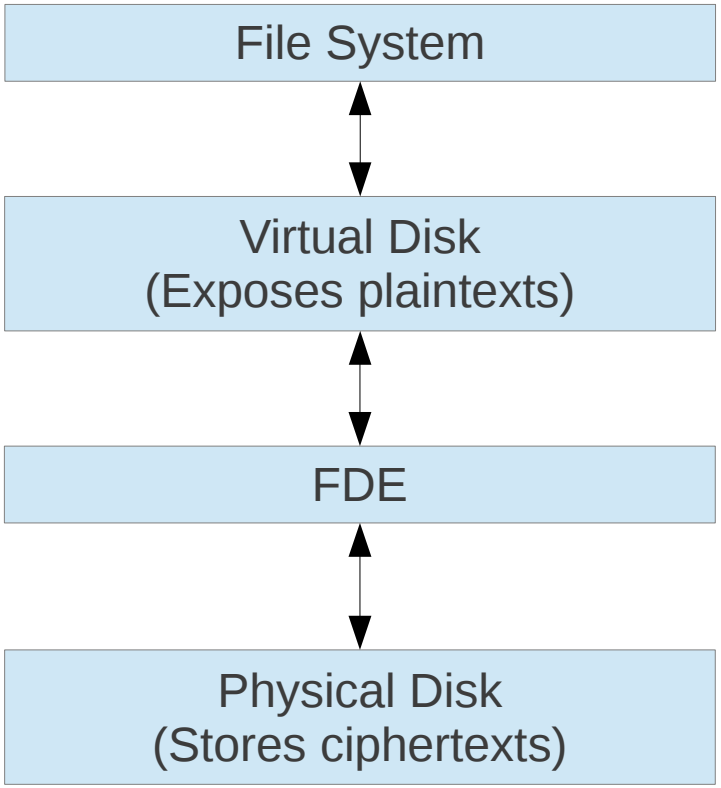
A Modular Framework for Building Variable-Input-Length Tweakable Ciphers

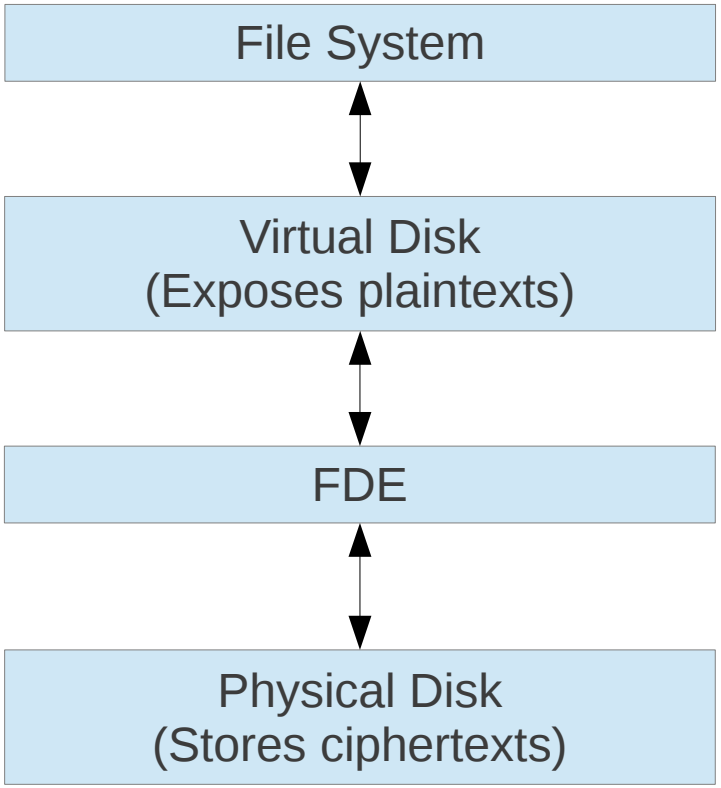
Thomas Shrimpton and **Seth Terashima**

Portland State University

Motivation:
Full Disk Encryption

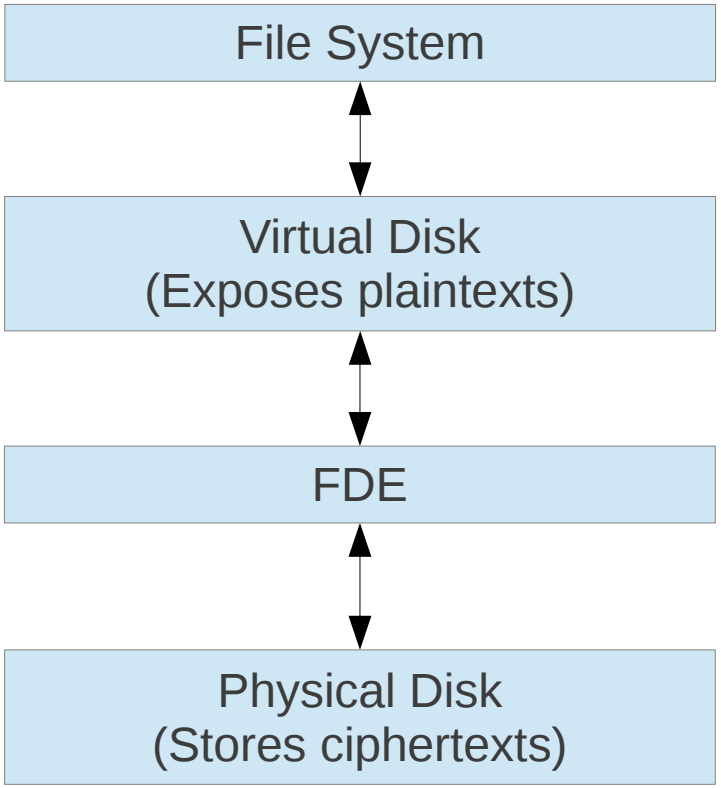


Motivation:
Full Disk Encryption



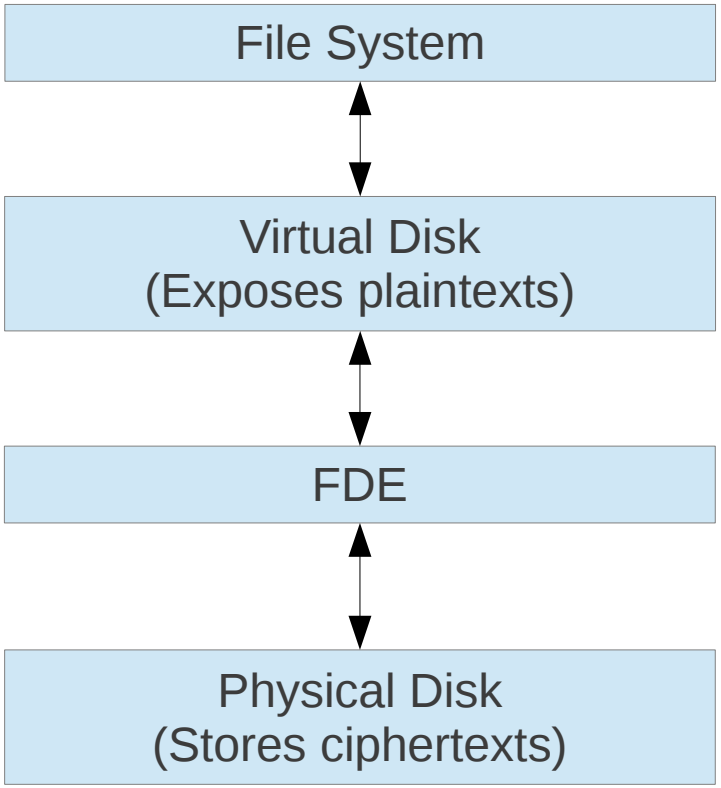
- Disks encrypted sector-by-sector
  - Plaintexts are sectors
  - No “file” abstraction

Motivation:
Full Disk Encryption



- Disks encrypted sector-by-sector
  - Plaintexts are sectors
  - No “file” abstraction
- Accessing a plaintext shouldn't result in accessing multiple HW sectors

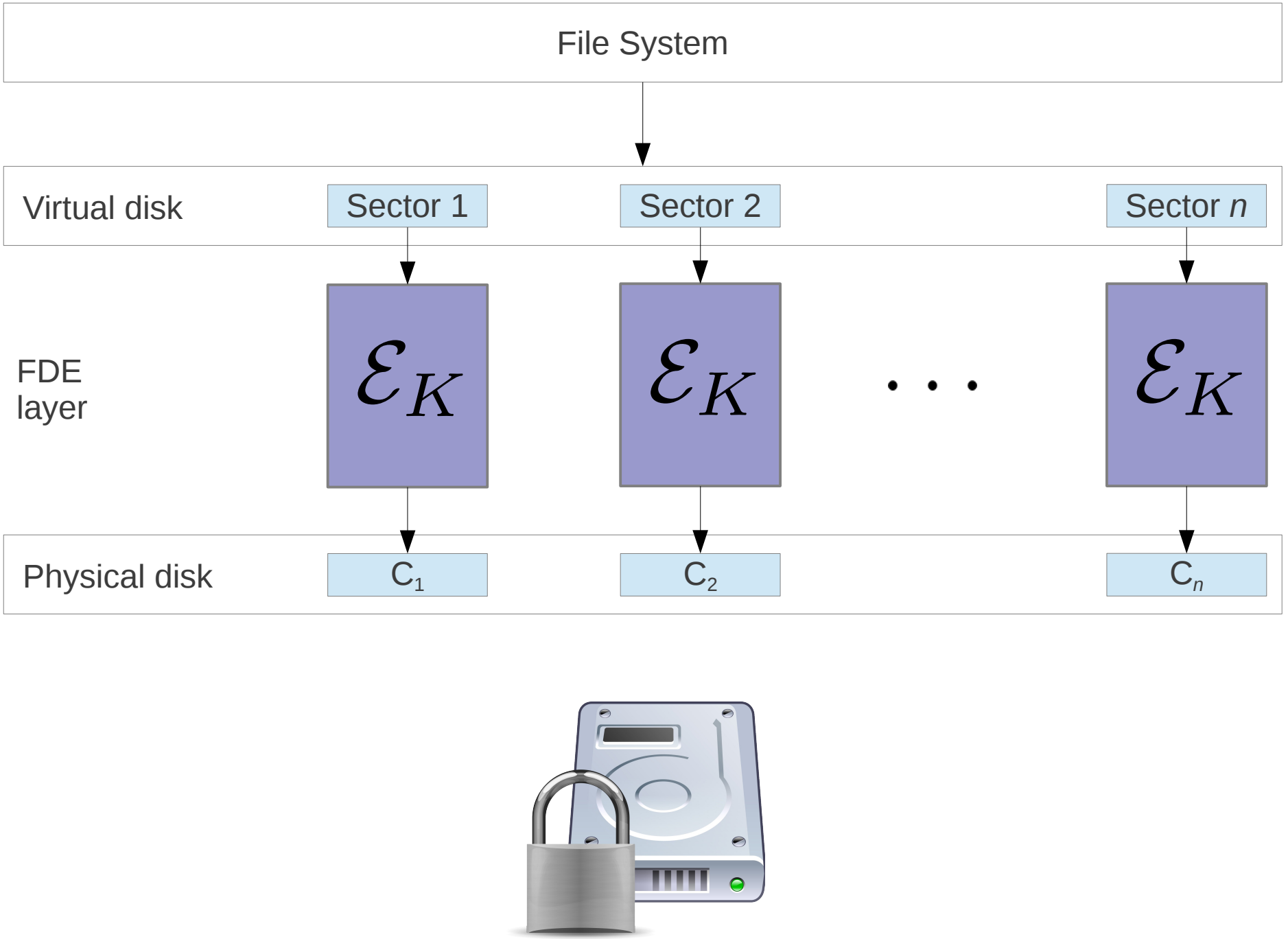
Motivation: Full Disk Encryption

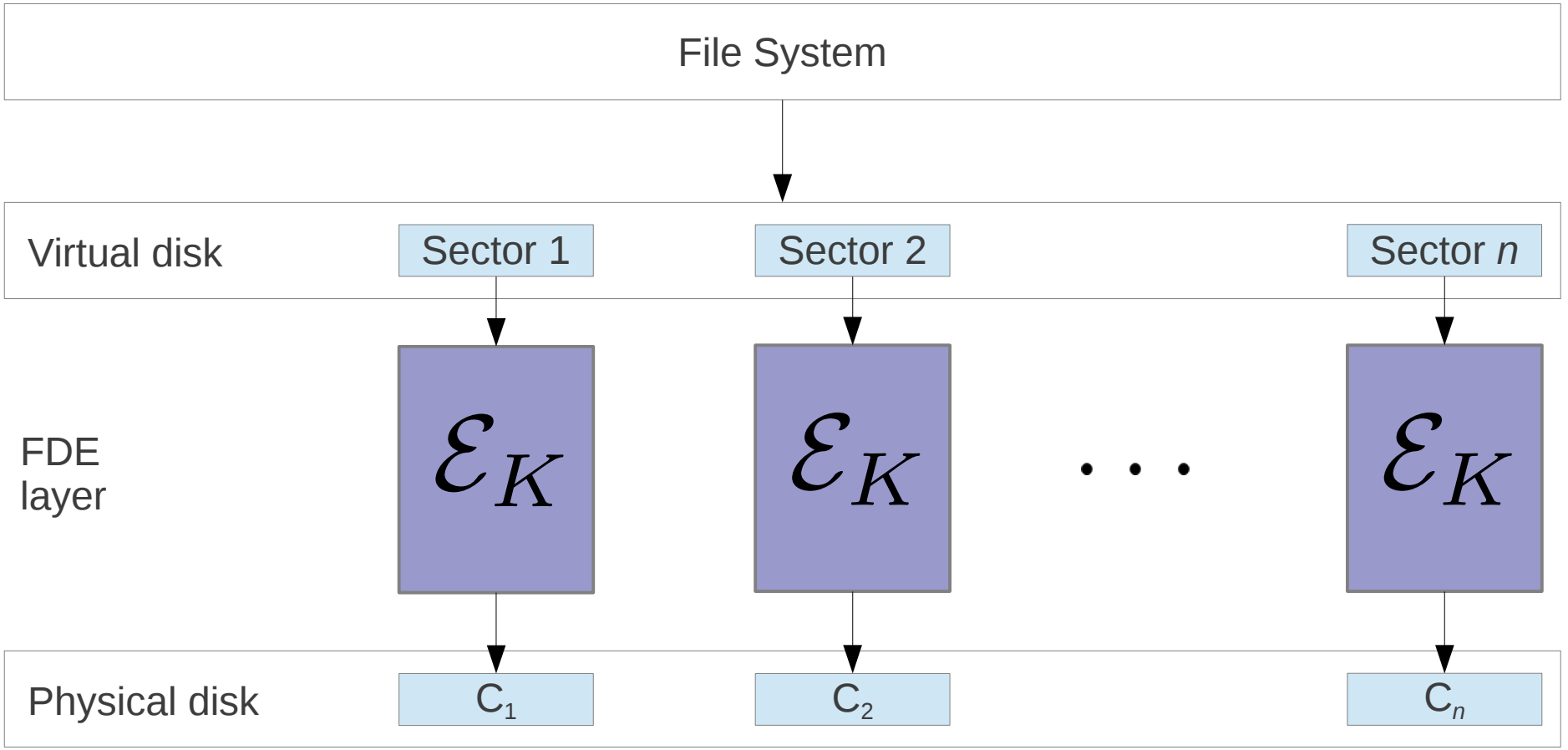


- Disks encrypted sector-by-sector
  - Plaintexts are sectors
  - No “file” abstraction
- Accessing a plaintext shouldn't result in accessing multiple HW sectors

