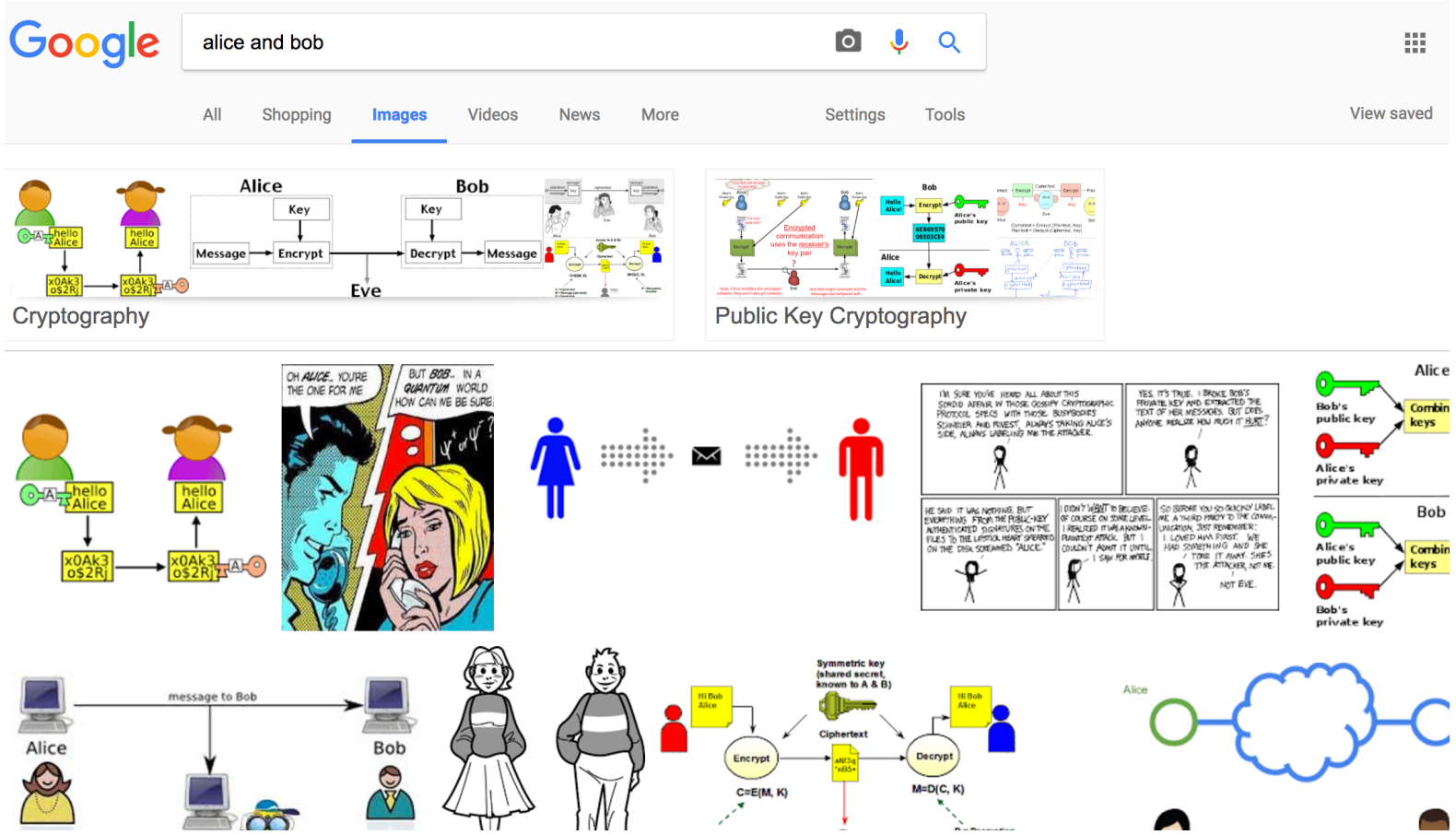
Protecting Small Data Items

Terence Spies

Chief Technologist, HPE Data Security

Evolving Models for Encryption



Classic Encryption Model

- Trusted party to trusted party
- Performance is measured by packets/bytes/sectors per second
- Stream mode performance is vital

Decrypt-before-use

Application layer security

- Two new-ish problem domains:
  - Layering encryption into existing applications
  - Building data substitutes that can be used without decryption

- Security model is trusted-to-semitrusted endpoint (cloud, hadoop)
- Performance is about ability to accommodate data characteristics
- Item-by-item rather than stream mode encryption

