MAC Reforgeability

John Black[1] and Martin Cochran[2]

Fast Software Encryption 2009

[1]University of Colorado, Boulder
[2]Google Inc.

Outline

- Problem setting - “reforgeability”
  - Appropriate scenarios
- Application to current MACs
- Propose new MAC with good tradeoffs
  - small tags
  - fast
  - flexible security
  - security reduction

Message Authentication: setting

- Alice and Bob share a secret key K
- Adversary Eve has access to communication channel
  - Can inject/modify messages
- Goal (informally): all adversarial modifications to channel are detectable

Message Authentication Codes (stateless)

- Append Tag = F(K, M) to each message M
- Eve should not be able to find new message M’ and Tag’ such that Tag’ = F(K, M’)

Message Authentication Codes (stateful)

- Append Tag = F(K, M, n) to each message M
- Eve should not be able to find new tuple (M', Tag', n') such that Tag' = F(K, M', n')

Current Options

- Essentially there are three types of MACs
  - Blockcipher based (CBC-MAC)
  - Compression-function based (HMAC)
  - Wegman-Carter based (Poly1305, VMAC)

Wegman-Carter

Let $\epsilon \in \mathbb{R}^+$ and fix a domain $\mathcal{D}$ and range $\mathcal{R}$. A finite multiset of hash functions $\mathcal{H} = \{h : \mathcal{D} \to \mathcal{R}\}$ is said to be ϵ-**Almost Universal** (ϵ-AU) if for every $x, y \in \mathcal{D}$ with $x \neq y$, $\Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq \epsilon$.

Building Blocks:

F_K

Fixed $h \in \mathcal{H}$

Wegman-Carter

Let $\epsilon \in \mathbb{R}^+$ and fix a domain $\mathcal{D}$ and range $\mathcal{R}$. A finite multiset of hash functions $\mathcal{H} = \{h : \mathcal{D} \to \mathcal{R}\}$ is said to be ϵ-**Almost Universal** (ϵ-AU) if for every $x, y \in \mathcal{D}$ with $x \neq y$, $\Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq \epsilon$.

Building Blocks:

