Flexible Authenticated Encryption

Sanketh Menda, Julia Len, Viet Tung Hoang, Mihir Bellare, and Thomas Ristenpart

The Third NIST Workshop on Block Cipher Modes of Operation 2023

One scheme to rule them all?

???

One scheme to rule them all?

???

Fast on
AES-NI servers

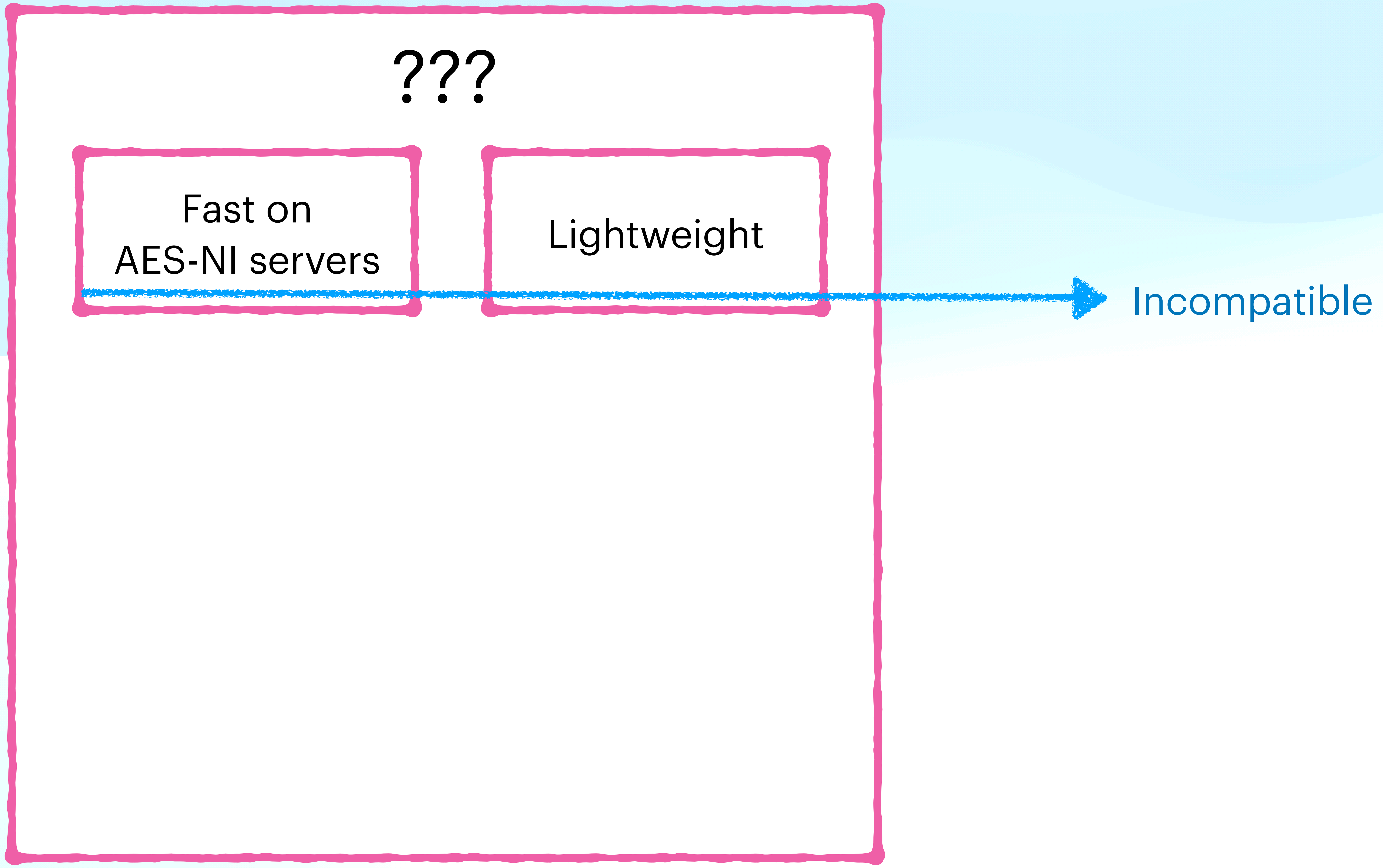
One scheme to rule them all?

???

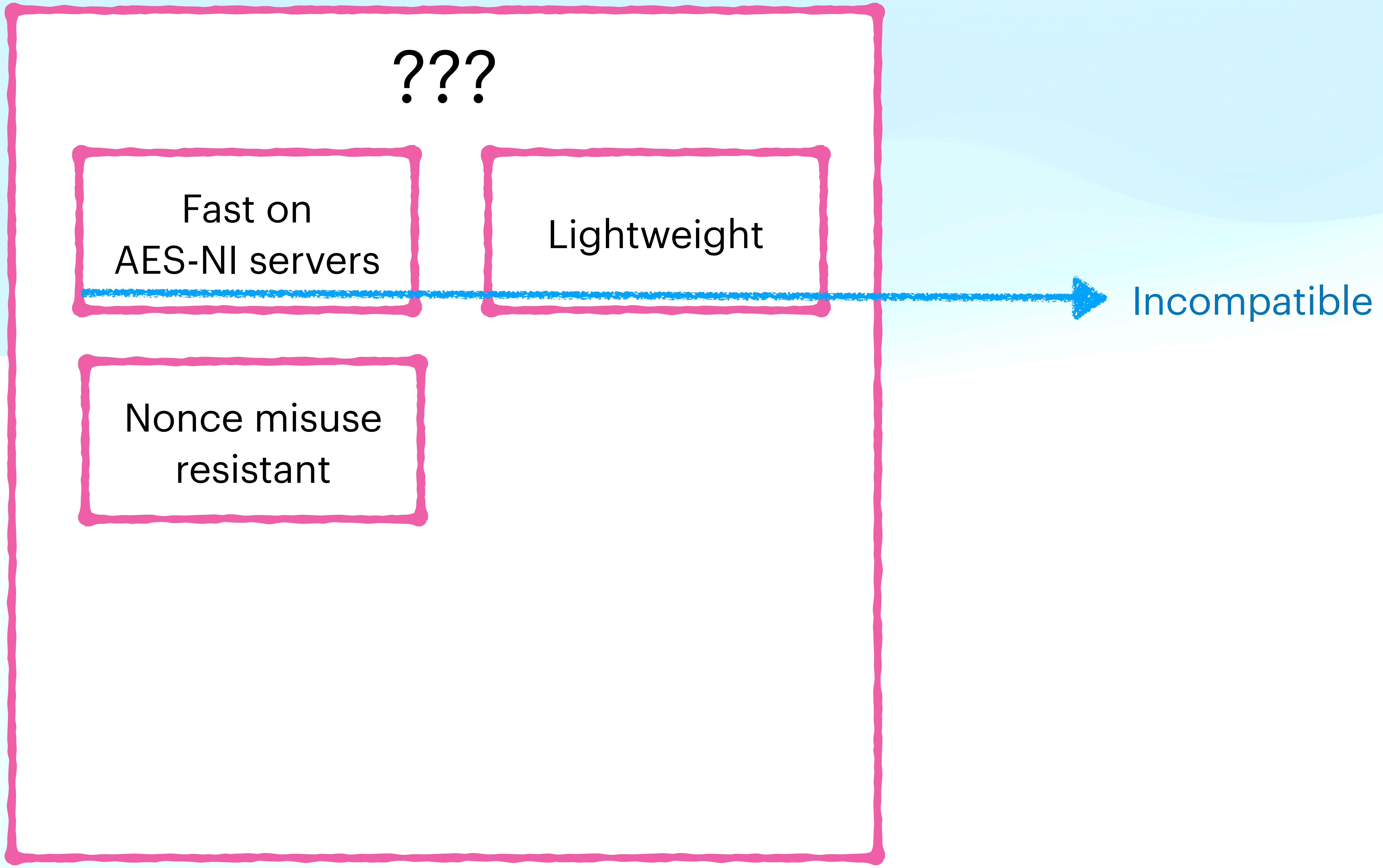
Fast on
AES-NI servers

Lightweight

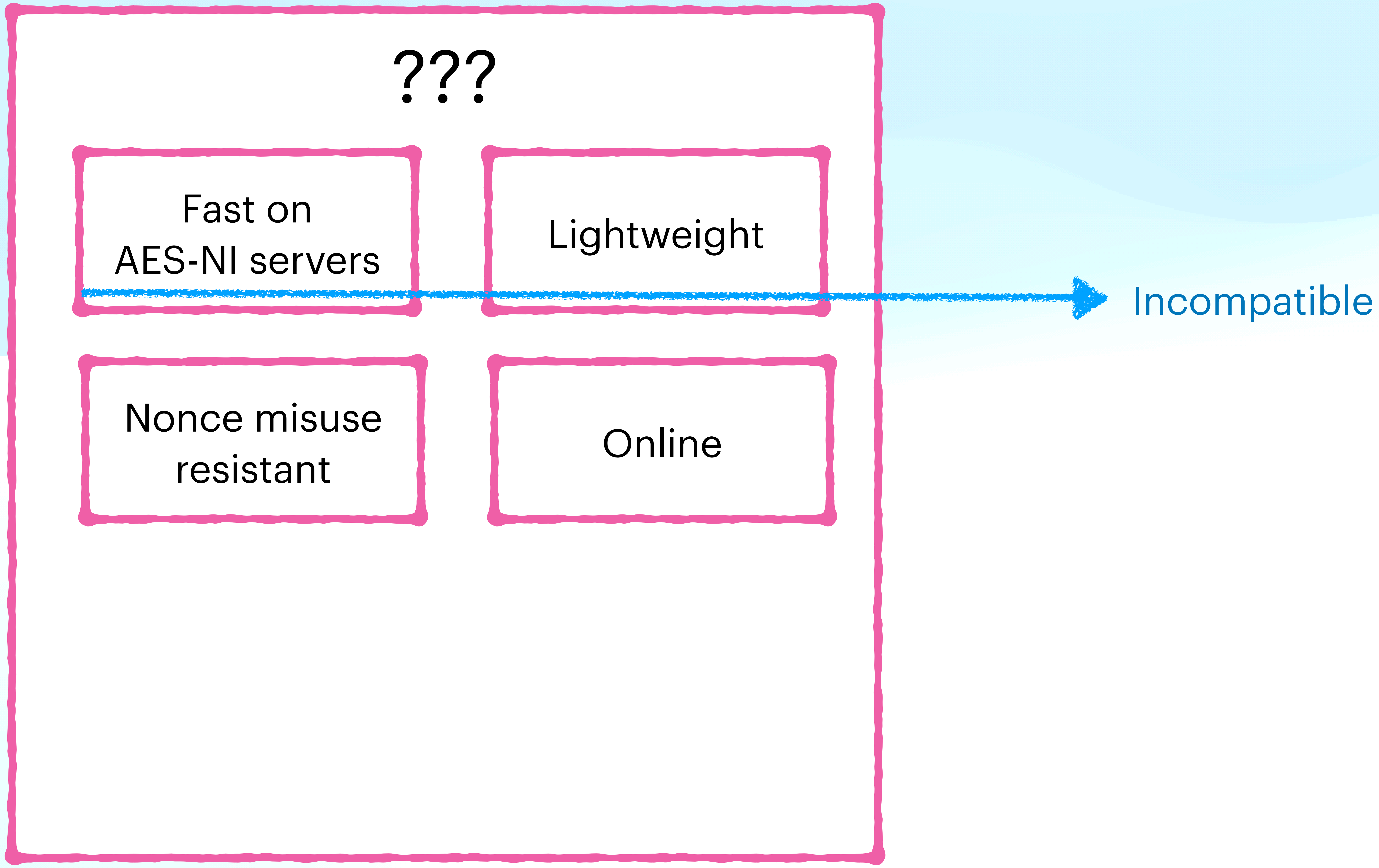
One scheme to rule them all?



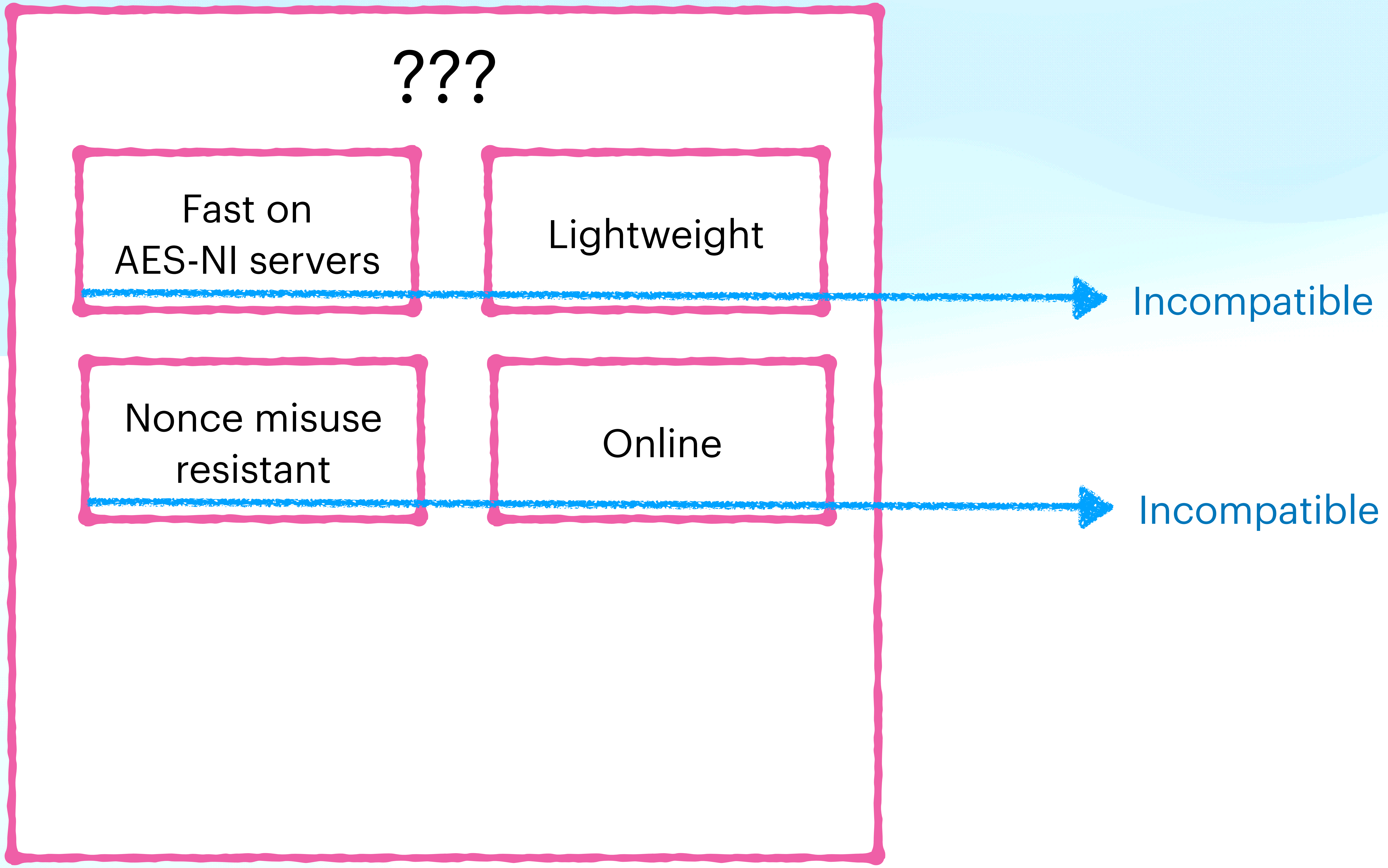
One scheme to rule them all?