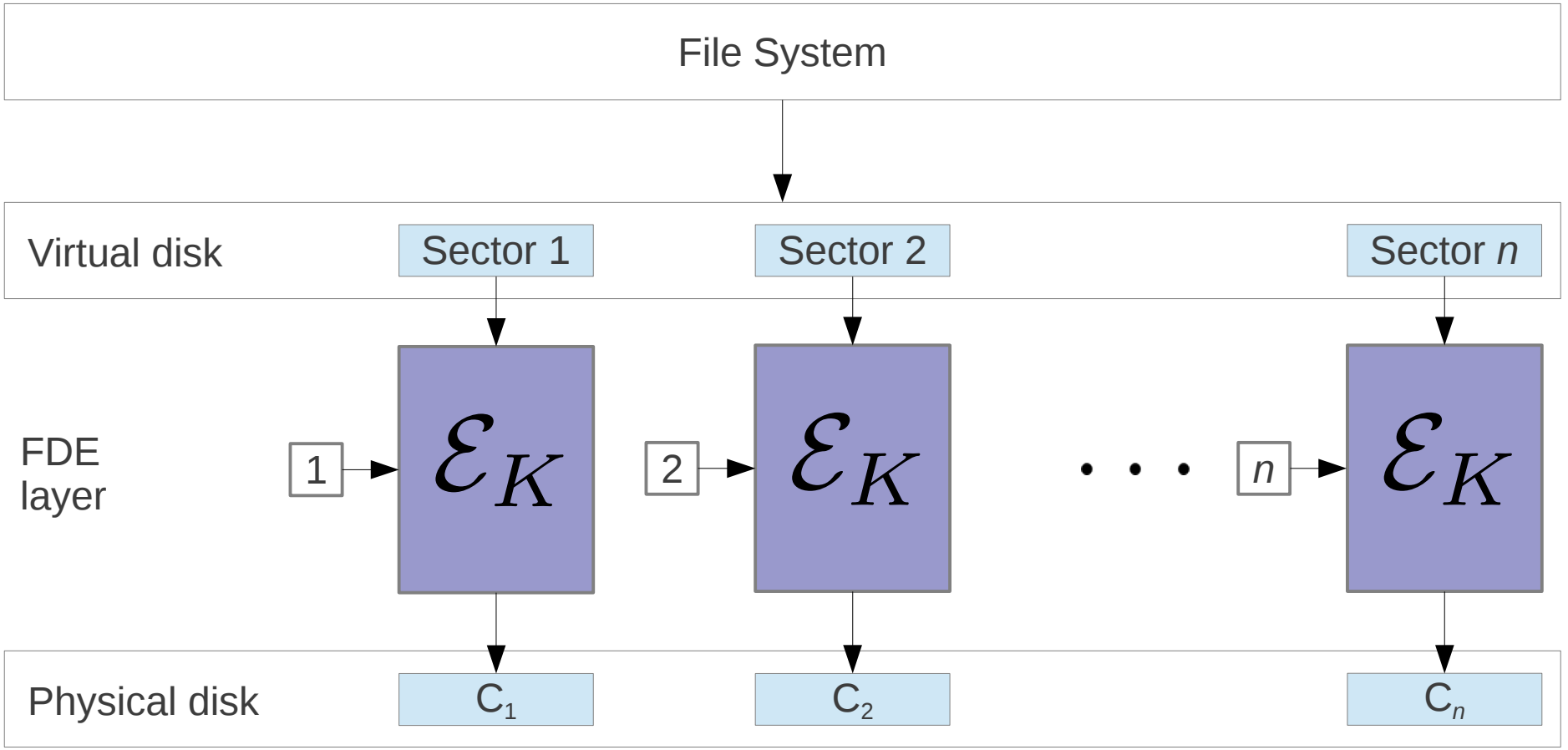
Therefore plaintext length = ciphertext length

- No room for IV bits
- No room for MAC bits


File System
Virtual disk
Sector 1
Sector 2
Sector n
FDE layer
$\mathcal{E}_K$
$\mathcal{E}_K$
$\mathcal{E}_K$
Physical disk
C_1
C_2
C_n



Problem: This looks uncomfortably like ECB (albeit with 4kB blocks)...



Problem: This looks uncomfortably like ECB (albeit with 4kB blocks)...

Solution (?): Use Sector IDs as IVs.

Nonce-based encryption isn't enough.

- What if an attacker images the disk at two different times?

Nonce-based encryption isn't enough.

- What if an attacker images the disk at two different times?

  Should only leak equality of plaintexts

  $\mathcal{E}_K(n, \cdot)$ should look like a random permutation

Nonce-based encryption isn't enough.

- What if an attacker images the disk at two different times?

  Should only leak equality of plaintexts

  $\mathcal{E}_K(n, \cdot)$ should look like a random permutation

- What if an attacker tampers with a ciphertext?

Nonce-based encryption isn't enough.

- What if an attacker images the disk at two different times?

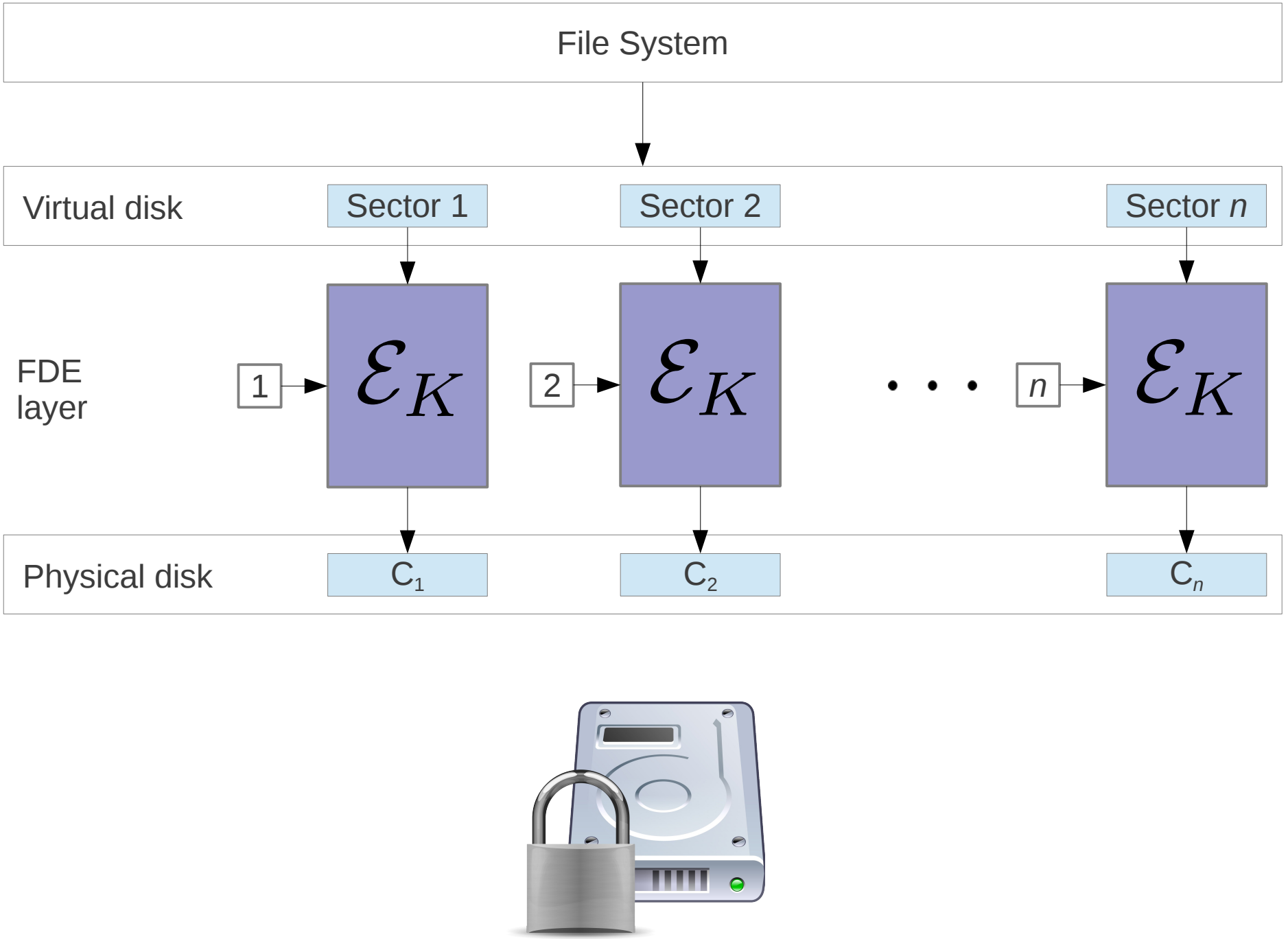
  Should only leak equality of plaintexts

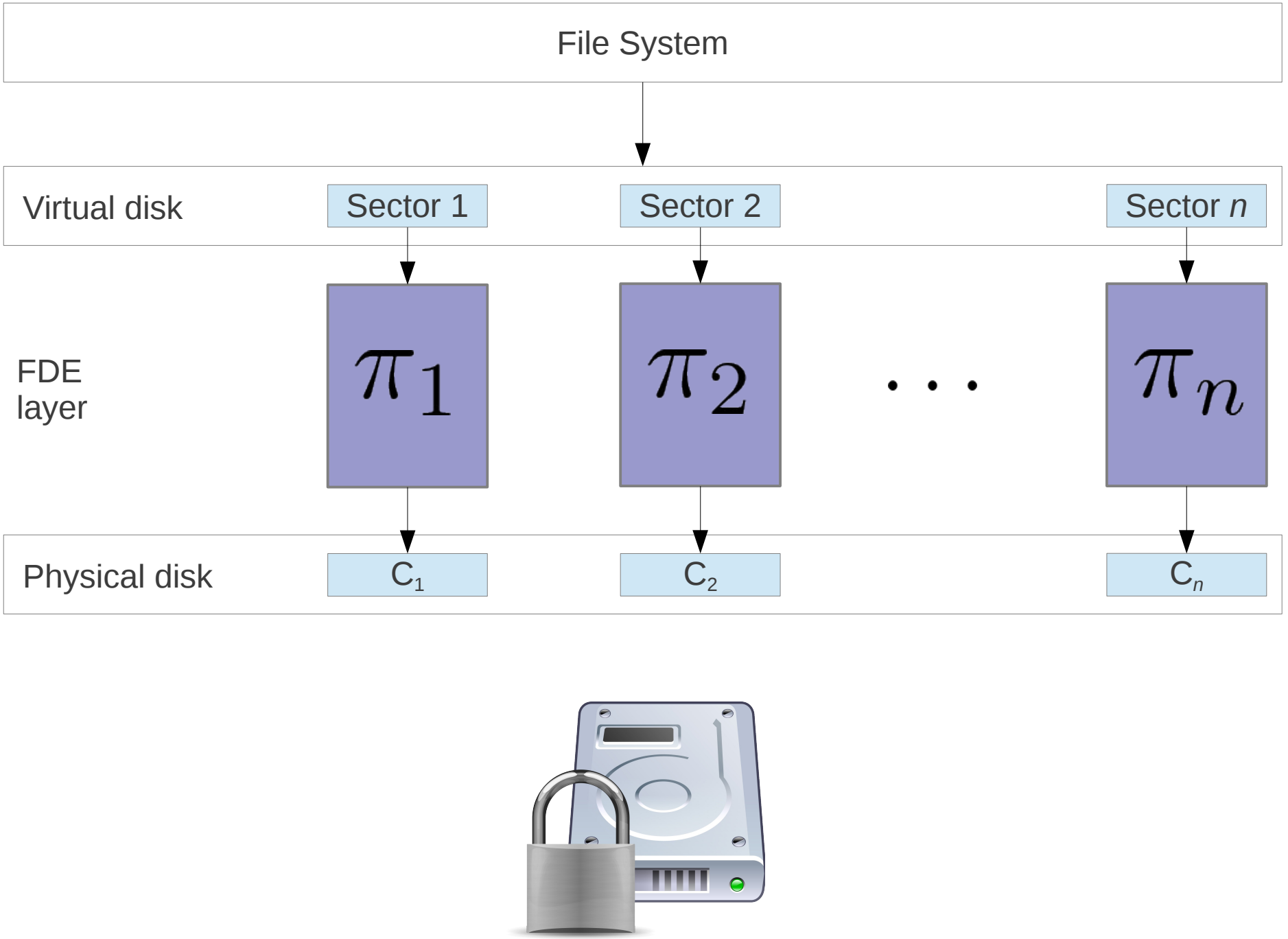
  $\mathcal{E}_K(n, \cdot)$ should look like a random permutation

- What if an attacker tampers with a ciphertext?