Use-before-decrypt

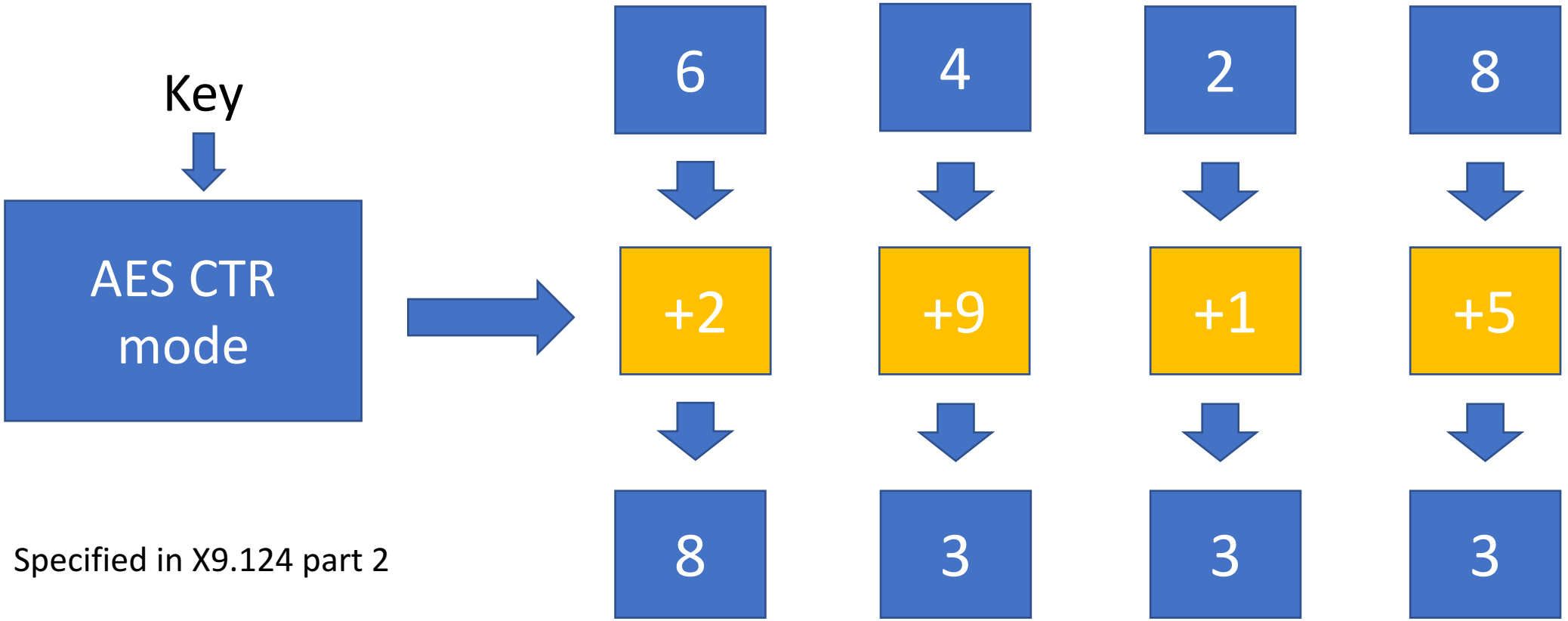
Today's Talk

- Initial solutions to application layer encryption: SP800-38G
- The Feistel security model and proofs
- Challenges and fixes for 38G methods
- Other models for building very-tiny domain ciphers

