The Summation-Truncation Hybrid: Reusing Discarded Bits for Free

Aldo Gunsing and Bart Mennink

Crypto 2020

PRP vs. PRF

- Many symmetric cryptographic schemes are based on pseudorandom permutations (PRPs) like AES

PRP vs. PRF

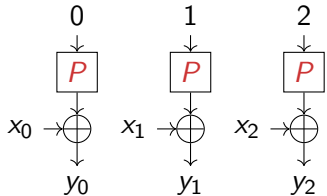
- Many symmetric cryptographic schemes are based on pseudorandom permutations (PRPs) like AES
- A lot of modes only use the forward direction, not making use of the invertibility

PRP vs. PRF

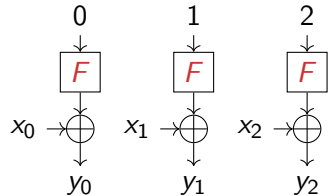
- Many symmetric cryptographic schemes are based on pseudorandom permutations (PRPs) like AES
- A lot of modes only use the forward direction, not making use of the invertibility
- In this case using a pseudorandom function (PRF) is often more secure

PRP vs. PRF

- Many symmetric cryptographic schemes are based on pseudorandom permutations (PRPs) like AES
- A lot of modes only use the forward direction, not making use of the invertibility
- In this case using a pseudorandom function (PRF) is often more secure
- Prominent example: CTR mode



$n/2$-bit security



n-bit-security

PRP-to-PRF Conversion

- We could design a dedicated PRF
- However, we have little understanding in how to design one

PRP-to-PRF Conversion

- We could design a dedicated PRF
- However, we have little understanding in how to design one
- Alternatively, we can design a PRP-to-PRF conversion

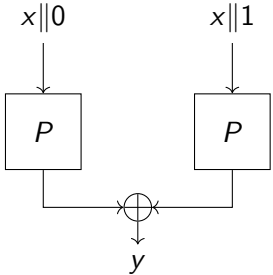
PRP-to-PRF Conversion

- We could design a dedicated PRF
- However, we have little understanding in how to design one
- Alternatively, we can design a PRP-to-PRF conversion
  - PRP-PRF switch: PRP behaves like a PRF up to the birthday bound

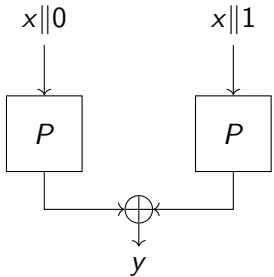
PRP-to-PRF Conversion

- We could design a dedicated PRF
- However, we have little understanding in how to design one
- Alternatively, we can design a PRP-to-PRF conversion
  - PRP-PRF switch: PRP behaves like a PRF up to the birthday bound
  - Conversions like summation and truncation achieve beyond birthday bound security