  Entire plaintext sector should be corrupted

  $\mathcal{E}_K^{-1}(n, \cdot)$ should look like a random permutation


File System
Virtual disk
Sector 1
Sector 2
Sector n
FDE layer
1
2
n
$\mathcal{E}_K$
$\mathcal{E}_K$
$\mathcal{E}_K$
Physical disk
C_1
C_2
C_n


File System
Virtual disk
Sector 1
Sector 2
Sector n
FDE layer
π_1
π_2
π_n
Physical disk
C_1
C_2
C_n

Tweakable (block)ciphers

- A good *tweakable* blockcipher “looks like” a *family* of independent, random permutations

Tweakable (block)ciphers

- A good *tweakable* blockcipher “looks like” a *family* of independent, random permutations

$$\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^n$$

Tweakable (block)ciphers

- A good *tweakable* blockcipher “looks like” a *family* of independent, random permutations

$$\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^n$$

Tweak

Tweakable (block)ciphers

- A good *tweakable* blockcipher “looks like” a *family* of independent, random permutations

$$\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^n$$

Tweak

$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{E}}(A) = \Pr\left[A^{\widetilde{E}_K, \widetilde{E}_K^{-1}} \Rightarrow 1 \right] - \Pr\left[A^{\Pi, \Pi^{-1}} \Rightarrow 1 \right]$$

Family of independent, random permutations

Tweakable (block)ciphers

- A good *tweakable* blockcipher “looks like” a *family* of independent, random permutations

$$\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$$

Tweak

$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{E}}(A) = \Pr\left[A^{\widetilde{E}_K, \widetilde{E}_K^{-1}} \Rightarrow 1 \right] - \Pr\left[A^{\Pi, \Pi^{-1}} \Rightarrow 1 \right]$$

Family of independent, random permutations

- FDE demands a “wideblock” STPRP (512 or 4096 byte blocks)

VIL Tweakable Ciphers

- VIL = Variable input length
  - Still preserves length of input
  - Random permutation for each length and tweak

VIL Tweakable Ciphers

- VIL = Variable input length
  - Still preserves length of input
  - Random permutation for each length and tweak
- Existing constructions
  - CMC, EME*, PEP, TET, HEH, HCTR, …
  - Security reduction to underlying *n*-bit blockcipher
  - Birthday-bound security (wrt *n*)
  - Either:

    2 blockcipher calls *or*

    1 blockcipher call, 1 GF multiply

    per *n* bits of input

Results

PIV: A new approach to VIL TCs

AEAD from VIL TCs

Results

PIV: A new approach to VIL TCs

AEAD from VIL TCs

Results

PIV: A new approach to VIL TCs

- TCT2: First to break the birthday bound

AEAD from VIL TCs

Results

PIV: A new approach to VIL TCs

- TCT2: First to break the birthday bound
- TCT1: First to require a single blockcipher call (and no finite field multiplications) for each *n* bits of input

AEAD from VIL TCs

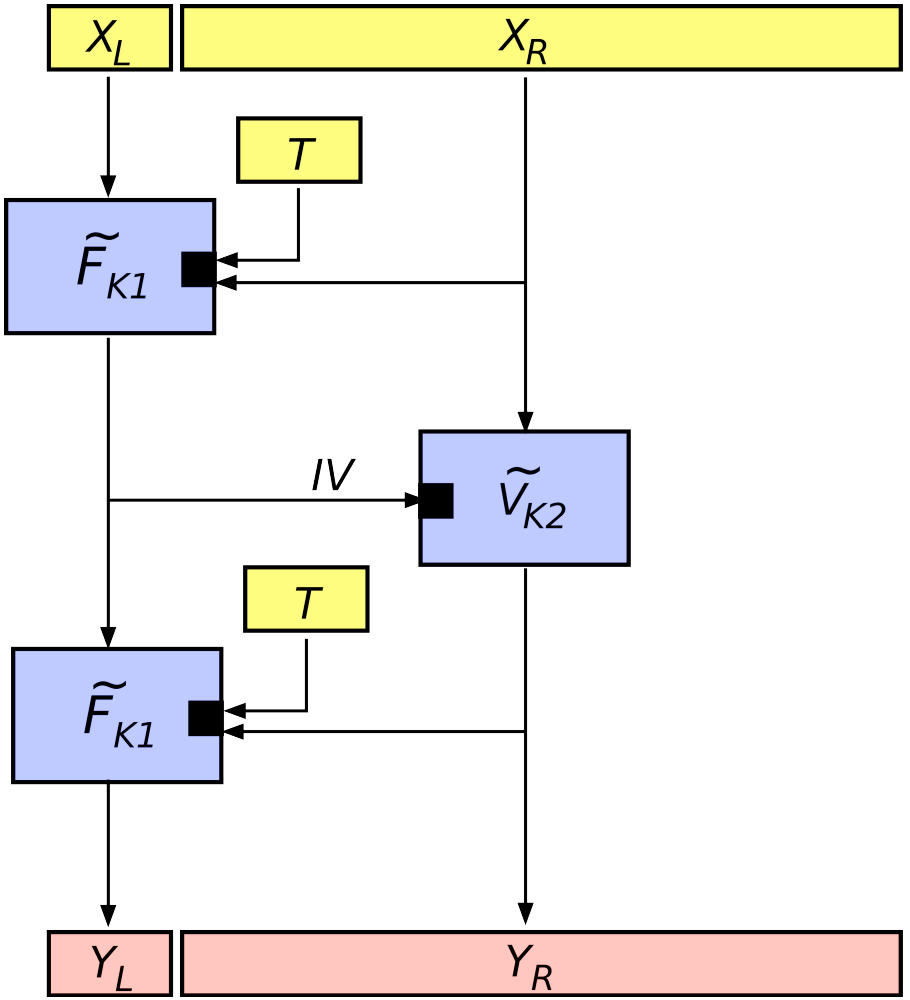
Results

PIV: A new approach to VIL TCs

- TCT2: First to break the birthday bound
- TCT1: First to require a single blockcipher call (and no finite field multiplications) for each *n* bits of input
- Simple, easily verified security proof