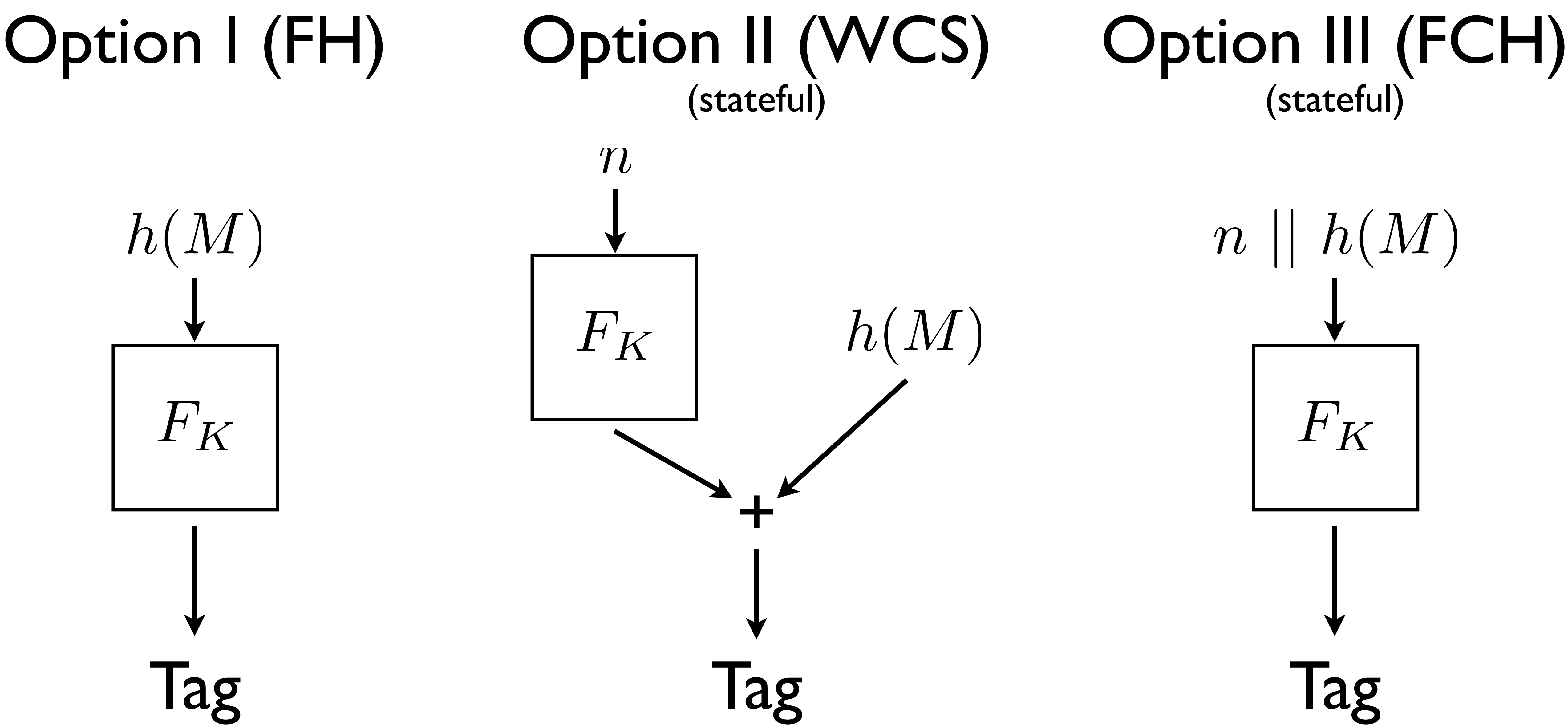
F_K

Fixed $h \in \mathcal{H}$

Key: $\{K, h\}$

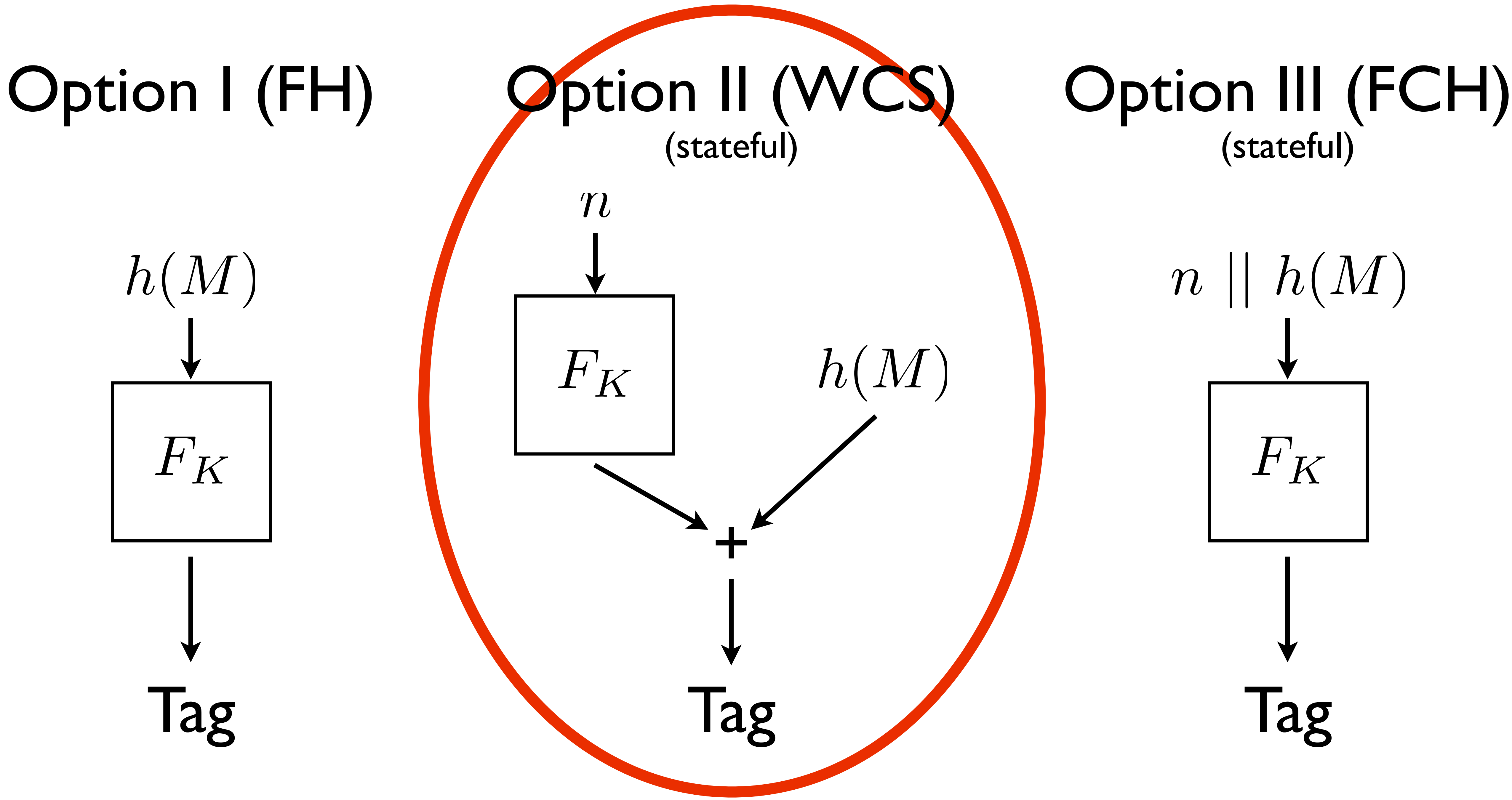
Wegman-Carter

n - nonce, M - message



Wegman-Carter

n - nonce, M - message



Wegman-Carter

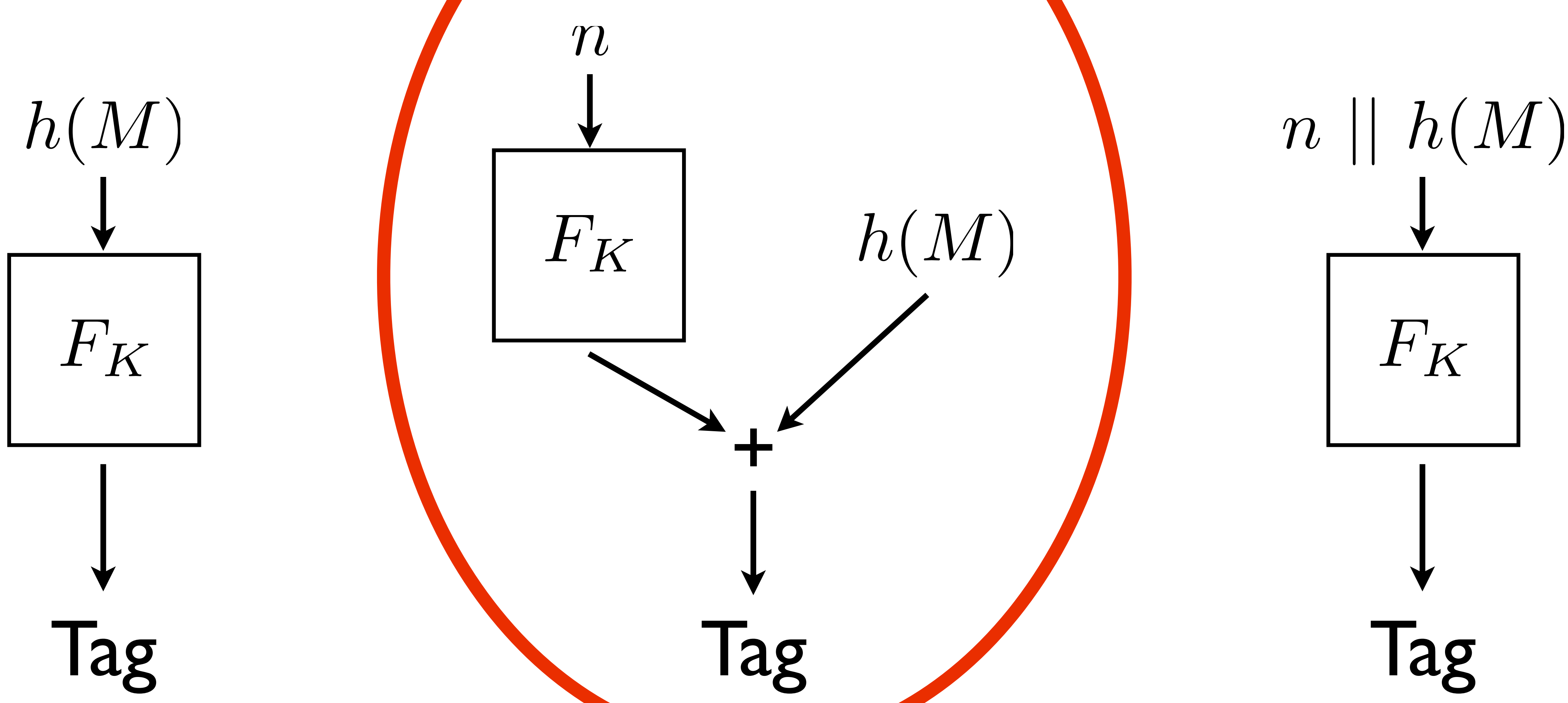
n - nonce, M - message

nonce must be unique!

Option I (FH)

Option II (WCS)
(stateful)

Option III (FCH)
(stateful)



Formal Model

- Oracle for MAC, oracle for verifications
- Adversary can query messages of her choice and receive tags
- Adversary wins if she can produce valid tag for unqueried message (valid verification query)

Security of typical MACs

- Security usually measured in terms of tag length, queries
- Most stateless MACs have chance of forgery of around $\frac{q_s^2}{2^n}$ (ϵq_s^2)
- Stateful MACs are better: more like $\frac{q_v}{2^n}$ (ϵq_v)

What happens after security is lost?

- Security bound measures chance of first forgery
