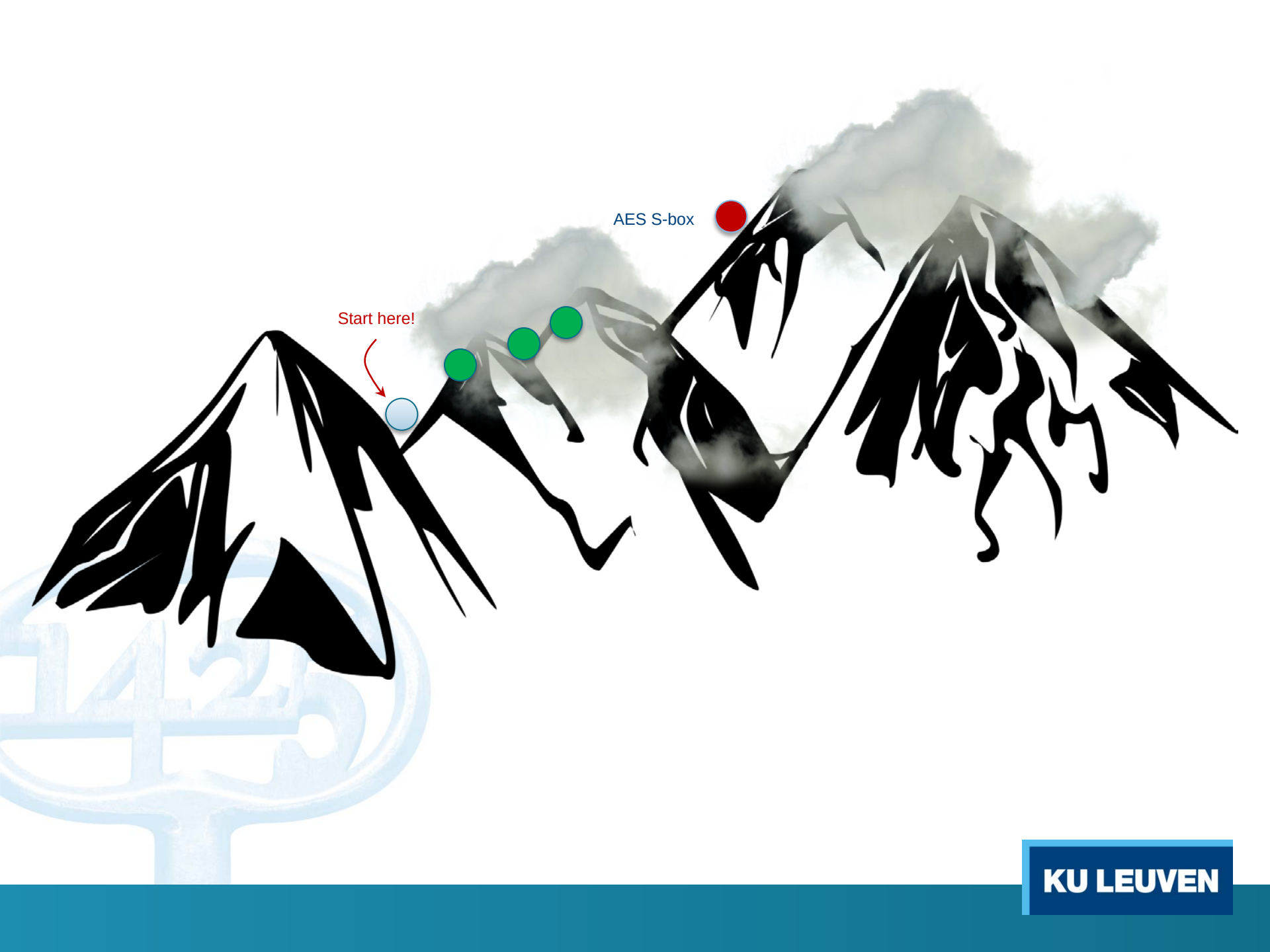
- Use a Genetic Algorithm working in a reverse way in order to save time and memory.