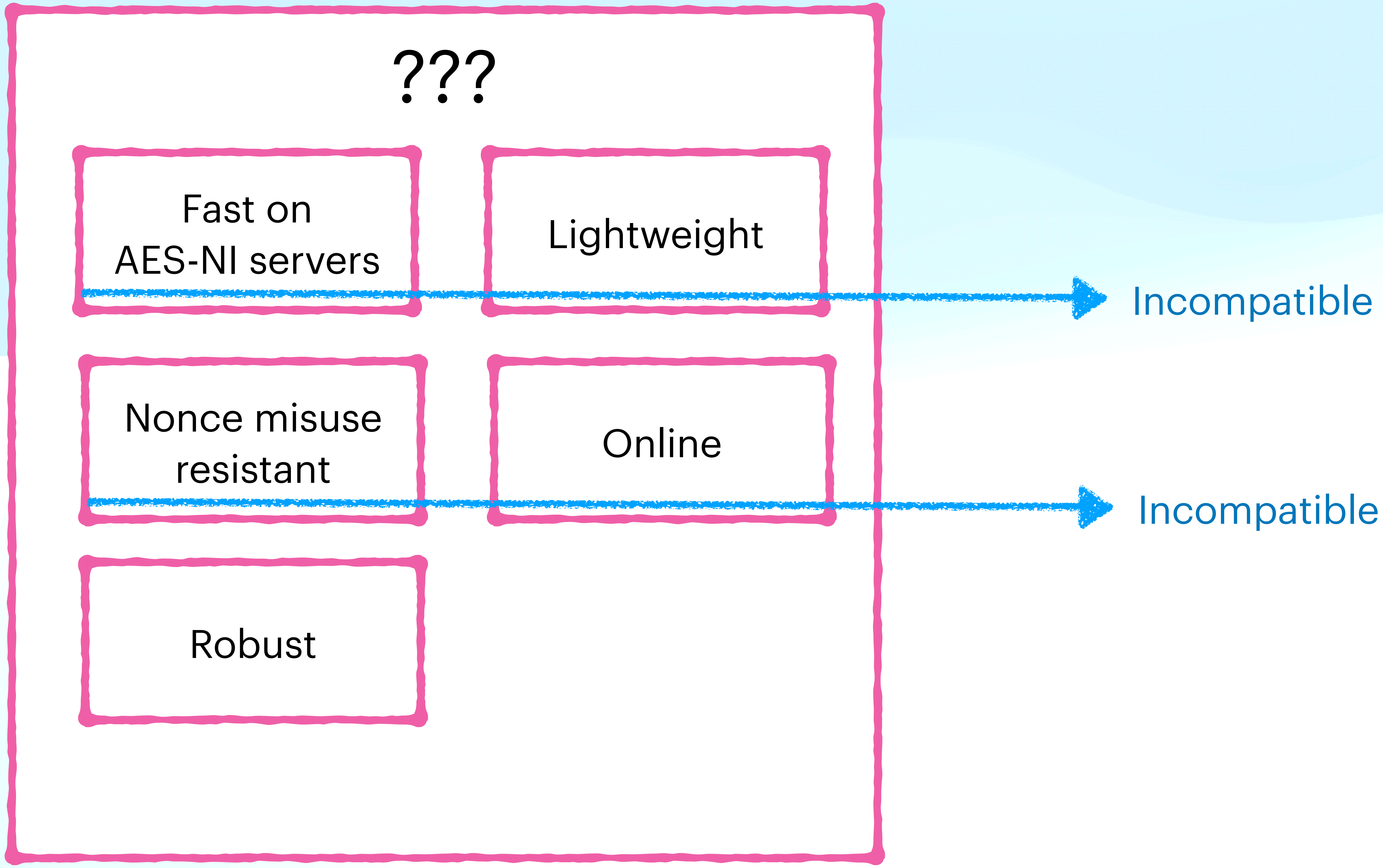
One scheme to rule them all?



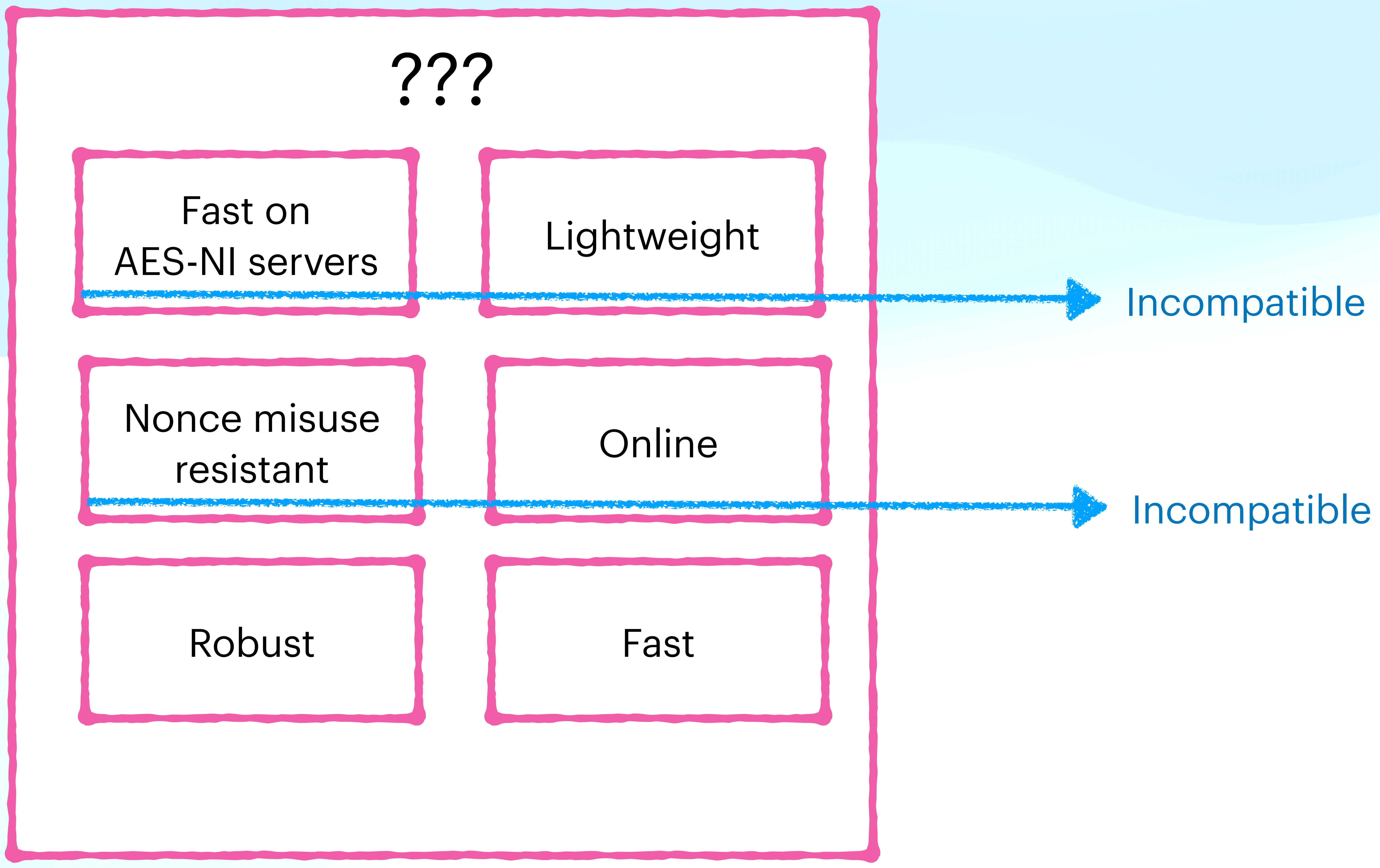
One scheme to rule them all?



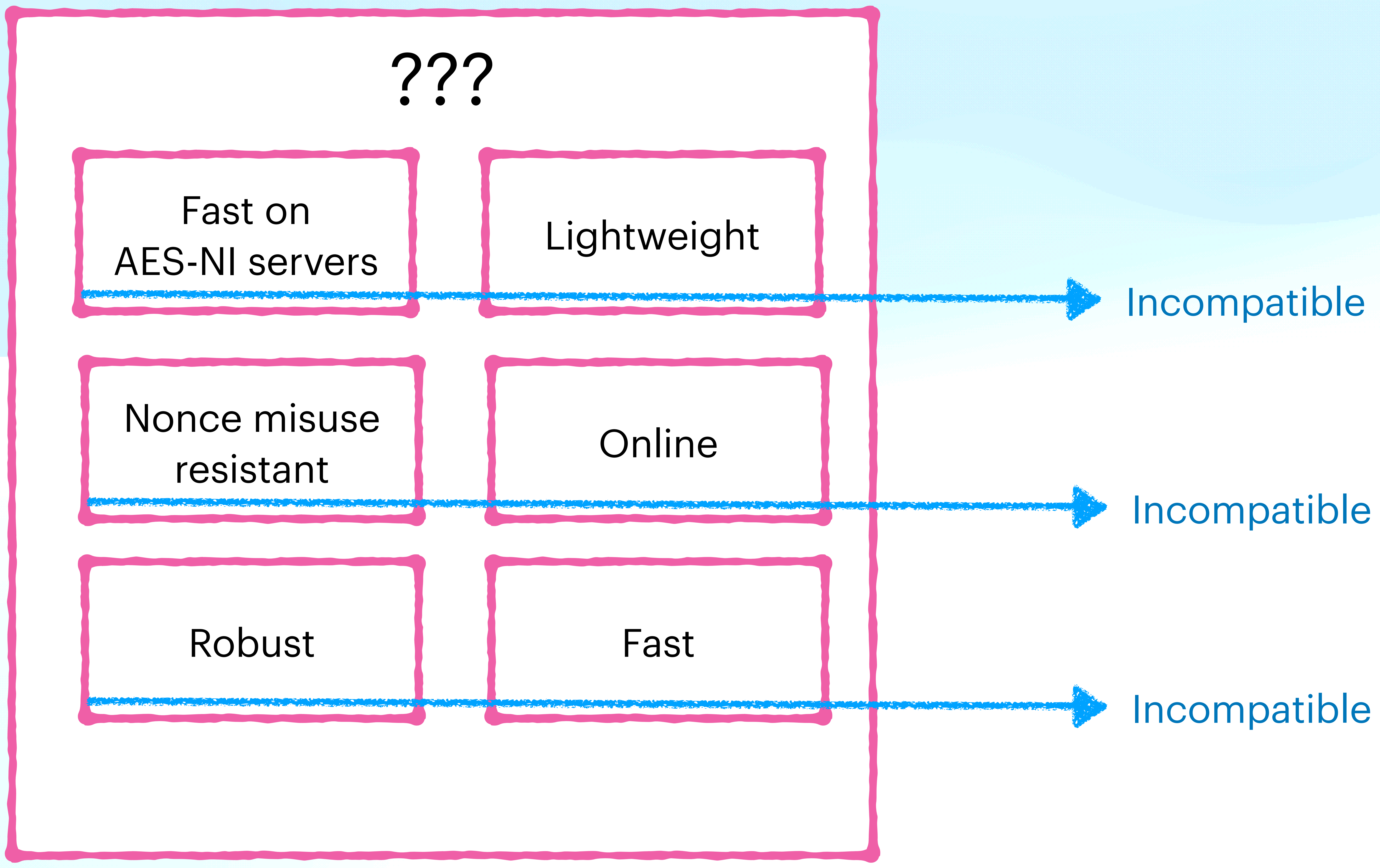
One scheme to rule them all?