Format-Preserving Encryption

- One solution to this issue is FPE, which NIST adopted under SP800-38G in 2016

Two FPE models: Streaming

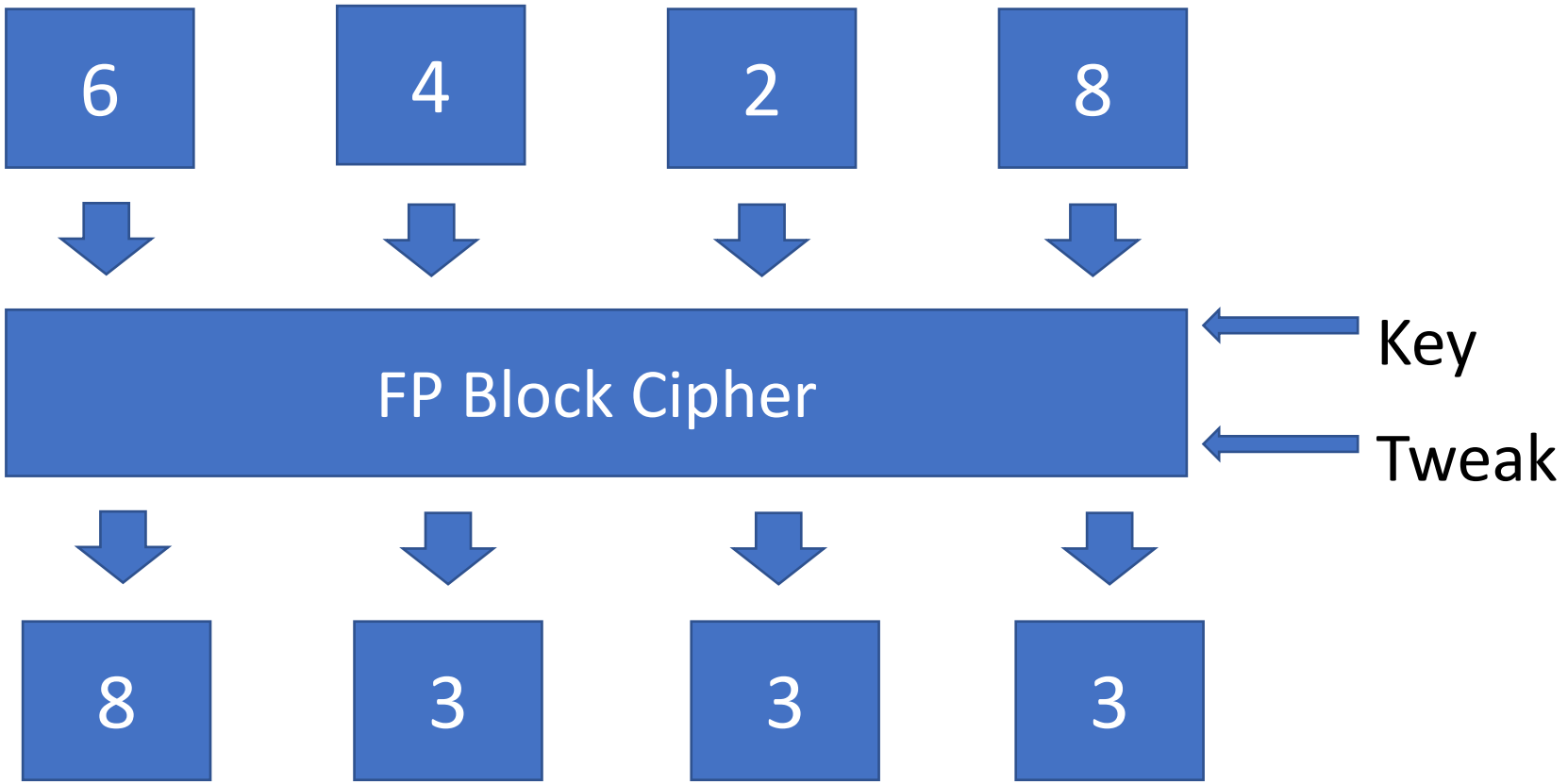


Specified in X9.124 part 2

Streaming

- Advantages:
  - Very fast, parallelizable
  - Equal security at all plaintext sizes
- Disadvantage:
  - Cannot encrypt multiple data items per key
  - Mallebility
- Useful in point-to-point contexts

Two FPE models: Variable sized permutation



Variable sized permutation

- Advantages:
  - Creates useful random maps (aka tokens or psuedonyms)
  - No mallebility
- Disadvantages:
  - Slower (requires multiple cipher queries per encryption)
  - Small data sizes are harder to secure
- Useful in storage contexts

- One way to implement FPE is via databases…

Tweaking

- Odd name, critical to encrypting small data items
- Key = private entropy, tweak = public entropy
- Can be used to keep subfields in the clear, removing the need for decryption

The FPE Security Model

- Standard FPE methods are based on Feistel networks
  - Well-studied construction, used as the basis for DES, other ciphers
  - Famous proofs show Feistel networks strong at 3 or 4 rounds

