How to Sign with White-Boxed AES

Latincrypt 2019

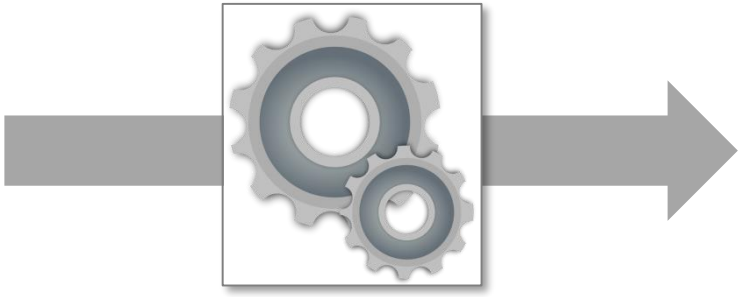
Marc Fischlin

joint work with
Helene Haagh

Obfuscation

program

obfuscator



„unintelligible“ but functionally equivalent program


```
alert('Hello, World!')
```

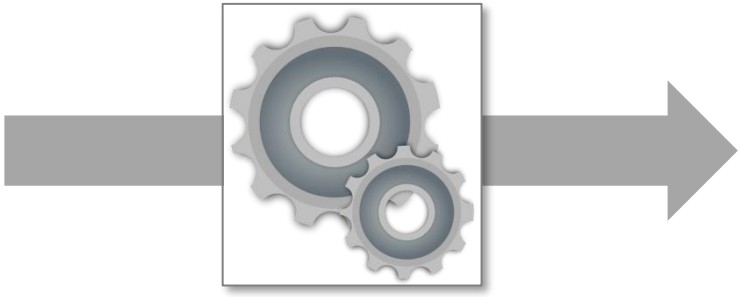
```
eval(function(p,a,c,k,e,d){e=function(c){return
c};if(!''.replace(/^/,String)){while(c--
){d[c]=k[c]||c}k=[function(e){return
d[e]}];e=function(){return'\\w+'};c=1};while(c--
){if(k[c]){p=p.replace(new
RegExp('\\b'+e(c)+'\\b','g'),k[c])}}return
p}('0(\'1,
2!\')',3,3,'alert|Hello|World'.split('|'),0,{}))
```

White-Boxing AES

Chow, Eisen, Johnson, van Oorschot (SAC 2002)

program

obfuscator



„unintelligible“ but functionally equivalent program

$\mathbf{AES}_k(\cdot)$

$\mathbf{WBAES}_k(\cdot)$

In particular, white-box version shall hide the secret key

White-Boxing is hard…

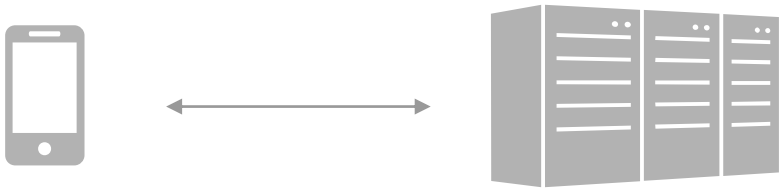
WhibOx Contests in 2017 and 2019: Competitors only lasts days or weeks

Even unclear if theoretical solutions exist at all

…but not the topic of this talk

Instead: How to use a good white-boxing

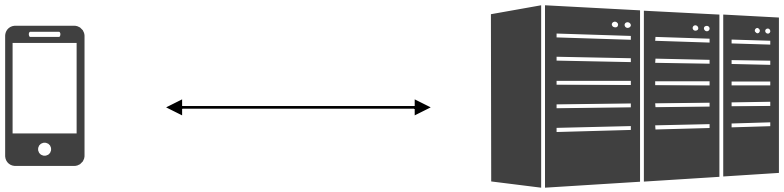
When to White-Box?