- Are more forgeries possible?
- Perfect MAC - random function

Low-security applications

Low-security applications

- Video streaming

Low-security applications

- Video streaming
- VOIP

Low-security applications

- Video streaming
- VOIP
- {power, CPU, bandwidth}-limited environments (sensor networks, eg)

Breaking Point

- All MACs examined have some breaking point, after which many forgeries are possible

Summary of Attacks

MAC scheme	Expected queries for j forgeries	Succumbs to padding attack	Succumbs to other attack	Message freedom
CBC MAC	$C_1 + j$		$\surd$	$m-2$
EMAC	$C_1 + j$	$\surd$	$\surd$	$m-2$
XCBC	$C_1 + j$	$\surd$	$\surd$	$m-2$
PMAC	$C_1 + j$		$\surd$	1
ANSI retail MAC	$C_1 + j$	$\surd$	$\surd$	$m-2$
HMAC	$\sum_i C_i/2^i + j$	$\surd$		$m-1$

C_i is the i-th observed collision (no truncation of tags)

Summary of Attacks

UHF in FH mode	Expected queries for j forgeries	Reveals key	Queries for key recovery
hash127/Poly1305	$C_1 + \log m + j$	$\surd$	$C_1 + \log m$
VMAC	$C_1 + 2j$		
Square Hash	$C_1 + 2j$	$\surd$	mC_1
Topelitz Hash	$C_1 + 2j$		
Bucket Hash	$C_1 + 2j$		
MMH/NMH	$C_1 + 2j$		

UHF in WCS mode with nonce misuse	Expected queries for j forgeries	Repeated nonce	Reveals key	Queries for key recovery
hash127/Poly1305	$2 + \log m + j$	1	$\surd$	$2 + \log m$
VMAC	$C_1 + 2j$	$C_1 + j$		
Square Hash	$3m + j$	m	$\surd$	$3m$
Topelitz Hash	$2j + 2$	1		
Bucket Hash	$2j + 2$	1		
MMH/NMH	$2m + j$	m	$\surd$	$2m$

There's more

- Preneel and Handschuh found much more severe attacks, many involving only verification queries

OK. Now what?

- Can we fix this?
- Probably, but at what cost?
  - F(F(K, M), M) would probably work but twice as much computation
  - Look for better tradeoffs

OK. Now what?

What if F(K,M) = F(K,M’) and F(F(K,M),M) = F(F(K,M’),M’)?