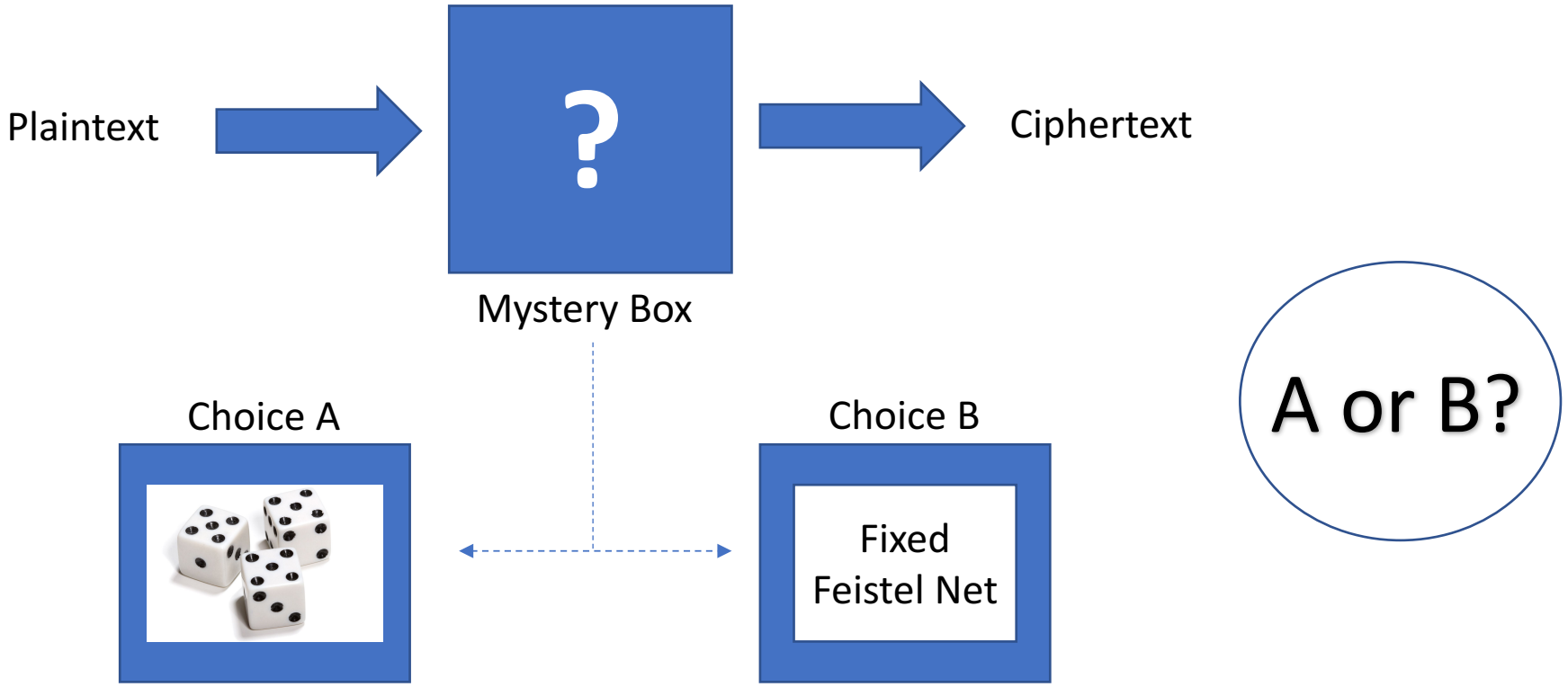
- Feistel is an iterated set of round functions, with the input divided into a right and left half, where each round computes a new right and left half. Each round n uses a distinct inner function fn()

  - R’ = L
  - L’ = R + fn(L)

- Easy to show that this forms a permutation, even if fn() isn’t

Luby-Rackoff Results

- Basic notion: indistinguishability from a random permutation



Luby-Rackoff Results

- If fn() is a set of pseudo-random functions, then:
  - A 3 round Feistel network using fn() is indistinguishable from a pseudo-random permutation by any attacker using up to X known plaintext queries
  - A 4 round Feistel network using fn() is indistinguishable form a psuedo-random permutation by any attacker using up to Y known plaintext queries
  - X and Y are bounded by the sqrt(size of fn()) – the birthday bound
- That bound is fine for larger block ciphers, and we know the bound increases with round count
- Can we use round count to overcome the effect of small fn() size?

Two Important Results

Luby - Rackoff: 5 Rounds are Enough for $2^{n-\epsilon}$ CPCA Security

Jacques Patarin
Université de Versailles
45 avenue des Etats-unis
78035 Versailles Cedex
France
Jacques.Patarin@prism.uvsq.fr

Abstract

We study adaptive chosen plaintext attacks (CPA) and adaptive chosen plaintext and chosen ciphertext attacks (CPCA) on random Feistel schemes.We denote by m the number of plaintext/ciphertext pairs, and by k the number of rounds.In their famous paper [2], M. Luby and C. Rackoff have completely solved the cases $m \ll 2^{n/2}$: the schemes are secure against all CPA attacks when $k \geq 3$ and against all CPCA attacks when $k \geq 4$.

Indifferentiability of 10-Round Feistel Networks

Yuanxi Dai and John Steinberger

shusdtc@gmail.com, jpsteinb@gmail.com

Abstract. We prove that a (balanced) 10-round Feistel network is indifferentiable from a random permutation. In a previous seminal result, Holenstein et al. [17] had established indifferentiability of Feistel at 14 rounds. Our simulator achieves security $O(q^8/2^n)$, runtime $O(q^4)$ and query complexity $O(q^4)$, to be compared with security $O(q^{10}/2^n)$, runtime $O(q^4)$ and query complexity $O(q^4)$ for the 14-round simulator of Holenstein et al.

What more could you want?