Summation and Truncation

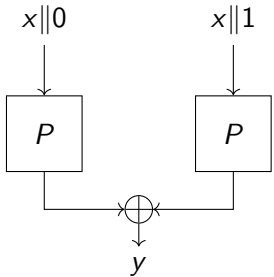


Summation and Truncation



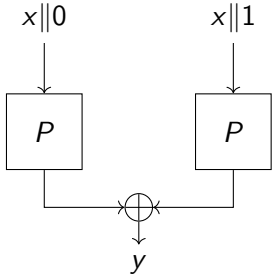
- Sums two consecutive calls

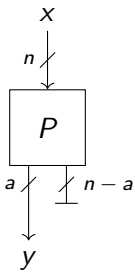
Summation and Truncation



- Sums two consecutive calls
- n-bit security

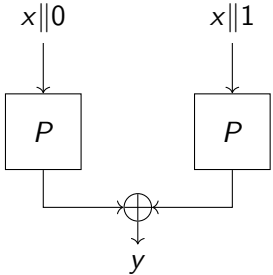
Summation and Truncation



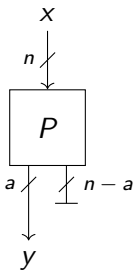


- Sums two consecutive calls
- n-bit security

Summation and Truncation

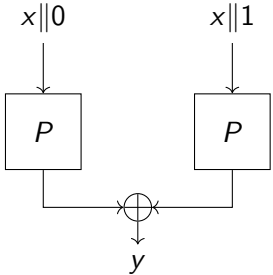


- Sums two consecutive calls
- n-bit security

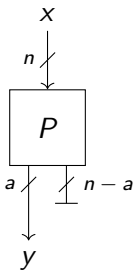


- Truncates a call to the first a bits

Summation and Truncation

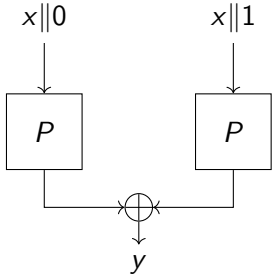


- Sums two consecutive calls
- n-bit security

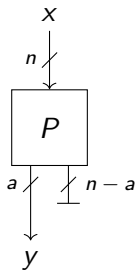


- Truncates a call to the first a bits
- Discards the other $n - a$ bits

Summation and Truncation

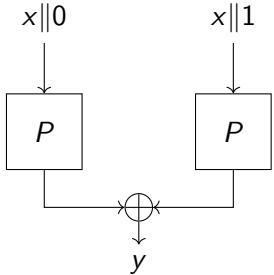


- Sums two consecutive calls
- n-bit security

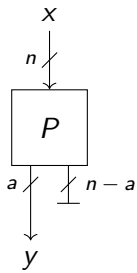


- Truncates a call to the first a bits
- Discards the other $n - a$ bits
- Used in the key derivation function of GCM-SIV

Summation and Truncation



- Sums two consecutive calls
- n-bit security

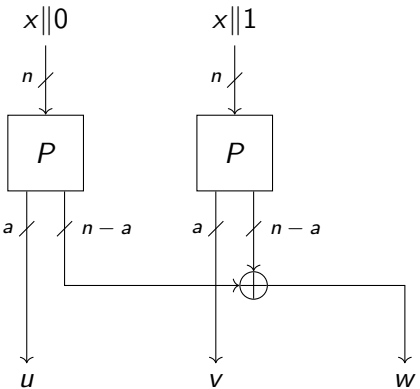


- Truncates a call to the first a bits
- Discards the other $n - a$ bits
- Used in the key derivation function of GCM-SIV
- $n - a/2$-bit security

Summation-Truncation Hybrid

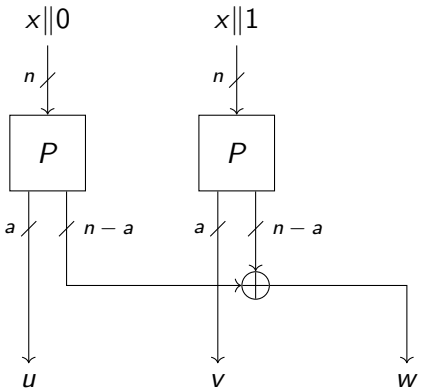
- Instead of discarding bits, we can reuse them by applying summation

Summation-Truncation Hybrid



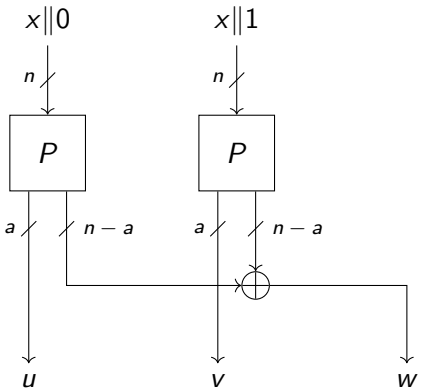
- Instead of discarding bits, we can reuse them by applying summation
- This leads to the Summation-Truncation Hybrid (STH)