$\mathtt{WBAES}_k(\cdot)$ $\mathtt{AES}_k(\cdot)$

protects key $\mathtt{k}$
against leakage on device

schemes based on
shared symmetric key $\mathtt{k}$



$\mathtt{AES}_k(\cdot)$ $\mathtt{WBAES}^{-1}_k(\cdot)$

This talk

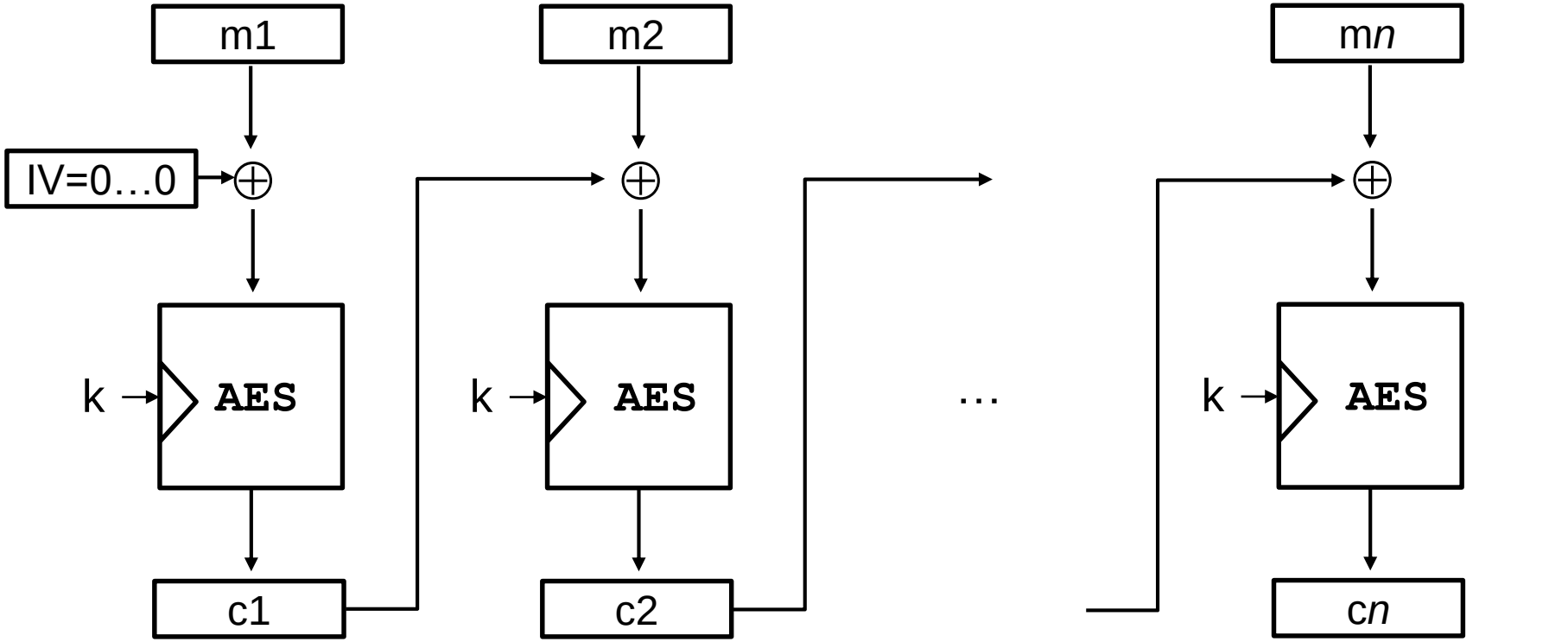
$\mathtt{AES}_k(\cdot)$, $\mathtt{WBAES}^{-1}_k(\cdot)$
= secret-public key pair

fast signing with key $\mathtt{k}$ on device
slow(er) verification on server

How not to Sign with White-boxed AES

CBC-MAC as Signature Scheme?

