PROJECT **STRONGBOX**

Capitalizing on Security, Performance, and Energy Tradeoffs in Full Drive/Disk Encryption Schemes for Fun and Profit

BERNARD DICKENS III **HARYADI S. GUNAWI** **DAVID CASH** **ARIEL J. FELDMAN** **HENRY HOFFMANN**

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

StrongBox II (April 2019)

StrongBox III (Q4 2019)

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

- ❖ We overturn the conventional wisdom regarding FDE: stream ciphers *are* suitable for FDE!
- ❖ We develop and implement a secure approach to FDE based on the ChaCha20 stream cipher, F2FS LFS, and RPMB eMMC trusted hardware

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

- ❖ We overturn the conventional wisdom regarding FDE: stream ciphers *are* suitable for FDE!
- ❖ We develop and implement a secure approach to FDE based on the ChaCha20 stream cipher, F2FS LFS, and RPMB eMMC trusted hardware

StrongBox II (April 2019)

- ❖ Building on the success of the StrongBox approach, we consider the wider tradeoff space between ciphers beyond ChaCha20 and the use cases they motivate

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

- ❖ We overturn the conventional wisdom regarding FDE: stream ciphers *are* suitable for FDE!
- ❖ We develop and implement a secure approach to FDE based on the ChaCha20 stream cipher, F2FS LFS, and RPMB eMMC trusted hardware

StrongBox II (April 2019)

- ❖ Building on the success of the StrongBox approach, we consider the wider tradeoff space between ciphers beyond ChaCha20 and the use cases they motivate

StrongBox III (Q4 2019)

- ❖ We present a new theoretical framework for FDE, define the cryptographic goals of StrongBox, and formally evaluate the security properties of our construction with respect to these goals

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

- ❖ We overturn the conventional wisdom regarding FDE: stream ciphers *are* suitable for FDE!
- ❖ We develop and implement a secure approach to FDE based on the ChaCha20 stream cipher, F2FS LFS, and RPMB eMMC trusted hardware

StrongBox II (April 2019)

- ❖ Building on the success of the StrongBox approach, we consider the wider tradeoff space between ciphers beyond ChaCha20 and the use cases they motivate

StrongBox III (Q4 2019)

- ❖ We present a new theoretical framework for FDE, define the cryptographic goals of StrongBox, and formally evaluate the security properties of our construction with respect to these goals

FULL DRIVE (DISK) ENCRYPTION?

FULL DRIVE (DISK) ENCRYPTION?



Industry standard for FDE:

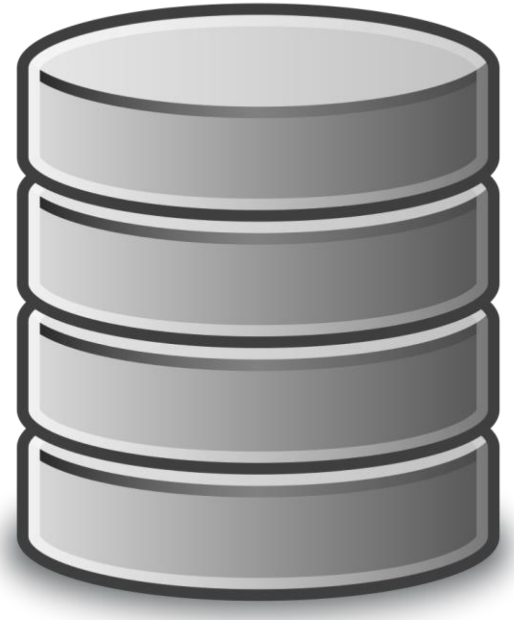
AES-XTS

AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher

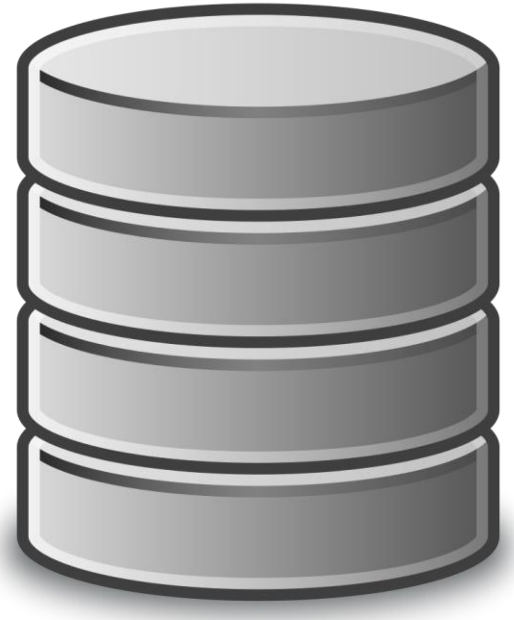
AES BLOCK CIPHER IN XTS MODE

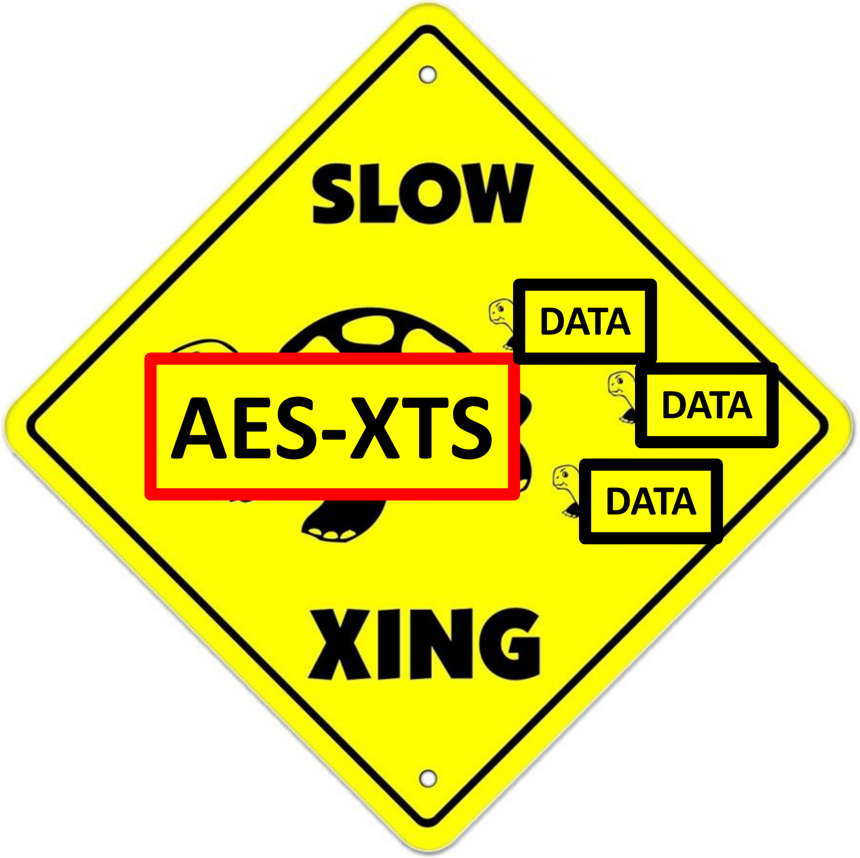
- AES is a block cipher
- Inspired by traditional spinning disk storage



AES BLOCK CIPHER IN XTS MODE

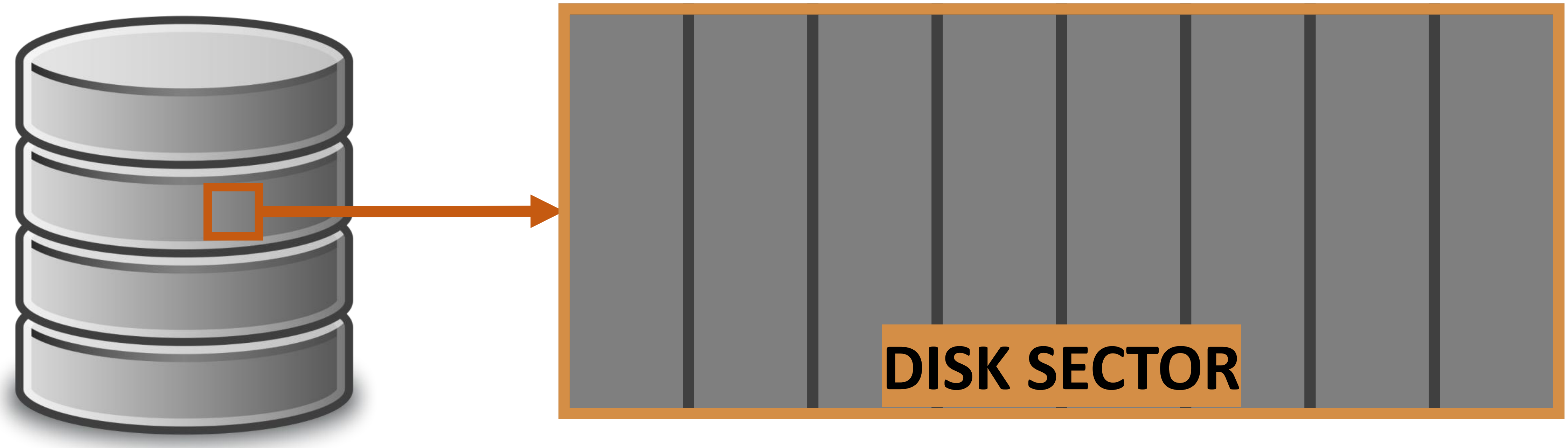
- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**