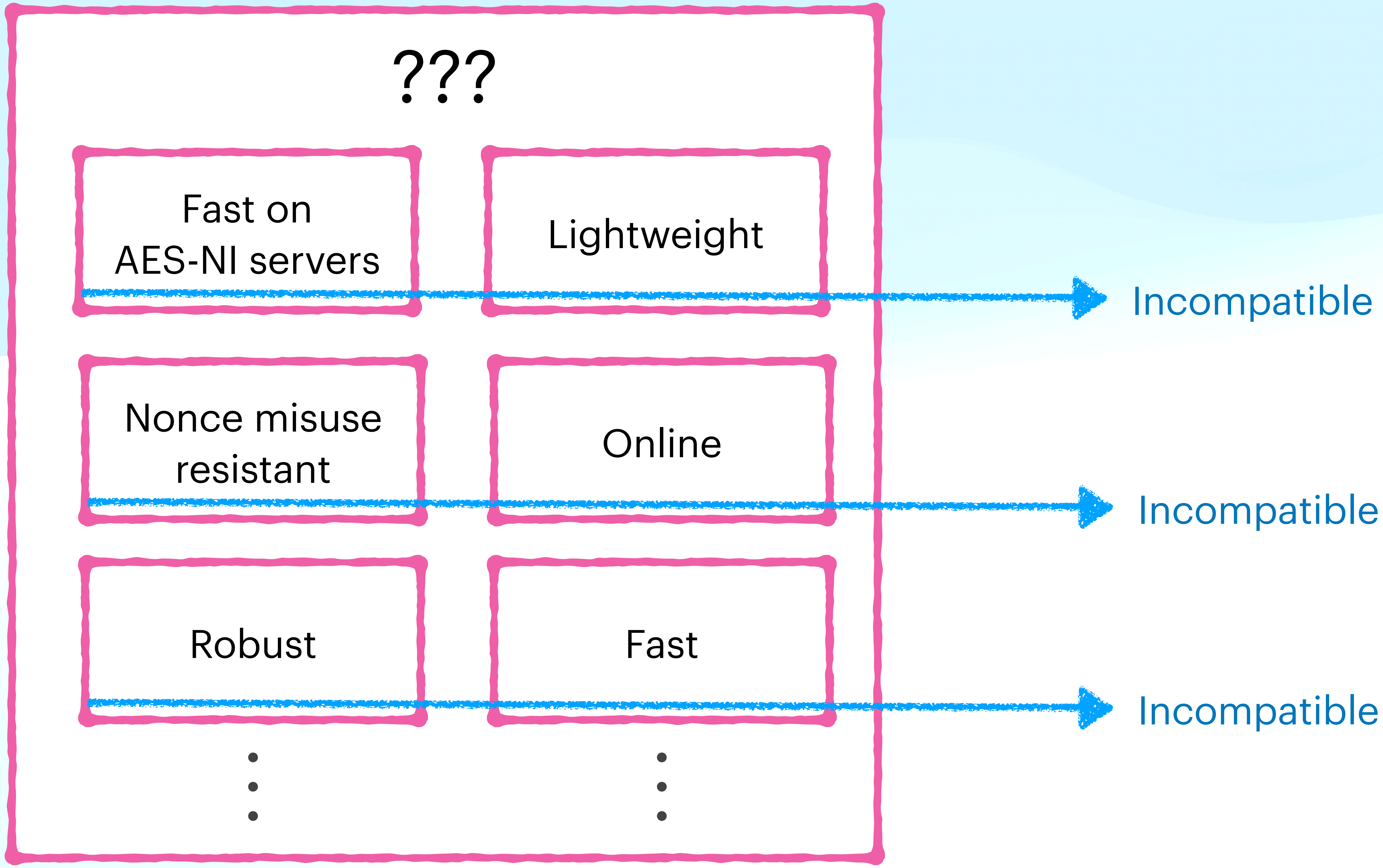
One scheme to rule them all?



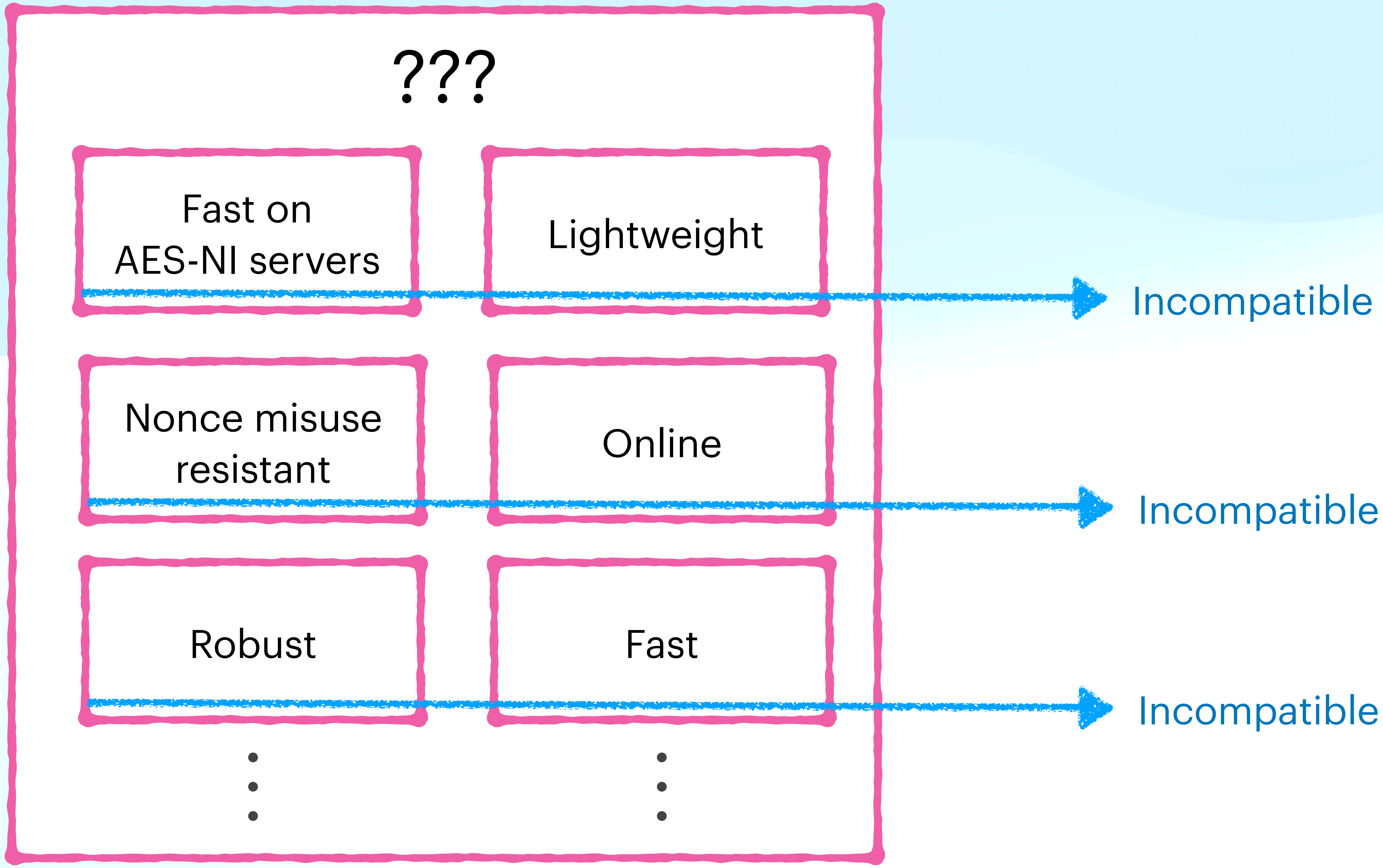
One scheme to rule them all?



One scheme to rule them all?



One scheme to rule them all?



Applications are asking for more features.

But these features are incompatible.

We cannot build one scheme with all these features.

Lots of different schemes

	Target hardware	Misuse resistant	Robust
AES—GCM	AES-NI	❌	❌
AES-GCM-SIV	AES-NI	✅	❌
Ascon	Lightweight	❌	❌
AES-AEZ	AES-NI	✅	✅

Lots of different schemes

	Target hardware	Misuse resistant	Robust
AES—GCM	AES-NI	❌	❌
AES-GCM-SIV	AES-NI	✅	❌
Ascon	Lightweight	❌	❌
AES-AEZ	AES-NI	✅	✅

Each of these schemes supports a different feature set.

Only getting more complicated. Ascon-SIV? Ascon-AEZ?

Up to developers to pick the most appropriate scheme.

Designing many different schemes scales poorly!

openssl / crypto /

Add file

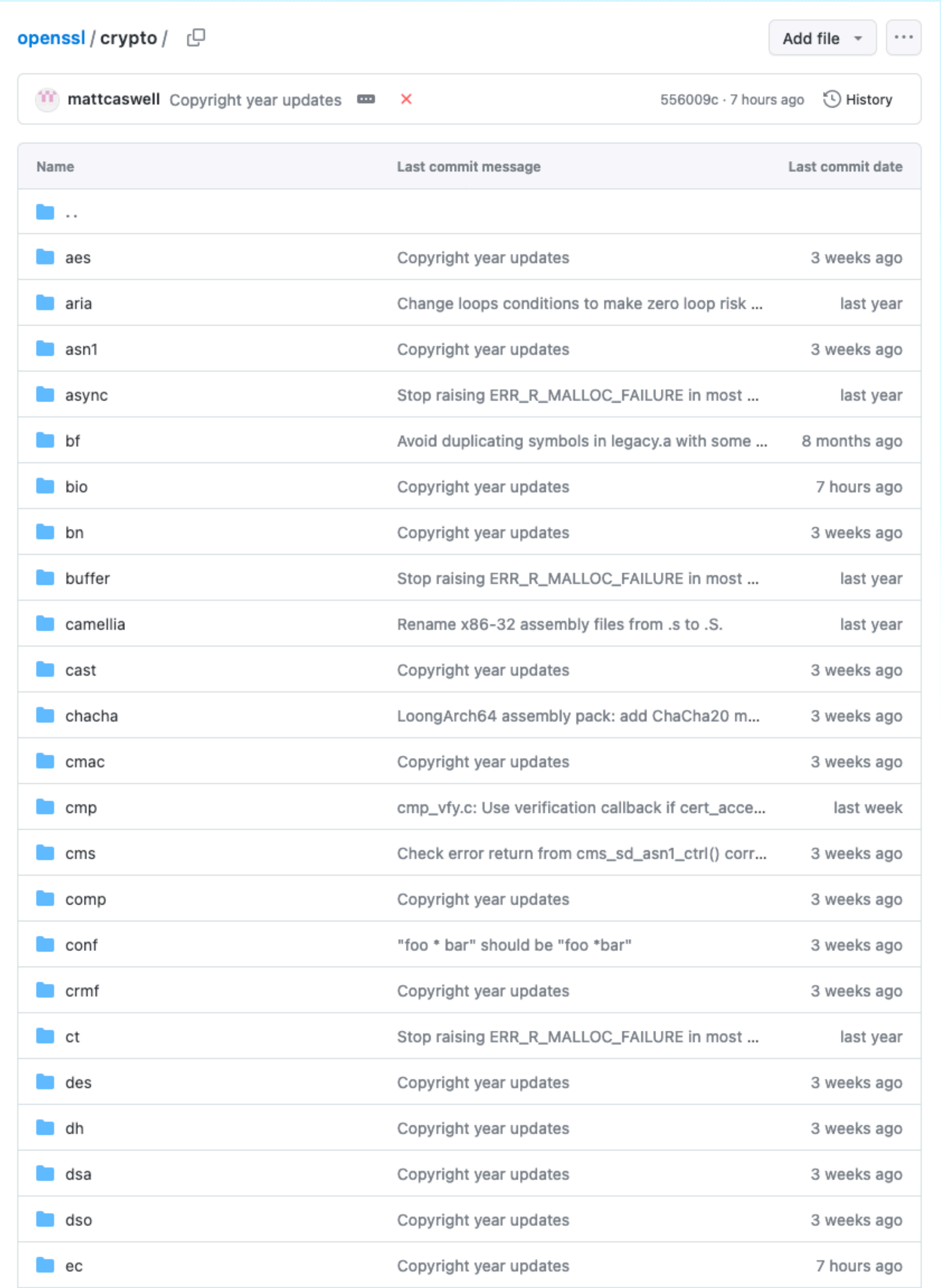
mattcaswell Copyright year updates 556009c · 7 hours ago History

Name	Last commit message	Last commit date
..		
aes	Copyright year updates	3 weeks ago
aria	Change loops conditions to make zero loop risk ...	last year
asn1	Copyright year updates	3 weeks ago
async	Stop raising ERR_R_MALLOC_FAILURE in most ...	last year
bf	Avoid duplicating symbols in legacy.a with some ...	8 months ago
bio	Copyright year updates	7 hours ago
bn	Copyright year updates	3 weeks ago
buffer	Stop raising ERR_R_MALLOC_FAILURE in most ...	last year
camellia	Rename x86-32 assembly files from .s to .S.	last year
cast	Copyright year updates	3 weeks ago
chacha	LoongArch64 assembly pack: add ChaCha20 m...	3 weeks ago
cmac	Copyright year updates	3 weeks ago
cmp	cmp_vfy.c: Use verification callback if cert_acce...	last week
cms	Check error return from cms_sd_asn1_ctrl() corr...	3 weeks ago
comp	Copyright year updates	3 weeks ago
conf	"foo * bar" should be "foo *bar"	3 weeks ago
crmf	Copyright year updates	3 weeks ago
ct	Stop raising ERR_R_MALLOC_FAILURE in most ...	last year
des	Copyright year updates	3 weeks ago
dh	Copyright year updates	3 weeks ago
dsa	Copyright year updates	3 weeks ago
dso	Copyright year updates	3 weeks ago
ec	Copyright year updates	7 hours ago

http	Remove repeated words	3 weeks ago
idea	Avoid duplicating symbols in legacy.a with some ...	8 months ago
kdf	Deprecate ERR_load_KDF_strings()	4 years ago
lhash	Copyright year updates	7 hours ago
md2	Avoid duplicating symbols in legacy.a with some ...	8 months ago
md4	Avoid duplicating symbols in legacy.a with some ...	8 months ago
md5	Copyright year updates	3 weeks ago
mdc2	Avoid duplicating symbols in legacy.a with some ...	8 months ago
modes	Copyright year updates	3 weeks ago
objects	Copyright year updates	3 weeks ago
ocsp	Copyright year updates	3 weeks ago
pem	Copyright year updates	7 hours ago
perlasm	Copyright year updates	3 weeks ago
pkcs12	Copyright year updates	7 hours ago
pkcs7	Copyright year updates	3 weeks ago
poly1305	Copyright year updates	3 weeks ago
property	Copyright year updates	3 weeks ago
rand	Copyright year updates	7 hours ago
rc2	Copyright year updates	3 weeks ago
rc4	Copyright year updates	3 weeks ago
rc5	Copyright year updates	3 weeks ago
ripemd	Avoid duplicating symbols in legacy.a with some ...	8 months ago
rsa	Fix a possible memleak in rsa_pub_encode	3 weeks ago
seed	Avoid duplicating symbols in legacy.a with some ...	8 months ago
sha	Copyright year updates	7 hours ago
siphash	crypto/*: Fix various typos, repeated words, alig...	last year
sm2	Copyright year updates	3 weeks ago
sm3	Copyright year updates	3 weeks ago

Subset of folders in https://github.com/openssl/openssl/tree/219bd6ac7061c40bd24f896f8652994d62d109de/crypto

Designing many different schemes scales poorly!