- Two issues:
  - Constants become important when the domain size is very small
  - Adding the ability to tweak the construction is a non-trivial step…
- Tweaks are critical in encryption of small data items in important applications….

Bellare-Hoang-Tessaro Attack

- When the FPE domain is very small, messages under many tweaks may allow message recovery.
- How small, and how efficient?

$2n$	$r = 8$ (FF3)		$r = 10$ (FF1)	
	ϵ	Q	ϵ	Q
4	1/2	2^{21}	1/2	2^{25}
8	14/16	2^{32}	14/16	2^{40}
14	63/64	2^{53}	63/64	2^{67}

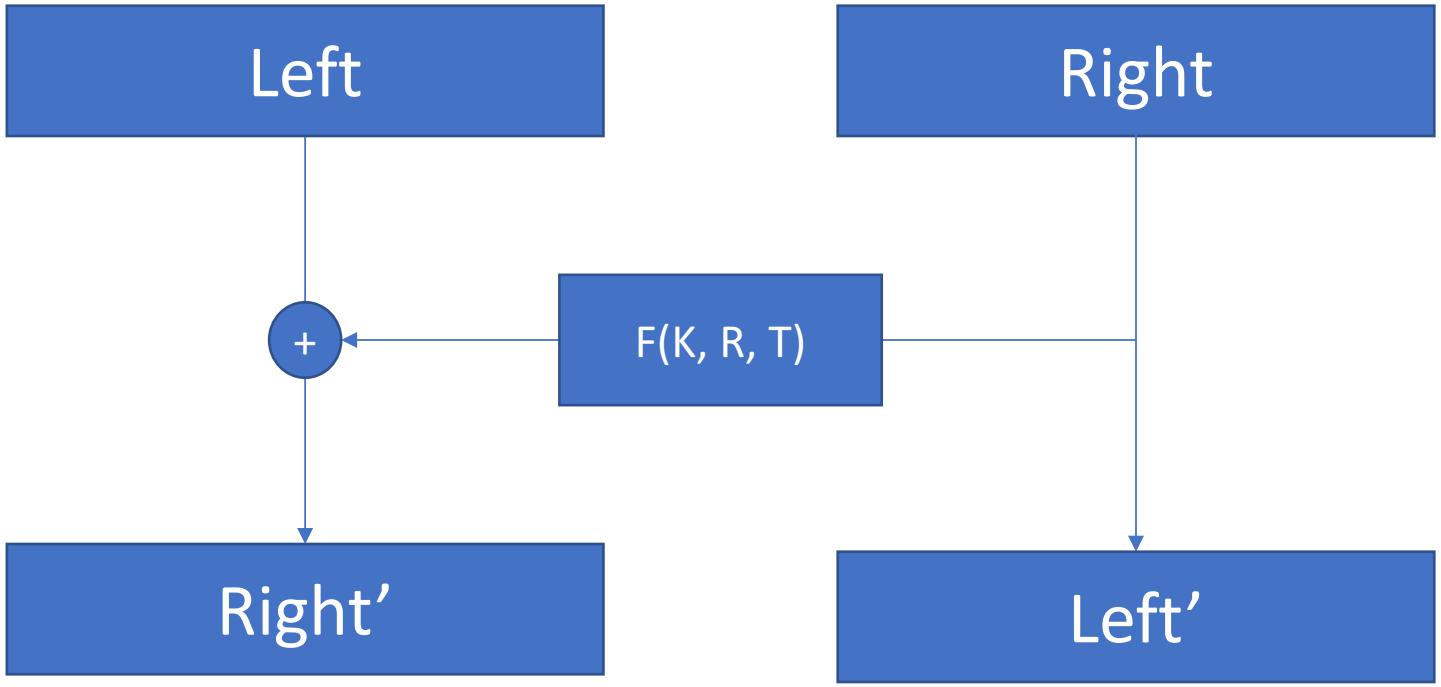
Figure 2: **Attack numbers.** We show the advantage and number of examples for the FMR attack for various input lengths $2n$ and the number of rounds of the standards.

From Bellare, Hoang, Tessaro, “Message Recovery Attacks on Feistel-based FPE”, CCS 2016

Resolving the BHT Attack

- Two solutions:
  - Disallow tiny domain sizes (standards already disallow the smallest cases)
  - Recommend double encryption for small domains
- Attack parameters:
  - Attack difficulty is super-exponential in domain size
  - Exponential in round count (sadly)
  - Double encryption makes attack infeasible even for tiny sizes