Summation-Truncation Hybrid



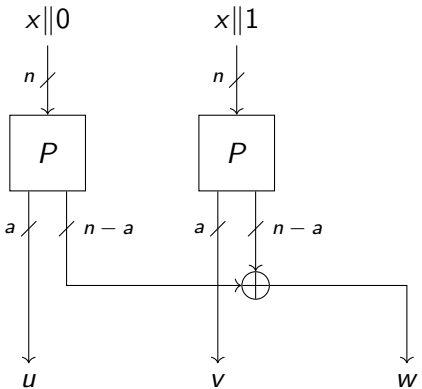
- Instead of discarding bits, we can reuse them by applying summation
- This leads to the Summation-Truncation Hybrid (STH)
- Outputs $n - a$ extra bits compared to truncation

Summation-Truncation Hybrid



- Instead of discarding bits, we can reuse them by applying summation
- This leads to the Summation-Truncation Hybrid (STH)
- Outputs $n - a$ extra bits compared to truncation
- But we show that it has equal security!

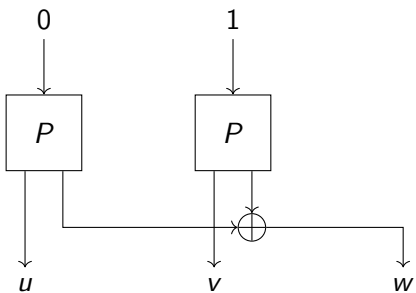
Summation-Truncation Hybrid



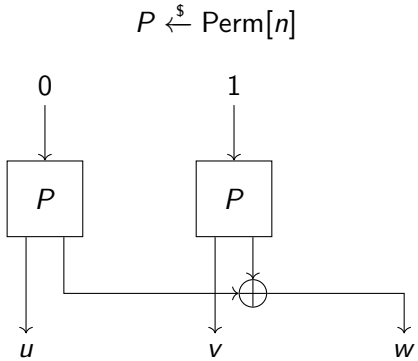
- Instead of discarding bits, we can reuse them by applying summation
- This leads to the Summation-Truncation Hybrid (STH)
- Outputs $n - a$ extra bits compared to truncation
- But we show that it has equal security!
- Identical to summation when $a = 0$