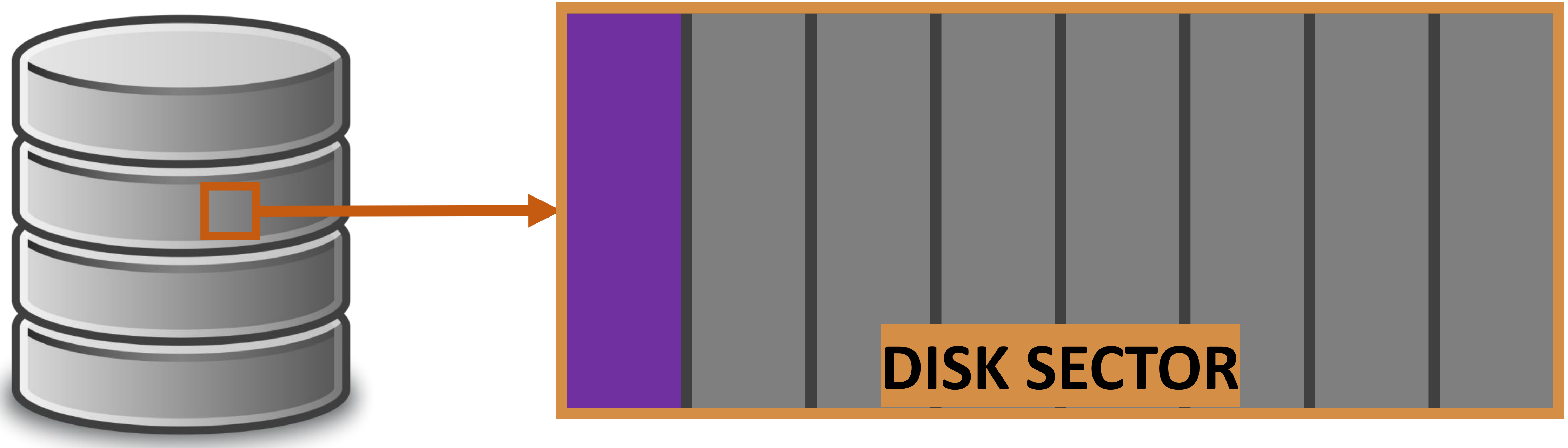
AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**
- Cause: narrow AES “blocks,” wide disk sectors



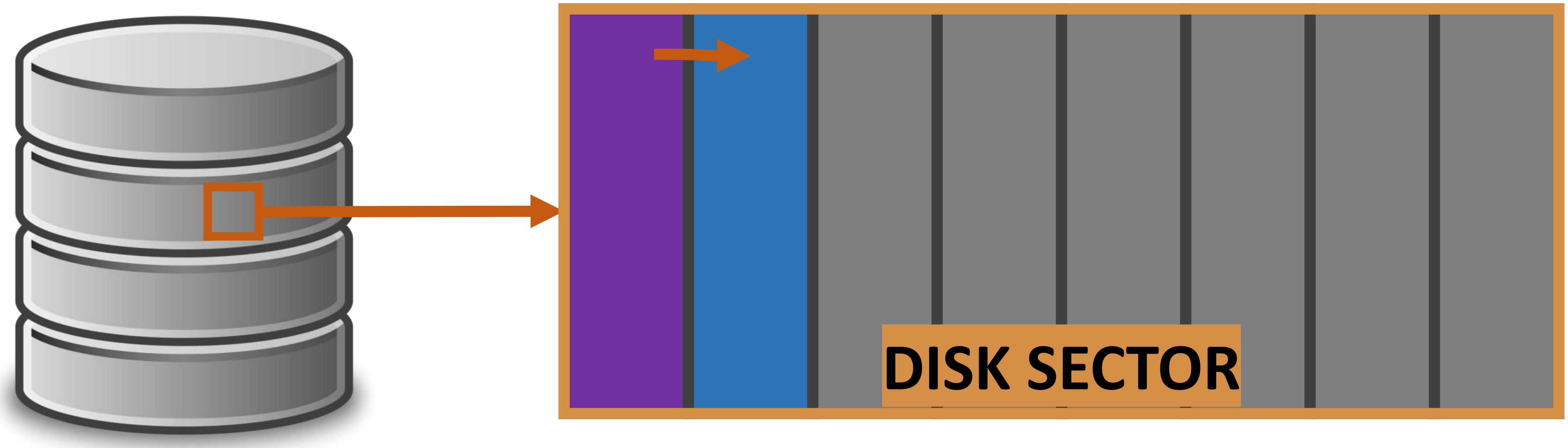
AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**
- Cause: narrow AES "blocks," wide disk sectors



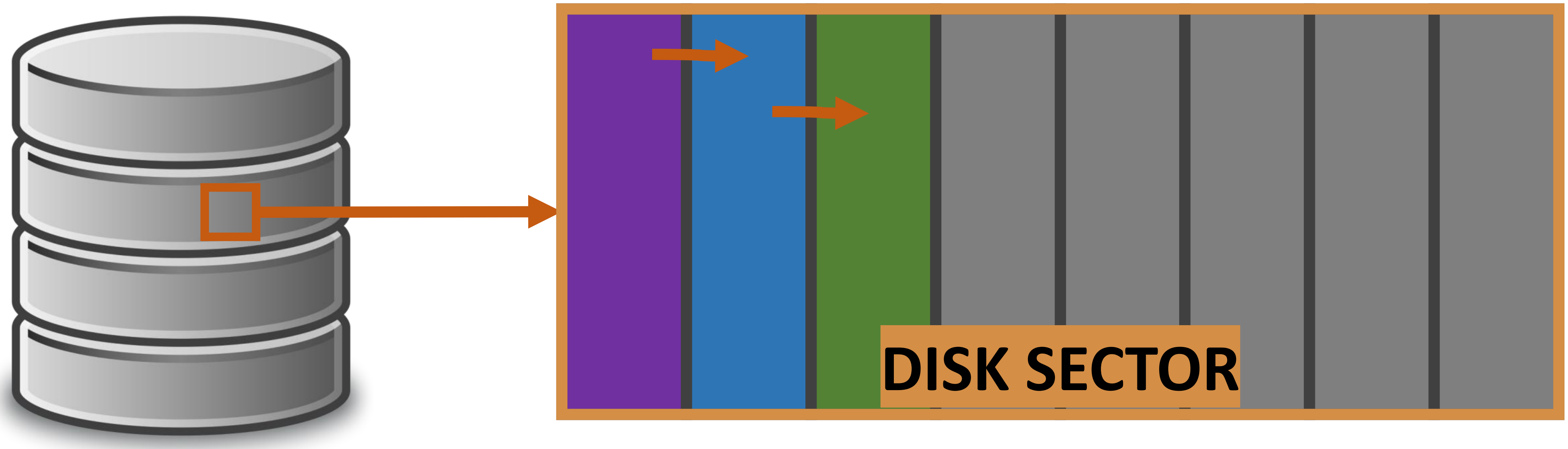
AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**
- Cause: narrow AES “blocks,” wide disk sectors



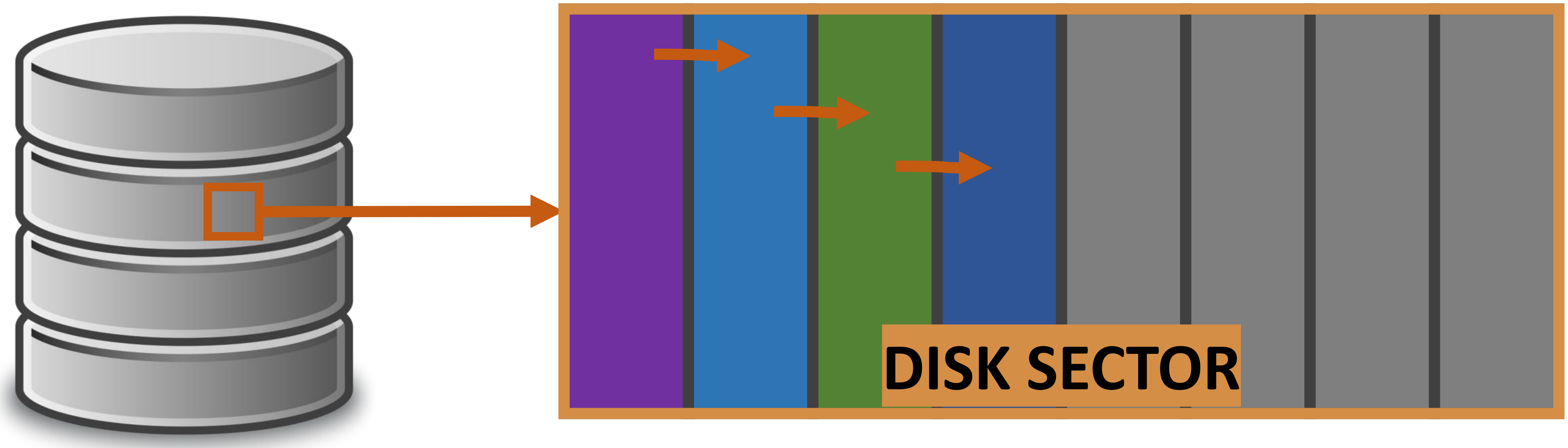
AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**
- Cause: narrow AES “blocks,” wide disk sectors