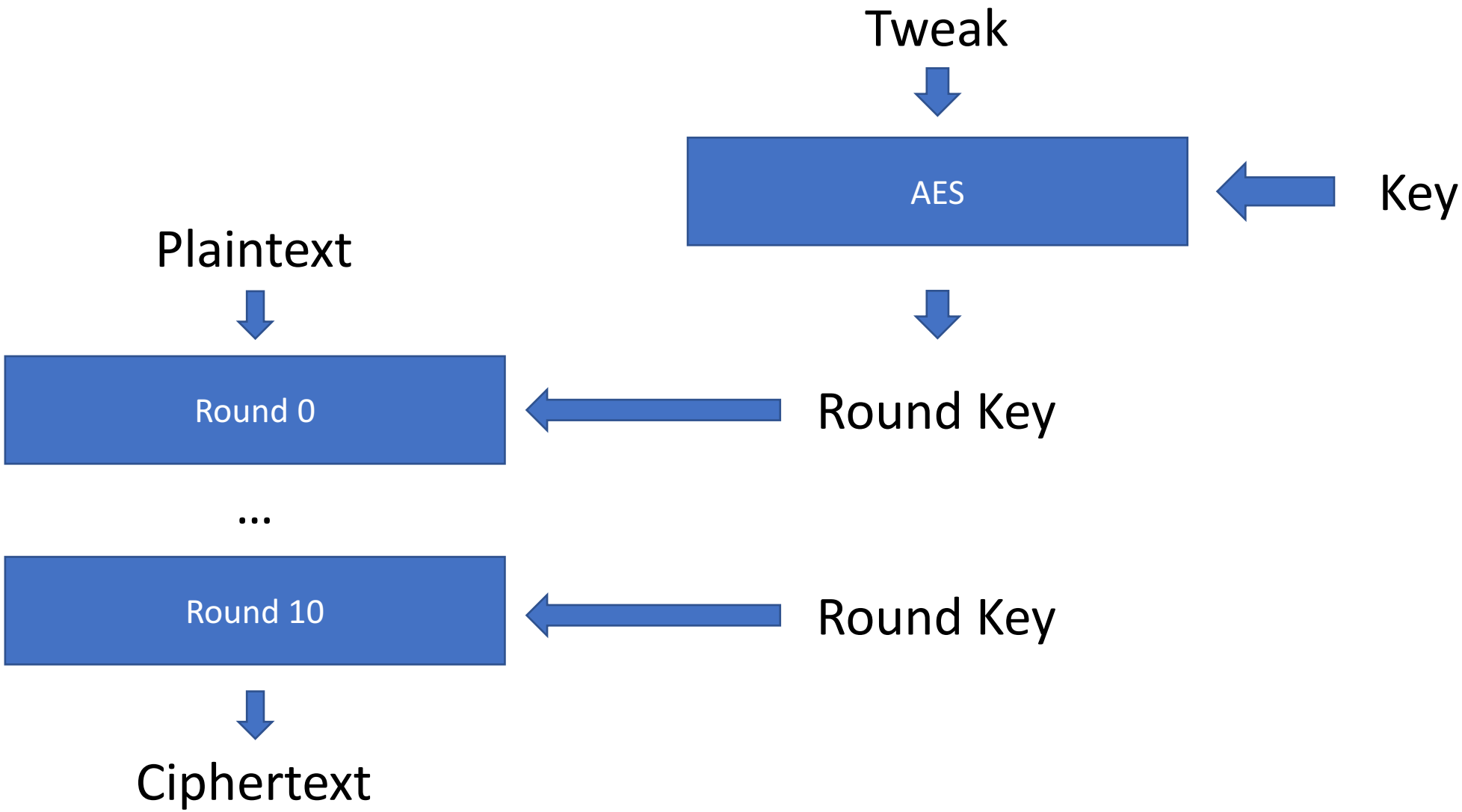
Tweak Inclusion Methods



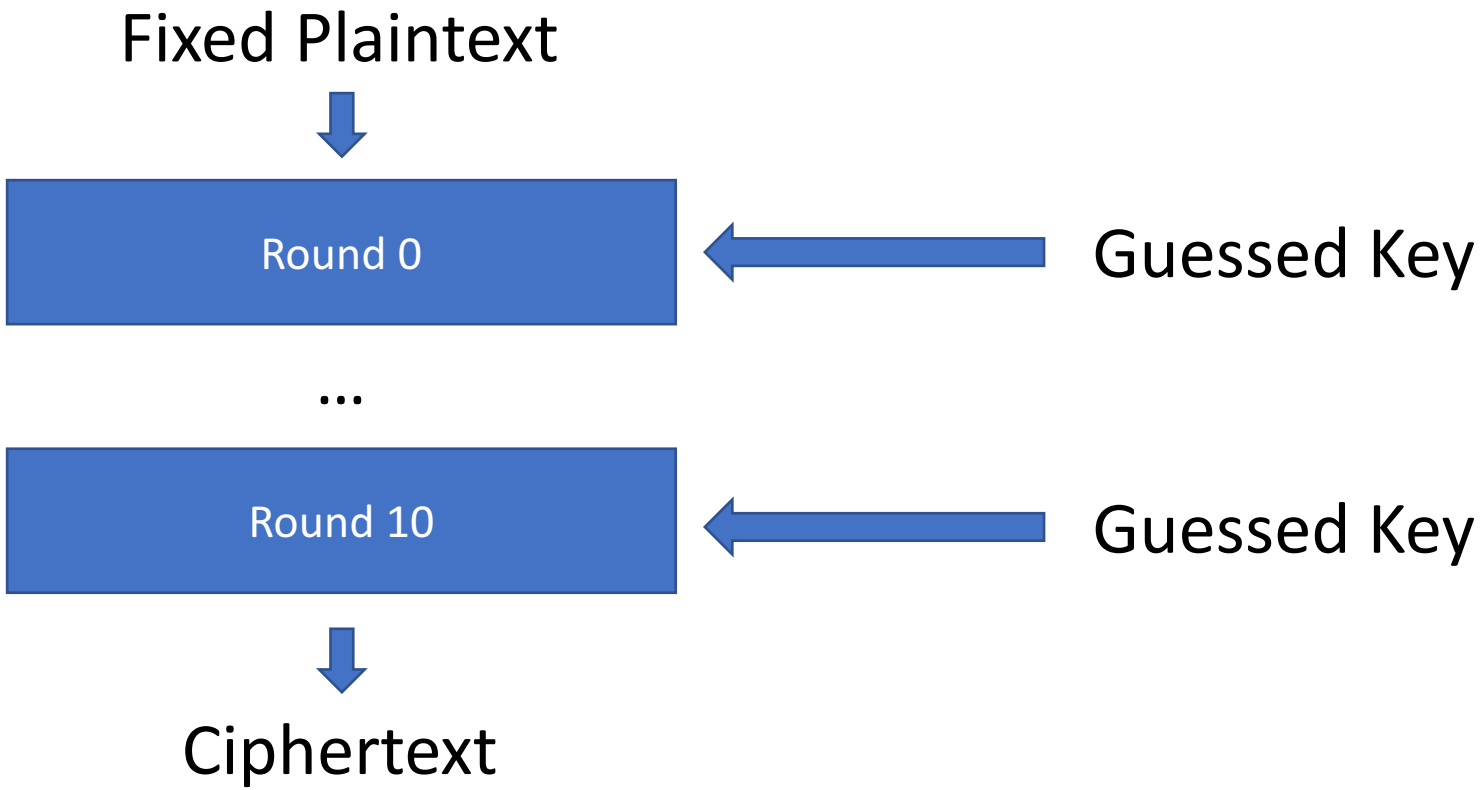
Three Methods

- FF1:
  - F(K, R, T, N) = CMAC-AES(K, T | N | R) (with domain separators)
- FF2:
  - F(K, R, T, N) = AES(AES(K, T), N | R) (with domain separators)
- FF3:
  - F(K, R, T, N) = AES(K, (T XOR N) XOR R)

