For Rebuttal: Details of the New Detection Algorithm

No Author Given

No Institute Given

In Section 6 of our paper, we show a new detection algorithm for key-independent property. The new algorithm surpasses the capability but achieves the similar efficiency with the two-subset bit-based division property. Here we show more details of the new algorithm and hope it is helpful.

For a derived function $\boldsymbol{x}^{(r)} = \boldsymbol{f}_d(\boldsymbol{x}^{(0)})$ with $\boldsymbol{\Gamma}^0, \boldsymbol{\Gamma}^1, \boldsymbol{\Gamma}^p, \boldsymbol{\Gamma}^s$ where $\boldsymbol{\Gamma}^s$ indicates the secret keys, we consider the monomials in

$$\mathbb{S}_0 = \{\pi_{\boldsymbol{\Gamma}^p \oplus \boldsymbol{w}}(\boldsymbol{x}^{(0)}) : \mathbf{0} \prec \boldsymbol{w} \preceq \boldsymbol{\Gamma}^s\}.$$

Since $\mathbf{0} \prec \boldsymbol{w} \preceq \boldsymbol{\Gamma}^s$, $\pi_{\boldsymbol{w}}(\boldsymbol{x}^{(0)}) \notin \{0, 1\}$. Furthermore, $\pi_{\boldsymbol{\Gamma}^p \oplus \boldsymbol{w}}(\boldsymbol{x}^{(0)}) = \pi_{\boldsymbol{\Gamma}^p}(\boldsymbol{x}^{(0)}) \cdot \pi_{\boldsymbol{w}}(\boldsymbol{x}^{(0)})$ is a monomial related to the secret key bits and $\mathbb{S}_0$ is the set of all such key-related monomials.

Let $\mathbb{X} = \{\boldsymbol{x} \oplus \boldsymbol{\Gamma}^1 : \boldsymbol{x} \preceq \boldsymbol{\Gamma}^p\}$. If we are interested in whether the value of

$$\lambda = \bigoplus_{\boldsymbol{x} \in \mathbb{X}} \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$$

is key-independent, we should check whether the ANF of $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ contains any monomial in $\mathbb{S}_0$.

For each monomial $\pi_{\boldsymbol{\Gamma}^p \oplus \boldsymbol{w}}(\boldsymbol{x}^{(0)}) \in \mathbb{S}_0$, we can construct a model searching for $\pi_{\boldsymbol{\Gamma}^p \oplus \boldsymbol{w}}(\boldsymbol{x}^{(0)}) \rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$. If the model is infeasible, there is no trails connecting $\pi_{\boldsymbol{\Gamma}^p \oplus \boldsymbol{w}}(\boldsymbol{x}^{(0)})$ and $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$, i.e., $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ does not contain $\pi_{\boldsymbol{\Gamma}^p \oplus \boldsymbol{w}}(\boldsymbol{x}^{(0)})$. If there are no trails to $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ from any monomial in $\mathbb{S}_0$, then $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ does not contain any monomials related to the secret keys.

Assume we have constructed the search model for monomials trails from $\pi_{\boldsymbol{\Gamma}^p}(\boldsymbol{x}^{(0)})$, which is equivalent to the case of the two-subset bit-based division property. It means we want to check if the ANF of $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ contains any monomials related to $\pi_{\boldsymbol{\Gamma}^p}(\boldsymbol{x}^{(0)})$, if it contains any such monomial, then λ is sure not zero. We can add an additional constraint to it as

$$\sum_i k_i \geq 1.$$

Then our goal becomes to check whether the ANF of $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ contains any monomial in $\mathbb{S}_0$, i.e., any monomial related to the secret keys.

We illustrate the new detection algorithm by an example of Trivium. The classical model of the two-subset bit-based division property is shown in Algorithm 1. Our new detection algorithm is illustrated in Algorithm 2. The only difference is we add one more constraints about the keys, but Algorithm 2 can find more key-independent property than Algorithm 1.