Proof Sketch: Idea

$P \xleftarrow{\$} \mathrm{Perm}[n]$



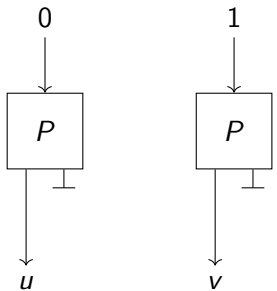
Proof Sketch: Idea

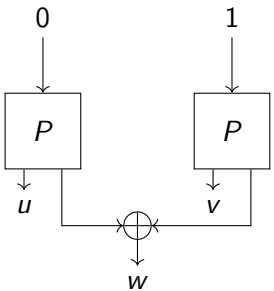


- Try to separate the truncation and summation parts

Proof Sketch: Separating STH

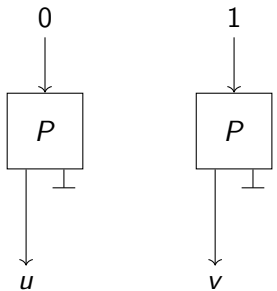
$P \xleftarrow{\$} \mathsf{Perm}[n]$

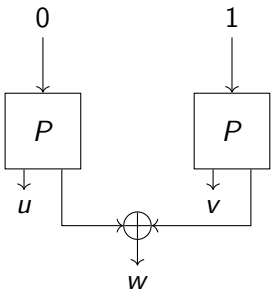




Proof Sketch: Separating STH

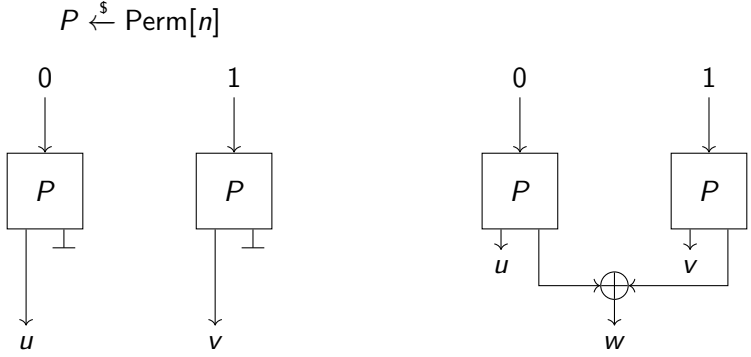
$P \xleftarrow{\$} \mathrm{Perm}[n]$





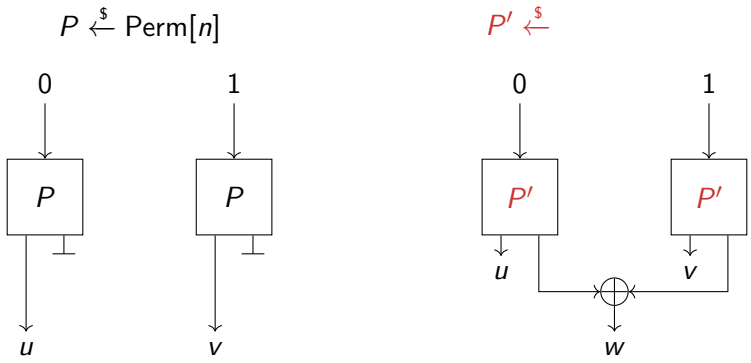
- Just write the two parts separately

Proof Sketch: Separating STH

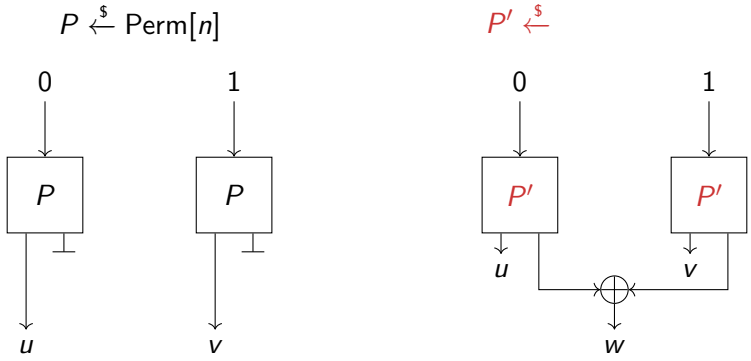


- Just write the two parts separately
- Problem: there is a shared secret P

Proof Sketch: Permutation-Separated STH

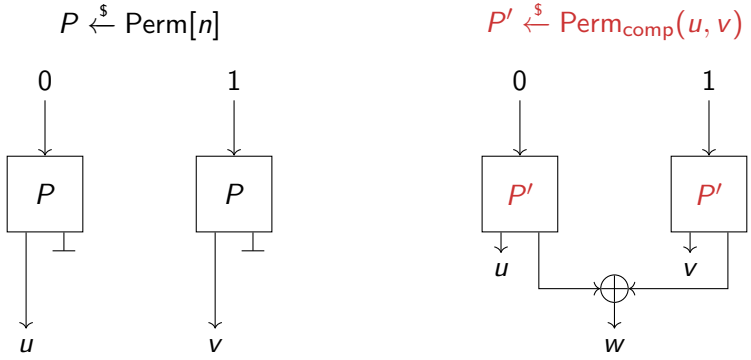


Proof Sketch: Permutation-Separated STH



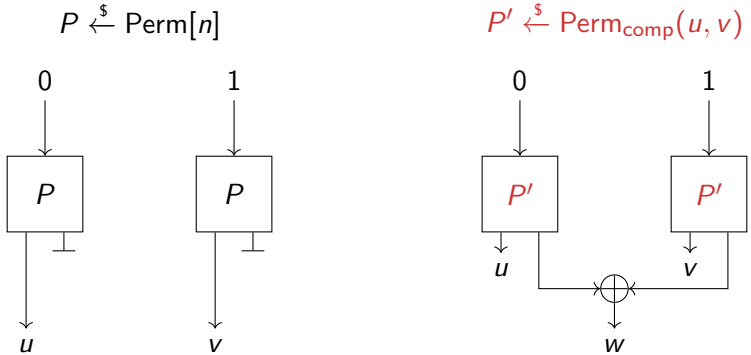
- We re-choose the permutation, but keep the same distribution