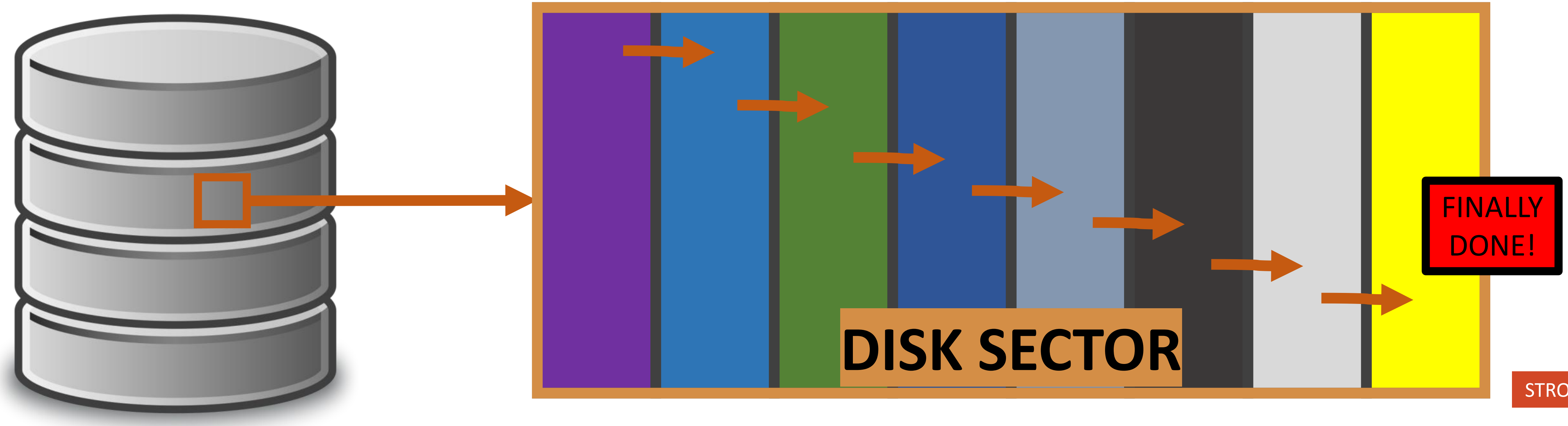
AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**
- Cause: narrow AES “blocks,” wide disk sectors



AES BLOCK CIPHER IN XTS MODE

- AES is a block cipher
- Inspired by traditional spinning disk storage
- Problem: AES-XTS is **SLOW**
- Cause: narrow AES "blocks," wide disk sectors

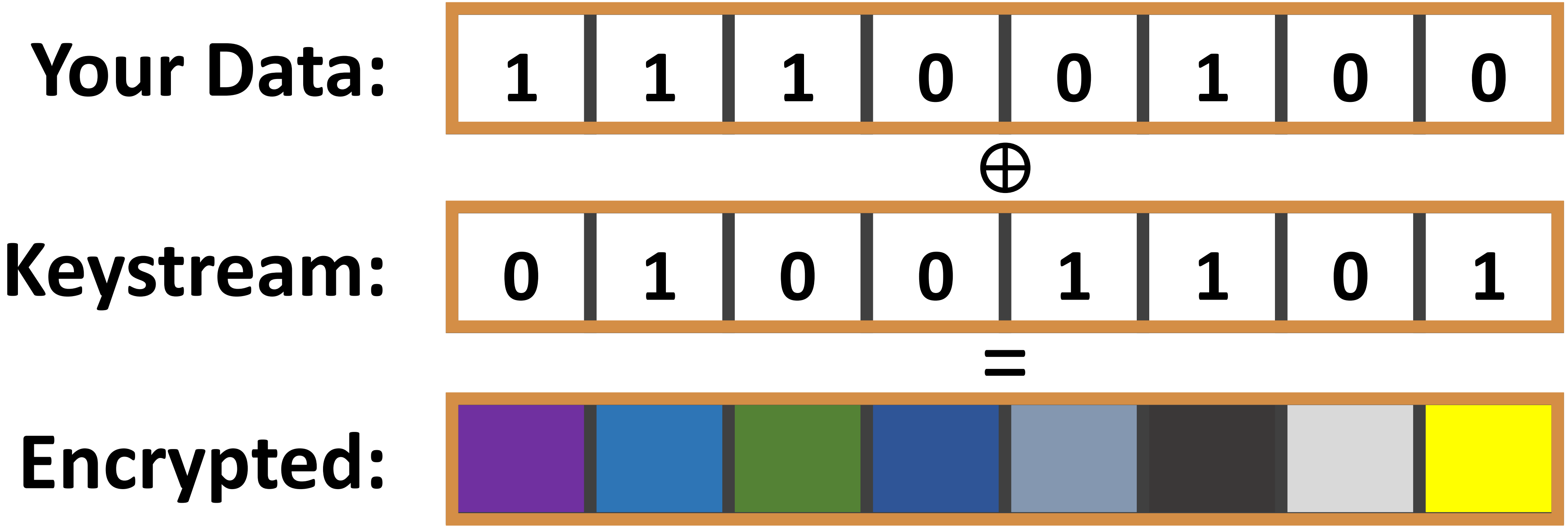


BLOCKS VERSUS STREAMS

- Stream ciphers are also a thing

BLOCKS VERSUS STREAMS

- Stream ciphers are also a thing
- Data encrypted via XOR as a continuous stream

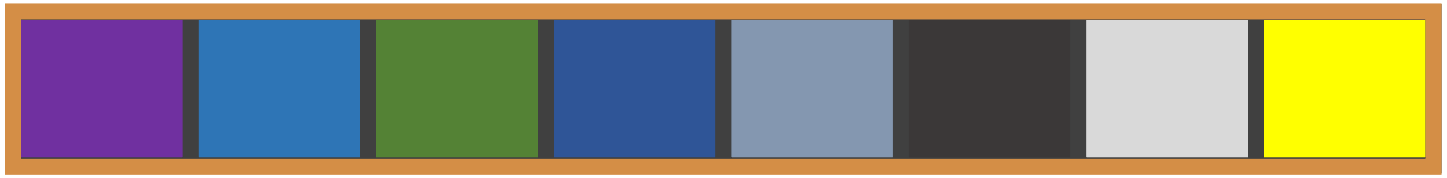


BLOCKS VERSUS STREAMS

- Stream ciphers are also a thing
- Data encrypted via XOR as a continuous stream
- Stream ciphers are **FAST**
- Cause: XOR (ARX) operation is simple, fast

Your Data:	1	1	1	0	0	1	0	0
				$\oplus$				
Keystream:	0	1	0	0	1	1	0	1
				=				

Encrypted:



NOT THE ONLY CIPHER IN TOWN

◊ There are *many* stream ciphers with various characteristics

NOT THE ONLY CIPHER IN TOWN

- There are *many* stream ciphers with various characteristics
  - High throughput ciphers
    - Sosemanuk and Rabbit

NOT THE ONLY CIPHER IN TOWN

- There are *many* stream ciphers with various characteristics
  - High throughput ciphers
    - Sosemanuk and Rabbit
  - Software/energy efficient ciphers
    - ChaCha and Salsa

NOT THE ONLY CIPHER IN TOWN

- There are *many* stream ciphers with various characteristics
  - High throughput ciphers
    - Sosemanuk and Rabbit
  - Software/energy efficient ciphers
    - ChaCha and Salsa
  - Tweakable overwrite-resistant ciphers
    - Freestyle

NOT THE ONLY CIPHER IN TOWN

- There are *many* stream ciphers with various characteristics
  - High throughput ciphers
    - Sosemanuk and Rabbit
  - Software/energy efficient ciphers
    - ChaCha and Salsa
  - Tweakable overwrite-resistant ciphers
    - Freestyle
  - Hardware-accelerated and other interesting ciphers
    - Reduced-round ChaCha (e.g. ARM Neon impl)
    - CTR mode block ciphers (e.g. AES-CTR)

SWITCH CIPHERS, PROBLEM SOLVED?

BIG PROBLEMS WITH STREAM CIPHERS

There are two big problems with using stream ciphers for FDE:

BIG PROBLEMS WITH STREAM CIPHERS

There are two big problems with using stream ciphers for FDE:

1. Data confidentiality can be trivially violated with overwrites

BIG PROBLEMS WITH STREAM CIPHERS

There are two big problems with using stream ciphers for FDE:

1. Data confidentiality can be trivially violated with overwrites

2. Rolling back the system to a previously valid state also leads to confidentiality violations

OVERWRITES ARE A BIG DEAL

◊ Why are overwrites such a problem?

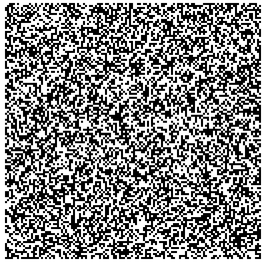
1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

◊ Why are overwrites such a problem?

Your Data:

Keystream:



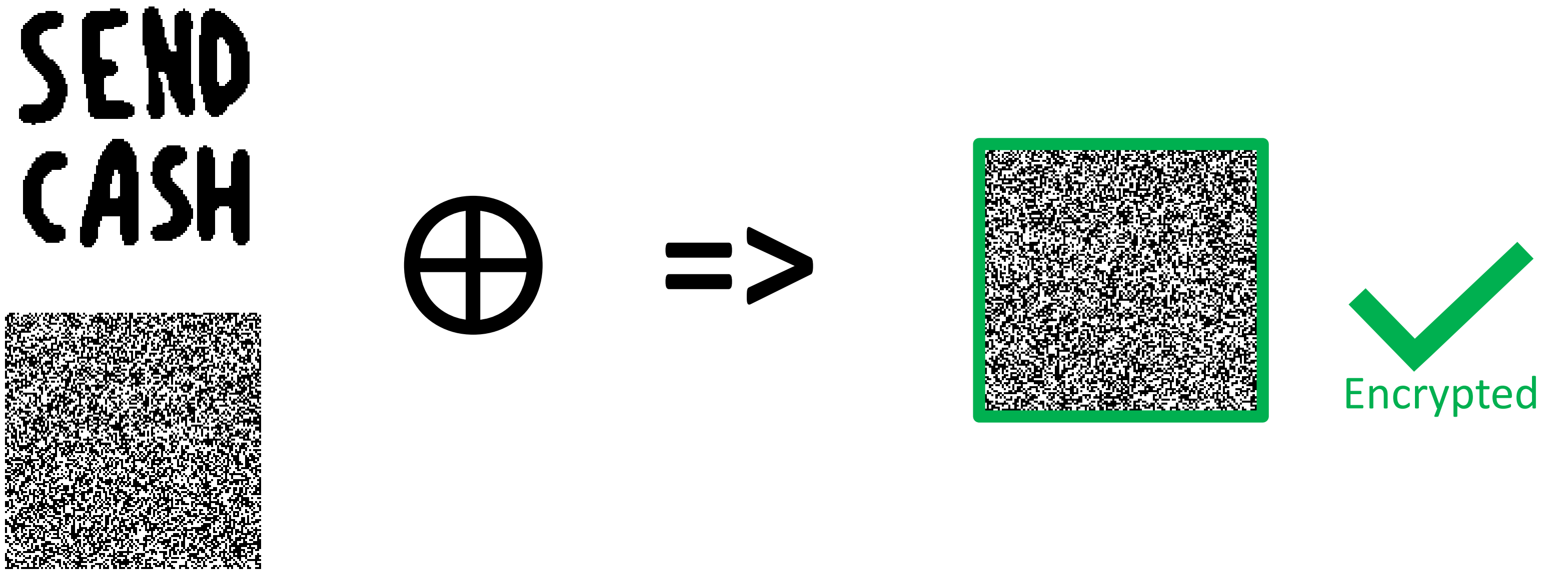
1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

- Why are overwrites such a problem?