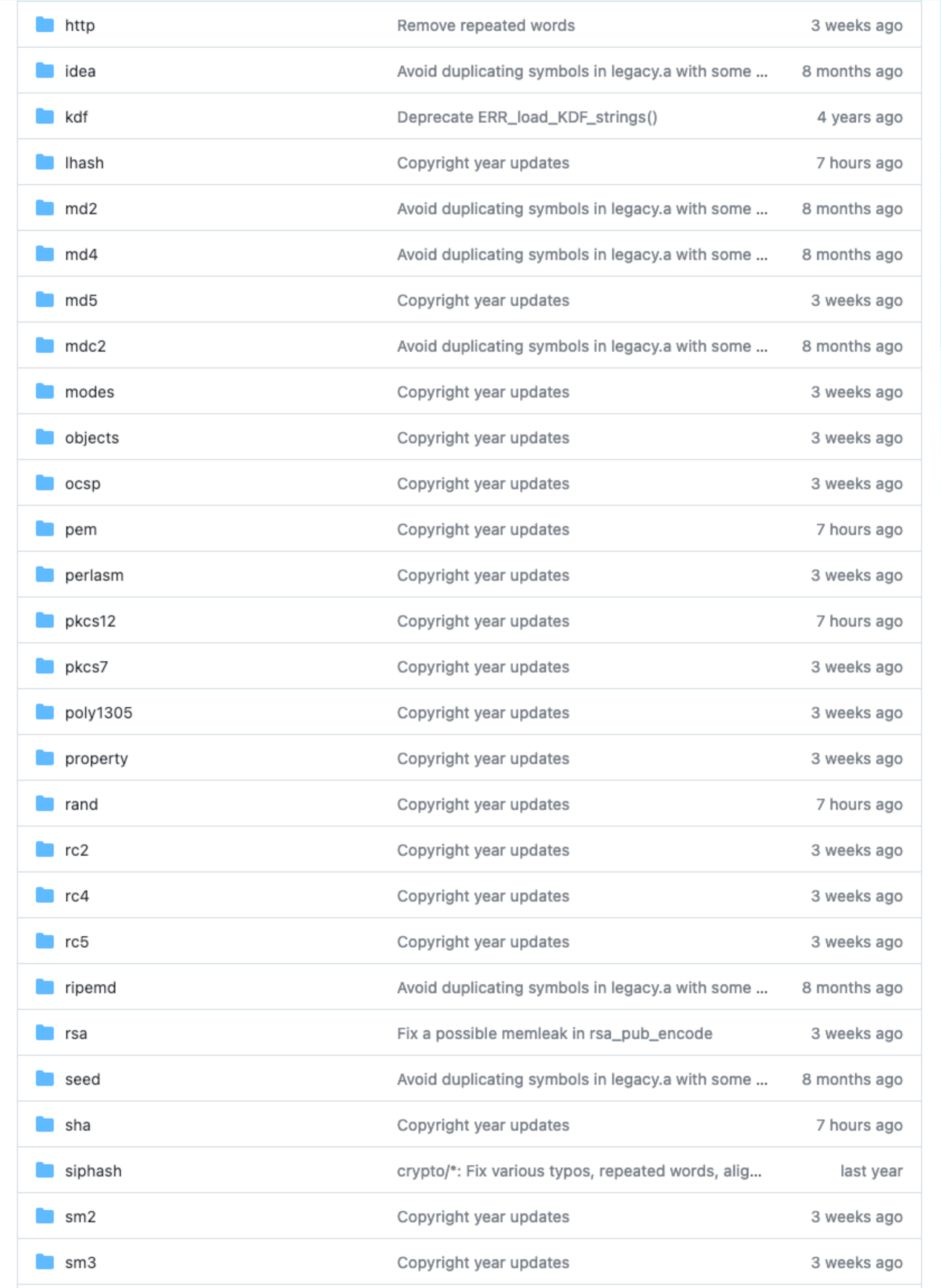


openssl / crypto /

Add file

mattcaswell Copyright year updates · 556009c · 7 hours ago · History

Name	Last commit message	Last commit date
..		
aes	Copyright year updates	3 weeks ago
aria	Change loops conditions to make zero loop risk ...	last year
asn1	Copyright year updates	3 weeks ago
async	Stop raising ERR_R_MALLOC_FAILURE in most ...	last year
bf	Avoid duplicating symbols in legacy.a with some ...	8 months ago
bio	Copyright year updates	7 hours ago
bn	Copyright year updates	3 weeks ago
buffer	Stop raising ERR_R_MALLOC_FAILURE in most ...	last year
camellia	Rename x86-32 assembly files from .s to .S.	last year
cast	Copyright year updates	3 weeks ago
chacha	LoongArch64 assembly pack: add ChaCha20 m...	3 weeks ago
cmac	Copyright year updates	3 weeks ago
cmp	cmp_vfy.c: Use verification callback if cert_acce...	last week
cms	Check error return from cms_sd_asn1_ctrl() corr...	3 weeks ago
comp	Copyright year updates	3 weeks ago
conf	"foo * bar" should be "foo *bar"	3 weeks ago
crmf	Copyright year updates	3 weeks ago
ct	Stop raising ERR_R_MALLOC_FAILURE in most ...	last year
des	Copyright year updates	3 weeks ago
dh	Copyright year updates	3 weeks ago
dsa	Copyright year updates	3 weeks ago
dso	Copyright year updates	3 weeks ago
ec	Copyright year updates	7 hours ago



http	Remove repeated words	3 weeks ago
idea	Avoid duplicating symbols in legacy.a with some ...	8 months ago
kdf	Deprecate ERR_load_KDF_strings()	4 years ago
lhash	Copyright year updates	7 hours ago
md2	Avoid duplicating symbols in legacy.a with some ...	8 months ago
md4	Avoid duplicating symbols in legacy.a with some ...	8 months ago
md5	Copyright year updates	3 weeks ago
mdc2	Avoid duplicating symbols in legacy.a with some ...	8 months ago
modes	Copyright year updates	3 weeks ago
objects	Copyright year updates	3 weeks ago
ocsp	Copyright year updates	3 weeks ago
pem	Copyright year updates	7 hours ago
perlasm	Copyright year updates	3 weeks ago
pkcs12	Copyright year updates	7 hours ago
pkcs7	Copyright year updates	3 weeks ago
poly1305	Copyright year updates	3 weeks ago
property	Copyright year updates	3 weeks ago
rand	Copyright year updates	7 hours ago
rc2	Copyright year updates	3 weeks ago
rc4	Copyright year updates	3 weeks ago
rc5	Copyright year updates	3 weeks ago
ripemd	Avoid duplicating symbols in legacy.a with some ...	8 months ago
rsa	Fix a possible memleak in rsa_pub_encode	3 weeks ago
seed	Avoid duplicating symbols in legacy.a with some ...	8 months ago
sha	Copyright year updates	7 hours ago
siphash	crypto/*: Fix various typos, repeated words, alig...	last year
sm2	Copyright year updates	3 weeks ago
sm3	Copyright year updates	3 weeks ago

Libraries are going to get even more complicated.

Need to write a new standard for each new scheme.

Need to analyze each scheme independently.

Subset of folders in https://github.com/openssl/openssl/tree/219bd6ac7061c40bd24f896f8652994d62d109de/crypto

Choosing an AEAD in BoringSSL

```
auto aead = EVP_aead_aes_128_gcm();

// Or AES-GCM-SIV or XChaCha20/Poly1305 or CTR-HMAC or ...

auto ctx = EVP_AEAD_CTX_new(aead, key, tag_len);

EVP_AEAD_CTX_seal(ctx, out, nonce, in, ad); // Encryption

EVP_AEAD_CTX_open(ctx, out, nonce, in, ad); // Decryption

(Slightly simplified)
```