Proof Sketch: Permutation-Separated STH



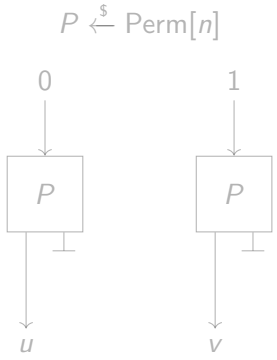
- We re-choose the permutation, but keep the same distribution
- $\mathrm{Perm}_{\mathrm{comp}}(u, v)$ is the set of all permutations that give (u, v) as truncation output

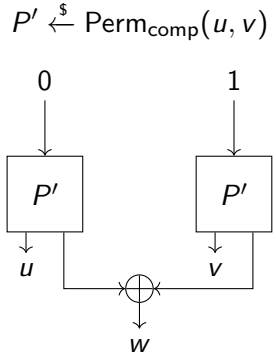
Proof Sketch: Permutation-Separated STH



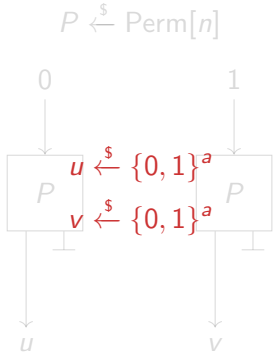
- We re-choose the permutation, but keep the same distribution
- $\mathrm{Perm}_{\mathrm{comp}}(u, v)$ is the set of all permutations that give (u, v) as truncation output
- No shared secret, as u and v are public!

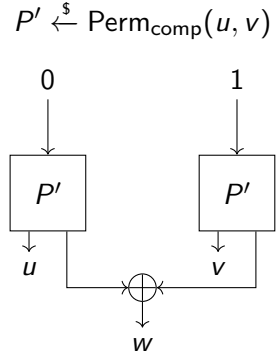
Proof Sketch: Isolating Truncation





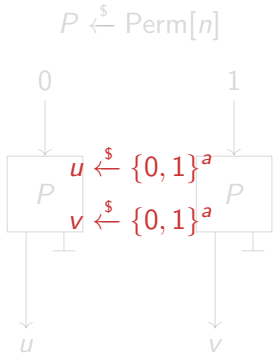
Proof Sketch: Isolating Truncation

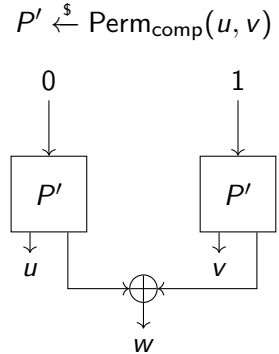




- Replace the truncation by a random function

Proof Sketch: Isolating Truncation

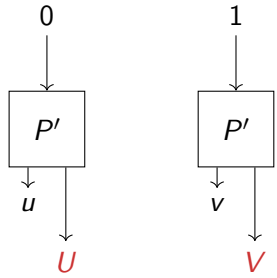




- Replace the truncation by a random function
- $\mathrm{Perm}_{\mathrm{comp}}(u, v)$ still well-defined, although u and v are generated differently

Proof Sketch: Summation Modifications

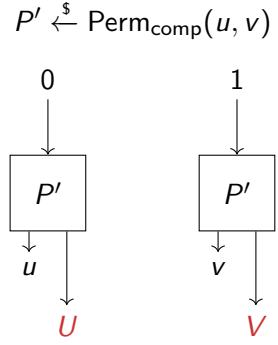
$P' \xleftarrow{\$} \mathsf{Perm}_{\mathsf{comp}}(u, v)$



$w = U \oplus V$

Proof Sketch: Summation Modifications

- Modified summation with subvalues U and V

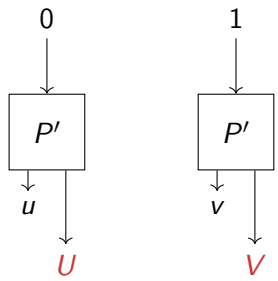


$$w = U \oplus V$$

Proof Sketch: Summation Modifications

- Modified summation with subvalues U and V
- What are their distributions?