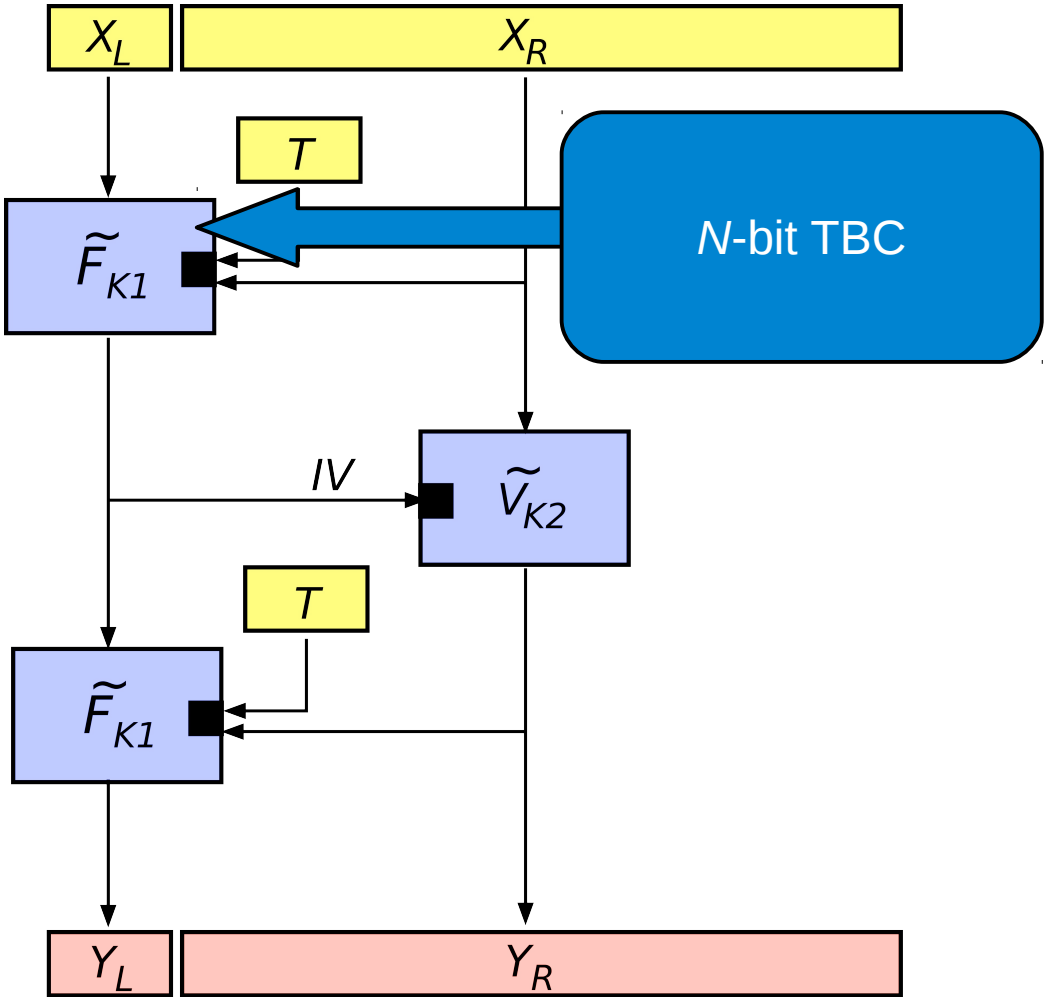
AEAD from VIL TCs

$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$



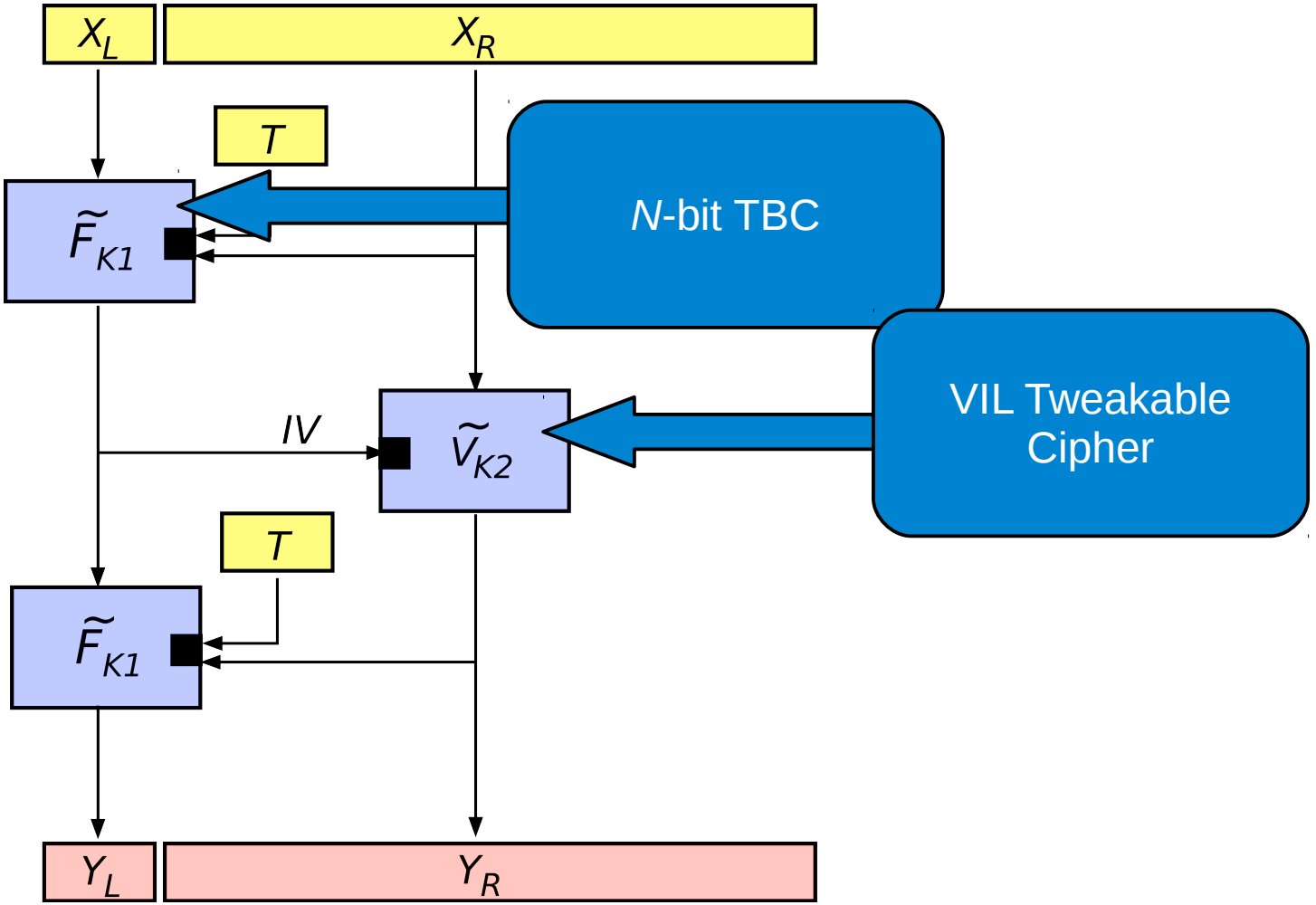
Protected IV Mode

$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$



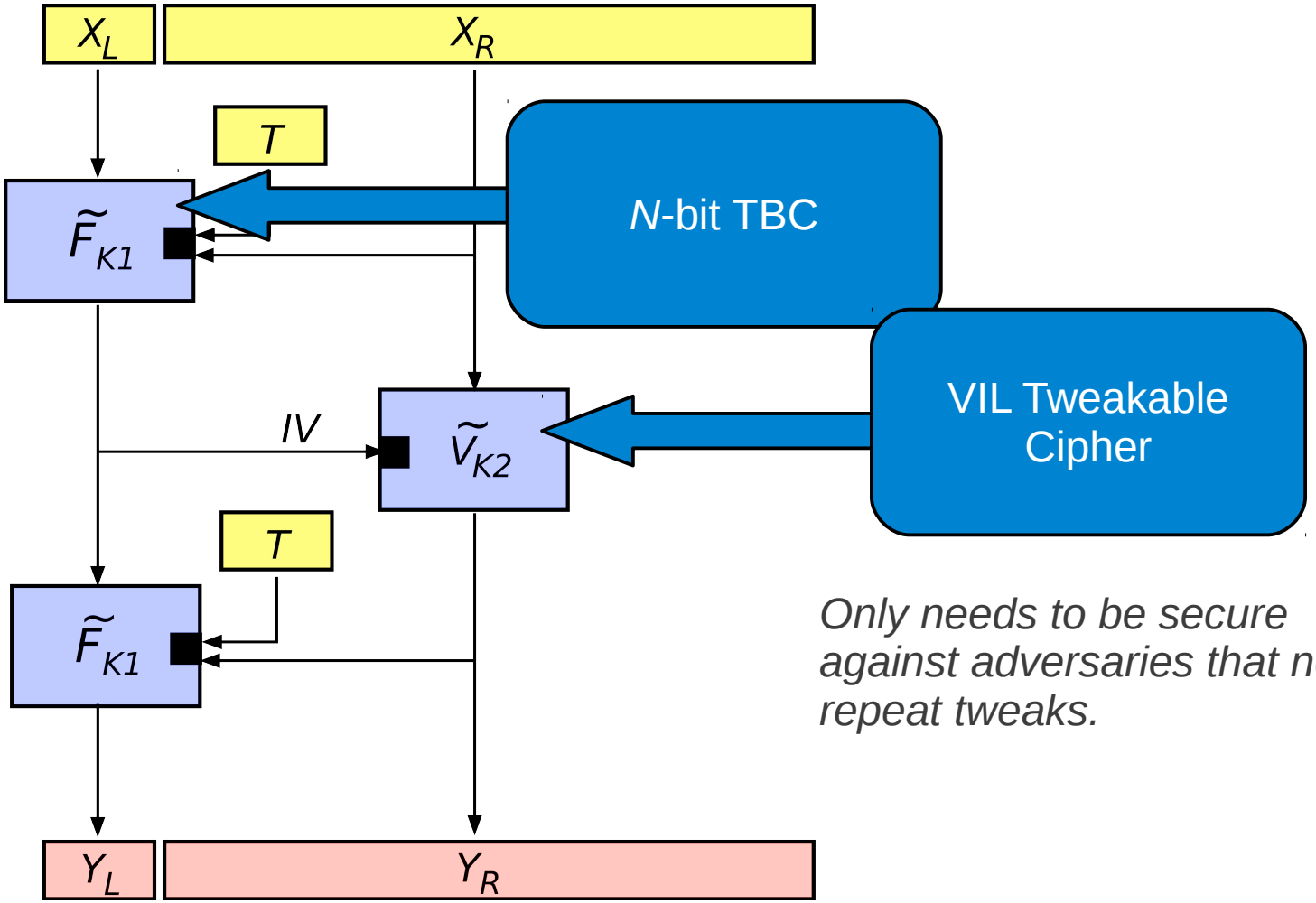
Protected IV Mode

$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$



Protected IV Mode

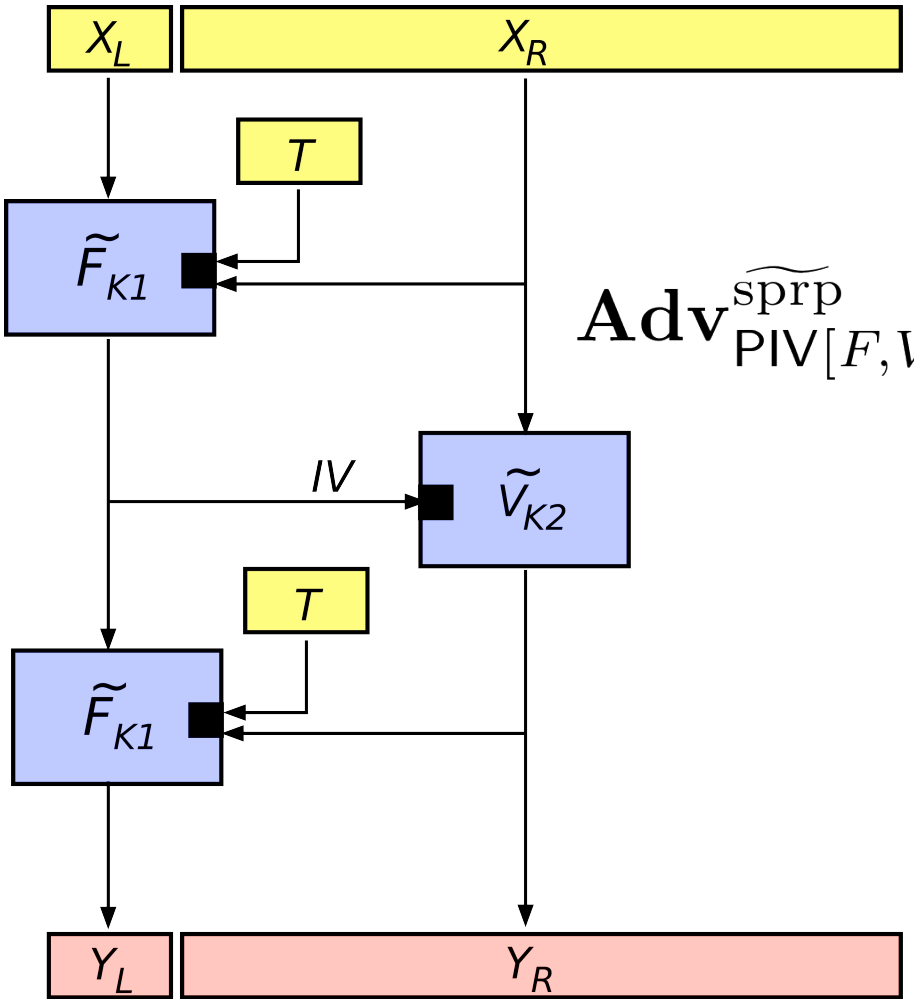
$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$



Only needs to be secure against adversaries that never repeat tweaks.

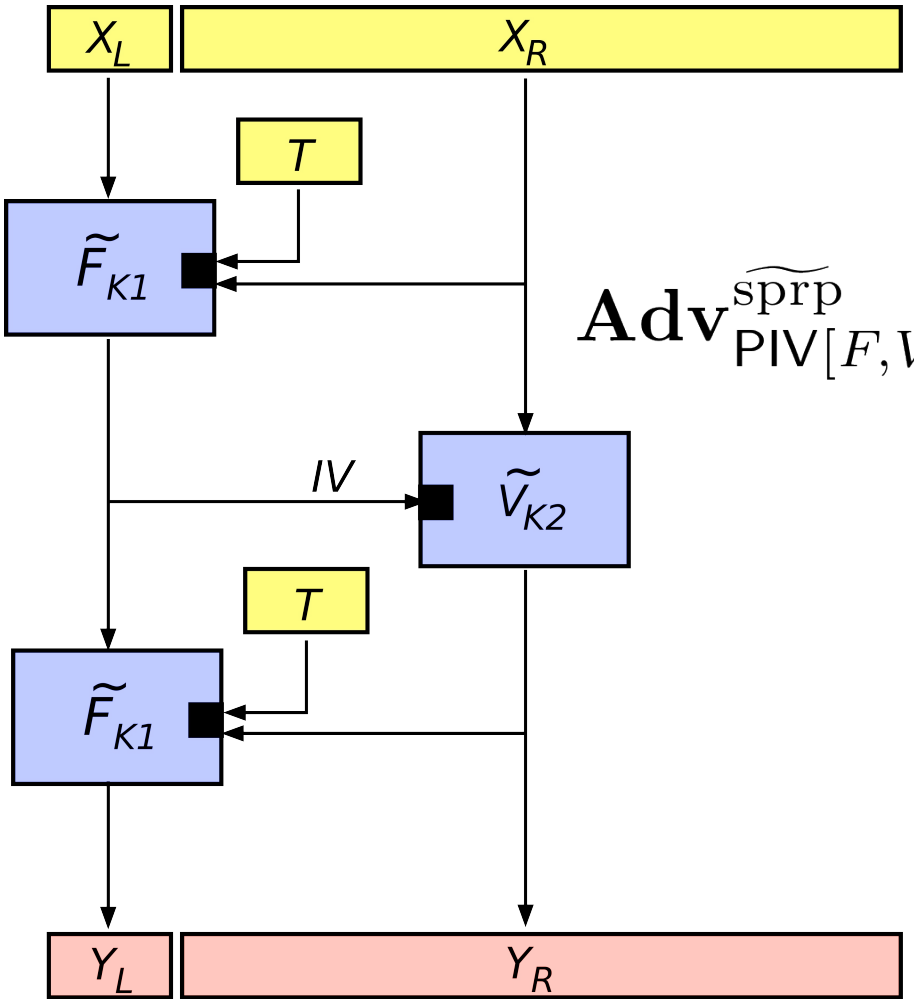
Protected IV Mode

$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$



$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \leq \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

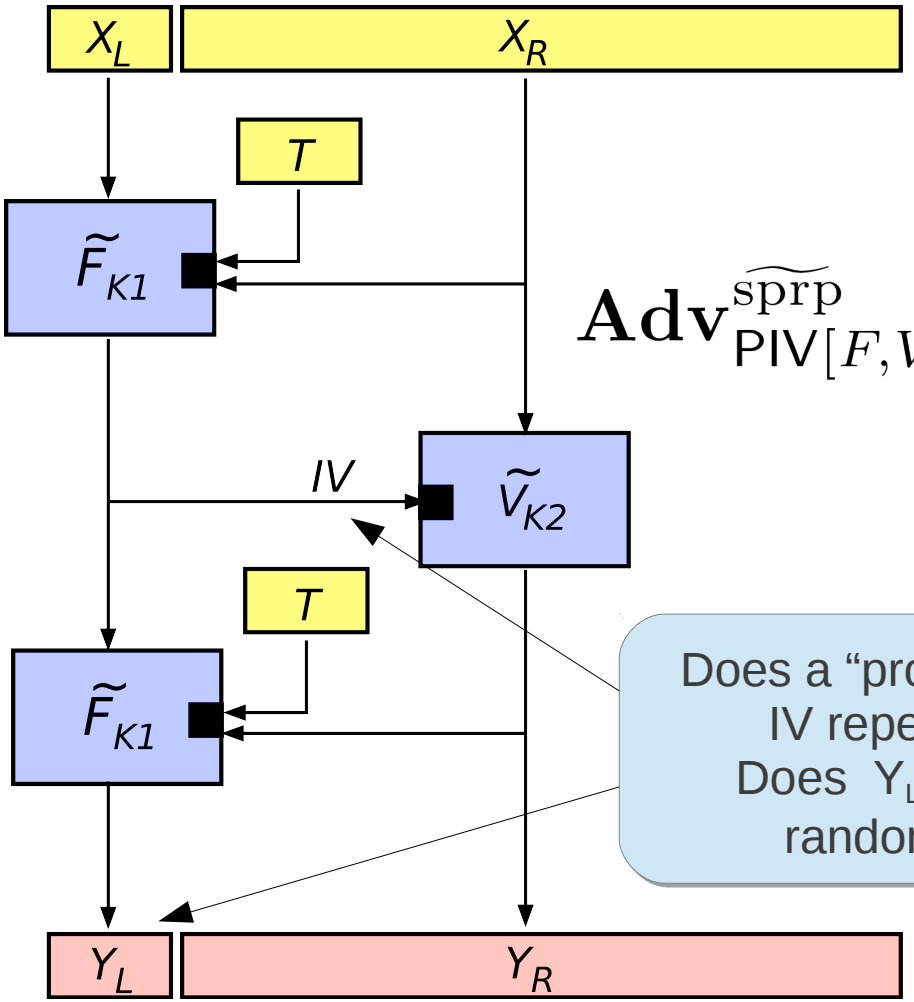
$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$



$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \leq \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

Doesn't repeat a tweak

$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$

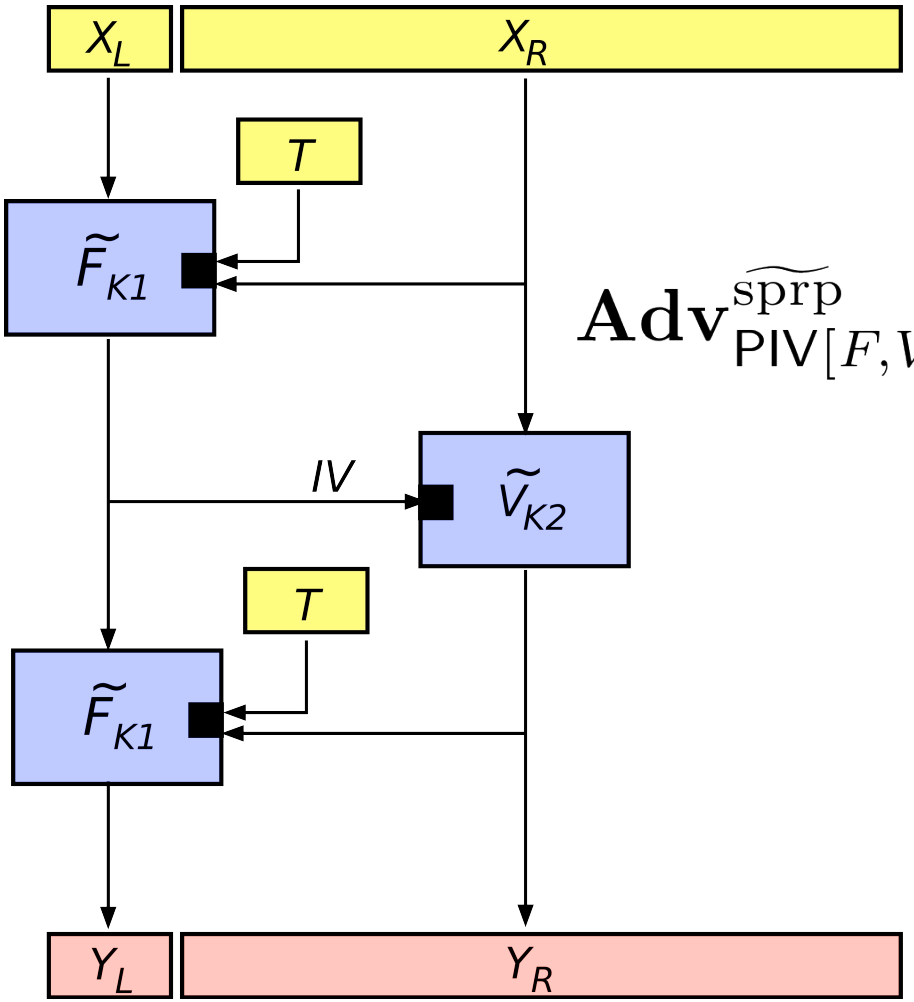


$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \leq \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

Does a “protected” IV repeat? Does Y_L look random?

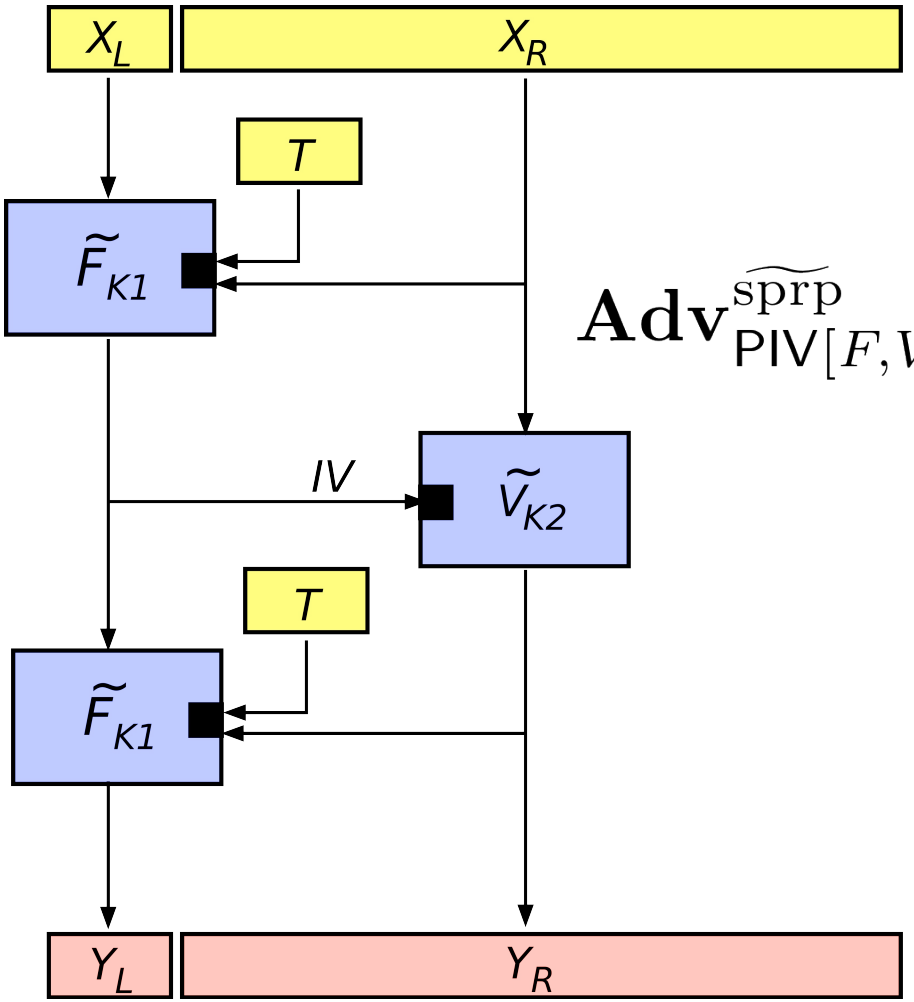
Doesn't repeat a tweak

$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$



$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \leq \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

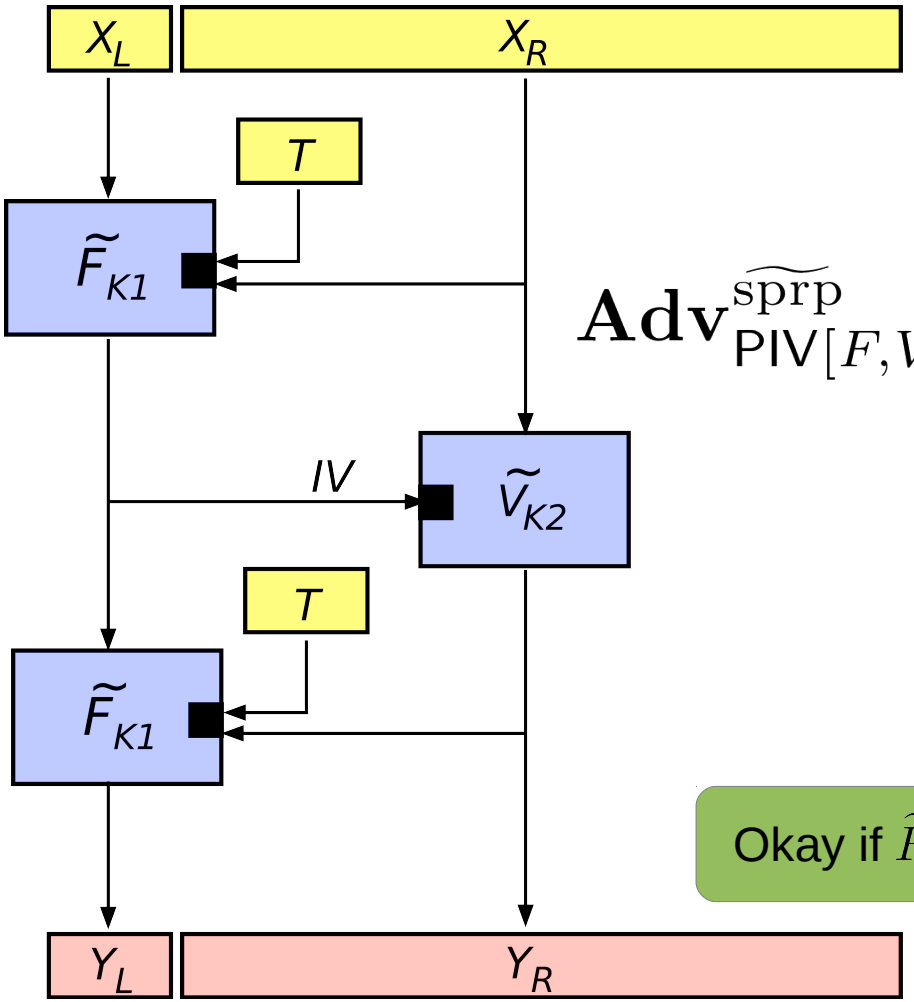
$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$



$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \leq \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

If we start with an *n*-bit blockcipher, we beat the b'day bound if $N > n$.