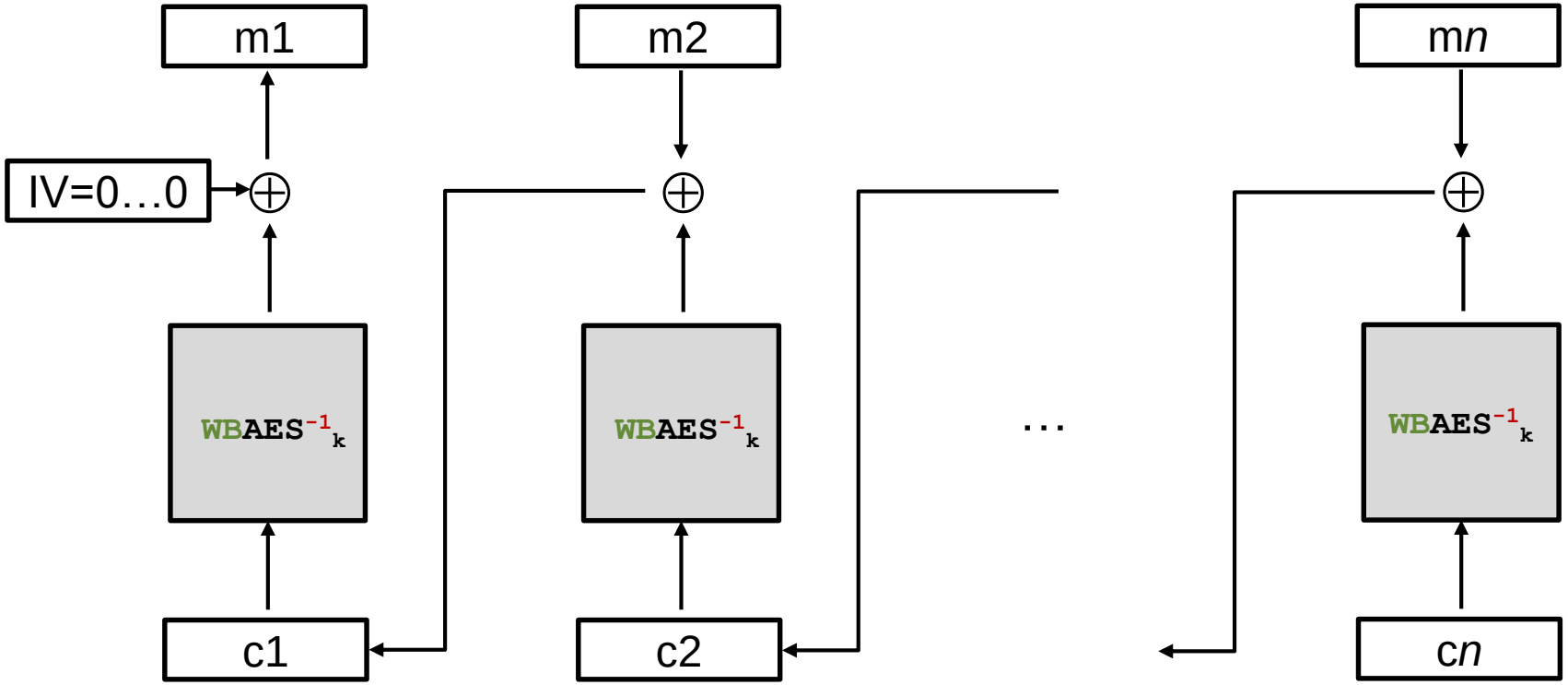
here: fixed-length, block-aligned messages