- Can we fix this?
- Probably, but at what cost?
  - F(F(K, M), M) would probably work but twice as much computation
  - Look for better tradeoffs

Good low security MACs

- Short tag
- Fast
- Guessing the tag is best adversarial strategy (up to a point!)
  - Attacker may get one right every now and then (one frame in video stream)

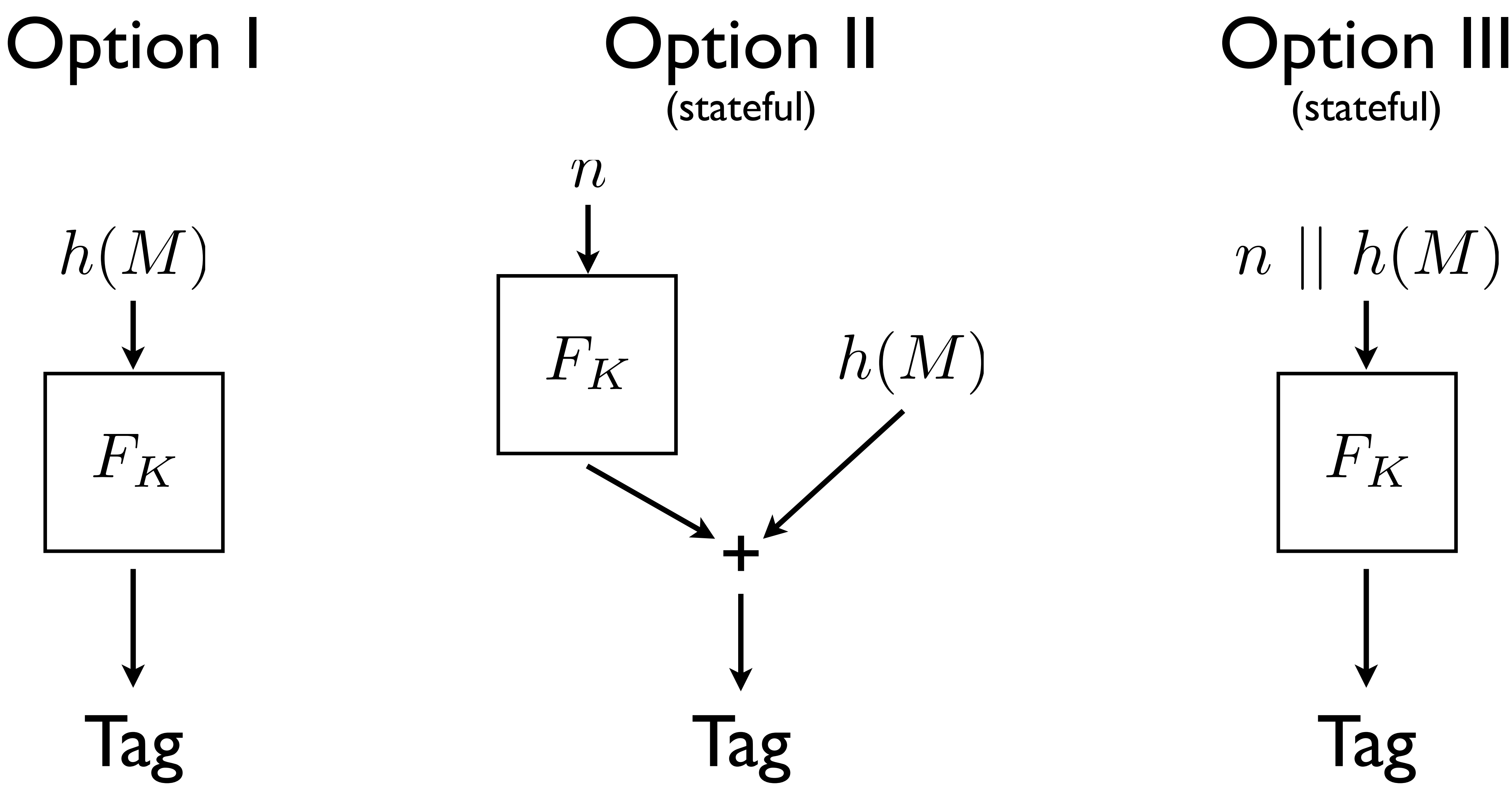
Countermeasures

- Truncate tags to desired length
- Use state to avoid reforgeability

	CBC-MAC	HMAC	WCS MACs
Use State?	✓	✓	✓
Truncate?	✓	✓	✗
Fast? (in software)	✗	✗	✓

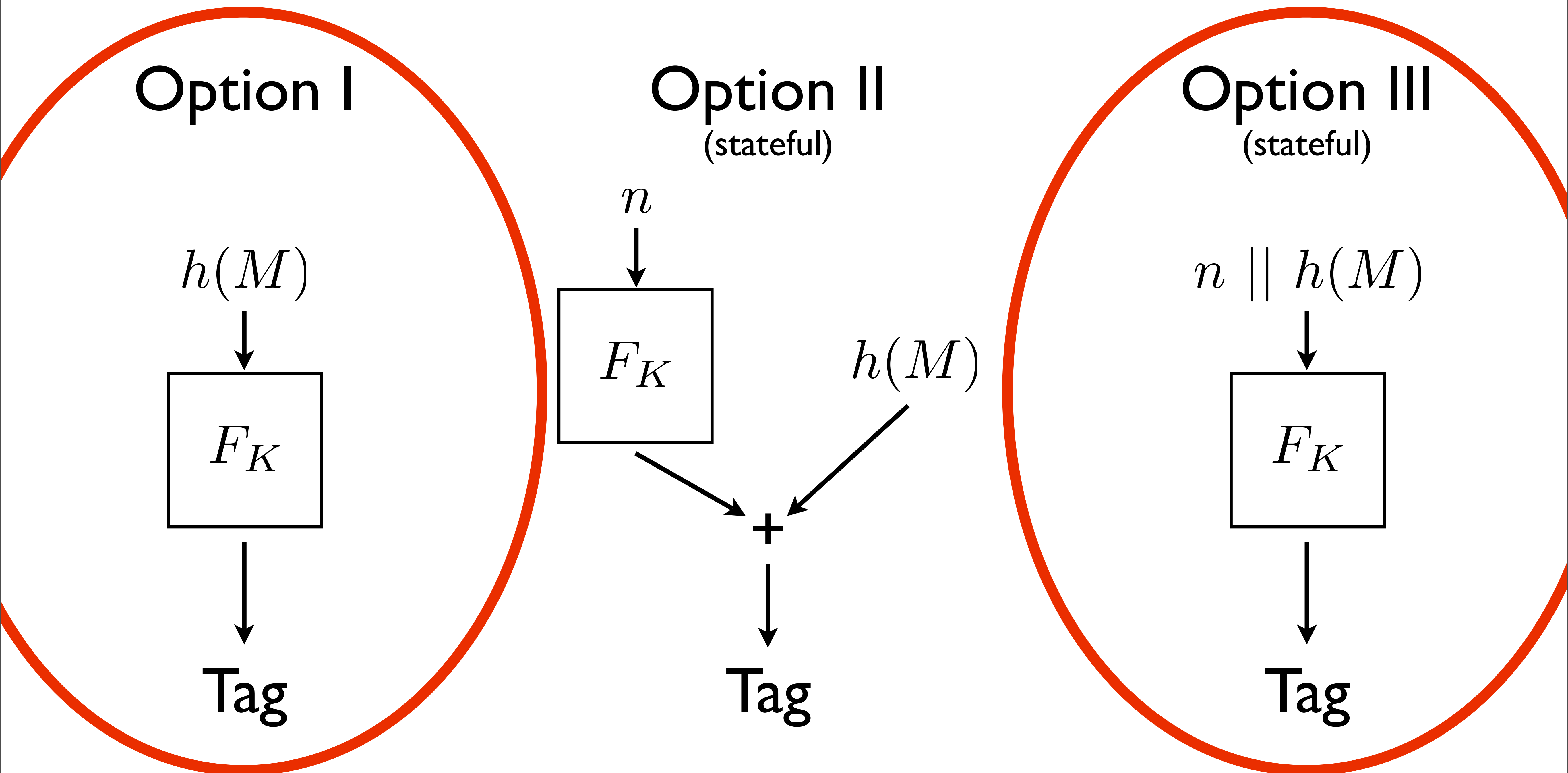
Wegman-Carter

n - nonce, M - message