$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$

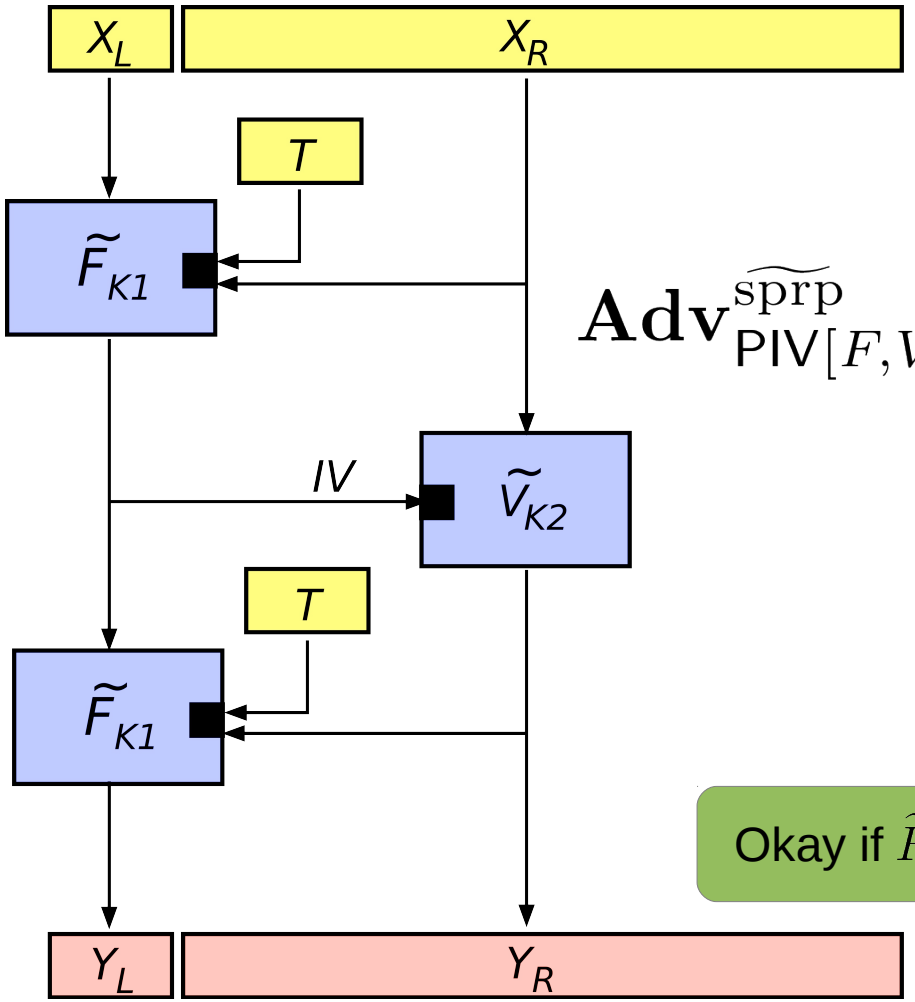


$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \leq \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

If we start with an *n*-bit blockcipher, we beat the b'day bound if $N > n$.

Okay if $\widetilde{F}$ is slow as long as $N \ll m$ and $\widetilde{V}$ is efficient

$$|X_L| = N \qquad |X_R| \in \{m, m+1, \ldots\}$$



$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[F,V]}(A) \le \frac{4q^2}{2^N} + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{F}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{V}(C)$$

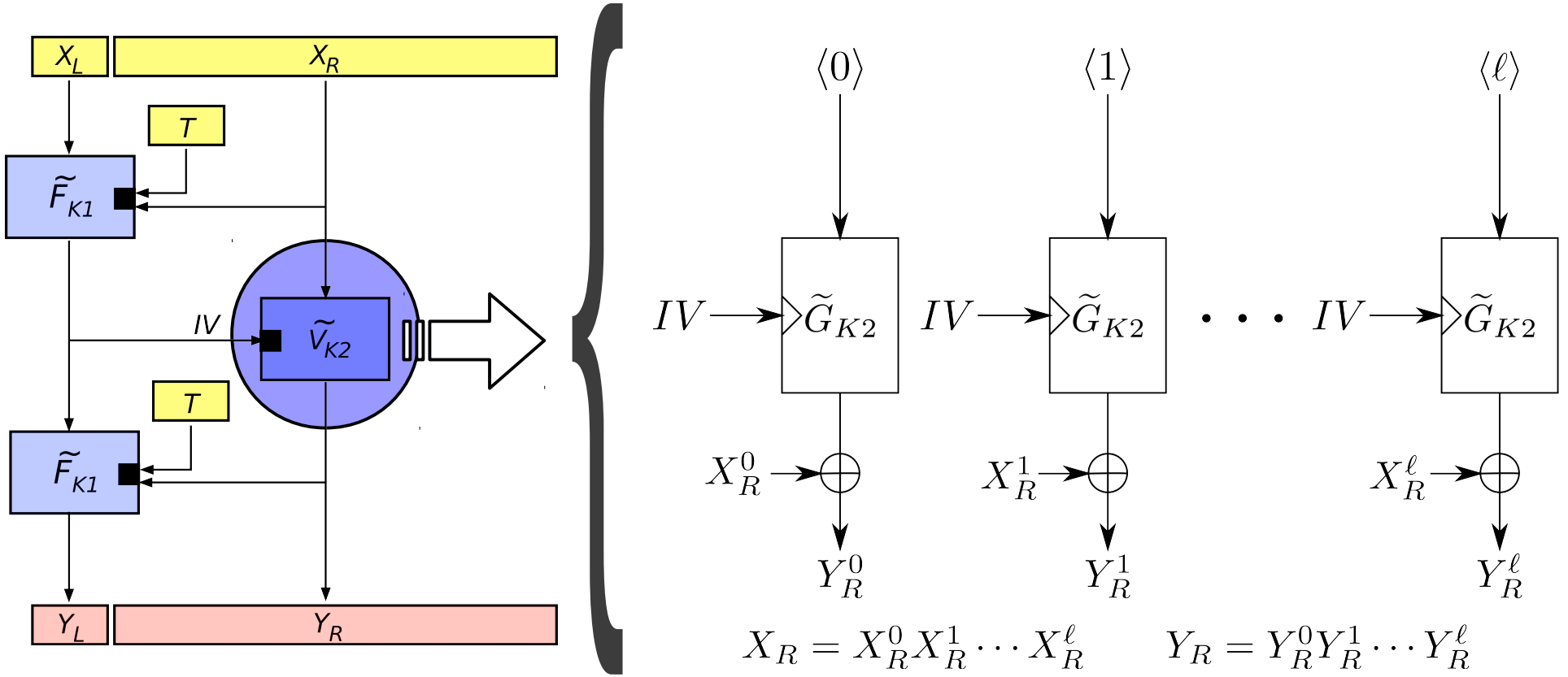
If we start with an *n*-bit blockcipher, we beat the b'day bound if $N > n$.

Okay if $\widetilde{F}$ is slow as long as $N \ll m$ and $\widetilde{V}$ is efficient

Standard 4KB disc sector, to scale (N = 256 bits)

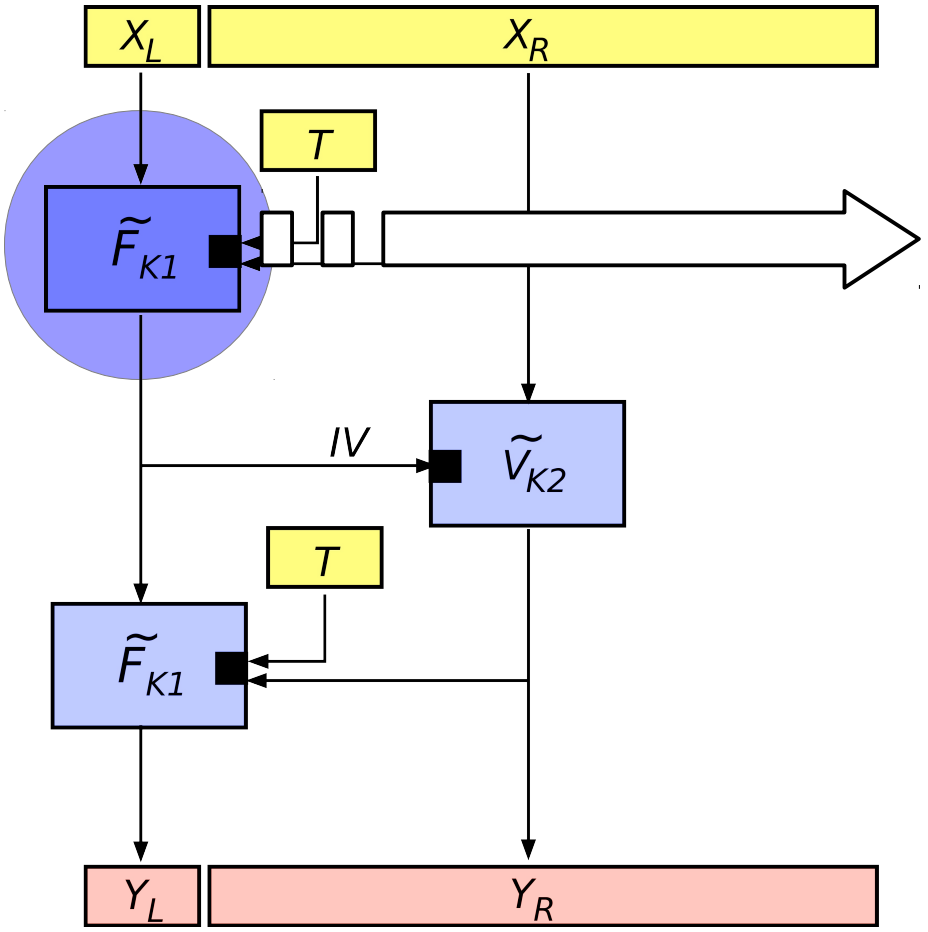
Y_L Y_R

TCT2: Constructing $\widetilde{V}$



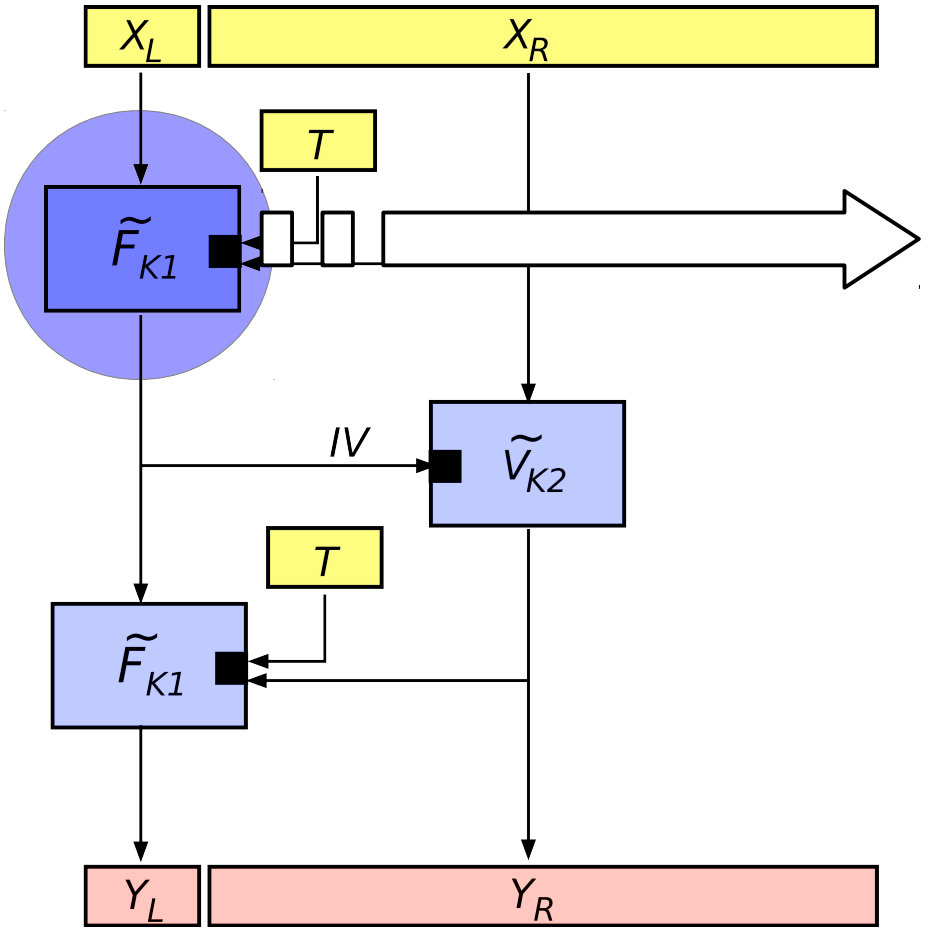
- Optimized for sector-sized messages (arbitrary length messages require incrementing the protected IV)
- Setting $\widetilde{G}$ = CLRW2 [LST '12] gives beyond b'day security
  - Makes two blockcipher calls per invocation