signature s=c*n*

Verification with WBAES

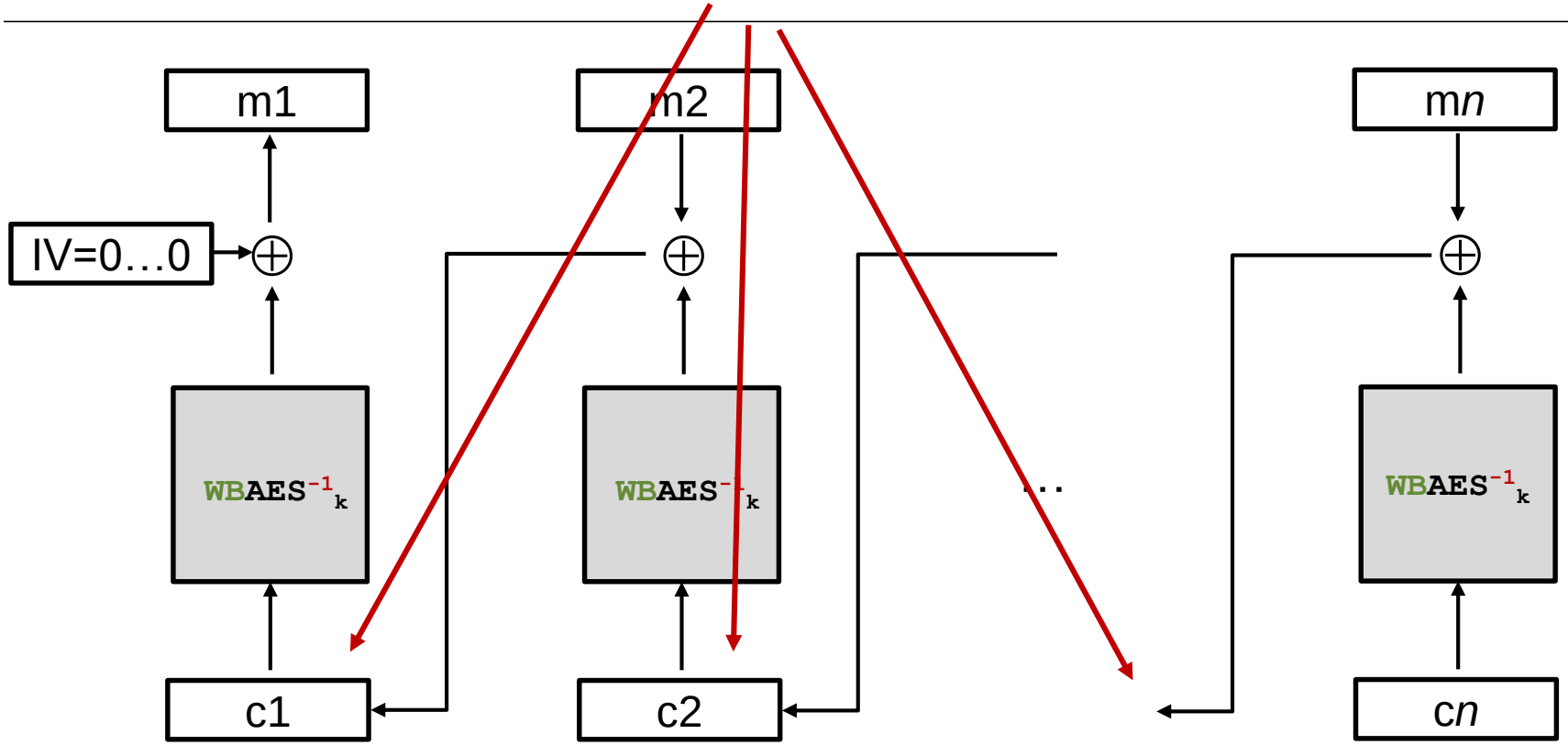
given signature s=c*n*
and message m



verification succeeds if „computing backwards“ yields m*1*

Security?

Adversary knows public $\mathbf{WBAES}^{-1}_{\mathbf{k}}$ and thus gets to see all intermediate results!