Goals for Flexible AEAD

Goals for Flexible AEAD

A. Minimize library complexity.

Goals for Flexible AEAD

A. Minimize library complexity.

B. Simplify analysis.

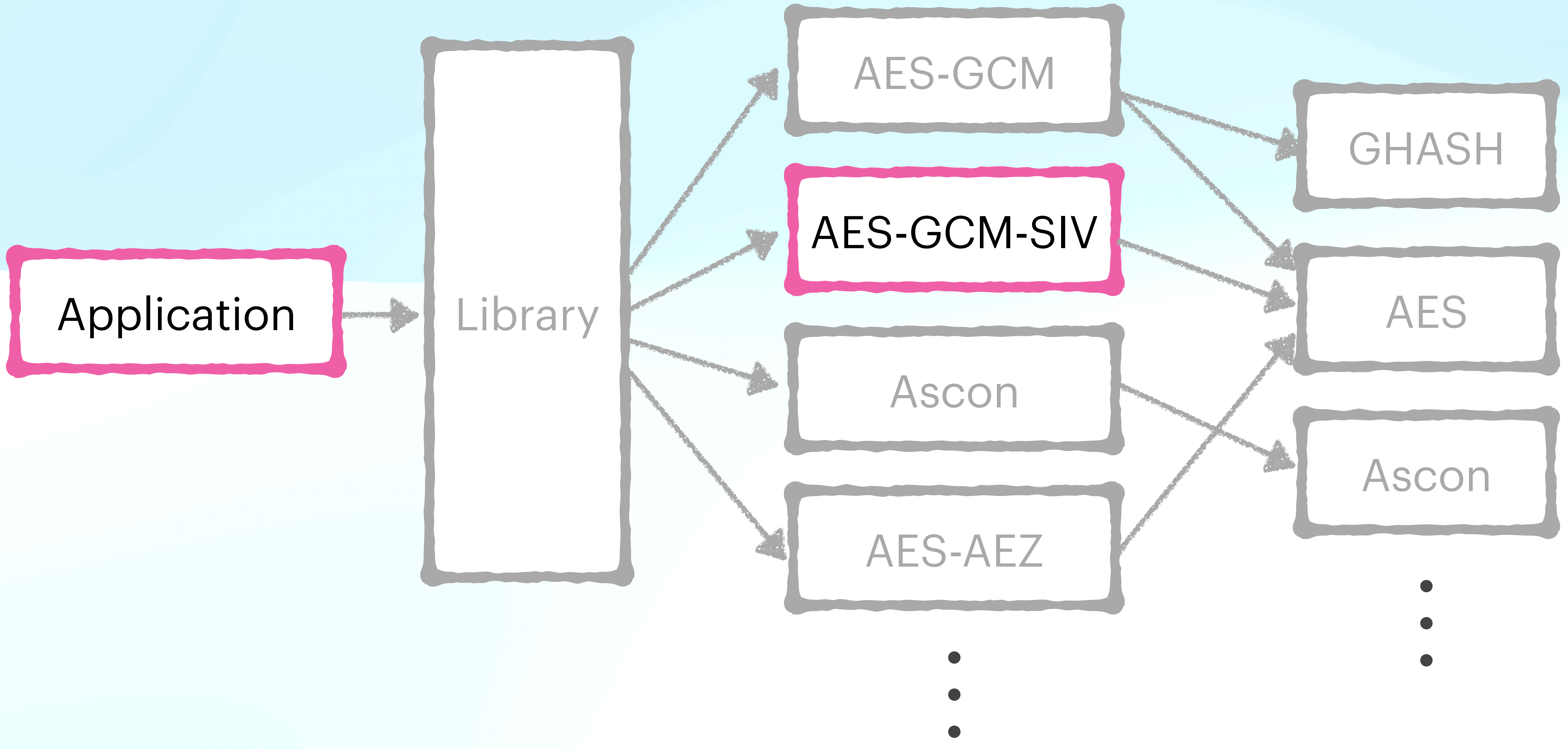
Goals for Flexible AEAD

A. Minimize library complexity.

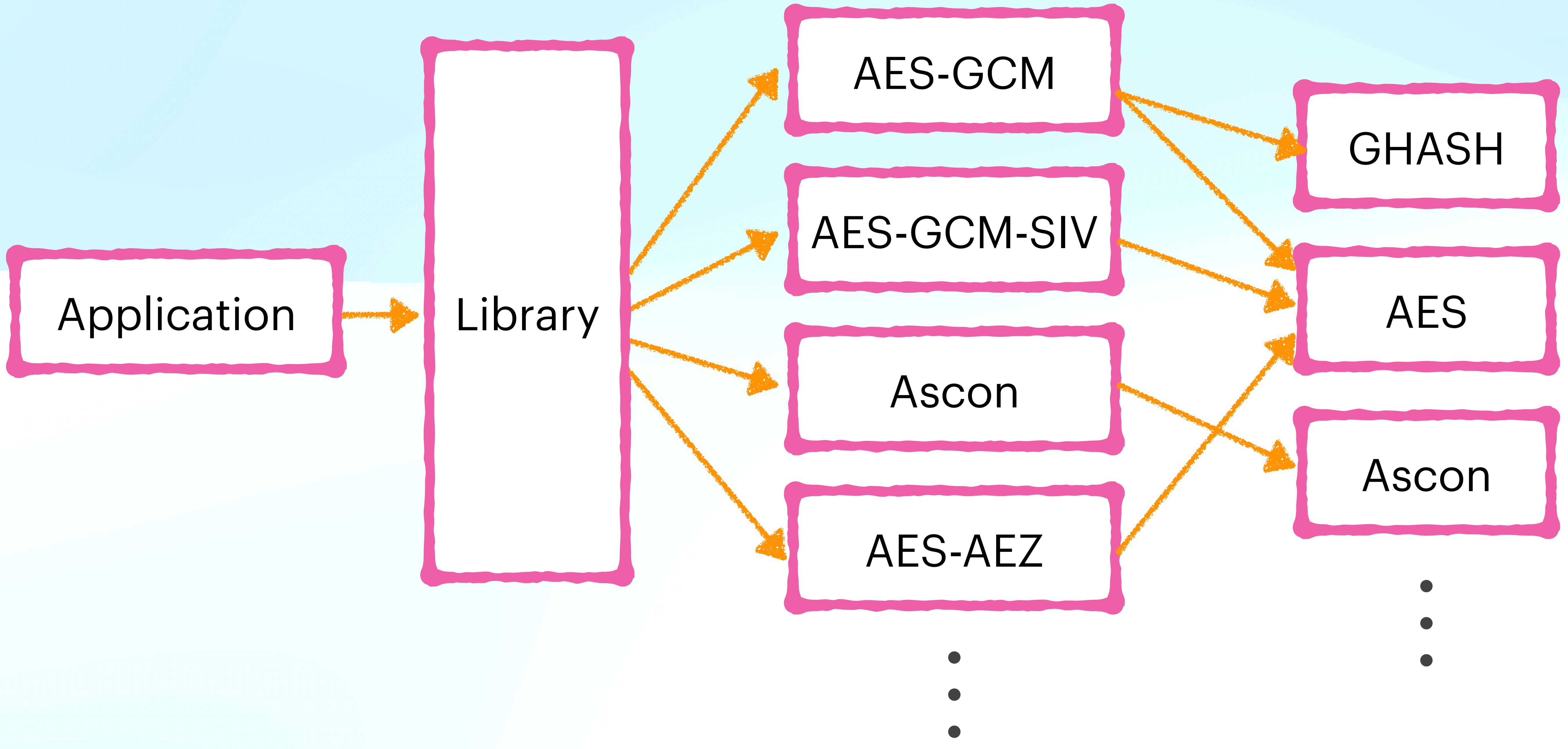
B. Simplify analysis.

C. Easy-to-use APIs.

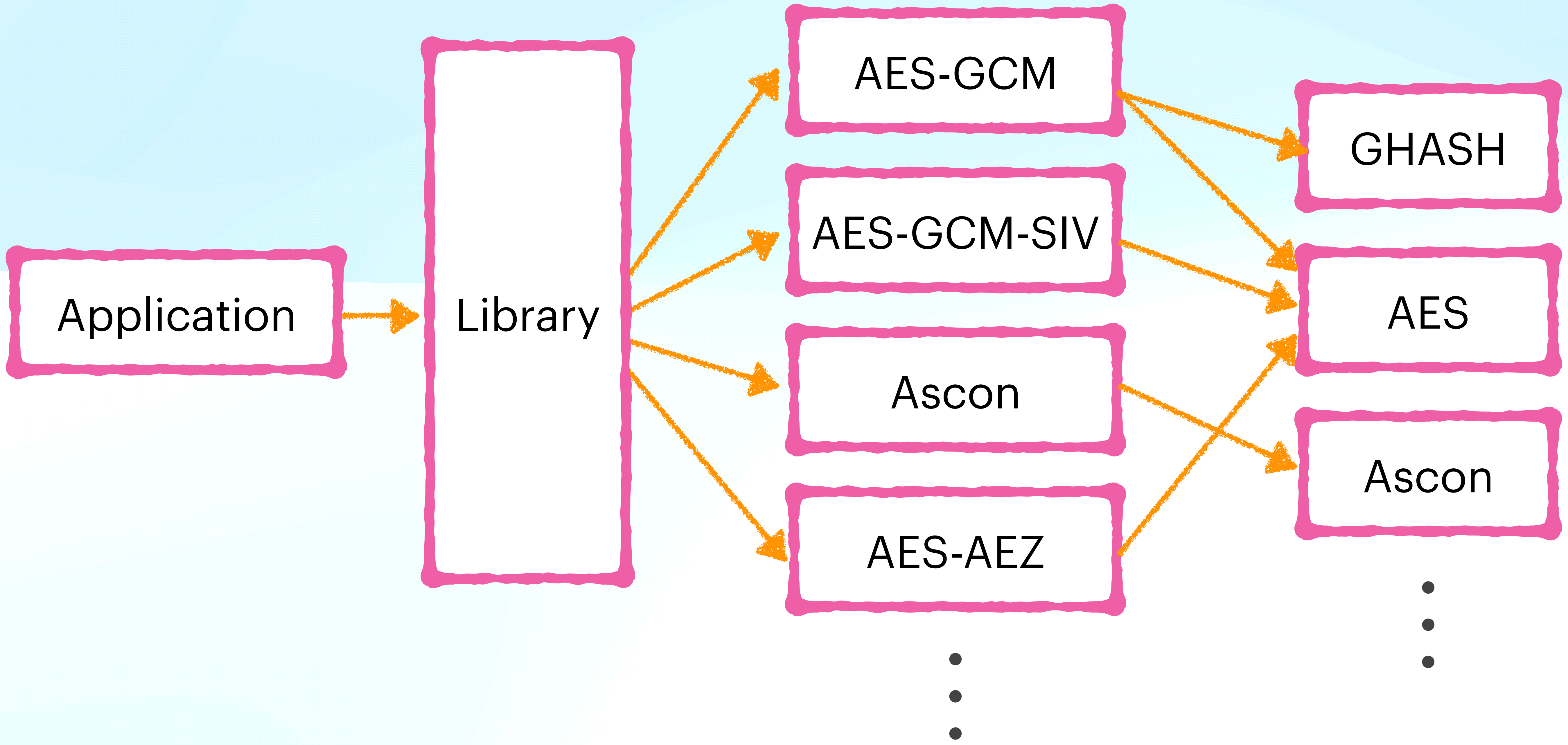
Real world AEAD implementations



Real world AEAD implementations



Real world AEAD implementations

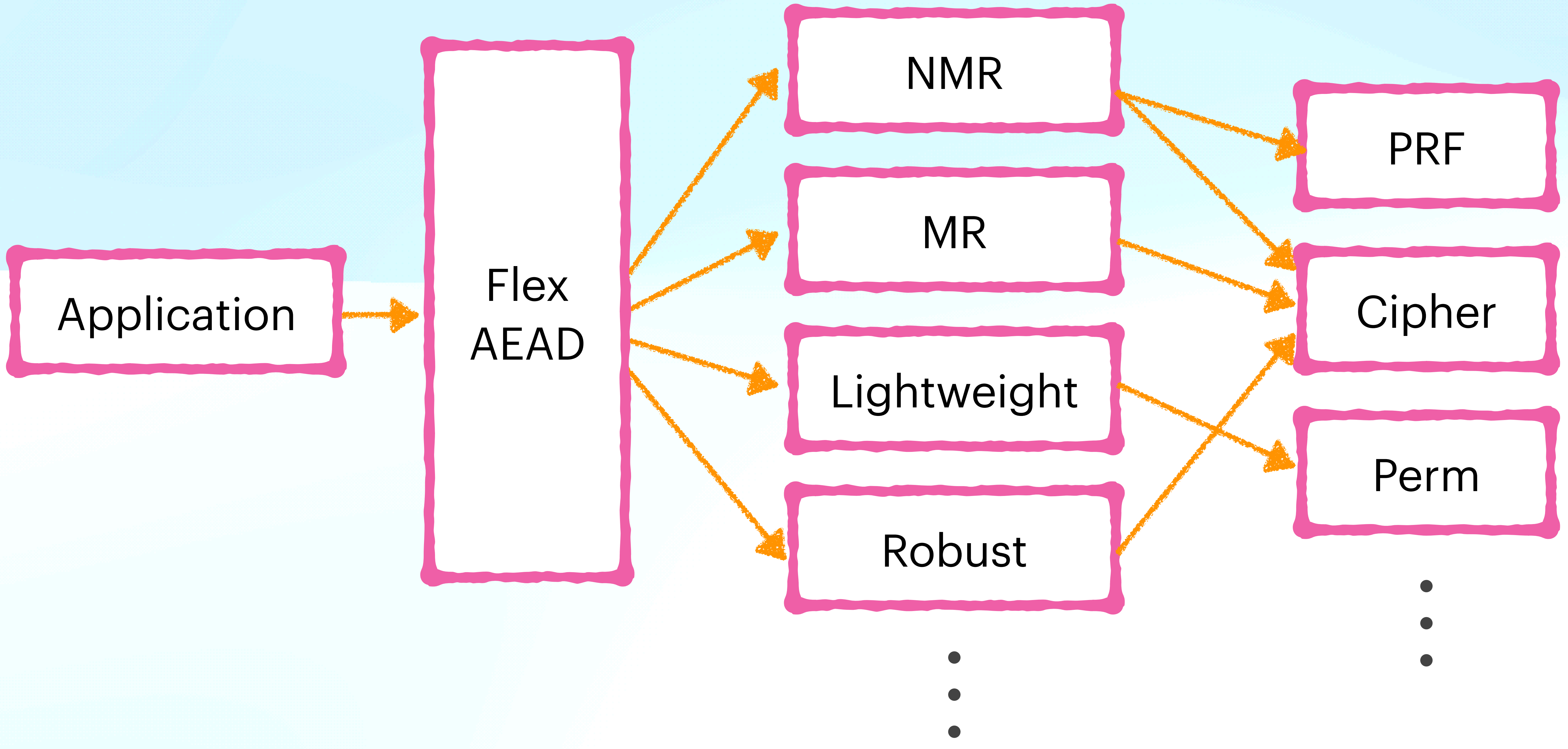


Applications use libraries not standalone schemes.

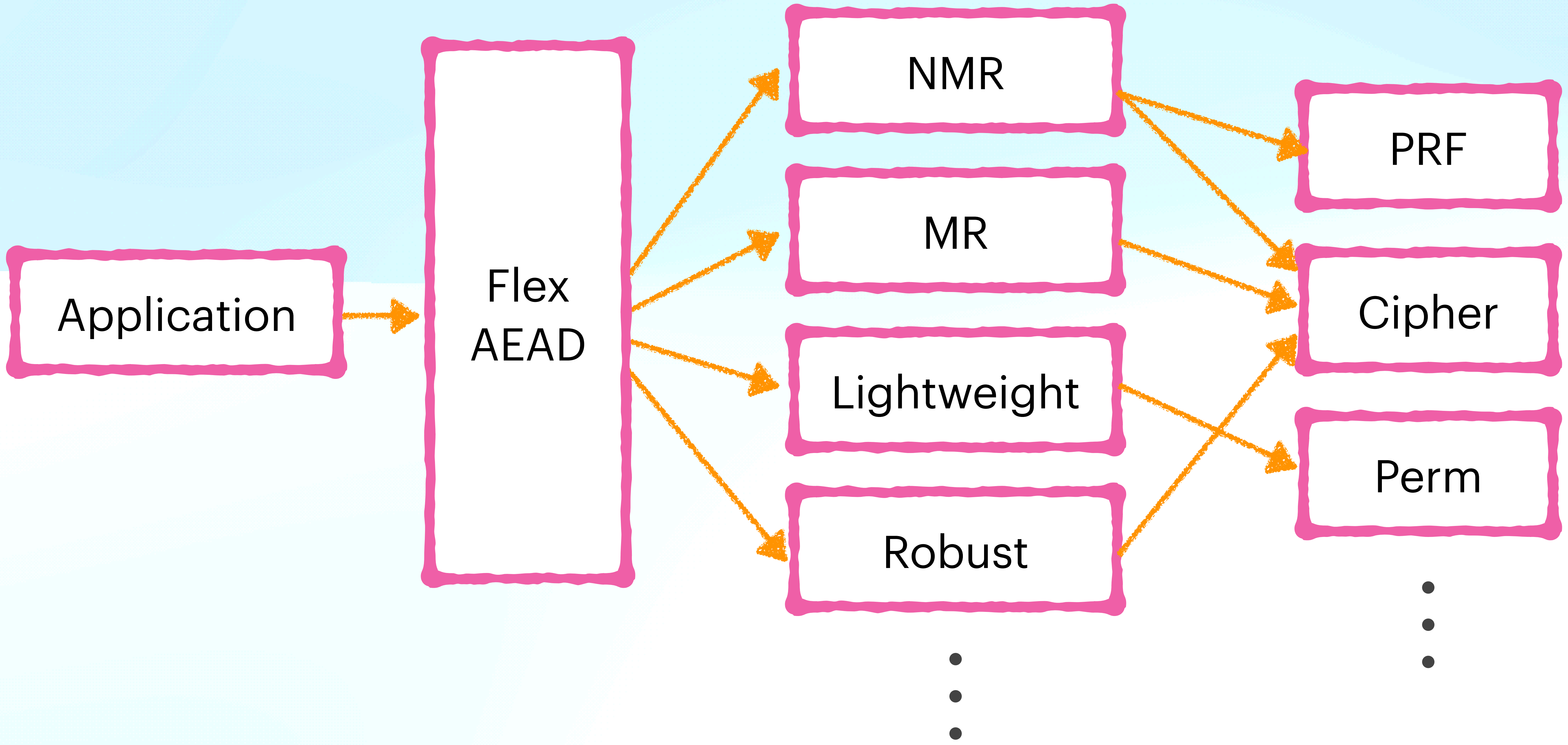
Applications use many AEAD schemes, not just the one scheme.