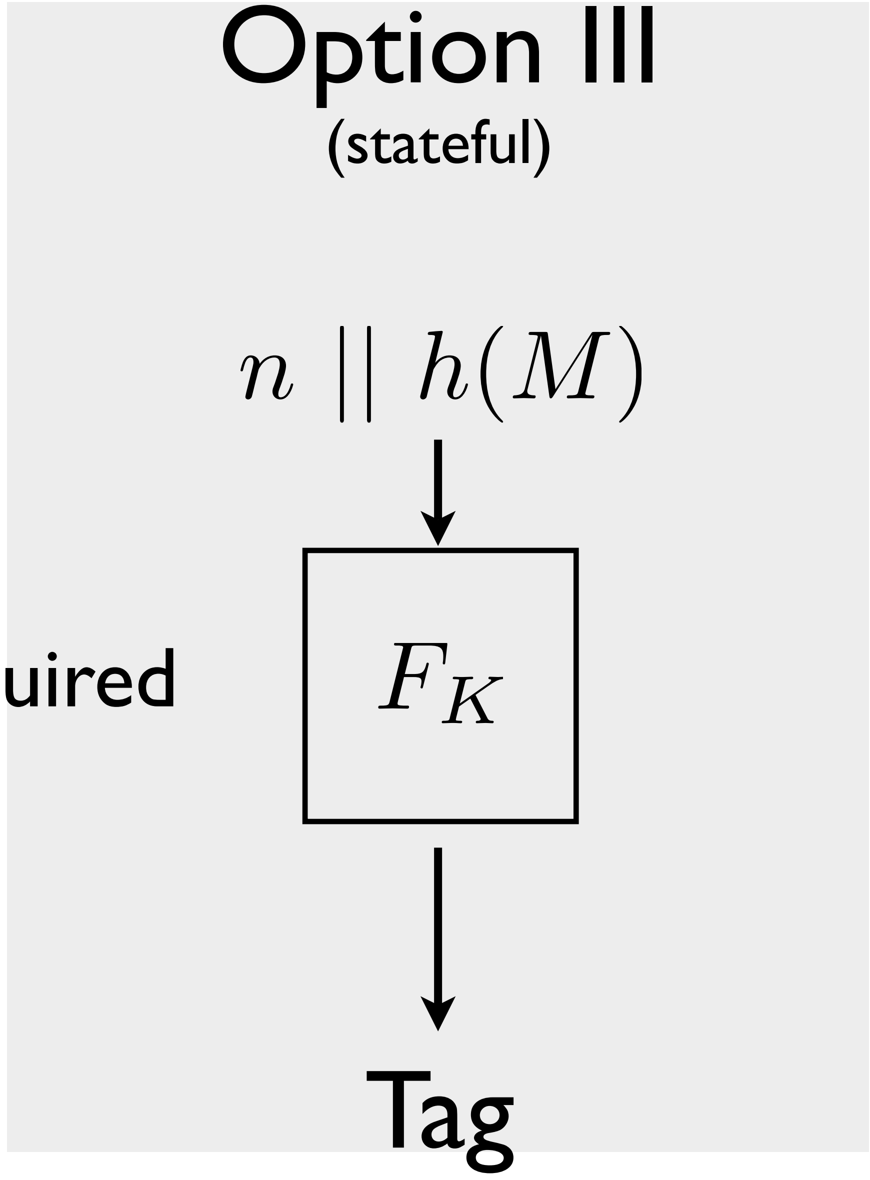
Wegman-Carter

n - nonce, M - message



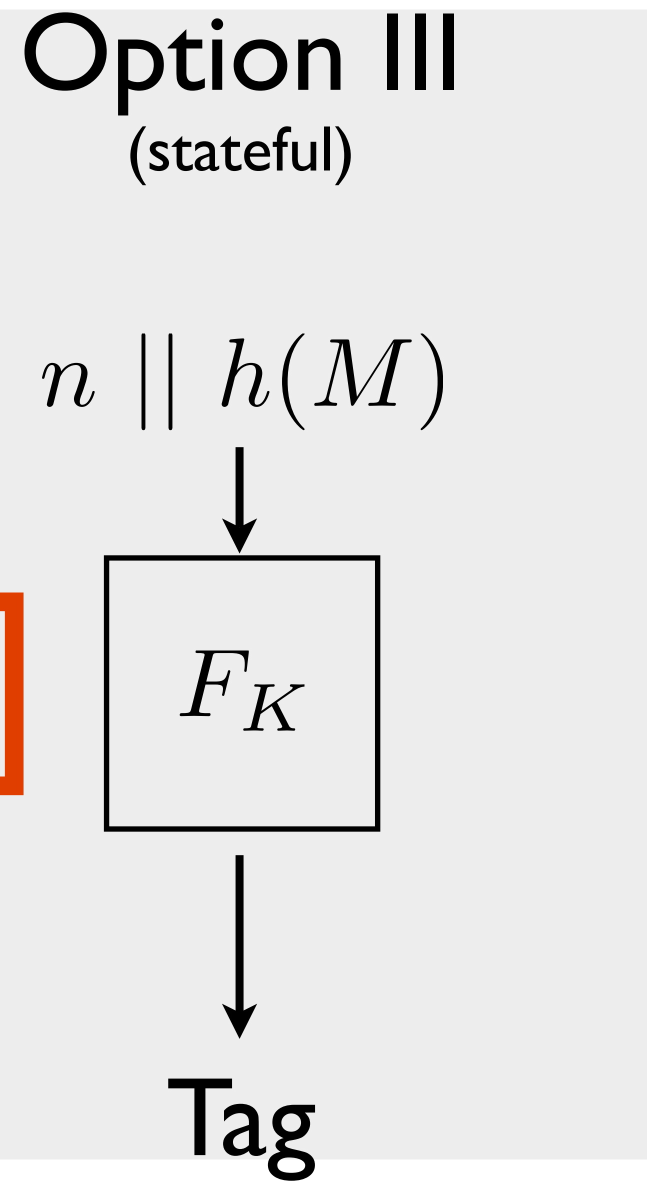
WMAC

- Generalization of options I and III
- State included, uniqueness not required



WMAC

- Generalization of options I and III
- State included, uniqueness not required



WMAC Benefits

- Fast, comparable to fastest WCS MACs
- Nonce reuse
  - Sliding scale of security
- Tags may be truncated safely
- Tight security reduction

WMAC tradeoffs

- No partial precomputation
- PRF must accept larger input (possible extra computation)
- Still has breaking point
- Limiting incorrect verification queries is important!