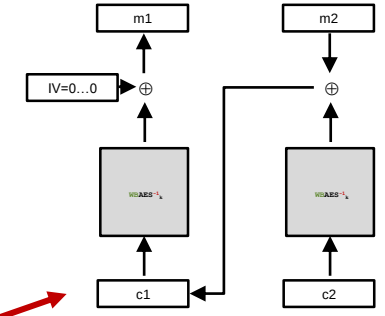


verification succeeds if „computing backwards“ yields m*1*

Breaking CBC

here: with two message blocks

goal: create forgery for m1||m2 with probability 1



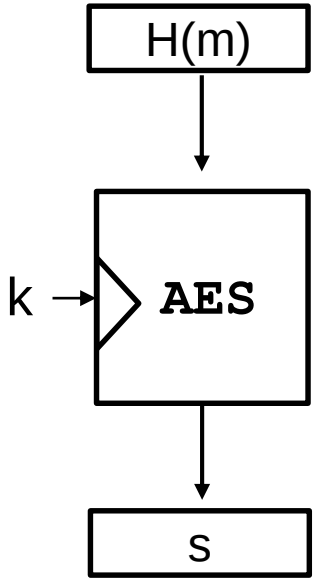
1. get signature for m1||x2, compute intermediate result $\mathbf{AES}_k(m1)$
2. get signature for x1||0..0, compute intermediate result $\mathbf{AES}_k(x1)$
3. get signature for x1|| $(m2 \oplus \mathbf{AES}_k(m1) \oplus \mathbf{AES}_k(x1))$

This is also a valid signature
for the (fresh) message m1||m2

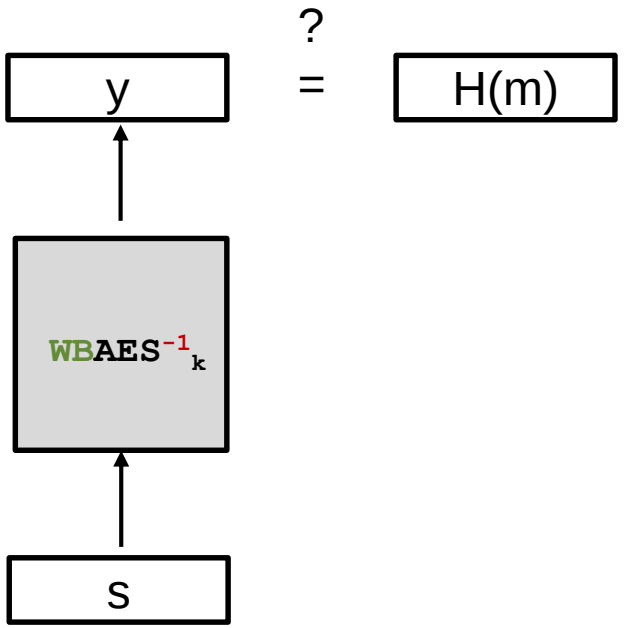
Signing with Full-Domain-Hashing

Full-Domain-Hash Signatures for WBAES

$\mathbf{AES}_k(\cdot)$ = secret signing key



$\mathbf{WBAES}^{-1}_k(\cdot)$ = public verification key



H = Hash function truncated to 128 Bits

Security Results for FDH-Signatures

Coron (Crypto 2000):

$\text{Adv}(\text{Forge}) \leq q_s \cdot \text{Adv}(\text{Unpred})$

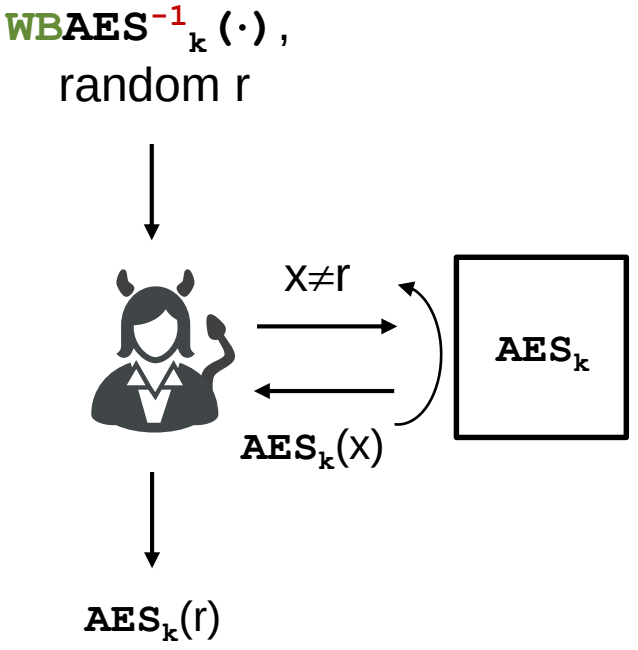
q_s=#signature queries

Problem:
trivial attack strategy against unpredictability, wins with prob 2^{-64} after 2^{64} WBAES evaluations and no AES queries

guarantees for 128-bit AES?