Your Data:

Keystream:



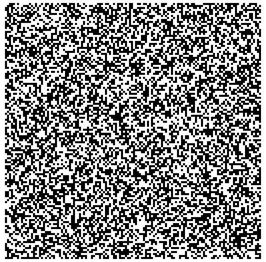
1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

◊ Why are overwrites such a problem?

Your Data:

(the same)
Keystream:



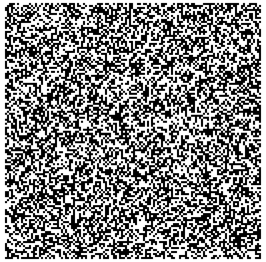
1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

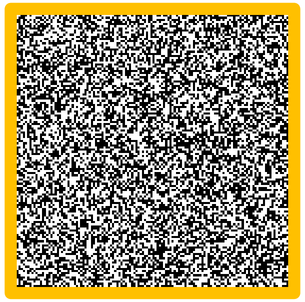
◊ Why are overwrites such a problem?

Your Data:

(the same)
Keystream:



$\oplus$ =>



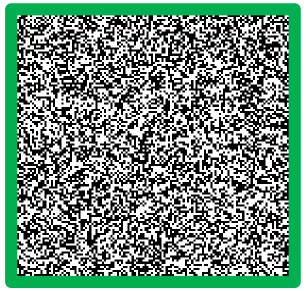
Encrypted?

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

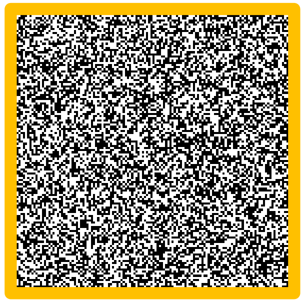
◊ Why are overwrites such a problem?

Original Write:



(i.e. same keystream)

Overwrite:

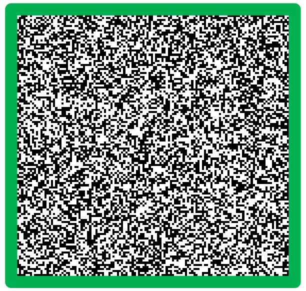


1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

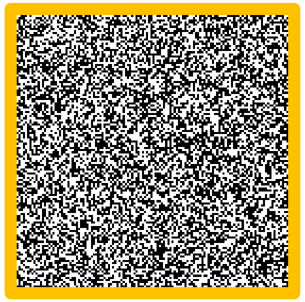
- Why are overwrites such a problem?

Original Write:



(i.e. same keystream)

Overwrite:



$\oplus$ => 

ENCRYPTION BROKEN!

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

- ◊ Why are overwrites such a problem?
- ◊ The solution: “rekey” the keystream on every overwrite

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

- ◊ Why are overwrites such a problem?
- ◊ The solution: "rekey" the keystream on every overwrite
- ◊ Another problem: rekeying is expensive!

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

OVERWRITES ARE A BIG DEAL

- Why are overwrites such a problem?
- The solution: "rekey" the keystream on every overwrite
- Another problem: rekeying is expensive!
  - Out of ~17,000 write ops, EXT4 makes ~10,000 overwrites
  - 10,000 **x** expensive operation **=** performance degradation
  - **This expense negates perf benefit of the stream cipher**

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

ROLLBACKS ARE COMPROMISING

◊ Rollbacks are even worse!

1. Overwrites violate confidentiality
2. **Rollbacks lead to overwrites**

ROLLBACKS ARE COMPROMISING

- ◊ Rollbacks are even worse!
- ◊ Take a snapshot of the system at valid state

1. Overwrites violate confidentiality
2. **Rollbacks lead to overwrites**

ROLLBACKS ARE COMPROMISING

- ◊ Rollbacks are even worse!
- ◊ Take a snapshot of the system at valid state
- ◊ Rollback the system to a previous valid state

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**

ROLLBACKS ARE COMPROMISING

- ◊ Rollbacks are even worse!
- ◊ Take a snapshot of the system at valid state
- ◊ Rollback the system to a previous valid state
- ◊ Allow writes as normal

1. Overwrites violate confidentiality
2. **Rollbacks lead to overwrites**

ROLLBACKS ARE COMPROMISING

- Rollbacks are even worse!
- Take a snapshot of the system at valid state
- Rollback the system to a previous valid state
- Allow writes as normal
- Contrast (XOR) encrypted snapshot with current "encrypted" state
  - **SAME AS AN OVERWRITE!**

1. **Overwrites violate confidentiality**
2. **Rollbacks lead to overwrites**


No!
You can only use
a block cipher for
Full Drive
Encryption!

THE OLD ASSUMPTION

- ◊ Stream ciphers are faster…
- ◊ … but the overhead of dealing with overwrites and rollbacks negates any potential perf benefit

- ◊ Hence, AES-XTS is the industry standard

THE OLD ASSUMPTION: INVALIDATED

Key insight: two technology trends invalidate old assumption

THE OLD ASSUMPTION: INVALIDATED

Key insight: two technology trends invalidate old assumption

1. **Non-volatile flash memory as secondary storage**
   - ◊ Governed by Flash Translation Layer (FTL)
   - ◊ FTL tends to avoid overwriting same cells (wear-leveling)

THE OLD ASSUMPTION: INVALIDATED

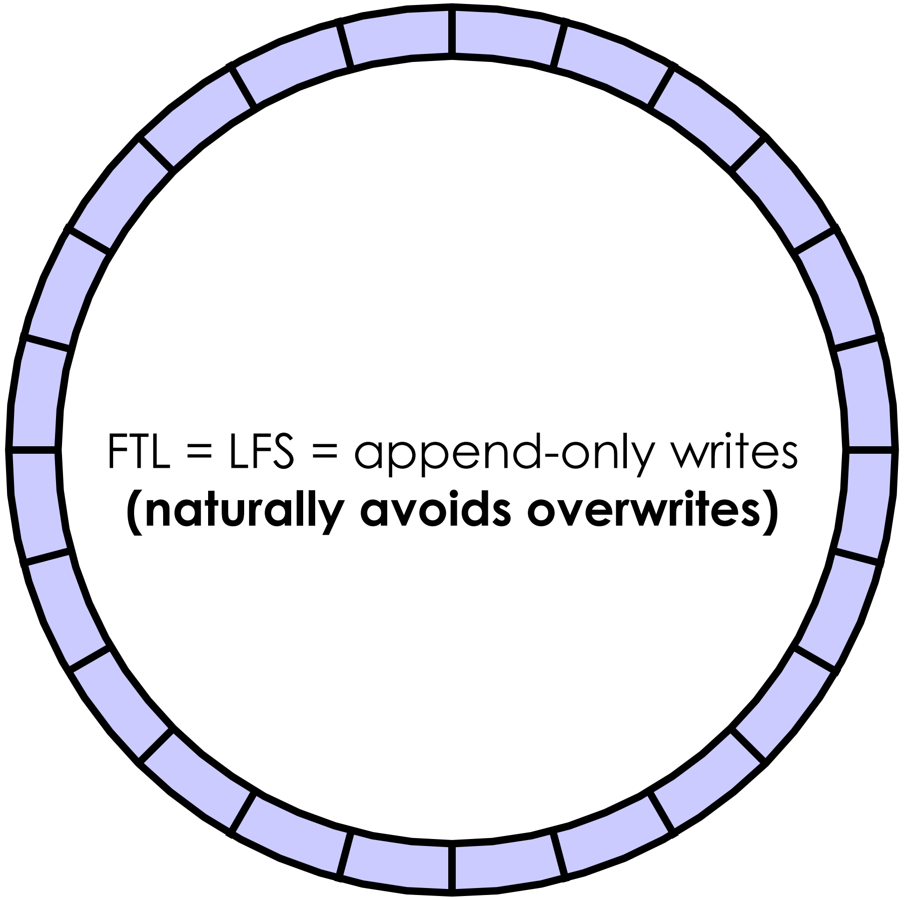
Key insight: two technology trends invalidate old assumption

1. **Non-volatile flash memory as secondary storage**
   - ◊ Governed by Flash Translation Layer (FTL)
   - ◊ FTL tends to avoid overwriting same cells (wear-leveling)

2. **Proliferation of secure hardware**
   - ◊ TPM/TEE implemented by nearly all computer and mobile device manufacturers
   - ◊ Secure hardware allows us to actively version system state and detect rollbacks

FTL <=> LFS

FTL operates like a Log-structured File System (LFS)



LFS MAKES FEW OVERWRITES

File System	Total Write Ops	Overwrites
ext4	16,756	10,787
LogFS	4,244	32
NILFS	4,199	24
F2FS	2.107	2

ACTIVELY VERSIONING SYSTEM STATE

- TPM/TEE allows us to actively version system state

ACTIVELY VERSIONING SYSTEM STATE

- TPM/TEE allows us to actively version system state
- Accomplished with fast secure monotonic counters

ACTIVELY VERSIONING SYSTEM STATE

- TPM/TEE allows us to actively version system state
- Accomplished with fast secure monotonic counters
- These hardware counters cannot be rolled back!