Libraries implement schemes using common components.

Formalizing real world AEAD implementations



Formalizing real world AEAD implementations



Reduce the number of components.

Modularly analyze the components.

Misuse resistant APIs and safe defaults.

Choosing an AEAD with Flexible AEAD

```
auto config = { mr: true, rob: false, hardware: aes-ni };

auto aead = aead_from_config(config);

auto ctx = EVP_AEAD_CTX_new(aead, key, tag_len);

EVP_AEAD_CTX_seal(ctx, out, nonce, in, ad); // Encryption

EVP_AEAD_CTX_open(ctx, out, nonce, in, ad); // Decryption
```

(Slightly simplified, reimagined from BoringSSL)

What is a configuration?

`config.nonce_hiding`	✅	❌
`config.misuse_resistance`	✅	❌
`config.key_length`	128	256
`config.target_hardware`	AES-NI	Lightweight
⋮	⋮	⋮

What is a configuration?

`config.nonce_hiding`	✅	❌
`config.misuse_resistance`	✅	❌
`config.key_length`	128	256
`config.target_hardware`	AES-NI	Lightweight
⋮	⋮	⋮

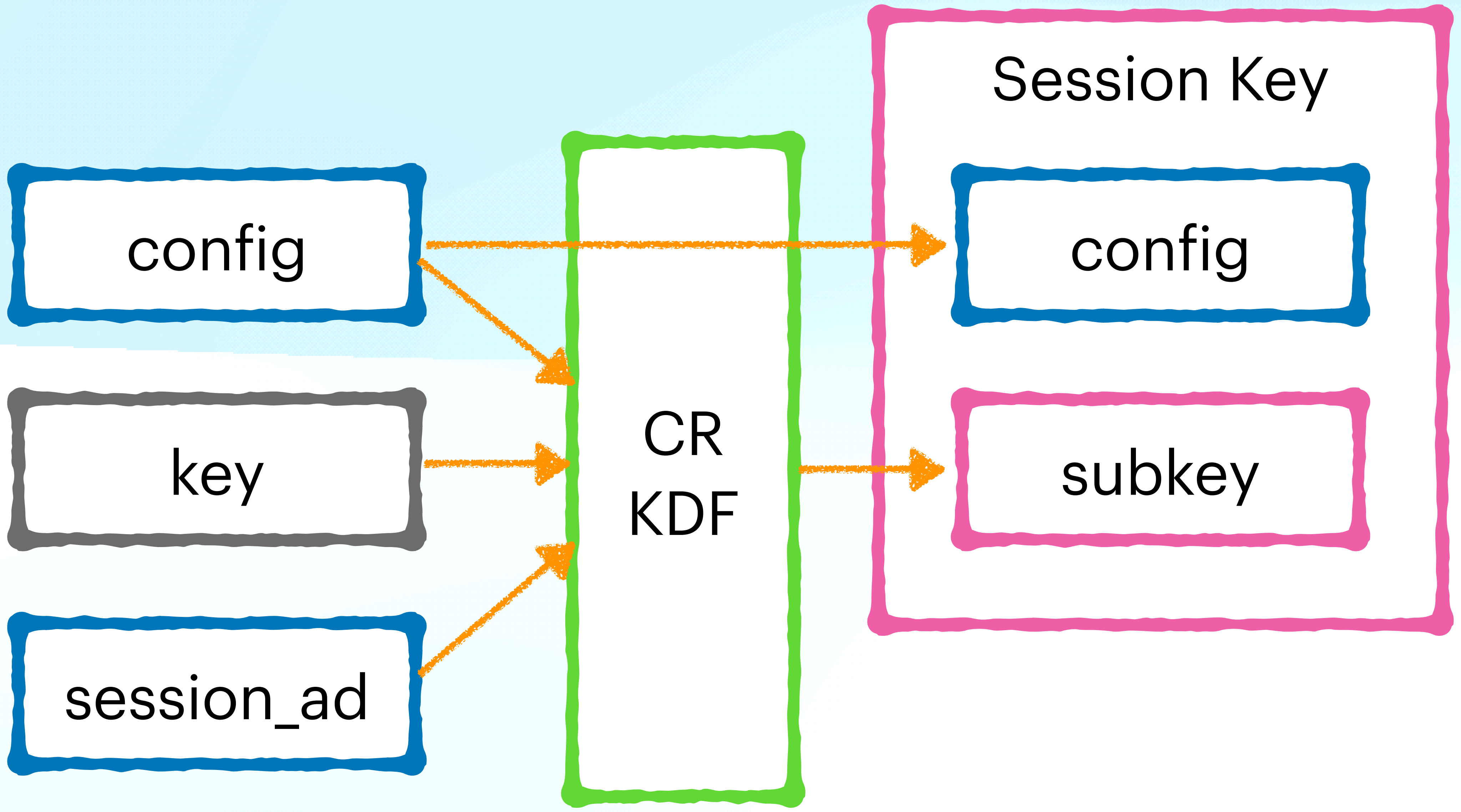
Encodes desired features as a dictionary.

Default to safer choices.

Tooling to generate configs.

Tooling to verify configs.

Implementing configurations



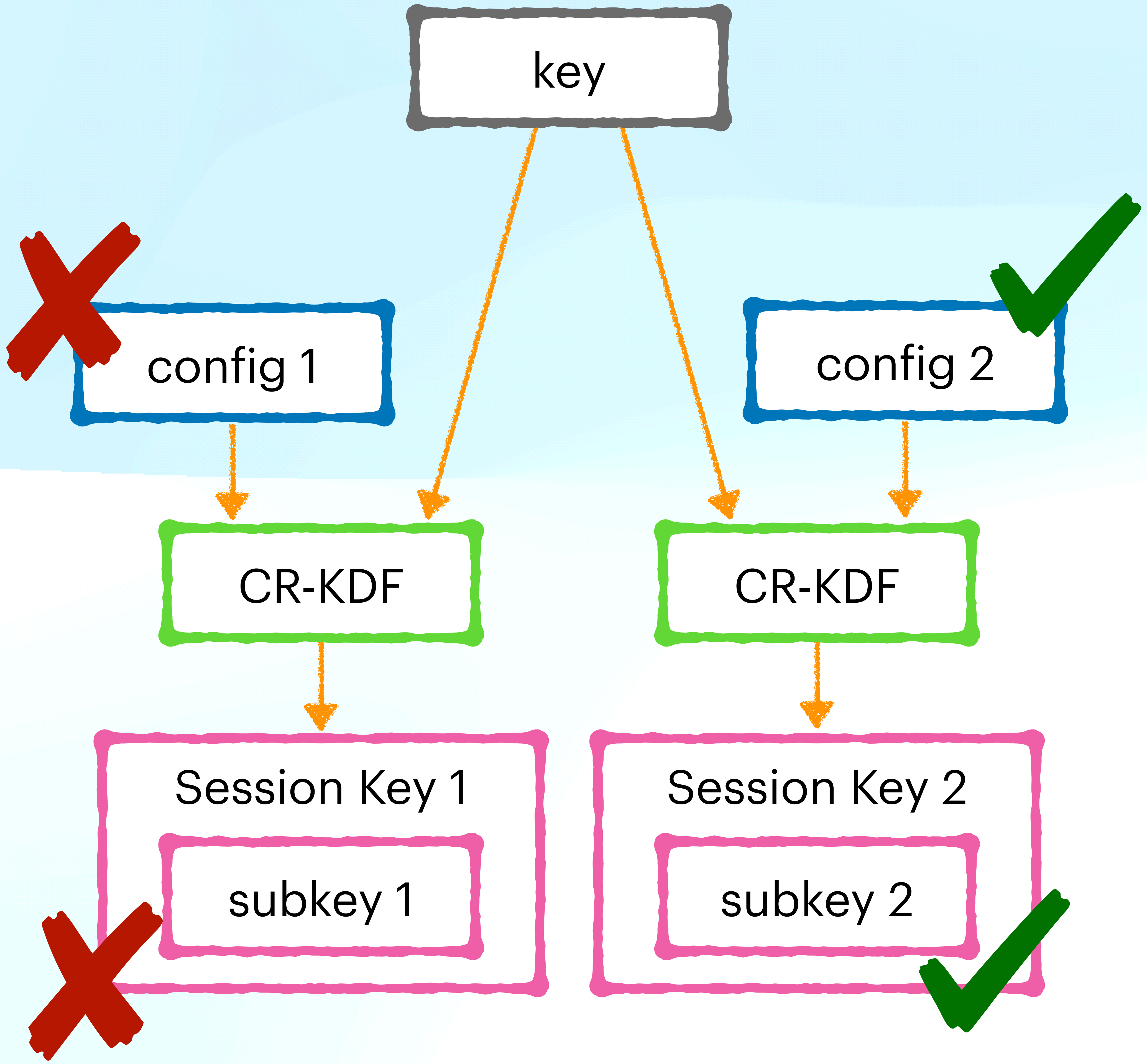
Session key encodes the config and a CR-KDF of the key, config, and session AD.

Supports session ADs by default.

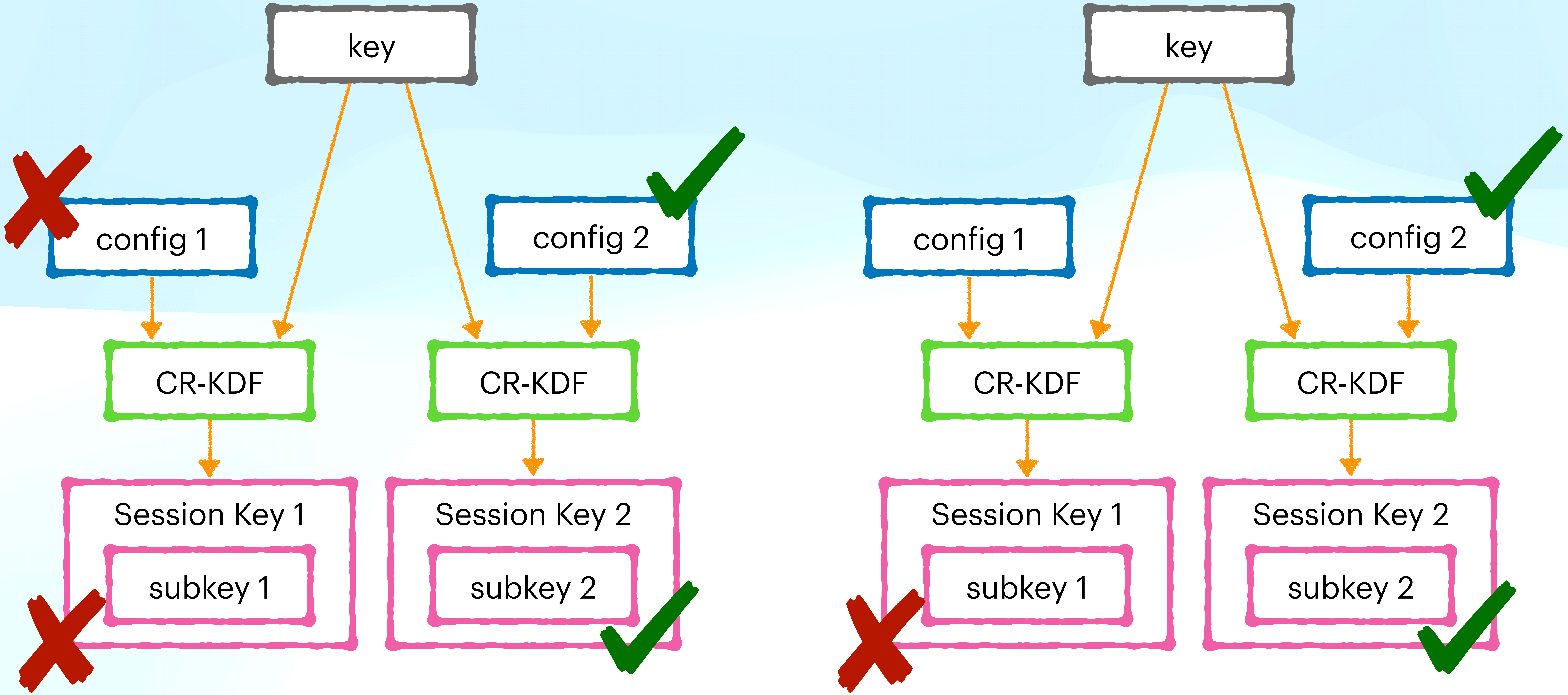
Can safely reuse key material across configs and sessions.

The session key is not exportable.

Gracefully handles broken configs and leaked keys



Gracefully handles broken configs and leaked keys



Public permutations: a natural starting point

We can build all of symmetric cryptography from a permutation.

Public permutations: a natural starting point

We can build all of symmetric cryptography from a permutation.