Other problem:
proof requires random oracle programming

Delerablee, Lepoint, Paillier, Rivain
Unpredictability game (SAC 2013):



CBC-Signatures with Random Oracles

Revisiting idea of CBC-Signing, but this time using random oracle:

Signature $s=CBC_k(H(m))$ for H outputting multiple of 128 bits

Verification: compute CBC backwards using H(m)

CBC with Random-Oracle-Hashing:

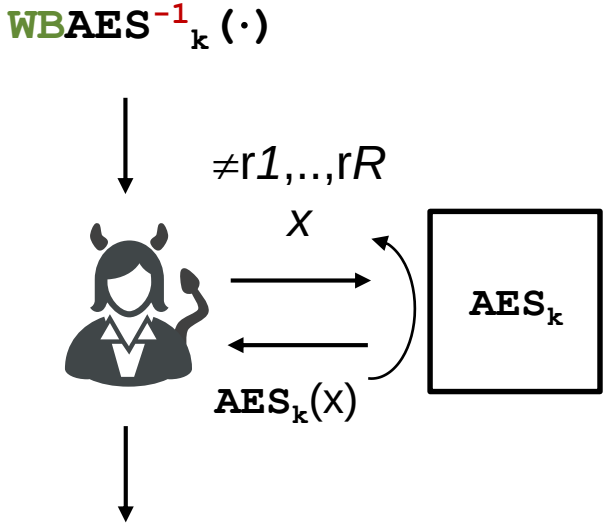
$$Adv(Forge) \leq q_H \cdot Adv(Unpred) + q_H \cdot (q_H + q_s) \cdot 2^{-128}$$

q_H=#random oracle queries

Problem with unpredictability inherent due to restricted input size of AES!

Correlation Intractability

Correlation Intractability



correlated r*1*, r*2*,…, r*R and*
AES$_k$(r*1*),**AES**$_k$(r*2*),…,**AES**$_k$(r*R*)

according to some fixed
(non-trivial) correlation criteria like
r*1*,r*2*,..r*R* are equal on leading 128-log R bits

Suzuki, Tonien, Kurosawa, Toyota (ICISC'06):

generic upper bound (no WBAES input) of
$(q_{AES})^{R} \cdot 2^{-128(R-1)}$ after q_{AES} AES queries

example: R=4 with $q_{AES}=2^{64}$ yields probability of less than 2^{-128}

Correlation Intractability vs. Unpredictability

Correlation intractability (for R=2) $\Rightarrow$ Unpredictability

Correlation intractability $\not\Rightarrow$ Unpredictability

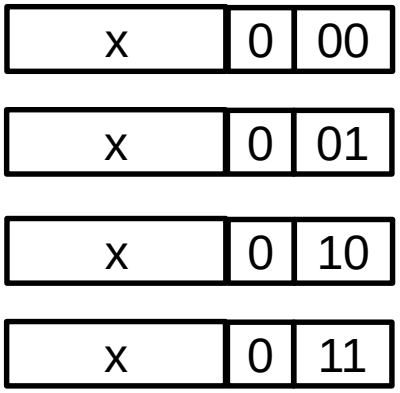
for general block ciphers,
unclear for AES