THE GRAND SOLUTION

- Leveraging the trends, we proposed **StrongBox**

- A drop-in replacement for AES-backed FDE providers such as dm-crypt
- Uses a stream cipher instead of the AES-XTS block cipher
- Uses an LFS to avoid overwrites and rekeying penalties
- Uses secure monotonic counters to prevent rollbacks

THREAT MODEL

Confidentiality

- ◊ dm-crypt (AES-XTS): AES block cipher
- ◊ StrongBox: ChaCha20 stream cipher

THREAT MODEL

Confidentiality

- dm-crypt (AES-XTS): AES block cipher
- StrongBox: ChaCha20 stream cipher

Overwrites (many-time pad)

- dm-crypt (AES-XTS): N/A
- StrongBox: append-only write property of FTL/LFS

THREAT MODEL (CONTINUED)

Rollbacks

- ◊ dm-crypt (AES-XTS): N/A
- ◊ StrongBox: secure monotonic counters

THREAT MODEL (CONTINUED)

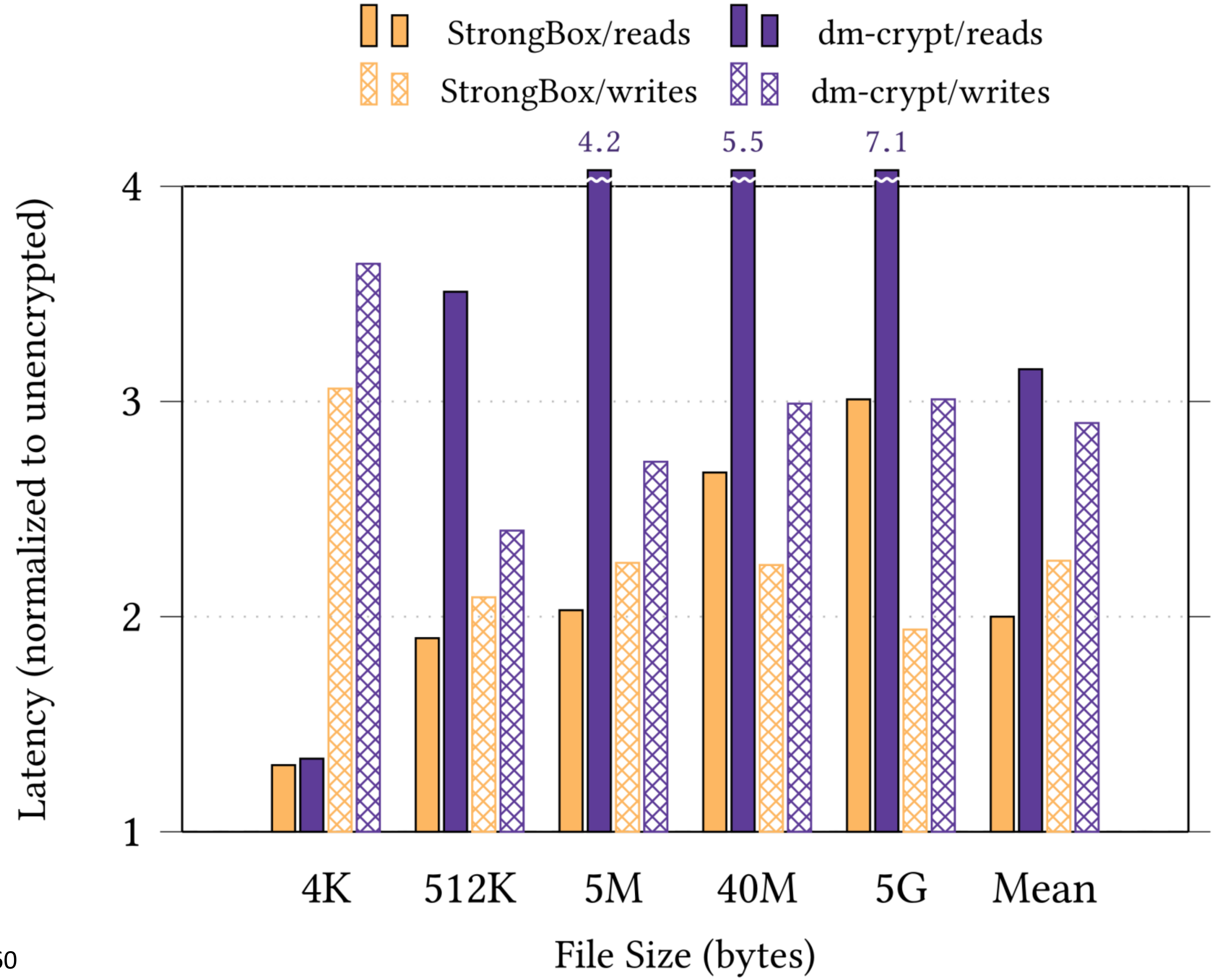
Rollbacks

- ◊ dm-crypt (AES-XTS): N/A
- ◊ StrongBox: secure monotonic counters

Integrity

- ◊ dm-crypt (AES-XTS): hopefully the user is paying attention
- ◊ StrongBox: hashing scheme to tie system state to counter

PERFORMANCE WIN



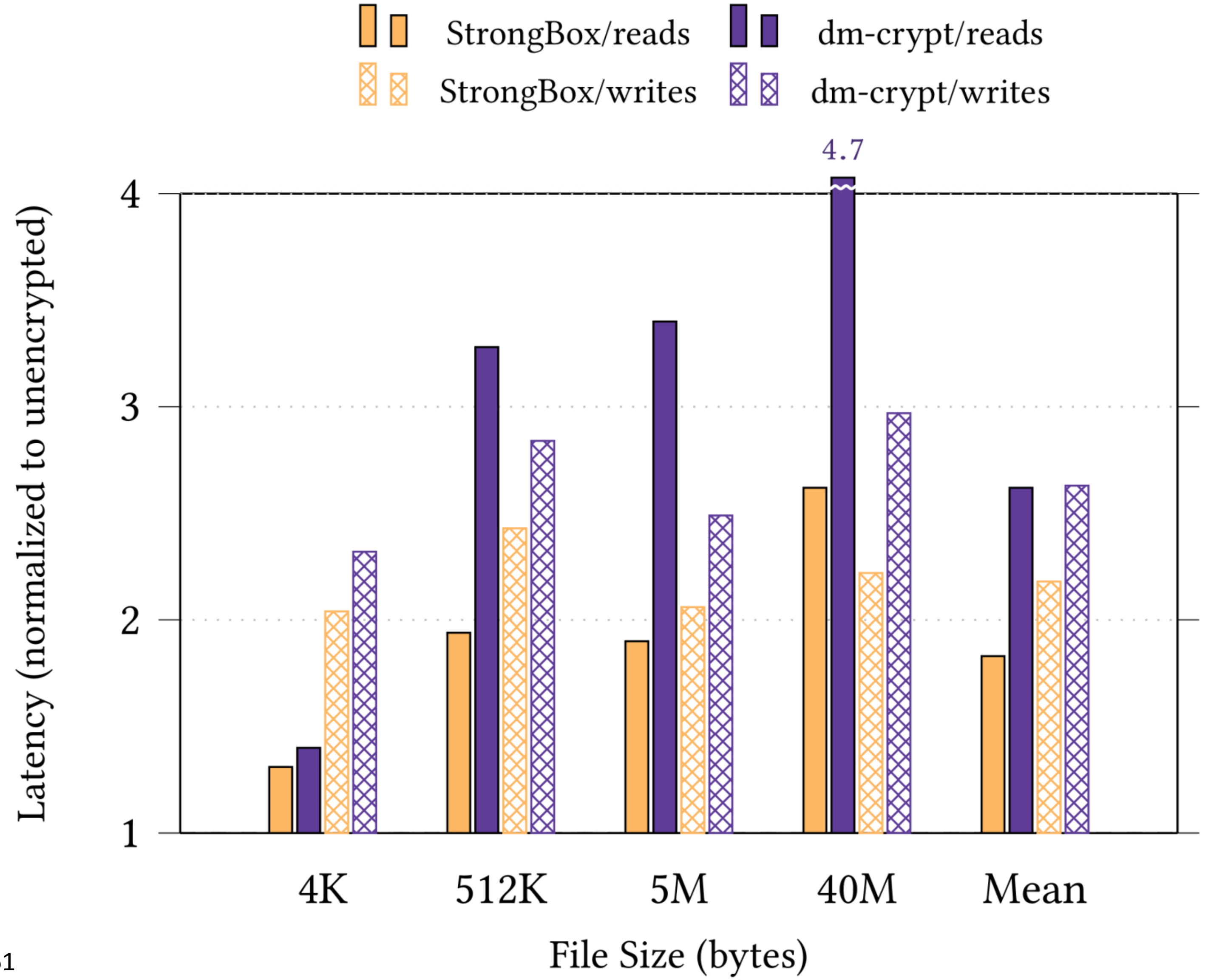
By harmonic mean:

- ◊ **1.7x** faster reads
- ◊ **1.3x** faster writes

Experimental setup:

- ◊ Odroid XU3 ARM big.LITTLE system
- ◊ eMMC 5.0HS400 backing store

PERFORMANCE WIN (RANDOM IO)



By harmonic mean:

- **1.7x** faster reads
- **1.3x** faster writes

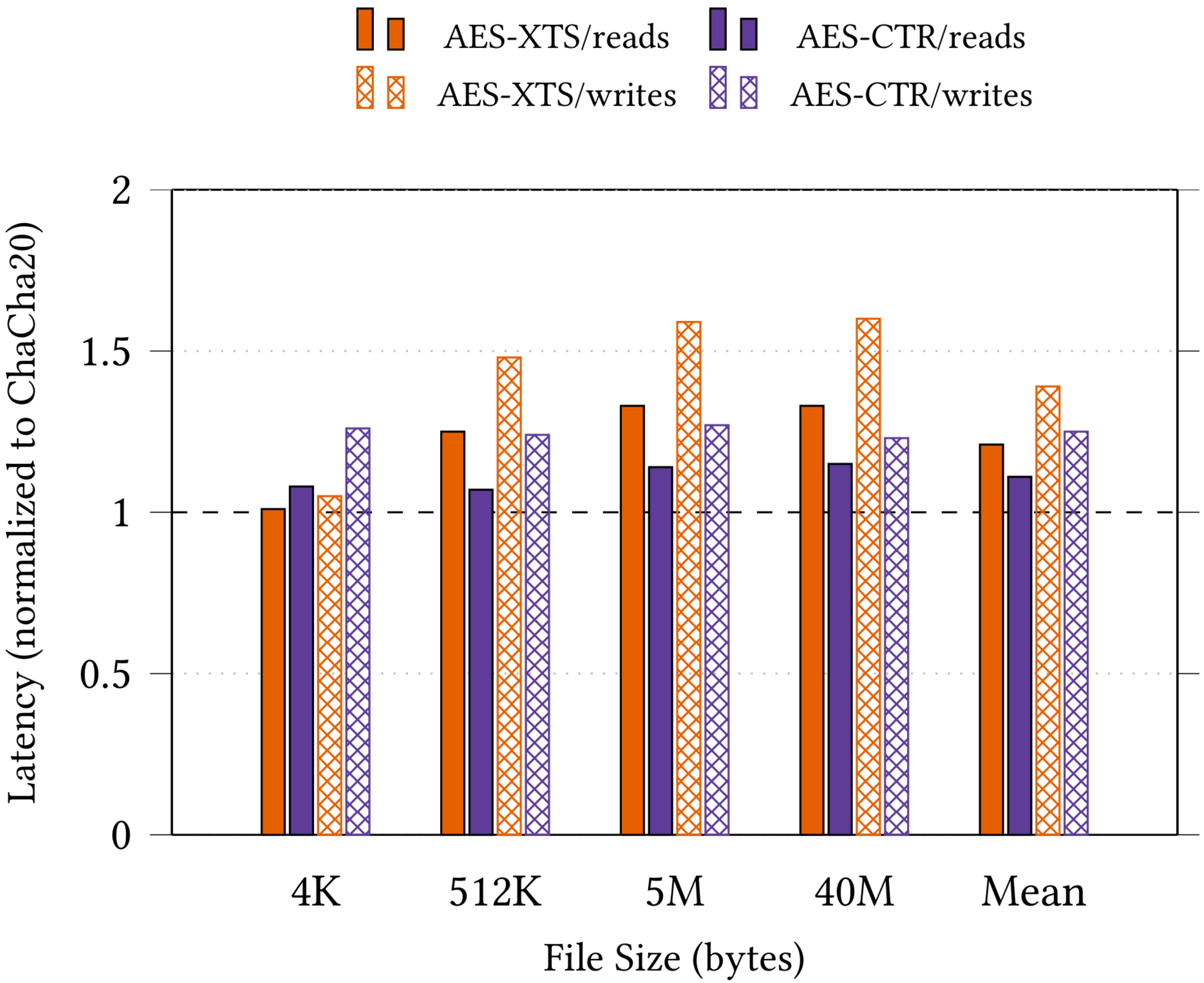
Experimental setup:

- Odroid XU3 ARM big.LITTLE system
- eMMC 5.0HS400 backing store

HARDWARE ACCELERATED AES?

What about hardware support for AES (i.e. AES-NI?)

SWITCHING CIPHERS: AES-CTR



USING OTHER STREAM CIPHERS?

What can we gain with ciphers other than ChaCha20?

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

- We overturn the conventional wisdom regarding FDE: stream ciphers *are* suitable for FDE!
- We develop and implement a secure approach to FDE based on the ChaCha20 stream cipher, F2FS LFS, and RPMB eMMC trusted hardware

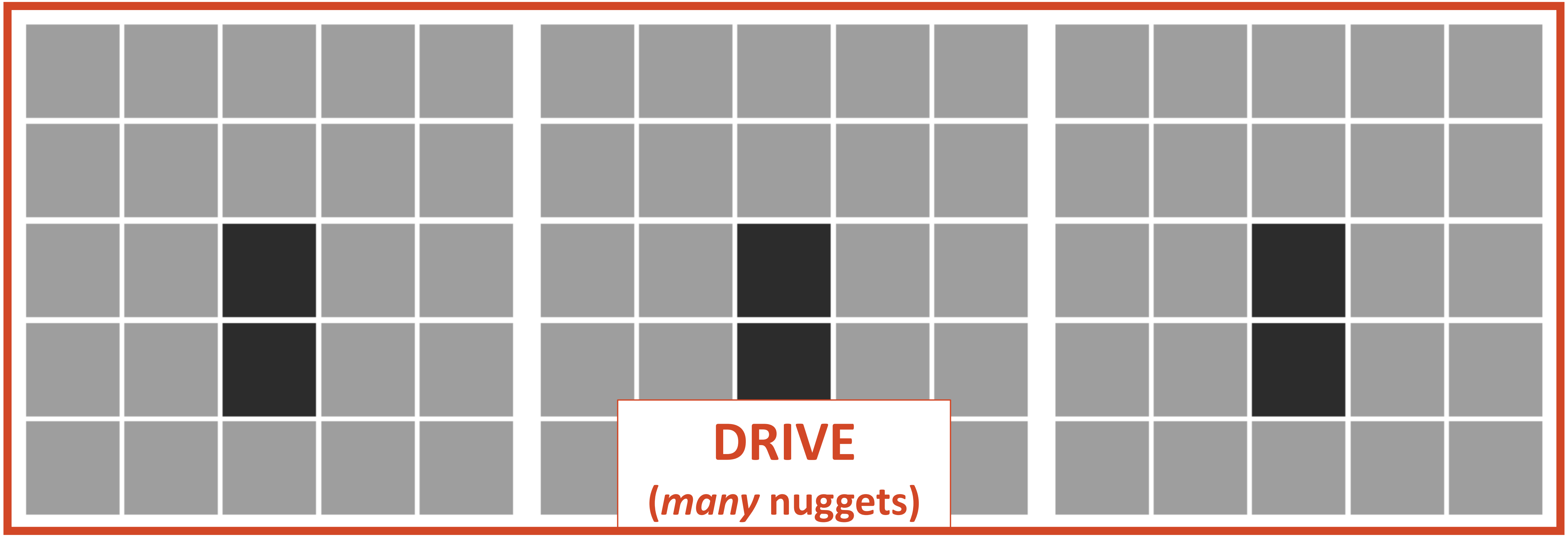
StrongBox II (April 2019)

- Building on the success of the StrongBox approach, we consider the wider tradeoff space between ciphers beyond ChaCha20 and the use cases they motivate

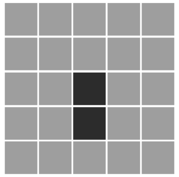
StrongBox III (Q4 2019)

- We present a new theoretical framework for FDE, define the cryptographic goals of StrongBox, and formally evaluate the security properties of our construction with respect to these goals

STRONGBOX INTERNAL DESIGN



 <= Flake

 <= Nugget (made of many flakes)

- ◊ Fine-grain flakes track overwrites
- ◊ Coarse-grain nuggets form logical storage units (like blocks) of many flakes
- ◊ A drive is partitioned into many nuggets

BUILT FOR CIPHER SWITCHING

- StrongBox I design methodology (flakes and nuggets) naturally lends itself to switching ciphers both offline *and online*

BUILT FOR CIPHER SWITCHING

- ◊ StrongBox I design methodology (flakes and nuggets) naturally lends itself to switching ciphers both offline *and online*
- ◊ Encryption and decryption of nuggets occur *independently of one another*

BUILT FOR CIPHER SWITCHING

- ◊ StrongBox I design methodology (flakes and nuggets) naturally lends itself to switching ciphers both offline *and online*
- ◊ Encryption and decryption of nuggets occur *independently of one another*
- ◊ We can leverage this compartmentalized approach and a consistent interface to securely crypt *different* nuggets with *any (valid) cipher*

CIPHERS BEYOND CHACHA20

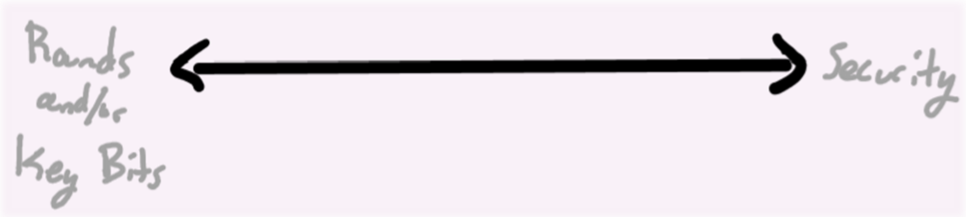
- Different ciphers behave differently under StrongBox
  - Various security guarantees offered
  - Various performance and power profiles for a given workload
  - Various storage requirements per block beyond StrongBox metadata storage

CIPHERS BEYOND CHACHA20

- Different ciphers behave differently under StrongBox
  - Various security guarantees offered

  - Various performance and power profiles for a given workload
  - Various storage requirements per block beyond StrongBox metadata storage

- Some cipher parameters can be explicitly tuned
  - e.g. rounds, algorithmic complexity, initialization steps