Security Reduction

Bad things happen with (approximate) probability:

$$\frac{\epsilon(\alpha-1)q_s}{2} + \frac{\epsilon}{2^{L-1}}\left(q_v^2 + q_v q_s\right) + 2\epsilon q_v$$

q_s - number of signing queries

q_v - number of verification queries

L - tag length in bits

α - max number of signing queries per nonce

ϵ - of the ϵ-AU family used

Security Reduction

Let α in $\{1, q_s\}$ for bound for {Option III, Option I}.

Bad things happen with (approximate) probability:

$$\frac{\epsilon(\alpha - 1)q_s}{2} + \frac{\epsilon}{2^{L-1}}\left(q_v^2 + q_v q_s\right) + 2\epsilon q_v$$

q_s - number of signing queries

q_v - number of verification queries

L - tag length in bits

α - max number of signing queries per nonce

ϵ - of the ϵ-AU family used

Example Parameters

- Truncated AES as PRF
- VHASH from VMAC
- Comparable speed to VMAC
- $\epsilon \leq 2^{-82}$, $L = 24$, $\alpha = 2^{24}$ (8-bit counter value)
- After 2^{32} queries, 2^{24} forgery attempts, one forgery is expected

Example Parameters

Tag + counter only 32 bits

- Truncated AES as PRF
- VHASH from VMAC
- Comparable speed to VMAC
- $\epsilon \leq 2^{-82}$, $L = 24$, $\alpha = 2^{24}$ (8-bit counter value)
- After 2^{32} queries, 2^{24} forgery attempts, one forgery is expected

Q&A