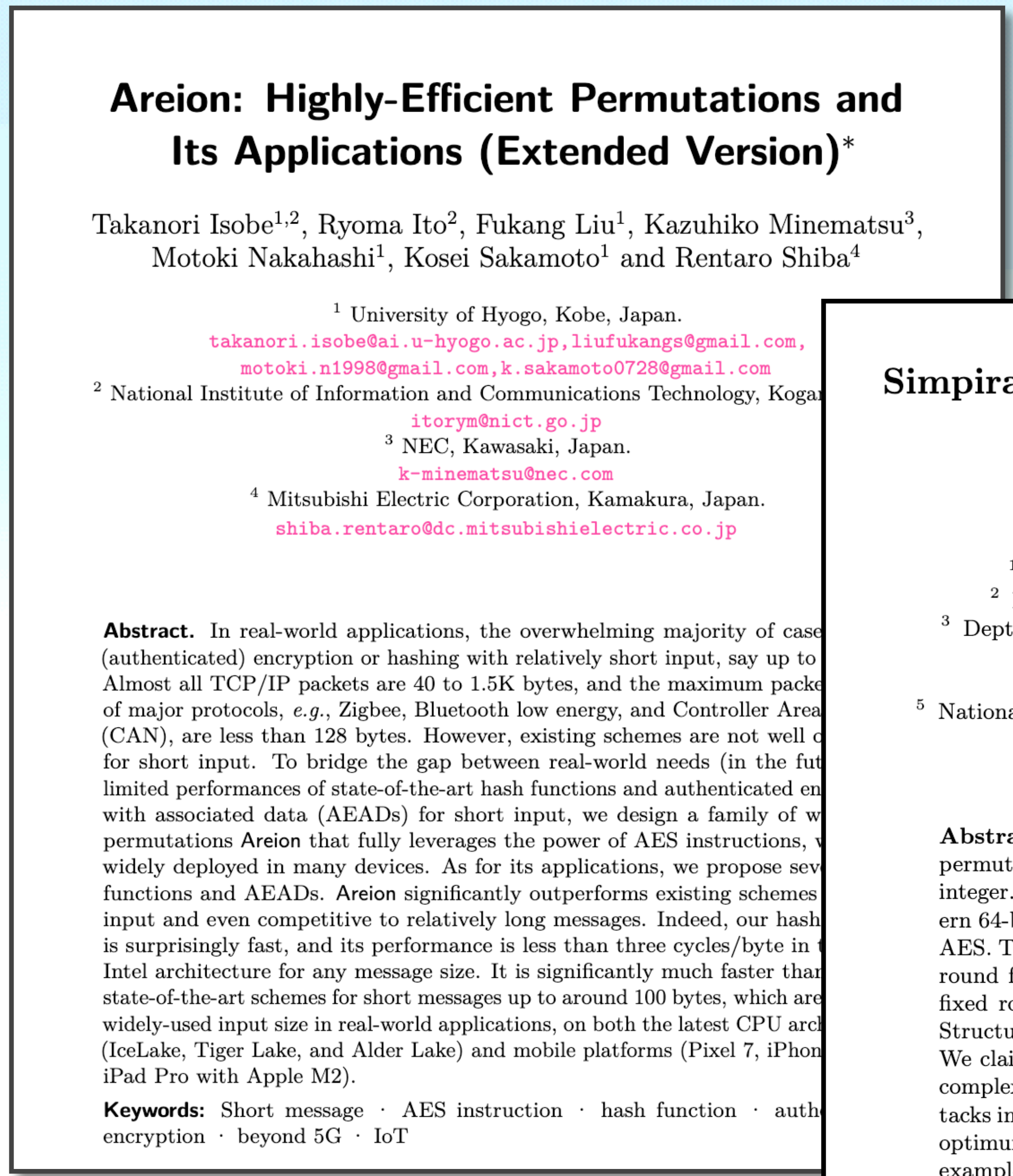
SHA3 and Ascon are based on permutations.

Public permutations: a natural starting point

We can build all of symmetric cryptography from a permutation.

SHA3 and Ascon are based on permutations.

Recent work on building permutations using AES-NI instructions.



Areion: Highly-Efficient Permutations and Its Applications (Extended Version)*

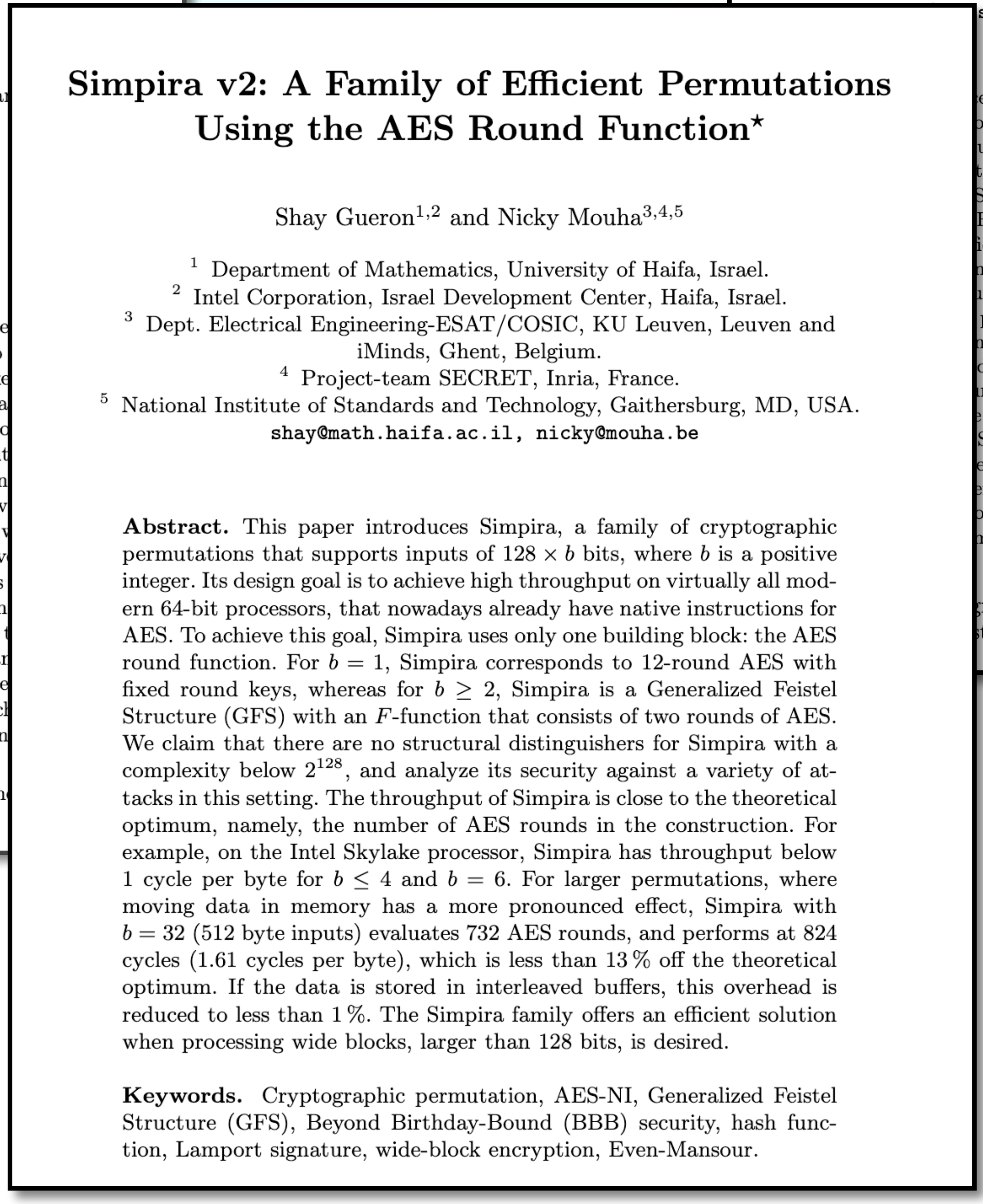
Takanori Isobe[1,2], Ryoma Ito[2], Fukang Liu[1], Kazuhiko Minematsu[3], Motoki Nakahashi[1], Kosei Sakamoto[1] and Rentaro Shiba[4]

[1] University of Hyogo, Kobe, Japan.
takanori.isobe@ai.u-hyogo.ac.jp, liufukangs@gmail.com, motoki.n1998@gmail.com, k.sakamoto0728@gmail.com
[2] National Institute of Information and Communications Technology, Kogan
itorym@nict.go.jp
[3] NEC, Kawasaki, Japan.
k-minematsu@nec.com
[4] Mitsubishi Electric Corporation, Kamakura, Japan.
shiba.rentaro@dc.mitsubishielectric.co.jp

Abstract. In real-world applications, the overwhelming majority of case (authenticated) encryption or hashing with relatively short input, say up to Almost all TCP/IP packets are 40 to 1.5K bytes, and the maximum packe of major protocols, *e.g.*, Zigbee, Bluetooth low energy, and Controller Area (CAN), are less than 128 bytes. However, existing schemes are not well c for short input. To bridge the gap between real-world needs (in the fut limited performances of state-of-the-art hash functions and authenticated en with associated data (AEADs) for short input, we design a family of w permutations Areion that fully leverages the power of AES instructions, w widely deployed in many devices. As for its applications, we propose sev functions and AEADs. Areion significantly outperforms existing schemes input and even competitive to relatively long messages. Indeed, our hash is surprisingly fast, and its performance is less than three cycles/byte in t Intel architecture for any message size. It is significantly much faster than state-of-the-art schemes for short messages up to around 100 bytes, which are widely-used input size in real-world applications, on both the latest CPU arch (IceLake, Tiger Lake, and Alder Lake) and mobile platforms (Pixel 7, iPhon iPad Pro with Apple M2).

Keywords: Short message · AES instruction · hash function · auth encryption · beyond 5G · IoT

ia.cr/2023/794



Simpira v2: A Family of Efficient Permutations Using the AES Round Function*

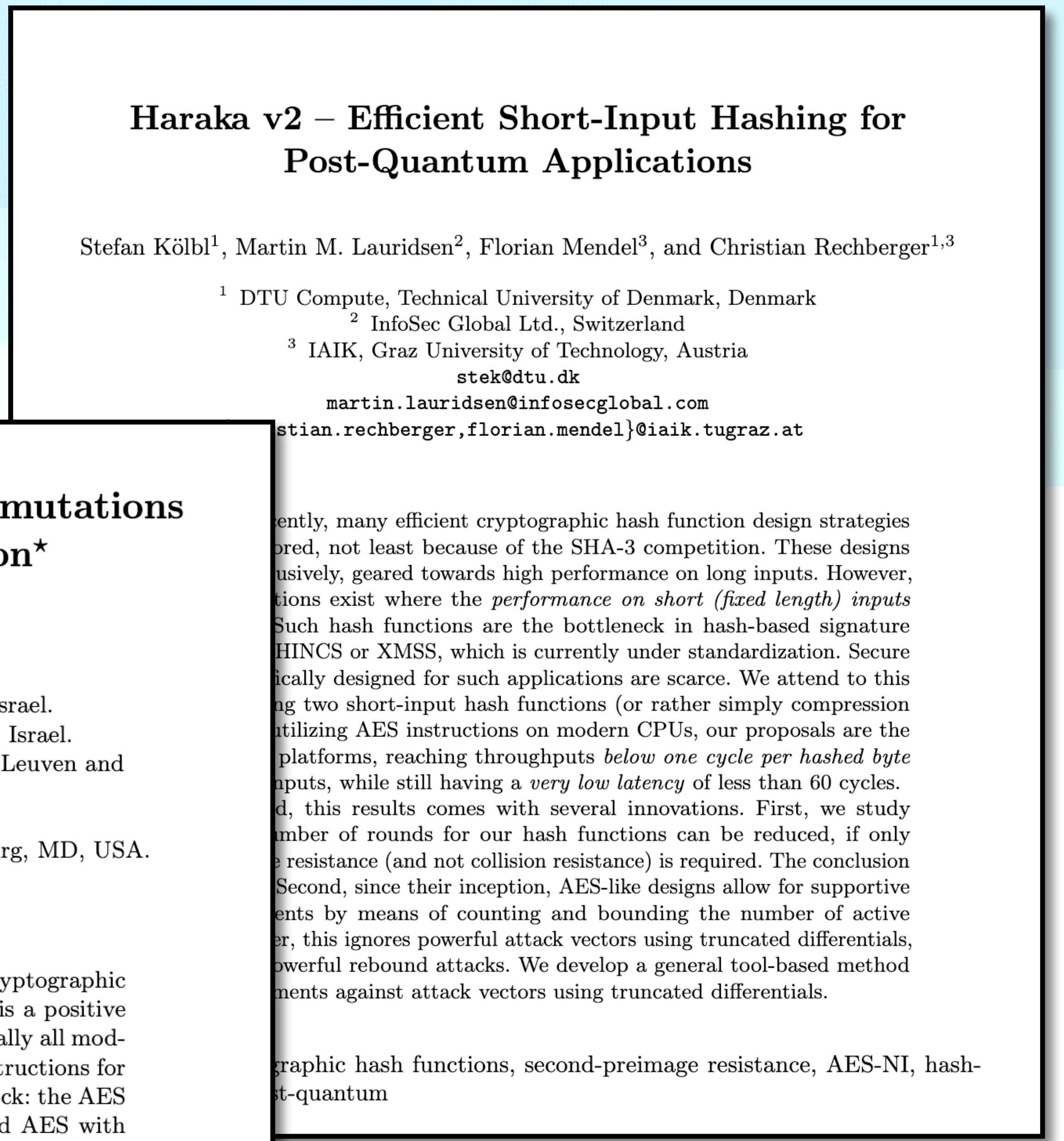
Shay Gueron[1,2] and Nicky Mouha[3,4,5]

[1] Department of Mathematics, University of Haifa, Israel.
[2] Intel Corporation, Israel Development Center, Haifa, Israel.
[3] Dept. Electrical Engineering-ESAT/COSIC, KU Leuven, Leuven and iMinds, Ghent, Belgium.
[4] Project-team SECRET, Inria, France.
[5] National Institute of Standards and Technology, Gaithersburg, MD, USA.
shay@math.haifa.ac.il, nicky@mouha.be

Abstract. This paper introduces Simpira, a family of cryptographic permutations that supports inputs of $128 \times b$ bits, where b is a positive integer. Its design goal is to achieve high throughput on virtually all modern 64-bit processors, that nowadays already have native instructions for AES. To achieve this goal, Simpira uses only one building block: the AES round function. For $b = 1$, Simpira corresponds to 12-round AES with fixed round keys, whereas for $b \geq 2$, Simpira is a Generalized Feistel Structure (GFS) with an F-function that consists of two rounds of AES. We claim that there are no structural distinguishers for Simpira with a complexity below 2^{128}, and analyze its security against a variety of attacks in this setting. The throughput of Simpira is close to the theoretical optimum, namely, the number of AES rounds in the construction. For example, on the Intel Skylake processor, Simpira has throughput below 1 cycle per byte for $b \leq 4$ and $b = 6$. For larger permutations, where moving data in memory has a more pronounced effect, Simpira with $b = 32$ (512 byte inputs) evaluates 732 AES rounds, and performs at 824 cycles (1.61 cycles per byte), which is less than 13 % off the theoretical optimum. If the data is stored in interleaved buffers, this overhead is reduced to less than 1 %. The Simpira family offers an efficient solution when processing wide blocks, larger than 128 bits, is desired.