Signing with Chaining and Correlation Intractability

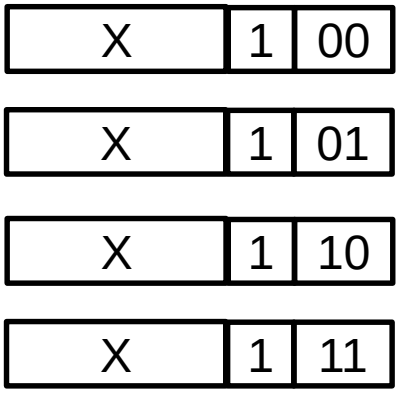
ROChain: Signing

R=4

x = H(0|m) truncated to 125 bits



X = H(1|s|m) truncated to 125 bits



$s=(AES_k(x|000),\ldots,AES_k(x|011))$

$S= (AES_k(X|100),\ldots,AES_k(X|111))$

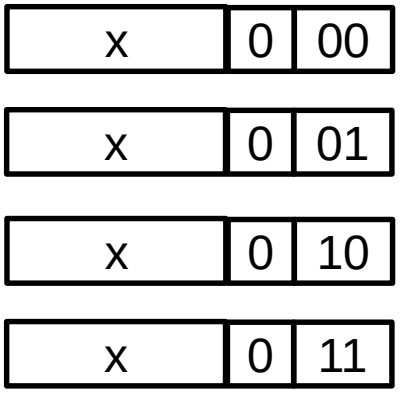
signature = (s, S)

Verification

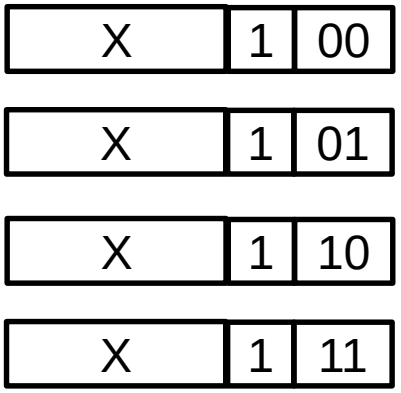
recompute x and X, check each AES value,
& that pre-images in s resp. S are correlated

R=4

x = H(0|m) truncated to 125 bits



X = H(1|s|m) truncated to 125 bits



$s=(AES_k(x|000),\ldots,AES_k(x|011))$

$S= (AES_k(X|100),\ldots,AES_k(X|111))$

signature = (s, S)

Security (Idea)

cannot query about valid s
for new x by corr.intractability

x = H(0|m) truncated to 125 bits

x	0	00
x	0	01
x	0	10
x	0	11

X = H(1|s|m) truncated to 125 bits

X	1	00
X	1	01
X	1	10
X	1	11

$s=(AES_k(x|000),\ldots,AES_k(x|011))$

$S= (AES_k(X|100),\ldots,AES_k(X|111))$

needs to re-use x from
signing query or one of
very few collisions ($\leq 2^6 \cdot q_s$)

most likely X* for m* different
from all previous values,
infeasbile to find valid S* by corr.intr.

Security bound for ROChainSign

Security bound for ROChainSign:

$$\mathrm{Adv}(\mathrm{Forge}) \leq 2 \cdot \mathrm{Adv}(\mathrm{Corr.Intr.}) + q_s^2 \cdot 2^{-113} + 3 \cdot 2^{-128}$$

in the non-programmable random oracle model

Extensions

Correlation intractability can be used to give counter-based and nonce-based construction without random oracles

but signatures become larger than in ROChainSig

Example (R=4, signing 256-bit messages)

Scheme	Signature Size	Random Oracle?
ROChainSign	8 AES values	non-programmable
CountSign (using counter)	16 AES values	no, but stateful
NonceSign (using nonces)	32 AES values	no, stateless

Conclusion

Conclusion (I)

Slowing down $\mathtt{WB}\mathbf{AES}^{-1}_{k}(\cdot)$ computations hinders attacks

similar to iterations in password-based hashing

Conclusion (II)

Limited block length of AES causes trouble

bypassing problems by switching to
correlation intractability assumption

constructions with reasonable bounds
in non-programmable random oracle model
and in standard model with nonces (larger signatures)

Thank you!