- Starting from the properties of the Finite Field Inversion-based S-boxes until reaching some threshold values chosen in advance

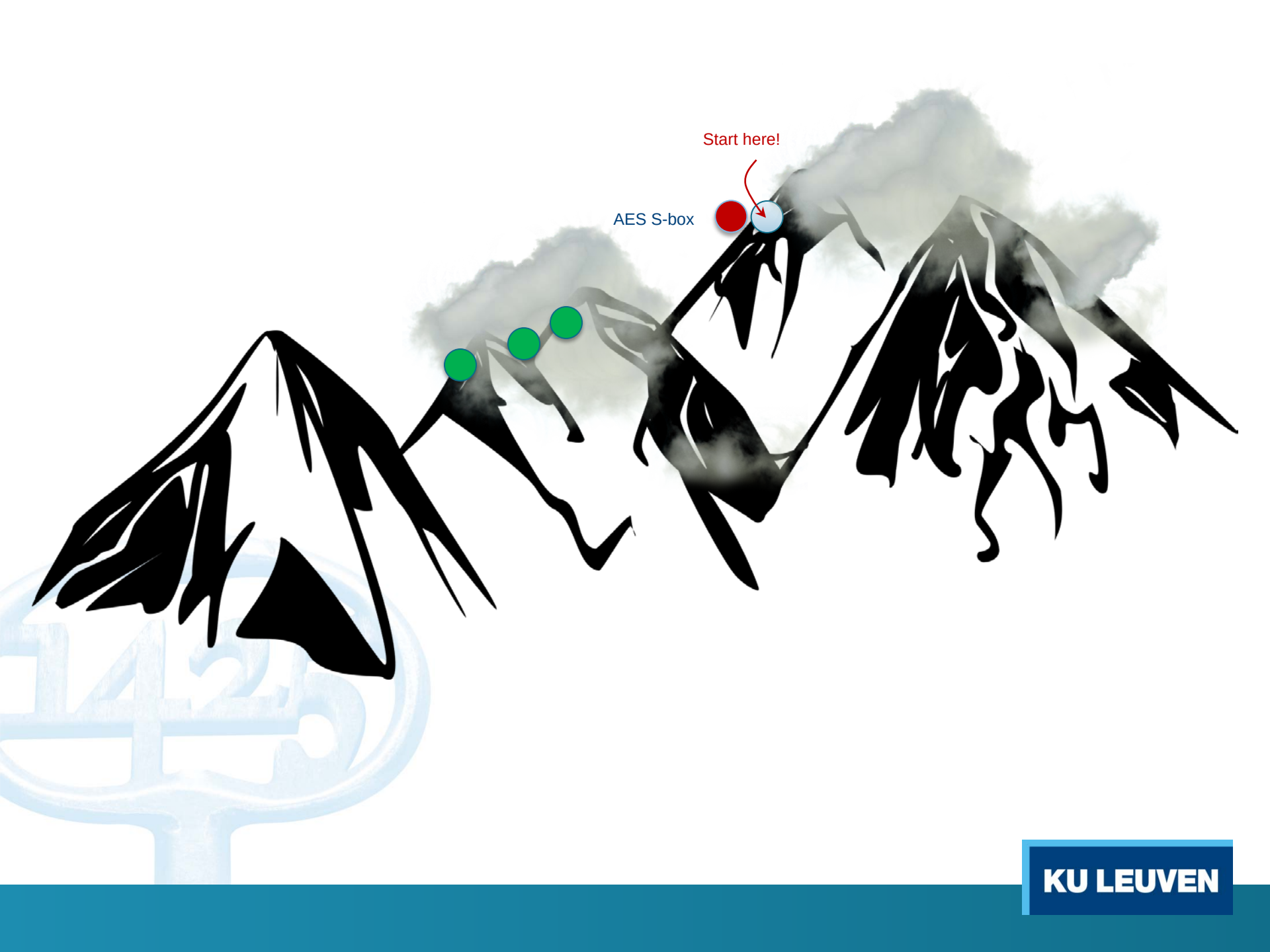


KU LEUVEN

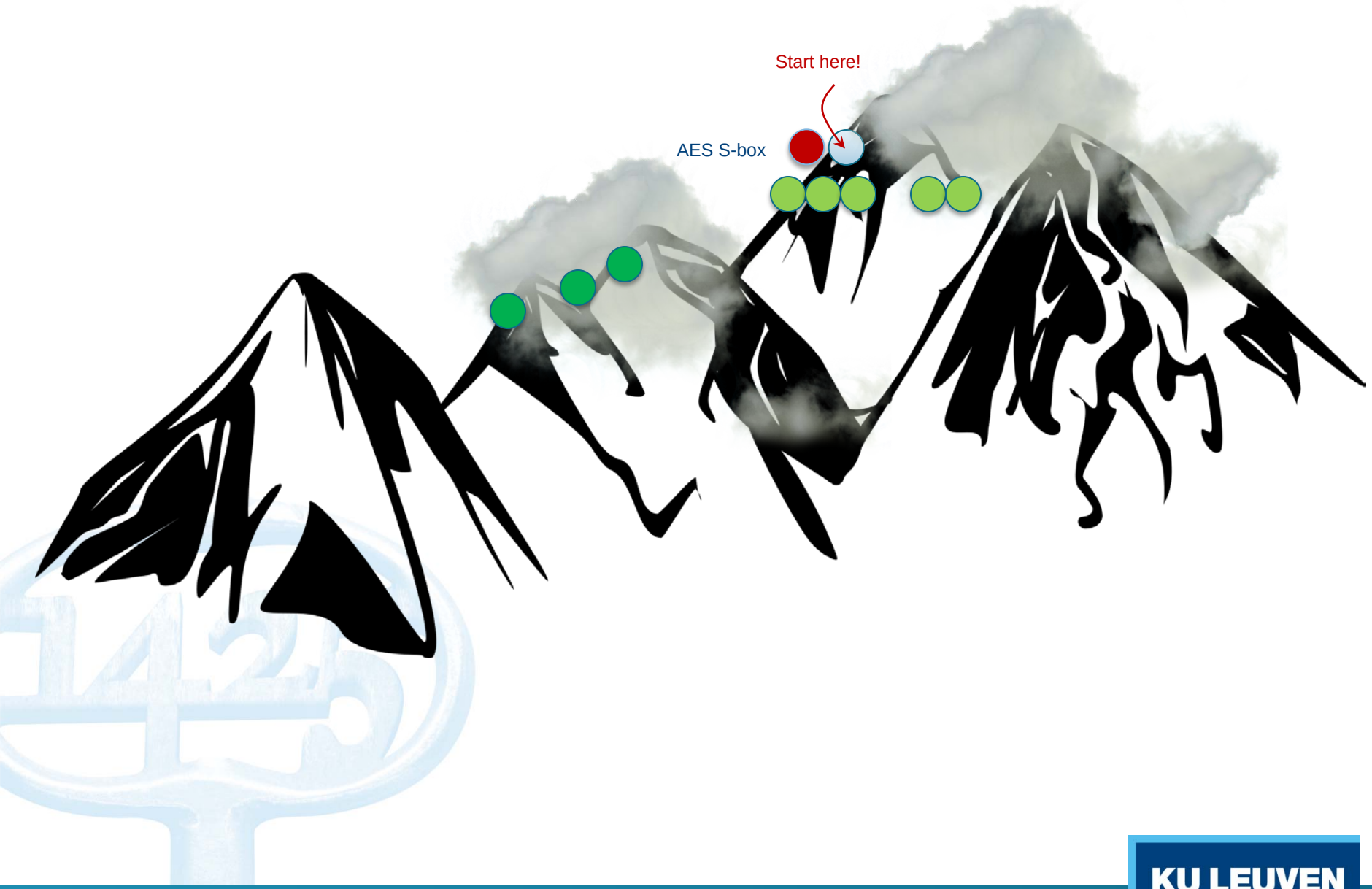

AES S-box
Start here!