CIPHERS BEYOND CHACHA20

- Different ciphers behave differently under StrongBox
  - Various security guarantees offered More Security ⟷ Higher Throughput More Energy Efficient ⟷ More Security
  - Various performance and power profiles for a given workload
  - Various storage requirements per block beyond StrongBox metadata storage More Security ⟷ Less Wasted Space
- Some cipher parameters can be explicitly tuned
  - e.g. rounds, algorithmic complexity, initialization steps Rounds and/or Key Bits ⟷ Security
- StrongBox itself has tunable parameters (e.g. flake size)

FPN ⟷ Throughput

Flake Size ⟷ Throughput

Energy Efficiency ⟷ Flake Size

CIPHERS BEYOND CHACHA20

- Different ciphers behave differently under StrongBox
  - Various security guarantees offered

    More Security ⟷ Higher throughput

    More Energy Efficient ⟷ More Security
  - Various performance and power profiles for a given workload
  - Various storage requirements per block beyond StrongBox metadata storage

    More Security ⟷ Less Wasted Space
- Some cipher parameters can be explicitly tuned
  - e.g. rounds, algorithmic complexity, initialization steps

    Rounds and/or Key Bits ⟷ Security
- StrongBox itself has tunable parameters (e.g. flake size)

  Flake Size ⟷ Throughput

  Energy Efficiency ⟷ Flake size

  FPN ⟷ Throughput
- Other choices

  Throughput ⟷ Switch Strategy

CIPHERS BEYOND CHACHA20

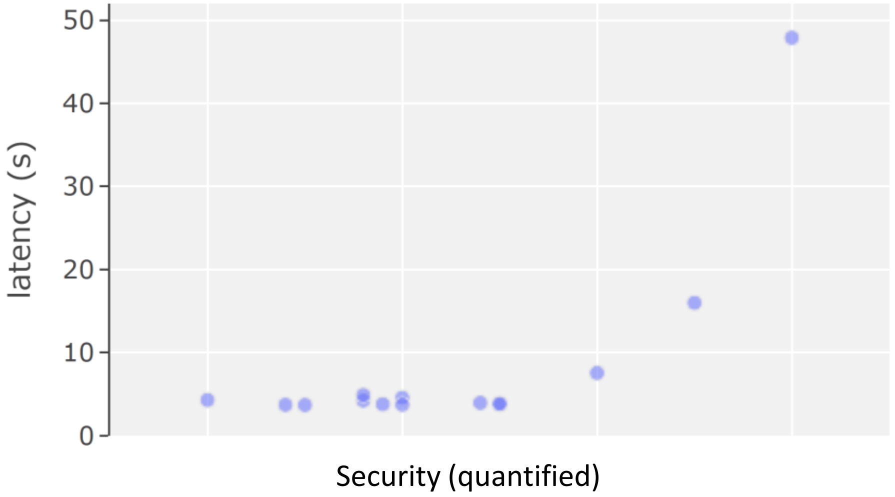
- ◊ Different ciphers behave differently under StrongBox
  - ◊ Various security guarantees offered — More Security ⟷ Higher Throughput
    More Energy Efficient ⟷ More Security
  - ◊ Various performance and power profiles for a given workload
  - ◊ Various storage requirements per block beyond StrongBox metadata storage — More Security ⟷ Less Wasted Space
- ◊ Some cipher parameters can be explicitly tuned
  - ◊ e.g. rounds, algorithmic complexity, initialization steps — Rounds and/or Key Bits ⟷ Security
- ◊ StrongBox itself has tunable parameters (e.g. flake size) — Flake Size ⟷ Throughput
- ◊ Other choices — Throughput ⟷ Switch Strategy
  Energy Efficiency ⟷ Flake Size
  FPN ⟷ Throughput
- ◊ **Can we leverage this diversity to our benefit? Yes!**

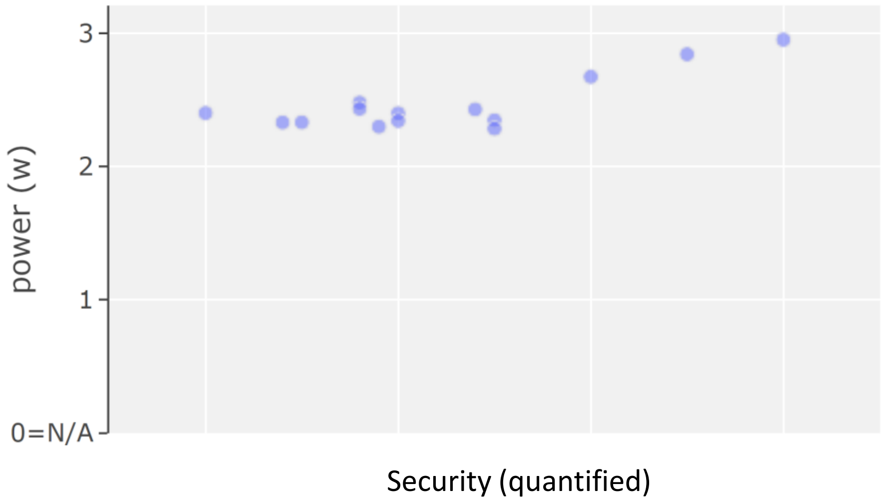
MOTIVATING CIPHER SWITCHING

- Dynamically navigating tradeoffs for profit is well-studied
  - Minimizing energy profiles (POET, CoPPer; Imes et al)
  - Approximation, self-aware systems (a lot; Hoffmann et al)
  - Learned models + control theory (CALOREE; Mishra et al)
  - Complex config optimization (SmartConf; Wang et al)
  - Many others!

MOTIVATING CIPHER SWITCHING (CONT.)

- Dynamically navigating tradeoffs for profit is well-studied
- Comparing ciphers, we observe non-linear relationships between tradeoff dimensions
- Tradeoff spaces are here! Need a “knob” to navigate them





Two examples: security vs latency, power; observed frontier curves for 19 ciphers in various configurations under StrongBox

"QUANTIFYING" SECURITY DIMENSION

- ◊ Quantifying features to determine **rank**
  - ◊ Output randomization (OR)
  - ◊ Resistance to cryptanalysis (CR)
  - ◊ Relative round/relative key length (RR/RK)

	OR	CR	RR/RK	RANK
CN8	0	0.5	0	0.5
CN12	0	0.5	0.5	1
CN20	0	0.5	1	1.5
C8	0	0.5	0	0.5
C12	0	0.5	0.5	1
C20	0	0.5	1	1.5
S8	0	0.4	0	0.4
S12	0	0.4	0.5	0.9

"QUANTIFYING" SECURITY DIMENSION

- Quantifying features to determine **rank**
  - Output randomization (OR)
  - Resistance to cryptanalysis (CR)
  - Relative round/relative key length (RR/RK)

	OR	CR	RR/RK	RANK
CN8	0	0.5	0	0.5
CN12	0	0.5	0.5	1
CN20	0	0.5	1	1.5
C8	0	0.5	0	0.5
C12	0	0.5	0.5	1
C20	0	0.5	1	1.5
S8	0	0.4	0	0.4
S12	0	0.4	0.5	0.9

- **Ideal**: cipher rank distributed evenly along a continuous "security gradient"
  - Allows us to communicate desired security to the system easily

"QUANTIFYING" SECURITY DIMENSION

- Quantifying features to determine **rank**
  - Output randomization (OR)
  - Resistance to cryptanalysis (CR)
  - Relative round/relative key length (RR/RK)

	OR	CR	RR/RK	RANK
CN8	0	0.5	0	0.5
CN12	0	0.5	0.5	1
CN20	0	0.5	1	1.5
C8	0	0.5	0	0.5
C12	0	0.5	0.5	1
C20	0	0.5	1	1.5
S8	0	0.4	0	0.4
S12	0	0.4	0.5	0.9

- **Ideal**: cipher rank distributed evenly along a continuous "security gradient"
  - Allows us to communicate desired security to the system easily

- **Reality**: ciphers ranked non-uniformly and discretely along "security axis" based on the above quantifying features

CHARACTERIZING CIPHERS

- Security (quantified)
  - Our primary knob!

CHARACTERIZING CIPHERS

- Security (quantified)
  - Our primary knob!
- Energy use
  - Strictly linear? Software vs hardware impl tradeoffs?

CHARACTERIZING CIPHERS

- ◊ Security (quantified)
  - ◊ Our primary knob!
- ◊ Energy use
  - ◊ Strictly linear? Software vs hardware impl tradeoffs?
- ◊ Performance
  - ◊ Read and write throughput under various StrongBox configurations for sequential and random workloads of varying sizes

CHARACTERIZING CIPHERS

- Security (quantified)
  - Our primary knob!
- Energy use
  - Strictly linear? Software vs hardware impl tradeoffs?
- Performance
  - Read and write throughput under various StrongBox configurations for sequential and random workloads of varying sizes
- Other characteristics
  - Area (i.e. metadata storage requirements)

SWITCH STRATEGIES

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?

SWITCH STRATEGIES

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?
- (partial) Answer: cipher switching *strategies*

SWITCH STRATEGIES

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?
- (partial) Answer: cipher switching *strategies*
  - **Immediate** (slowest total throughput)

SWITCH STRATEGIES

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?
- (partial) Answer: cipher switching *strategies*
  - **Immediate** (slowest total throughput)
  - **Forward Immediate** (best for random IO)

SWITCH STRATEGIES

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?
- (partial) Answer: cipher switching *strategies*
  - **Immediate** (slowest total throughput)
  - **Forward Immediate** (best for random IO)
  - **Aggressive Immediate** (sequential IO friendly)

SWITCH STRATEGIES

- ◊ StrongBox II research (April-May 2019) seeks to answer:
  - ◊ How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - ◊ What does this cipher configuration tradeoff space look like?
  - ◊ Exactly how much benefit is realizable here (use cases)?
- ◊ (partial) Answer: cipher switching *strategies*
  - ◊ **Immediate** (slowest total throughput)
  - ◊ **Forward Immediate** (best for random IO)
  - ◊ **Aggressive Immediate** (sequential IO friendly)
  - ◊ **Opportunistic Immediate** (reduces to forward-immediate)

SWITCH STRATEGIES

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?
- (partial) Answer: cipher switching *strategies*
  - **Immediate** (slowest total throughput)
  - **Forward Immediate** (best for random IO)
  - **Aggressive Immediate** (sequential IO friendly)
  - **Opportunistic Immediate** (reduces to forward-immediate)
  - **Mirrored Immediate** (fastest reads by far, slowest writes)
  - Others?