FF2 “Subkey” Attack



FF2 “Subkey” Attack



FF2 ”Subkey” Attack

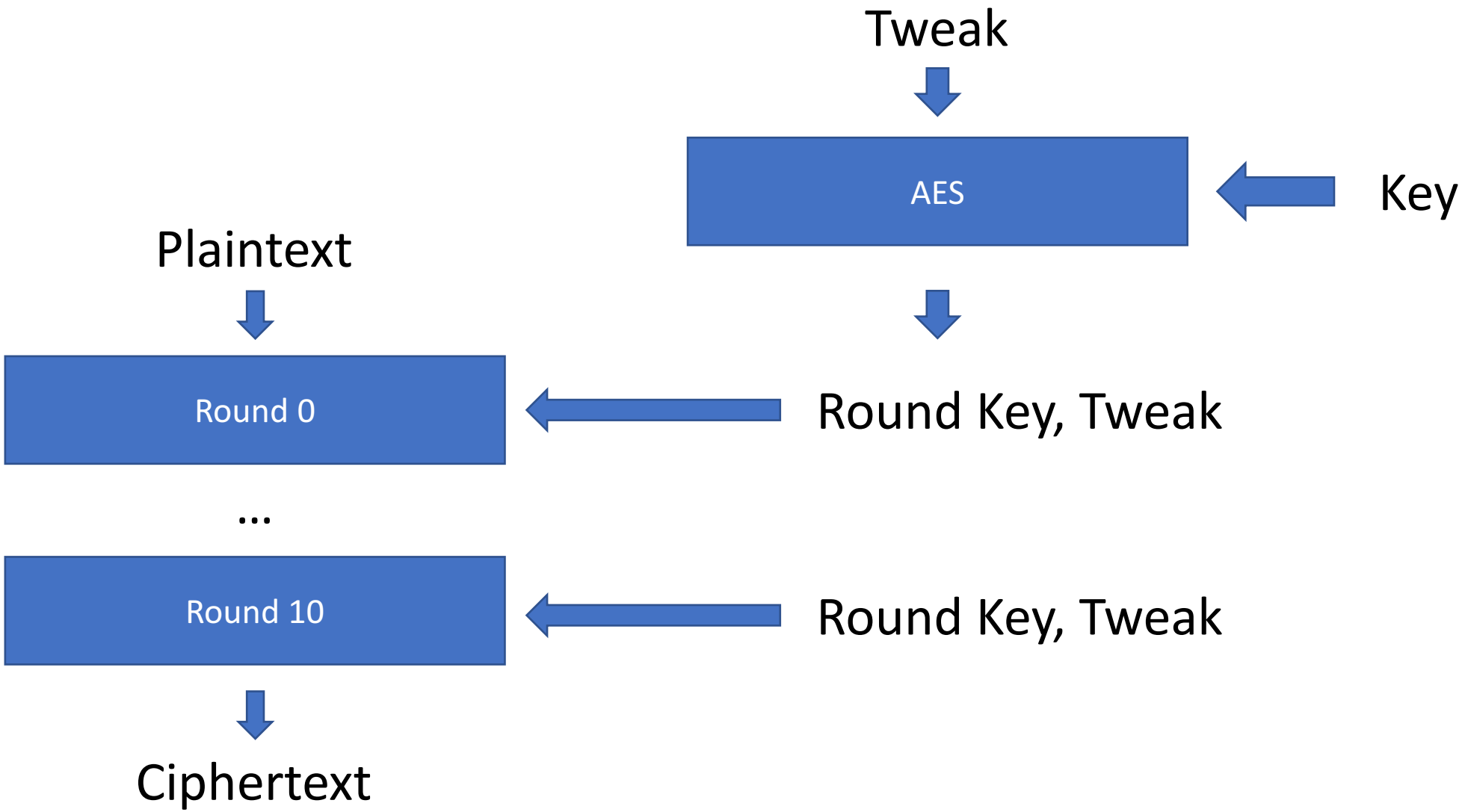
Tweak	Plaintext encrypted under unknown key
0	213 314 922
1	992 128 239
2	771 002 932
3	….
4	…..

Guessed Round Key	Plaintext encrypted under unknown key
7f4b0231…	981 231 001
ffb07343..	412 023 339
012bc43..	213 314 922
….	….
…	…..

FF2 “Subkey” Attack

- How effective is this?
  - 2^{128} / Q, where Q = number of encryptions under different tweaks
  - Effectively trades time for memory
- Can be mitigated at application by preventing large scale encryptions
- Straightforward to mitigate by a small change to FF2

FF2 “Subkey” Fix



FF3 “divide-and-conquer” Attack

- Due to Durak and Vaudenay (paper forthcoming)
- Observation: when the input is small, may be able to attack 3 or 4 round Feistel

- Because FF3 XORs the round number and the tweak, controlling tweak bytes allows the attacker cause round numbers to repeat, turning the 8 round cipher into two four round blocks, allowing the smaller attack to succeed.

FF3 ”divide-and-conquer” Attack

- How practical is this? Still looks like >2^{75} effort for practical cases.
- Fix is straightforward: restrict upper tweak bytes so that tweak and round counter cannot interact.

- X9 is working on incorporation of these fixes into X9.124, and building a structure that will allow future evolution.
  - Double encrypt small items
  - Fix subkey attack
  - Fix FF3 tweak/round interaction

Handling the really small stuff

- In many implementations, out-of-format data is a reality
- Ingesting transaction data with CCNs < 12 digits, empty fields, single character fields
- Need a few different solutions:
  - The ability to encrypt single character values safely
  - Application level protocols for hiding empty fields and short values
  - Need logic to hide/strip fields

Sub-FPE encryption

- How do we handle very small fields? (outside of padding)
- Shuffle methods can be effective
  - Write out every possible input
  - Use AES to produce random bits
  - Method 1: Knuth shuffle
  - Method 2: Encrypt with AES ECB, sort the outputs
- Would be useful to have a FIPS method for doing this…

What have we learned about data item privacy?

- Format preservation is useful, but only part of the solution
- Ultimate value: avoiding decryptions in semi-trusted enviroments
- This means surgically controlling properties reflected in ciphertext
  - Clear subfields
  - Referential integrity
  - Many other wishlist items, especially in analytic environments
- Feistel techniques are a flexible, efficient technique with established security arguments