Keywords. Cryptographic permutation, AES-NI, Generalized Feistel Structure (GFS), Beyond Birthday-Bound (BBB) security, hash function, Lamport signature, wide-block encryption, Even-Mansour.

ia.cr/2016/122



Haraka v2 – Efficient Short-Input Hashing for Post-Quantum Applications

Stefan Kölbl[1], Martin M. Lauridsen[2], Florian Mendel[3], and Christian Rechberger[1,3]

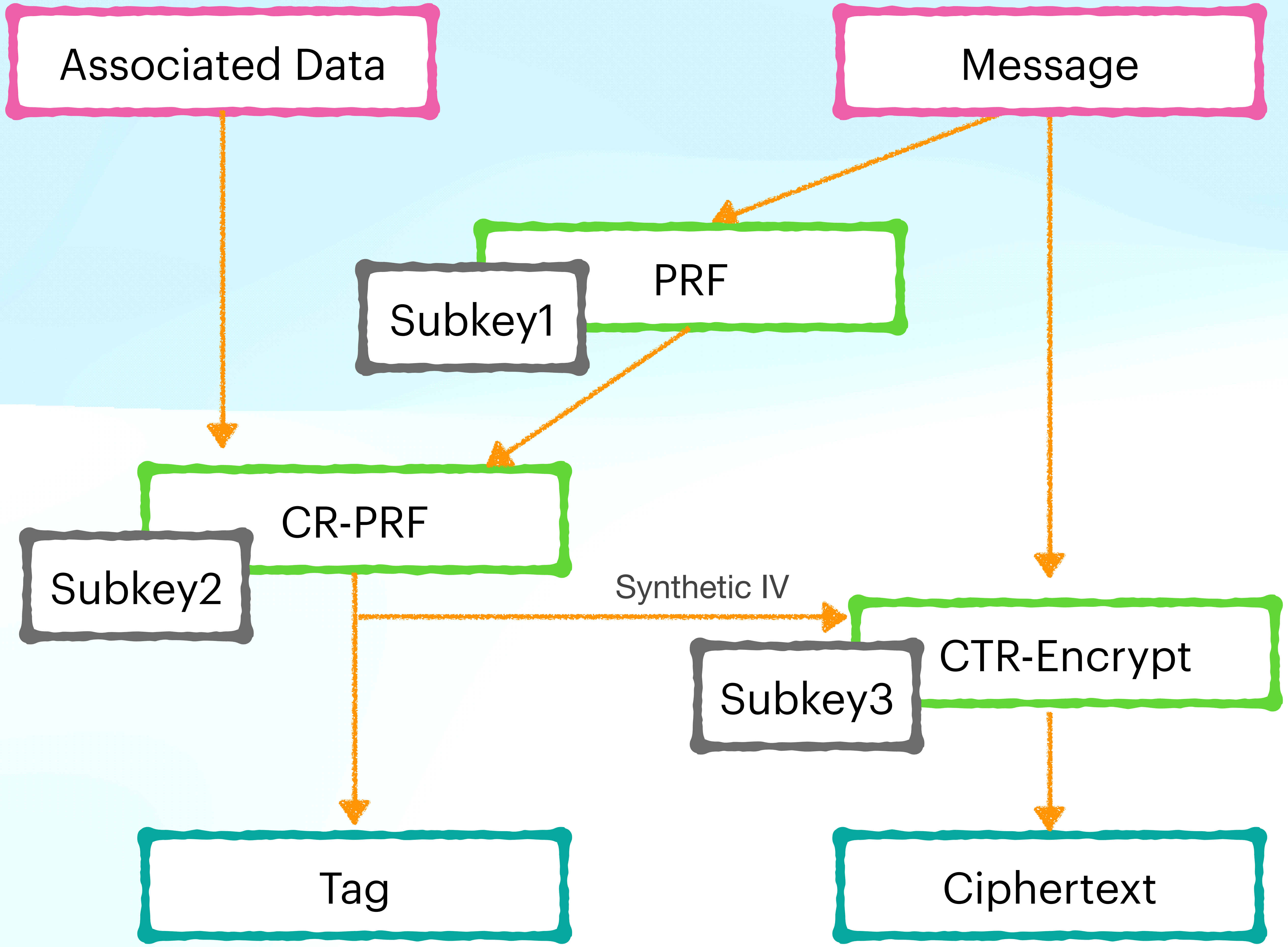
[1] DTU Compute, Technical University of Denmark, Denmark
[2] InfoSec Global Ltd., Switzerland
[3] IAIK, Graz University of Technology, Austria
stek@dtu.dk
martin.lauridsen@infosecglobal.com
stian.rechberger,florian.mendel}@iaik.tugraz.at

ently, many efficient cryptographic hash function design strategies ored, not least because of the SHA-3 competition. These designs usively, geared towards high performance on long inputs. However, tions exist where the *performance on short (fixed length) inputs* Such hash functions are the bottleneck in hash-based signature HINCS or XMSS, which is currently under standardization. Secure ically designed for such applications are scarce. We attend to this ng two short-input hash functions (or rather simply compression tilizing AES instructions on modern CPUs, our proposals are the platforms, reaching throughputs *below one cycle per hashed byte* nputs, while still having a *very low latency* of less than 60 cycles. d, this results comes with several innovations. First, we study mber of rounds for our hash functions can be reduced, if only resistance (and not collision resistance) is required. The conclusion Second, since their inception, AES-like designs allow for supportive ents by means of counting and bounding the number of active r, this ignores powerful attack vectors using truncated differentials, owerful rebound attacks. We develop a general tool-based method ments against attack vectors using truncated differentials.

raphic hash functions, second-preimage resistance, AES-NI, hash- t-quantum

ia.cr/2016/098

CIV: SIV-inspired MR context committing AEAD

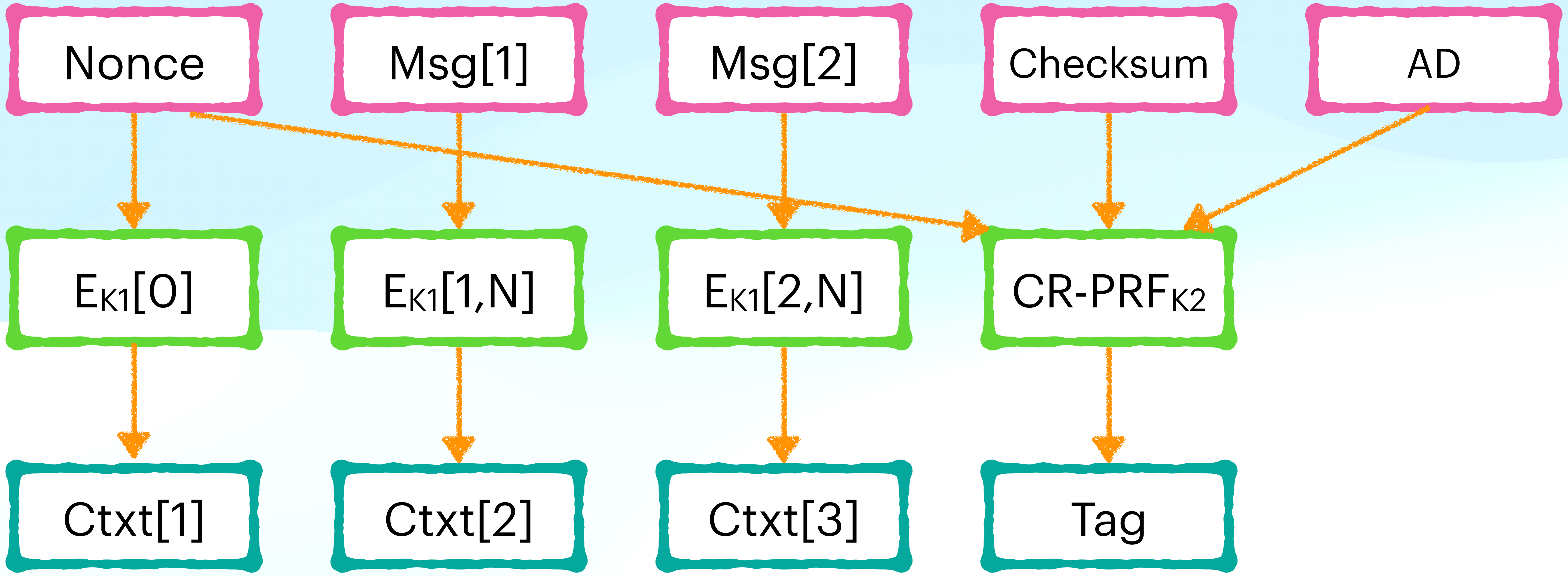


Build PRF, CR-PRF, and CTR from an underlying permutation.

Context committing and nonce misuse resistant.

Fast, competitive with AES-based misuse resistant schemes.

OCH: OCB3-inspired NMR context committing AEAD



Build TBC and CR-PRF from an underlying permutation.

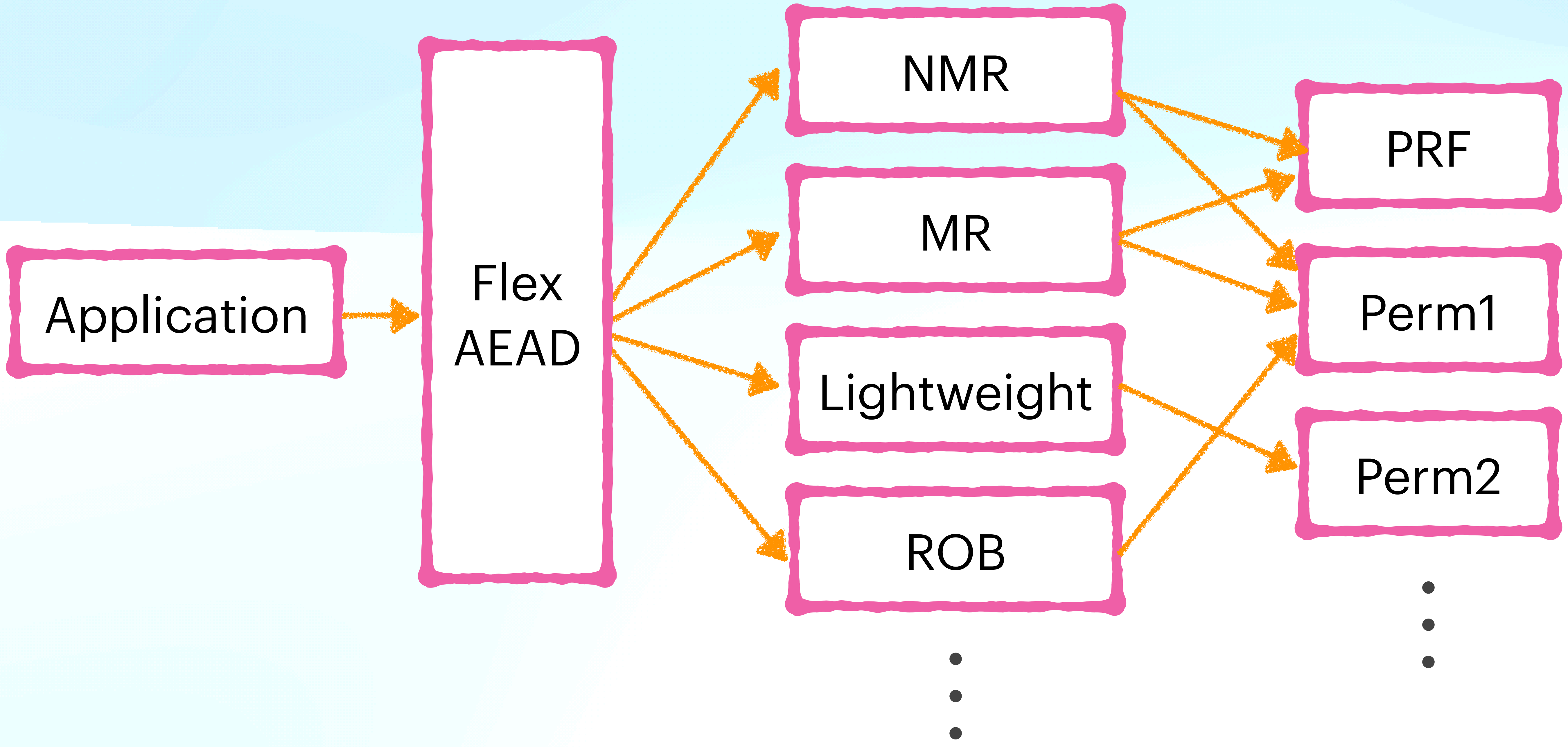
Context committing and nonce hiding.

Fast, competitive with AES-based AEAD schemes.

Len, Menda, Hoang, Bellare, and Ristenpart. The OCH Context-Committing AEAD Algorithm. Forthcoming.

Next Steps

Our Suggestion



What do y'all think of this?

Robust AEAD?
Compactly committing AEAD?

API design improvements?
Cross-language challenges?

Reach out!

snkth.com