TCT2: Constructing $\widetilde{F}$



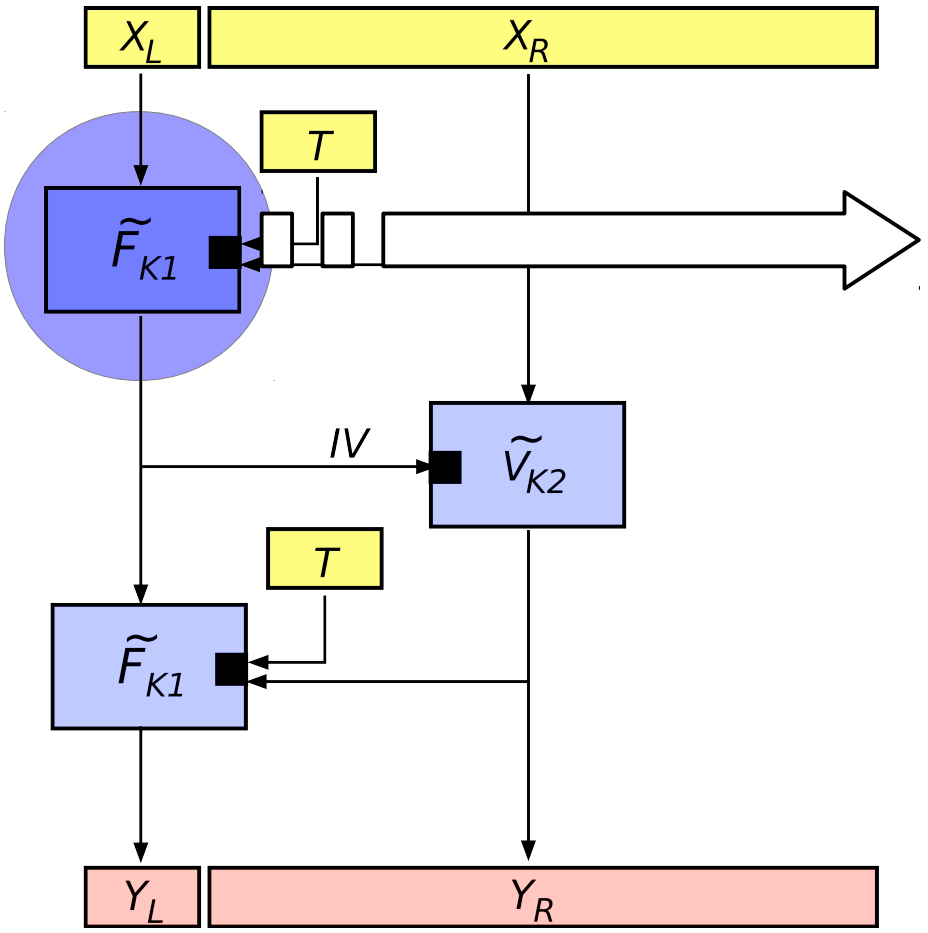
- Build an $N = 2n$-bit TBC out of an n-bit TBC [CDMS '10]

TCT2: Constructing $\widetilde{F}$



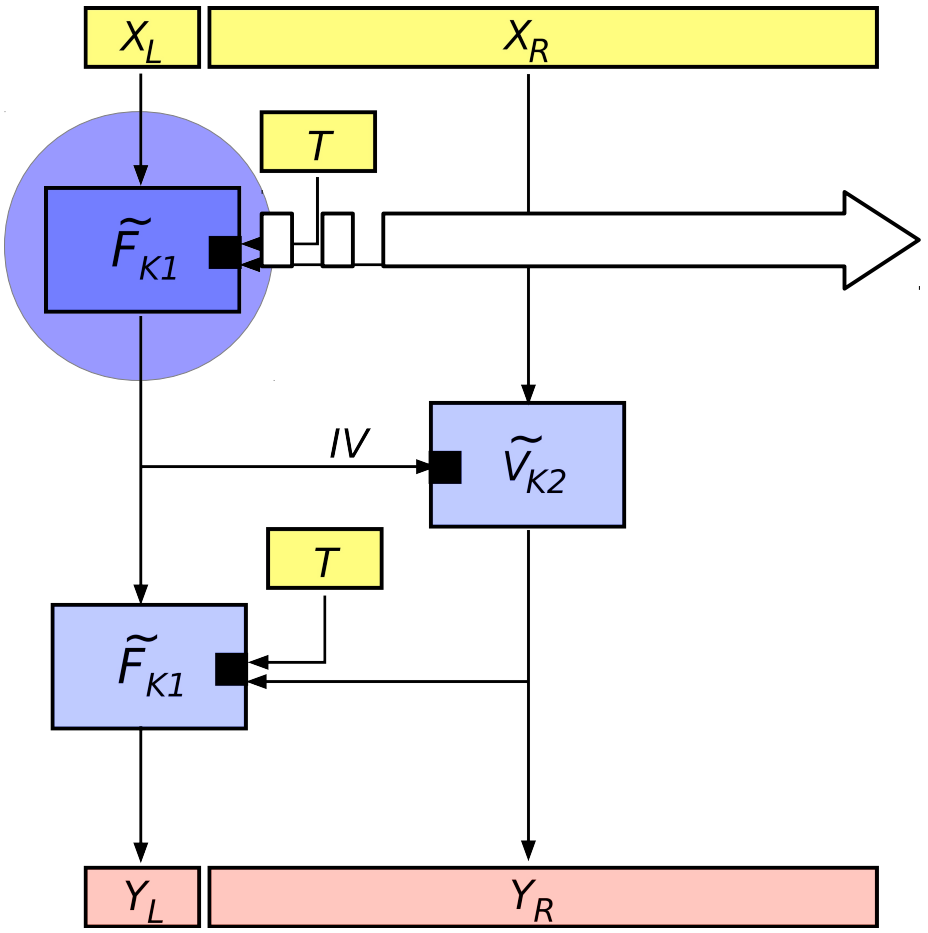
- Build an $N = 2n$-bit TBC out of an n-bit TBC [CDMS '10]
- Implement the n-bit TBC using CLRW2 [LST '12] over, e.g., AES

TCT2: Constructing $\widetilde{F}$



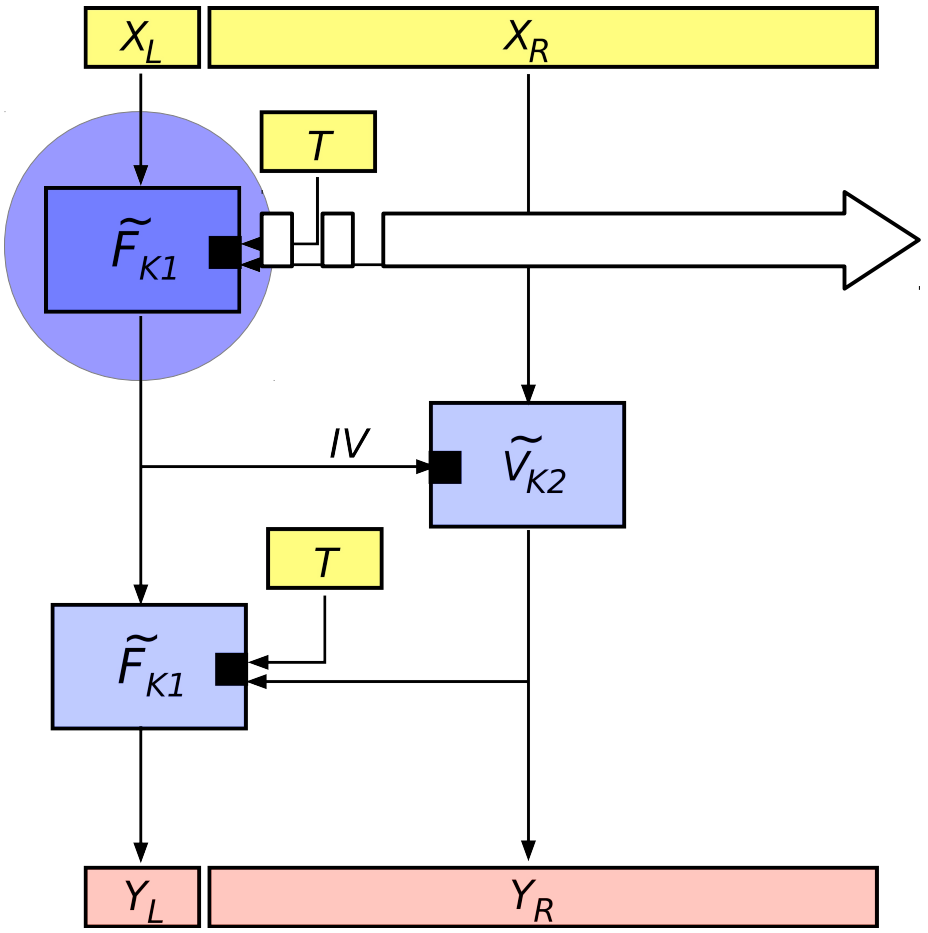
- Build an $N = 2n$-bit TBC out of an n-bit TBC [CDMS '10]
- Implement the n-bit TBC using CLRW2 [LST '12] over, e.g., AES
- Use NH [BHKKR '99] to extend the tweak length

TCT2: Constructing $\widetilde{F}$



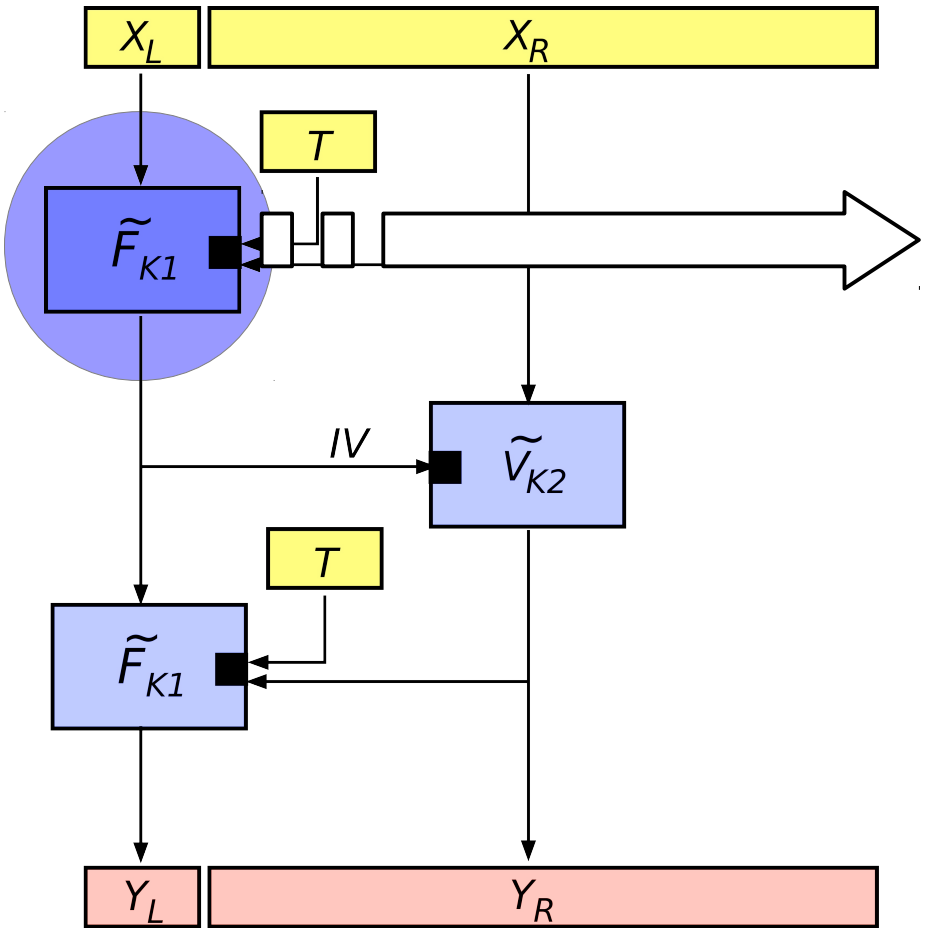
- Build an $N = 2n$-bit TBC out of an n-bit TBC [CDMS '10]
- Implement the n-bit TBC using CLRW2 [LST '12] over, e.g., AES
- Use NH [BHKKR '99] to extend the tweak length
- Secure to $O(2^{2n/3})$ queries

TCT2: Constructing $\widetilde{F}$



- Build an $N = 2n$-bit TBC out of an n-bit TBC [CDMS '10]
- Implement the n-bit TBC using CLRW2 [LST '12] over, e.g., AES
- Use NH [BHKKR '99] to extend the tweak length
- Secure to $O(2^{2n/3})$ queries
- The two F calls make a total:
  - 28 multiplies in GF_n
  - 12 n-bit blockcipher calls

TCT2: Constructing $\widetilde{F}$



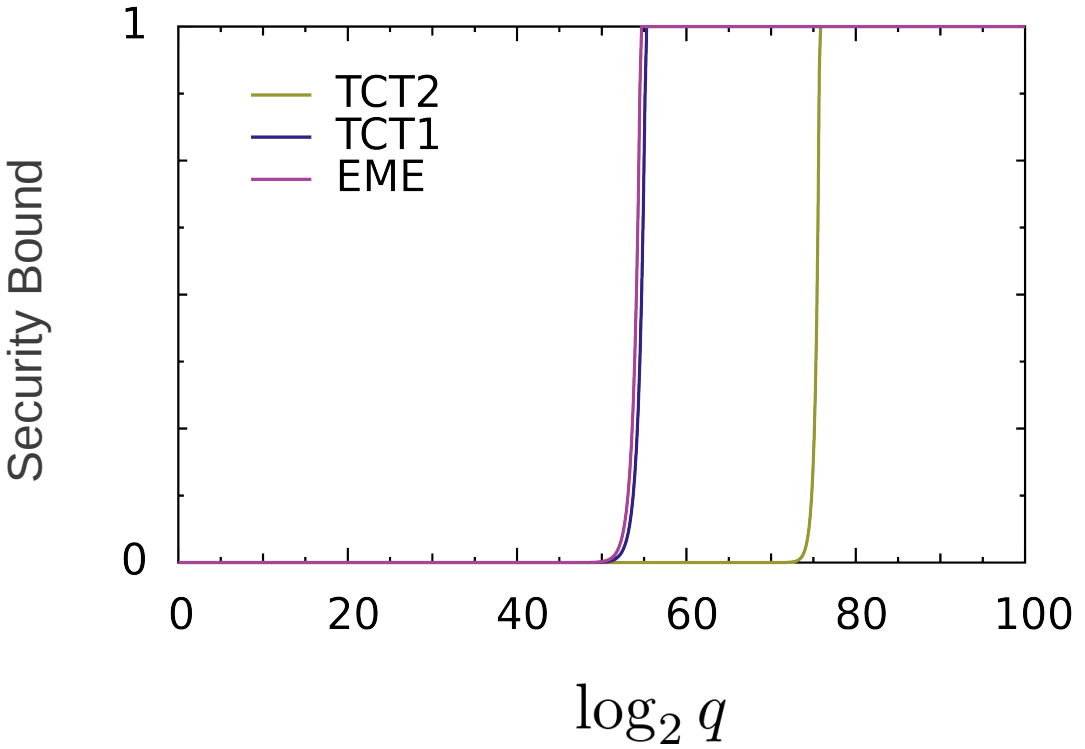
- Build an $N = 2n$-bit TBC out of an n-bit TBC [CDMS '10]
- Implement the n-bit TBC using CLRW2 [LST '12] over, e.g., AES
- Use NH [BHKKR '99] to extend the tweak length
- Secure to $O(2^{2n/3})$ queries
- The two F calls make a total:
  - 28 multiplies in GF_n
  - 12 n-bit blockcipher calls
- Potentially expensive for short inputs, fine for long ones