Algorithm 1 Model for two-subset bit-based division property of TRIVIUM

procedure $\texttt{TriviumEval}$(round R, $\boldsymbol{\Gamma}^p, \boldsymbol{\Gamma}^s, \boldsymbol{\Gamma}^1, \boldsymbol{\Gamma}^0$)
  Prepare empty MILP Model $\mathcal{M}$
  $\mathcal{M}.var \leftarrow \mathtt{s}_\mathtt{i}^0$ for $\mathtt{i} \in \{0, 1, \ldots, 287\}$
  for $\mathtt{i} = 0$ to 287 **do**
    if $\boldsymbol{\Gamma}_i^p = 1$ **then**
      $\mathtt{s}_\mathtt{i}^0 = 1$
    else
      $\mathtt{s}_\mathtt{i}^0 = 0$
  for $r = 1$ to R **do**
    $(\mathcal{M}, \mathtt{x}_1, \ldots, \mathtt{x}_{288}) = \texttt{TriviumCore}(\mathcal{M}, \mathtt{s}_1^{r-1}, \ldots, \mathtt{s}_{288}^{r-1}, 66, 171, 91, 92, 93)$
    $(\mathcal{M}, \mathtt{y}_1, \ldots, \mathtt{y}_{288}) = \texttt{TriviumCore}(\mathcal{M}, \mathtt{x}_1, \ldots, \mathtt{x}_{288}, 162, 264, 175, 176, 177)$
    $(\mathcal{M}, \mathtt{z}_1, \ldots, \mathtt{z}_{288}) = \texttt{TriviumCore}(\mathcal{M}, \mathtt{y}_1, \ldots, \mathtt{y}_{288}, 243, 69, 286, 287, 288)$
    $(\mathtt{s}_1^r, \ldots, \mathtt{s}_{288}^r) = (\mathtt{z}_{288}, \mathtt{z}_1, \ldots, \mathtt{z}_{287})$
  for all $\mathtt{i} \in \{1, 2, \ldots, 288\}$ w/o $66, 93, 162, 177, 243, 288$ **do**
    $\mathcal{M}.\mathtt{con} \leftarrow \mathtt{s}_\mathtt{i}^\mathtt{R} = 0$
  $\mathcal{M}.con \leftarrow (\mathtt{s}_{66}^\mathtt{R} + \mathtt{s}_{93}^\mathtt{R} + \mathtt{s}_{162}^\mathtt{R} + \mathtt{s}_{177}^\mathtt{R} + \mathtt{s}_{243}^\mathtt{R} + \mathtt{s}_{288}^\mathtt{R}) = 1$
  $\mathcal{M}.optimze()$
  if $M.status$ is infeasible **then**
    return 0
  else
    return unknown

Algorithm 2 Model for two-subset bit-based division property of TRIVIUM

procedure $\texttt{TriviumEval}$(round R, $\boldsymbol{\Gamma}^p, \boldsymbol{\Gamma}^s, \boldsymbol{\Gamma}^1, \boldsymbol{\Gamma}^0$)
  Prepare empty MILP Model $\mathcal{M}$
  $\mathcal{M}.var \leftarrow \mathtt{s}_\mathtt{i}^0$ for $\mathtt{i} \in \{0, 1, \ldots, 287\}$
  $\mathcal{M}.con \leftarrow \sum_{i=0}^{79} \mathtt{s}_\mathtt{i}^0 \geq 1$ ▷ add key-related constraints
  for $\mathtt{i} = 80$ to 287 **do**
    if $\boldsymbol{\Gamma}_i^p = 1$ **then**
      $\mathtt{s}_\mathtt{i}^0 = 1$
    else
      $\mathtt{s}_\mathtt{i}^0 = 0$
  for $r = 1$ to R **do**
    $(\mathcal{M}, \mathtt{x}_1, \ldots, \mathtt{x}_{288}) = \texttt{TriviumCore}(\mathcal{M}, \mathtt{s}_1^{r-1}, \ldots, \mathtt{s}_{288}^{r-1}, 66, 171, 91, 92, 93)$
    $(\mathcal{M}, \mathtt{y}_1, \ldots, \mathtt{y}_{288}) = \texttt{TriviumCore}(\mathcal{M}, \mathtt{x}_1, \ldots, \mathtt{x}_{288}, 162, 264, 175, 176, 177)$
    $(\mathcal{M}, \mathtt{z}_1, \ldots, \mathtt{z}_{288}) = \texttt{TriviumCore}(\mathcal{M}, \mathtt{y}_1, \ldots, \mathtt{y}_{288}, 243, 69, 286, 287, 288)$
    $(\mathtt{s}_1^r, \ldots, \mathtt{s}_{288}^r) = (\mathtt{z}_{288}, \mathtt{z}_1, \ldots, \mathtt{z}_{287})$
  for all $\mathtt{i} \in \{1, 2, \ldots, 288\}$ w/o $66, 93, 162, 177, 243, 288$ **do**
    $\mathcal{M}.\mathtt{con} \leftarrow \mathtt{s}_\mathtt{i}^\mathtt{R} = 0$
  $\mathcal{M}.con \leftarrow (\mathtt{s}_{66}^\mathtt{R} + \mathtt{s}_{93}^\mathtt{R} + \mathtt{s}_{162}^\mathtt{R} + \mathtt{s}_{177}^\mathtt{R} + \mathtt{s}_{243}^\mathtt{R} + \mathtt{s}_{288}^\mathtt{R}) = 1$
  $\mathcal{M}.optimze()$
  if $M.status$ is infeasible **then**
    return 0
  else
    return unknown