SWITCH STRATEGIES (CONT.)

- StrongBox II research (April-May 2019) seeks to answer:
  - How can StrongBox effectively switch between and utilize different ciphers on the same system?
  - What does this cipher configuration tradeoff space look like?
  - Exactly how much benefit is realizable here (use cases)?
- Partial answer: cipher switching *strategies*
- This implies there are higher level performance/energy tradeoffs between different cipher switching strategies

STRONGBOX II: GOALS

- Select interesting ciphers
- Quantify security, other “knobs”
- Determine characteristics of ciphers
- Implement cipher switching/strategies
- Explore contours of tradeoff space(s) of competing concerns
  - Measure impact of cipher metadata extra storage requirements
  - Experiment with *mirrored immediate* switching strategy
- Model/implement dynamic switching use case scenarios
  - Use switching strategies to navigate the space for fun and profit
  - Explicitly state threat models for use cases
- Determine full extent of realizable benefit

STRONGBOX II: PROGRESS

- ✓ Select interesting ciphers (2017)
- ✓ Quantify security, other “knobs” (Early 2018)
- ✓ Determine characteristics of ciphers (Early 2018)
- ✓ Implement cipher switching/strategies (1000+ LoC; Q4 2018)
- ✓ Explore contours of tradeoff space(s) of competing concerns
  - ✓ Measure impact of cipher metadata extra storage requirements
  - ◊ Experiment with *mirrored immediate* switching strategy (Now!)
- ◊ Model/implement dynamic switching use case scenarios (April)
  - ◊ Use switching strategies to navigate the space for fun and profit
  - ◊ Explicitly state threat models for use cases
- ◊ Determine full extent of realizable benefit (April-May)

STRONGBOX

StrongBox I (Published in ASPLOS 2018)

- ✓ We overturn the conventional wisdom regarding FDE: stream ciphers *are* suitable for FDE!
- ✓ We develop and implement a secure approach to FDE based on the ChaCha20 stream cipher, F2FS LFS, and RPMB eMMC trusted hardware

StrongBox II (April 2019)

- ✓ Building on the success of the StrongBox approach, we consider the tradeoff space and motivated use cases formed between ciphers beyond ChaCha20 (e.g. Freestyle, Rabbit, AES-CTR)

StrongBox III (Q4 2019)

- ✓ We present a new theoretical framework for FDE, define the cryptographic goals of StrongBox, and formally evaluate the security properties of our construction

NEW SECURITY NOTIONS FOR FDE?

- Prior generalized FDE security notions assume deterministic output due to sector number, PBA, et cetera
  - This is not valid for ciphers under StrongBox

NEW SECURITY NOTIONS FOR FDE?

- Prior generalized FDE security notions assume deterministic output due to sector number, PBA, et cetera
  - This is not valid for ciphers under StrongBox

- XOR FDE with stream ciphers/CTR mode are considered trivially vulnerable to known-plaintext attacks
  - StrongBox avoids this by avoiding overwrites entirely

NEW SECURITY NOTIONS FOR FDE?

- Prior generalized FDE security notions assume deterministic output due to sector number, PBA, et cetera
  - This is not valid for ciphers under StrongBox
- XOR FDE with stream ciphers/CTR mode are considered trivially vulnerable to known-plaintext attacks
  - StrongBox avoids this by avoiding overwrites entirely
- We want: well-defined cryptographic goals for StrongBox
  - E.g. security against chosen-ciphertext attack?

STRONGBOX III: GOALS

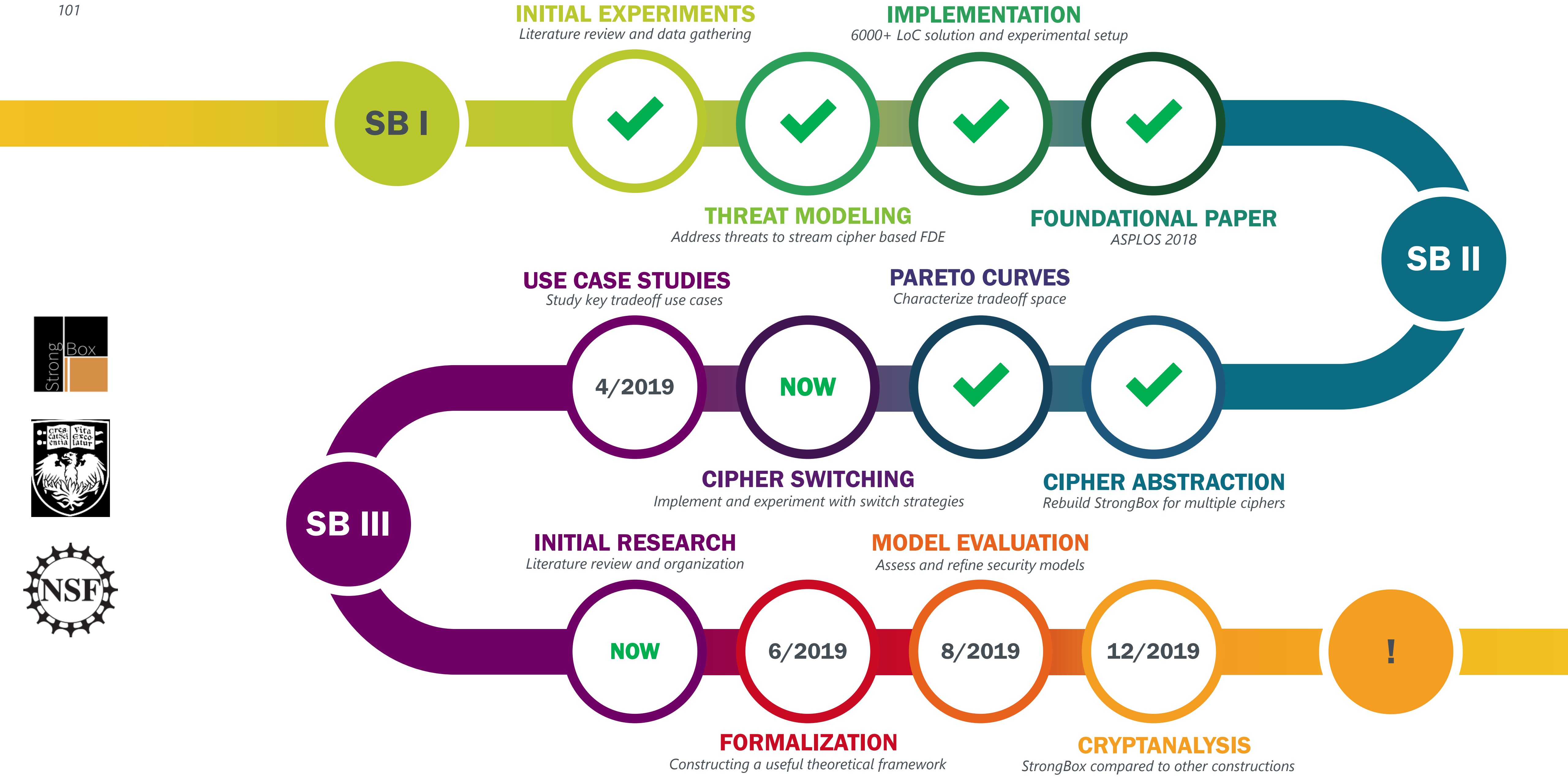
- The big question: **how secure is StrongBox FDE**? How much "security is left" in FDE context given StrongBox constraints?

STRONGBOX III: GOALS

- The big question: **how secure is StrongBox FDE**? How much "security is left" in FDE context given StrongBox constraints?
- Define a new set of formal FDE security notions (May-June)

STRONGBOX III: GOALS (CONT.)

- The big question: **how secure is StrongBox FDE**? How much "security is left" in FDE context given StrongBox constraints?
- Define a new set of formal FDE security notions (May-June)
- Use these notions to construct a theoretical framework to evaluate StrongBox FDE (subject to practical constraints)
- Refine security notions, models; evaluate other relevant constructions versus StrongBox using our framework (2019)
- Revisit AES-XTS formal security guarantees versus StrongBox with any valid cipher (2019)


SB I
INITIAL EXPERIMENTS
Literature review and data gathering
THREAT MODELING
Address threats to stream cipher based FDE
IMPLEMENTATION
6000+ LoC solution and experimental setup
FOUNDATIONAL PAPER
ASPLOS 2018
SB II
CIPHER ABSTRACTION
Rebuild StrongBox for multiple ciphers
PARETO CURVES
Characterize tradeoff space
CIPHER SWITCHING
Implement and experiment with switch strategies
NOW
USE CASE STUDIES
Study key tradeoff use cases
4/2019
SB III
INITIAL RESEARCH
Literature review and organization
NOW
FORMALIZATION
Constructing a useful theoretical framework
6/2019
MODEL EVALUATION
Assess and refine security models
8/2019
CRYPTANALYSIS
StrongBox compared to other constructions
12/2019
!
StrongBox

PROJECT STRONGBOX

- Questions?

BERNARD DICKENS III HARYADI S. GUNAWI DAVID CASH ARIEL J. FELDMAN HENRY HOFFMANN