Comparison with other modes

Computational cost on* sn-*bit inputs

Mode	BC Calls	GF Multiplies	Ring Ops	Queries	Reference
EME*	2*s* + 3	---	---	$2^{n/2}$	Halevi '04; Halevi, Rogaway '03
HEH	*s* + 1	*s* + 2	---	$2^{n/2}$	Sarkar '07, '09
TCT1	*s* + 1	5	16*s*	$2^{n/2}$	
TCT2	2*s* + 8	32	32*s*	$\mathbf{2^{2n/3}}$	

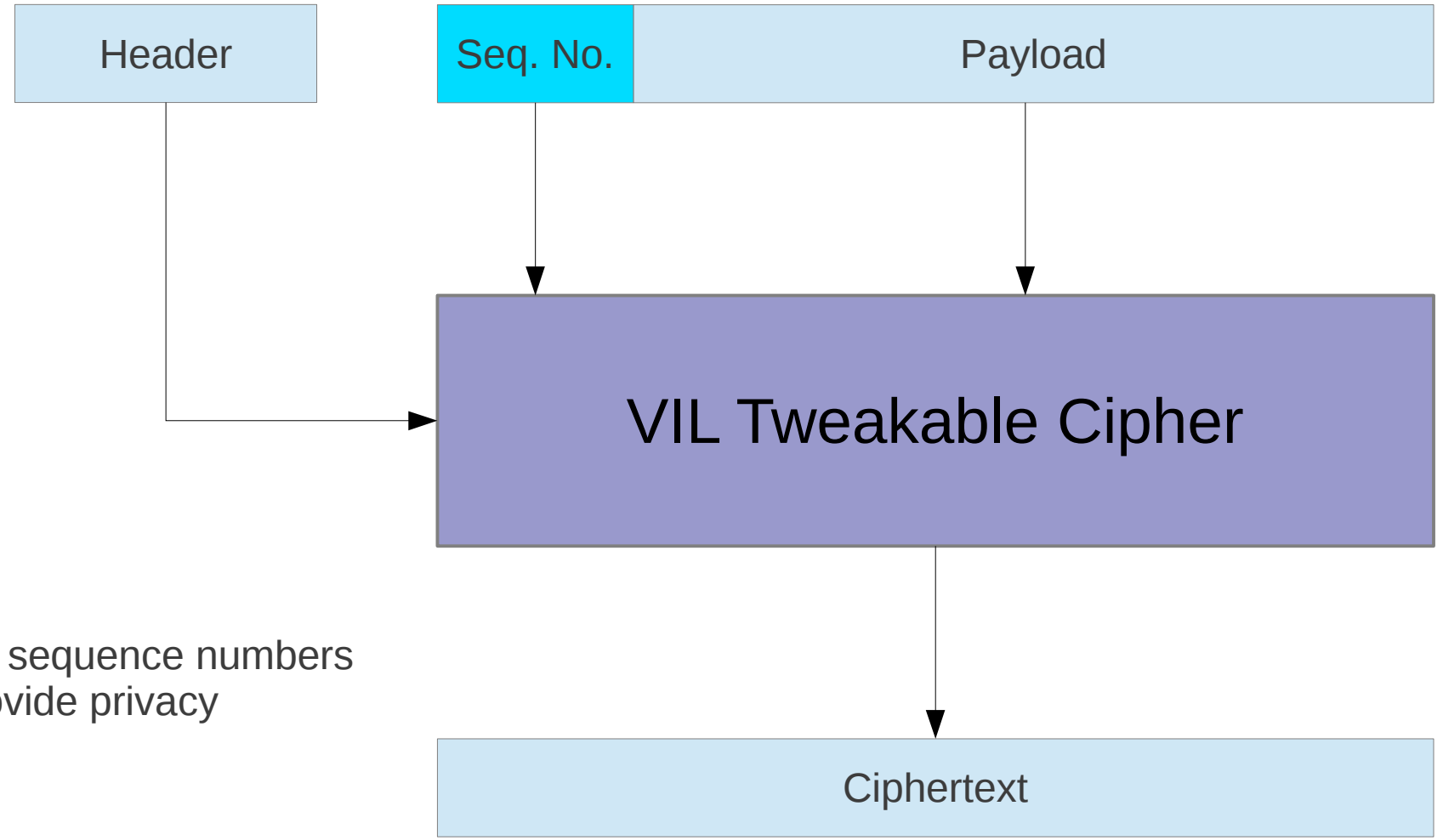
Typical: *s* = 256 (4KB sectors, AES)