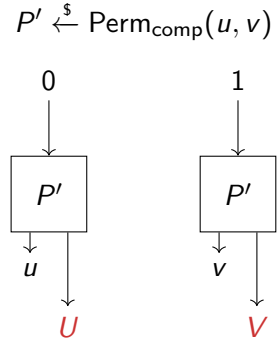
$$P' \xleftarrow{\$} \mathrm{Perm}_{\mathrm{comp}}(u, v)$$



$$w = U \oplus V$$

Proof Sketch: Summation Modifications

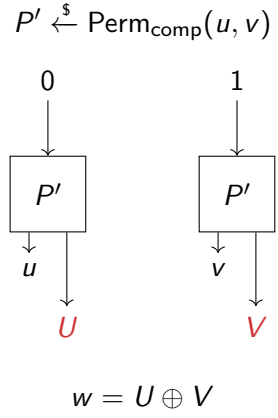
- Modified summation with subvalues U and V
- What are their distributions?
- U has a uniform distribution



$$w = U \oplus V$$

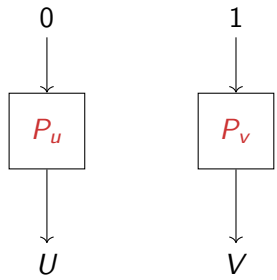
Proof Sketch: Summation Modifications

- Modified summation with subvalues U and V
- What are their distributions?
- U has a uniform distribution
- For V it depends:
  - If $v \neq u$, V is uniform from $\{0,1\}^b$
  - If $v = u$, V is uniform from $\{0,1\}^b \setminus \{U\}$



Proof Sketch: Alternative Description

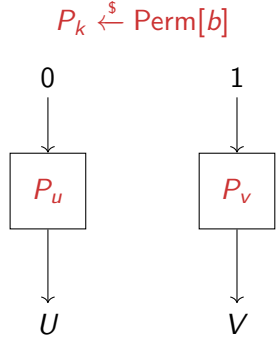
$P_k \xleftarrow{\$} \mathrm{Perm}[b]$



$w = U \oplus V$

Proof Sketch: Alternative Description

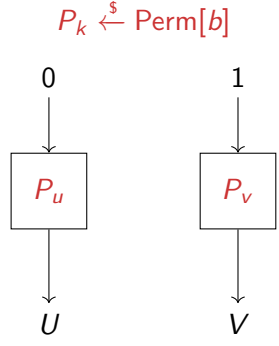
- We modify the construction to a family of permutations



$$w = U \oplus V$$

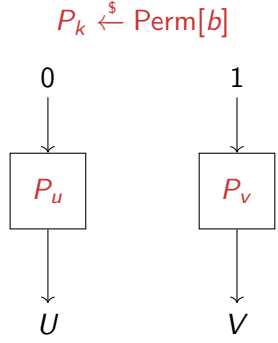
Proof Sketch: Alternative Description

- We modify the construction to a family of permutations
- We get the same distributions of U and V



Proof Sketch: Alternative Description

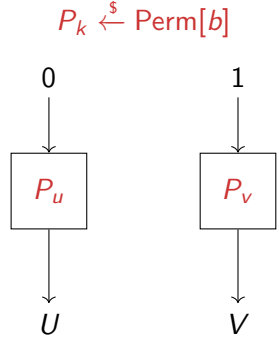
- We modify the construction to a family of permutations
- We get the same distributions of U and V
- U has a uniform distribution



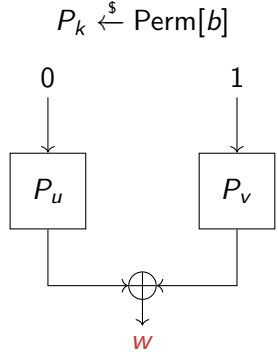
$$w = U \oplus V$$

Proof Sketch: Alternative Description

- We modify the construction to a family of permutations
- We get the same distributions of U and V
- U has a uniform distribution
- For V it depends:
  - If $v \neq u$, V is uniform from $\{0,1\}^b$
  - If $v = u$, V is uniform from $\{0,1\}^b \setminus \{U\}$