KU LEUVEN



KU LEUVEN



KU LEUVEN

Target Criteria

- MAX of S-BOX **nonlinearity** NS ⇔ MIN of the largest non-trivial value in LATS (LC)
- MAX of S-BOX (minimal) **algebraic degree** deg(S) (LOA)
- MIN of the largest non-trivial value δ in DDTS (**differential δ-uniformity**) (DC)
- MIN of the largest non-trivial **absolute autocorrelation value** |AC|max (DC)
- Non-existence of **fixed points** (SA)
- Non-possession of **linear redundancy** (AA)

GAs, Evolution and Natural Selection

- Population of parents interbreeds to produce children
- Mutation – helps in providing genetic variation
- Selection Process – only the fittest survive to become the next generation

KU LEUVEN

GAs Terminology

- Parent Pool (PP) – current set of t candidate solutions
- Parents – a pair of individuals in the PP chosen for breeding
- Breeding – the mating process of two parents to produce children
- Children – the offspring, resulting from the breeding
- Fitness – measure to ascertain surviving individuals
- Offspring Pool (OP) – set of t children passed the fitness test

Genetic Algorithm 1

Step 1: Initial **PP**

Generate a set of **t** bijective (**n** x **n**) S-BOXes, $\mathbf{P_1}$, $\mathbf{P_2}$, … , $\mathbf{P_t}$, representing the **PP**.

PP is generated using affine transformations of the finite field inversion.

PP - constructed as an (**t** x $2^{\mathbf{n}}$) array

KU LEUVEN

Genetic Algorithm 1

Step 2: Breeding

Choose the first pair of parents ($\mathbf{P_1}$,$\mathbf{P_2}$):

($\mathbf{Ch_1}$,$\mathbf{Ch_2}$) = **breeding** ($\mathbf{P_1}$,$\mathbf{P_2}$,$\mathbf{CoP_1}$,$\mathbf{CoP_2}$,**cnt**)

$\mathbf{CoP_1}$ & $\mathbf{CoP_2}$ – random numbers between 1 and 2^n, pointing out the breaking positions of parents genes.

cnt – a 5-valued counter, specifying the order (straight or reverse) in which the parent genes are copied into the children.

Example

Let $\mathbf{P_1}$, $\mathbf{P_2}$ are (8 x 8) bijective S-BOXes,
CoP_1 = 123, **CoP_2** = 210 and **cnt** = 1:

$\mathbf{P_1}$	V_1	V_2	. . .	V_{123}	V_{124}	V_{125}	. . .	V_{256}
$\mathbf{P_2}$	V_1	V_2	. . .	. . .	V_{210}	V_{211}	. . .	V_{256}
$\mathbf{Ch_1}$	V_1	V_2	. . .	V_{123}	V_{124}	V_{125}	. . .	V_{256}
$\mathbf{Ch_2}$	V_1	V_2	. . .	. . .	V_{210}	V_{211}	. . .	V_{256}

- Unwanted mutation – repeated genes?
- Restore bijection – **modeling** (**Ch**)

$\mathbf{Ch_1}$	V_1	. . .	V_{123}	V_{124}	. . .	V_k	. . .	V_{256}
$\mathbf{Ch_1}$	V_1	. . .	V_{123}	V_{124}	. . .	V_{rnd}	. . .	V_{256}

If for some $k > 123$ and $s \leq 123$: $V_k = V_s$, randomly generate V_{rnd} until $V_{rnd} \neq V_s$, $\forall s \leq 123$.

Replace V_k with V_{rnd} and repeat the same process from the right-hand neighbor of V_k.

At the end $\mathbf{Ch_1}$ & $\mathbf{Ch_2}$ are permutations.