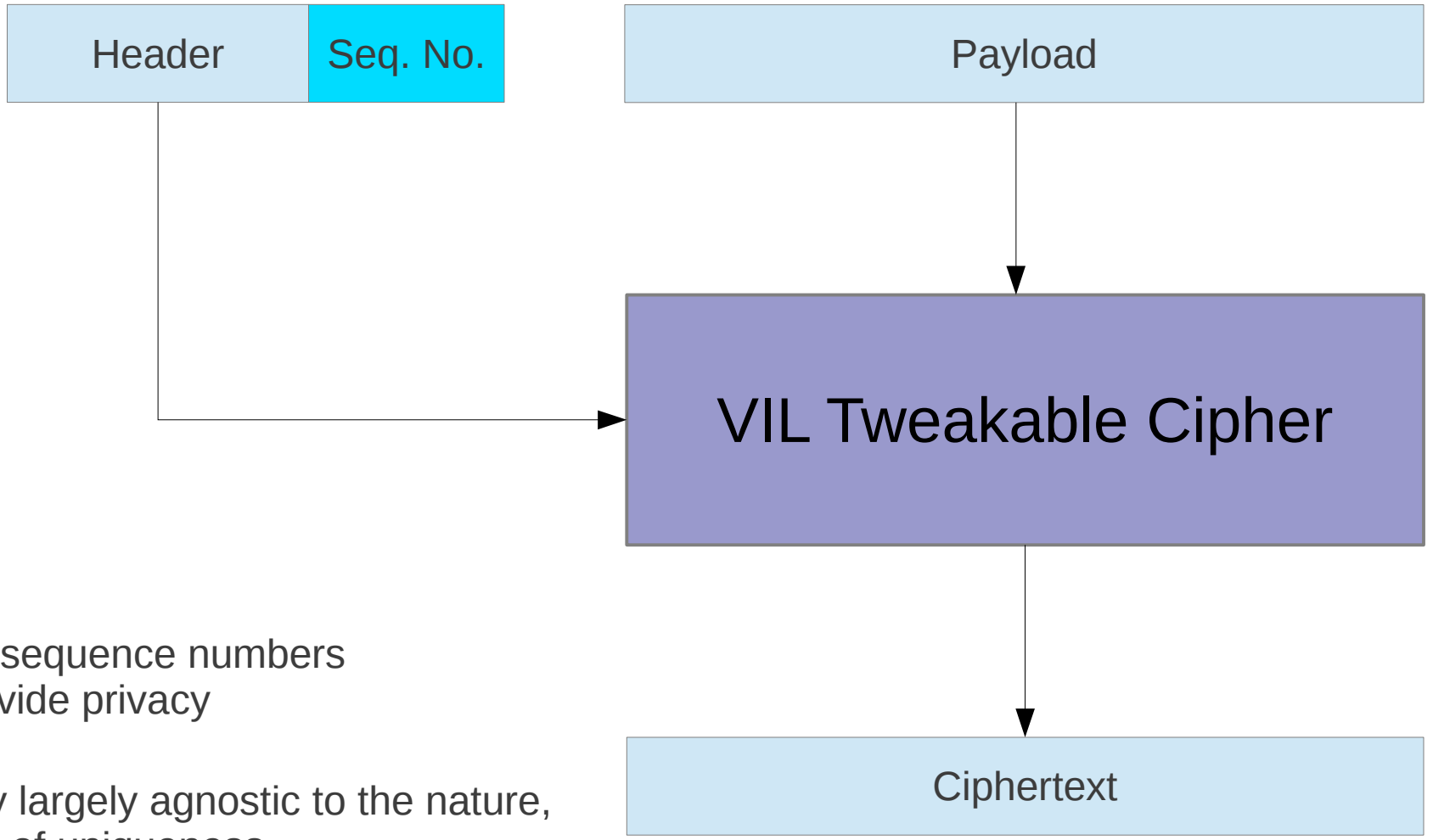
Results

PIV: A new approach to VIL TCs

AEAD from VIL TCs

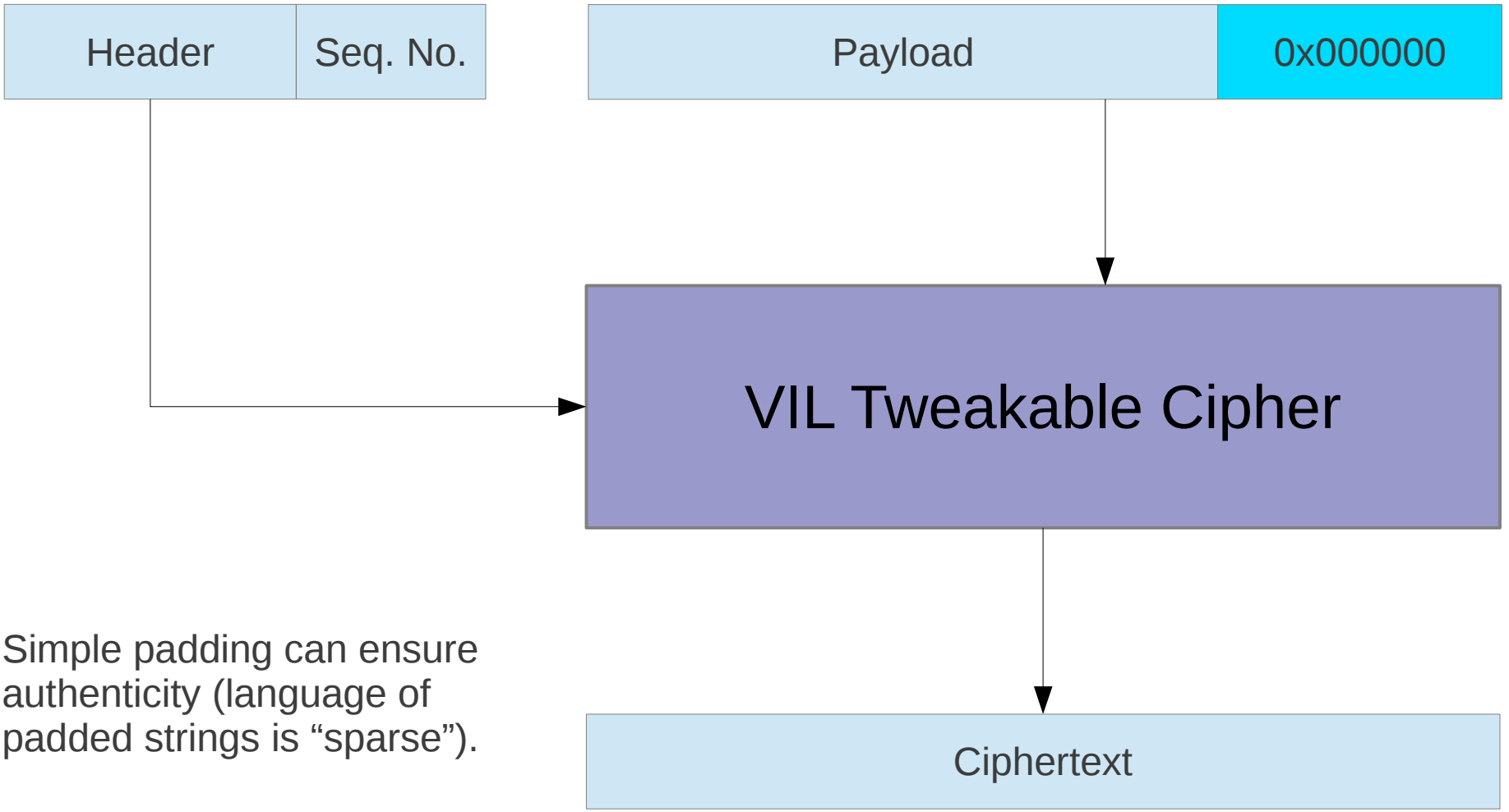


Unique sequence numbers
can provide privacy

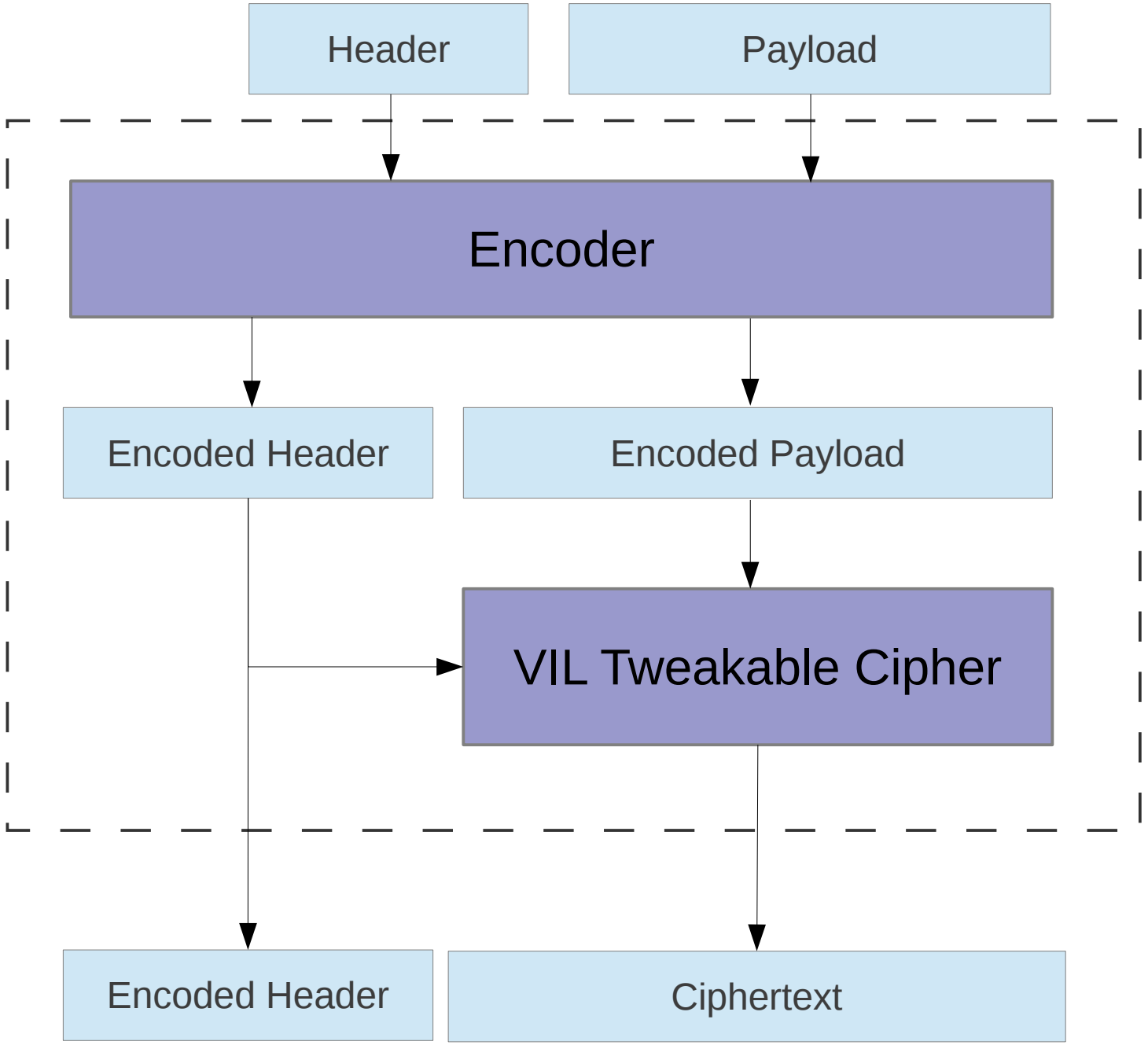


Unique sequence numbers
can provide privacy

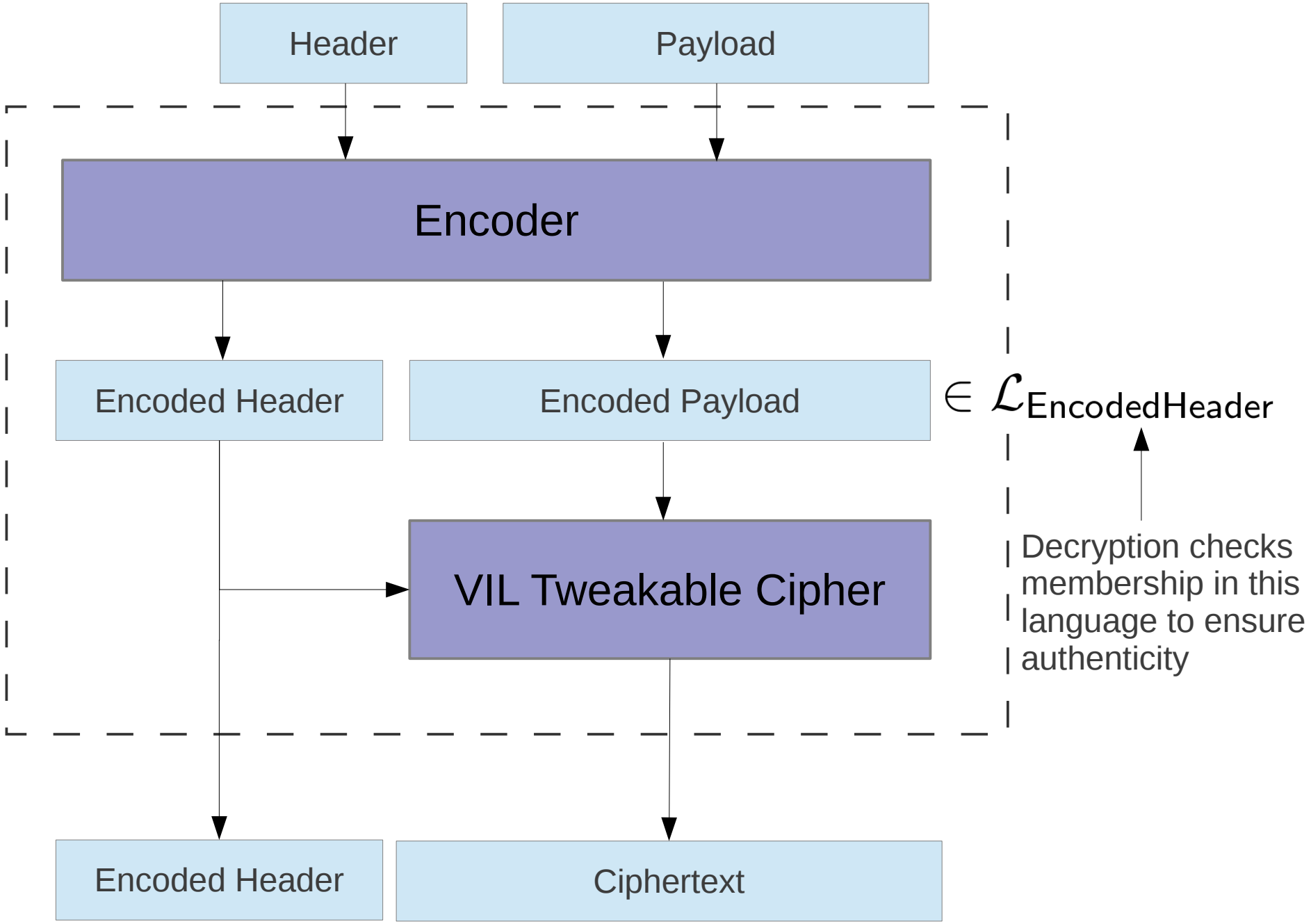
Security largely agnostic to the nature,
location of uniqueness



Simple padding can ensure authenticity (language of padded strings is “sparse”).



cf. “Encode then Encrypt” [Bellare and Rogaway '00]



cf. “Encode then Encrypt” [Bellare and Rogaway '00]

If Payload $\in \mathcal{L} \subseteq \{0,1\}^*$ and for all *n*,

$\frac{|\mathcal{L} \cap \{0,1\}^n|}{2^n} \leq 2^{-b}$, then we get *b* bits of authenticity.

- Payload may be mapped into $\mathcal{L}$ during an explicit encoding step (e.g., pad with 0x00..00)

If Payload $\in \mathcal{L} \subseteq \{0,1\}^*$ and for all *n*,

$\frac{|\mathcal{L} \cap \{0,1\}^n|}{2^n} \leq 2^{-b}$, then we get *b* bits of authenticity.

- Payload may be mapped into $\mathcal{L}$ during an explicit encoding step (e.g., pad with 0x00..00)
- Payload may already be in some “sparse” language (e.g., a protocol with human-readable fields, checksums)
  - No ciphertext stretch!

If Payload $\in \mathcal{L} \subseteq \{0,1\}^*$ and for all *n*,

$\frac{|\mathcal{L} \cap \{0,1\}^n|}{2^n} \leq 2^{-b}$, then we get *b* bits of authenticity.

- Payload may be mapped into $\mathcal{L}$ during an explicit encoding step (e.g., pad with 0x00..00)
- Payload may already be in some “sparse” language (e.g., a protocol with human-readable fields, checksums)
  - No ciphertext stretch!
- Remains secure even with multiple error messages
  - Errors can depend on encoded payload

If Payload $\in \mathcal{L} \subseteq \{0,1\}^*$ and for all *n*,

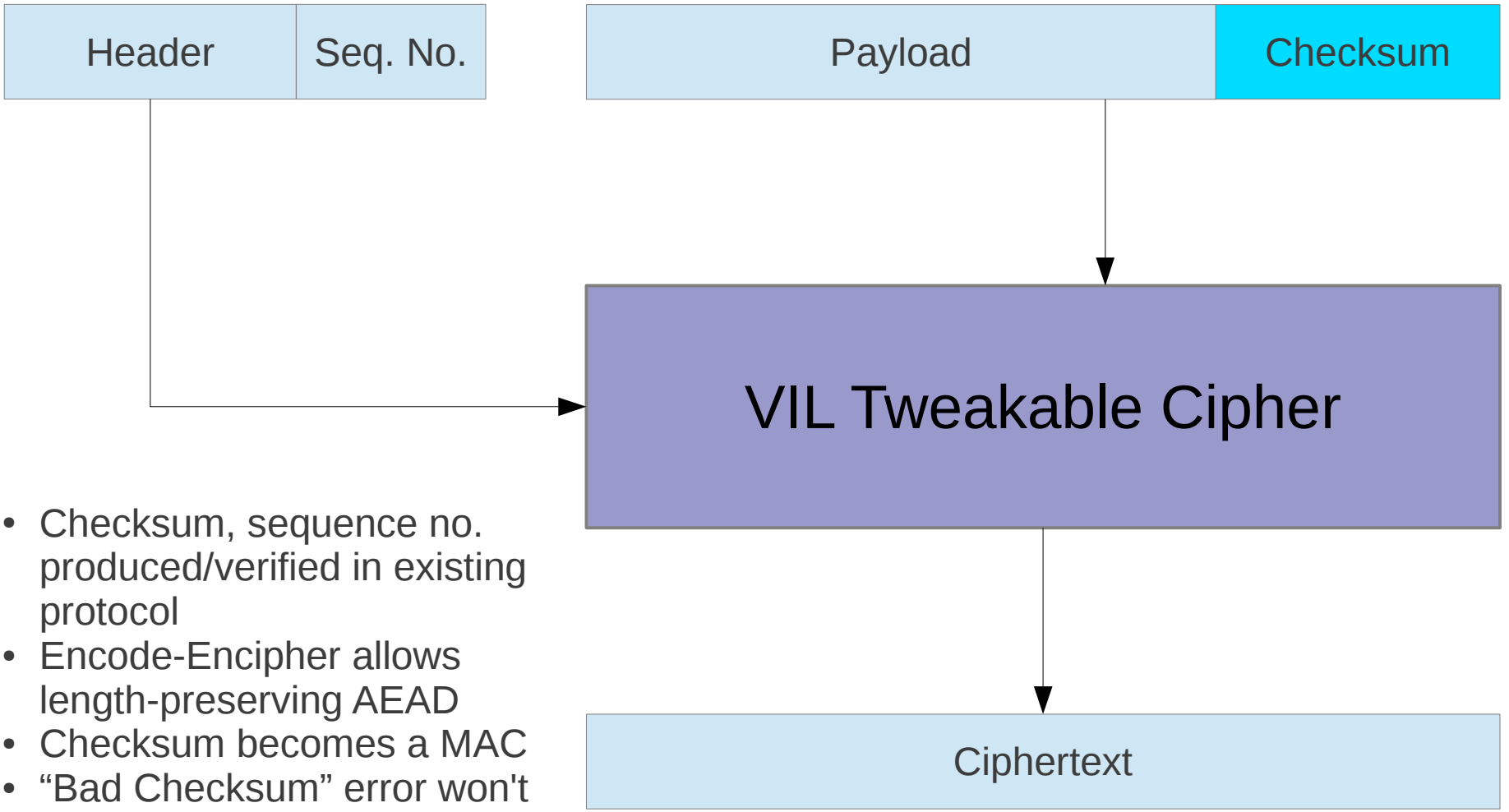
$\frac{|\mathcal{L} \cap \{0,1\}^n|}{2^n} \leq 2^{-b}$, then we get *b* bits of authenticity.

- Payload may be mapped into $\mathcal{L}$ during an explicit encoding step (e.g., pad with 0x00..00)
- Payload may already be in some “sparse” language (e.g., a protocol with human-readable fields, checksums)
  - No ciphertext stretch!
- Remains secure even with multiple error messages
  - Errors can depend on encoded payload
- Nonce-misuse resistance

If $\mathsf{Payload} \in \mathcal{L} \subseteq \{0,1\}^*$ and for all *n*,

$\frac{|\mathcal{L} \cap \{0,1\}^n|}{2^n} \leq 2^{-b}$, then we get *b* bits of authenticity.

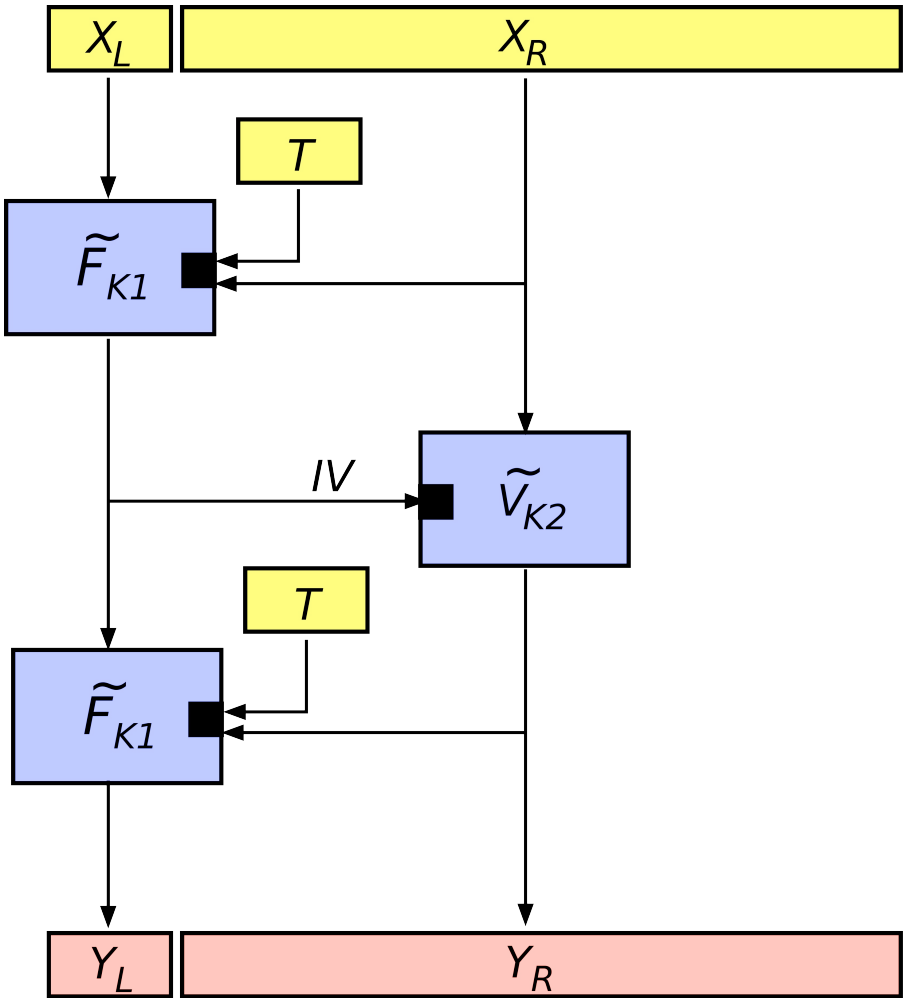
- Payload may be mapped into $\mathcal{L}$ during an explicit encoding step (e.g., pad with 0x00..00)
- Payload may already be in some “sparse” language (e.g., a protocol with human-readable fields, checksums)
  - No ciphertext stretch!
- Remains secure even with multiple error messages
  - Errors can depend on encoded payload
- Nonce-misuse resistance
- NM-CPA/IND-CCA not enough [AnBellare01]



- Checksum, sequence no. produced/verified in existing protocol
- Encode-Encipher allows length-preserving AEAD
- Checksum becomes a MAC
- “Bad Checksum” error won't leak info about original payload
- Possible use-case: low-power wireless networks

Wrapping up

- PIV: New VIL TC
  - Can beat b'day bound at little cost
- AEAD from a VIL TC
  - Privacy & authenticity from broad classes of encodings
  - Possibility of zero ciphertext stretch
  - Robust against multiple error messages



Questions?