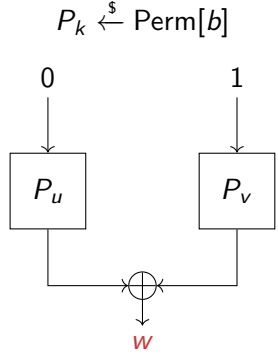


Proof Sketch: Generalized Summation



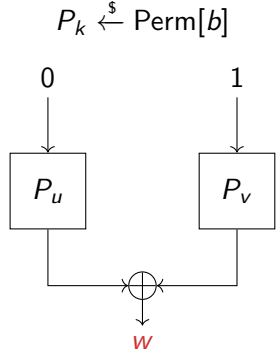
Proof Sketch: Generalized Summation

- This construction is a generalization of summation



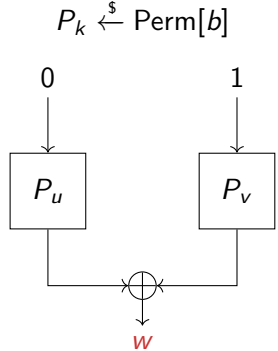
Proof Sketch: Generalized Summation

- This construction is a generalization of summation
- We modify the proof of summation to cover this variant

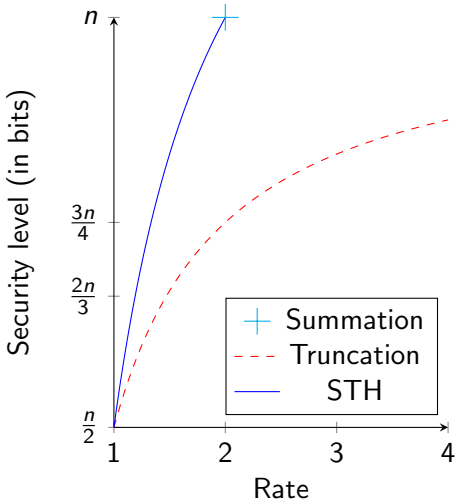


Proof Sketch: Generalized Summation

- This construction is a generalization of summation
- We modify the proof of summation to cover this variant
- Uses the χ^2-technique



Comparison



Construction	Rate (in/out)	Security level (in bits)
Summation	2	n
Truncation	n/a	$n - a/2$
STH	$2n/(n+a)$	$n - a/2$

Conclusion

- A lot of cryptographic constructions use PRFs

Conclusion

- A lot of cryptographic constructions use PRFs
- Dedicated PRFs are not conventional, block ciphers are more commonly built

Conclusion

- A lot of cryptographic constructions use PRFs
- Dedicated PRFs are not conventional, block ciphers are more commonly built
- An established PRP-to-PRF conversion is truncation

Conclusion

- A lot of cryptographic constructions use PRFs
- Dedicated PRFs are not conventional, block ciphers are more commonly built
- An established PRP-to-PRF conversion is truncation
- However, it discards valuable PRP-output

Conclusion

- A lot of cryptographic constructions use PRFs
- Dedicated PRFs are not conventional, block ciphers are more commonly built
- An established PRP-to-PRF conversion is truncation
- However, it discards valuable PRP-output
- We can reuse these bits without security loss with summation: STH

Conclusion

- A lot of cryptographic constructions use PRFs
- Dedicated PRFs are not conventional, block ciphers are more commonly built
- An established PRP-to-PRF conversion is truncation
- However, it discards valuable PRP-output
- We can reuse these bits without security loss with summation: STH
- Might be interesting to expand to more permutation calls

Conclusion

- A lot of cryptographic constructions use PRFs
- Dedicated PRFs are not conventional, block ciphers are more commonly built
- An established PRP-to-PRF conversion is truncation
- However, it discards valuable PRP-output
- We can reuse these bits without security loss with summation: STH
- Might be interesting to expand to more permutation calls

Thank you for your attention!