Step 3: **Fitness test** (**GA1**):

N_{Ch} = **fitness** (**Ch**)

- The test is passed if $N_{ch} > N_{thr}$.

Ch survives and is placed in the **OP**.

- The test is passed if $N_{ch} = N_{thr}$.

Ch is placed in the **OP** and in addition saved in a file.

- The test is not passed if $N_{ch} < N_{thr}$.

Ch is left off.

Step 3: (**Fitness test** GA2):

N_{ch} = **fitness** (**Ch**) and C_{ch} = **cost** (**Ch**), where:

- **cost** (**Ch**) = $\Sigma_{v \in GF(2^n)^*} \Sigma_{w \in GF(2^n)} |F_{v.\mathbf{Ch}}(w) - 21|^7$
- $F_{v.Ch}(w)$ is the WHT spectral coefficient of the component function of **Ch** = $(f_1, f_2, \ldots, f_n)$ corresponding to v:

v.**Ch** = $v_1 f_1 \oplus v_2 f_2 \oplus \ldots \oplus v_n f_n$.

- Test is passed if $N_{ch} > N_{thr}$ and $C_{ch} < C_P$.
Ch survives and is placed in the **OP**.

- Test is passed if $N_{ch} = N_{thr}$ and $C_{ch} < C_P$.
Ch is placed in the **OP** and saved in a file.

- Test is not passed if $N_{ch} < N_{thr}$ or $C_{ch} > C_P$.

KU LEUVEN

Step 4: Solution Pool

- Until **OP** gets full of children, repeat the breeding process with parental pairs $(\mathbf{P_1},\mathbf{P_3})\ldots(\mathbf{P_1},\mathbf{P_t})$, $(\mathbf{P_2},\mathbf{P_3})\ldots(\mathbf{P_2},\mathbf{P_t})\ldots(\mathbf{P_{t-1}},\mathbf{P_t})$
- If after the breeding of all **PP** pairs the **OP** is not totally full, the breeding process starts all over, again with $(\mathbf{P_1},\mathbf{P_2})$
- If **OP** is full:

  - If $\boldsymbol{N_{ch}} = \boldsymbol{N_{thr}}$, $\forall$ **Ch** $\in$ **OP**,

    the Algorithm stops

  - Otherwise, **PP** = **OP** and go to the next generation (step 2)

Results from GA1 and GA2

More than 200 8-bit S-boxes

n = 8	Non-linearity	deg	$\|AC\|_{max}$	δ	Fixed points	Linear redundancy
Inversion	112	7	32	4	2	complete
PRND Search	94-100	6-7	96-106	8	0	zero
4-bit to 8-bit	64-96	6-7	-	16-32	-	-
Heuristics	98-104	6-7	56-80	6	0-2	zero
GA1/GA2	104/106	7/6	64/48	6/6	2/0	zero
GA1/GA2	106/110	6/7	56/40	6/6	2/0	zero
GA1/GA2	108/112	6/7	48/32	6/6	0/0	zero

Results from GA1, **n** =16

N_{thr} = 32 400, **t** = 50 S-BOXes

	N	deg	$\|AC\|_{max}$
Inversion	32512	15	512
GA 1	32400	15	976
GA 1	32400	14	984
N_{thr} = 32428 and 32476			
GA1	32428	15	864
GA1	32476	14	616

KU LEUVEN

Algorithm variations and future work

- Execute the algorithm long after all children from the **OP** reach N_{thr}
- Add to initial pool some S-BOXes based on power mappings
- Modify the children modeling technique to speed-up the process
- Add more criteria to be measured in fitness function – for now not applicable for big **n**

Most recent results

- Differential-uniformity added to the fitness function

n = 8	Non-linearity	deg	$\|AC\|_{max}$	δ	Linear redundancy
Inversion	112	7	32	4	complete
δ in the fitness function	110	6	40	4	zero
only nonlinearity in the fitness function	110	7	40	6	zero

Thank you!

KU LEUVEN