Symmetric encryption

CS642:
Computer Security

Professor Ristenpart

http://www.cs.wisc.edu/~rist/

rist at cs dot wisc dot edu

US Army May Relax Physical Requirements To Recruit Cyber Warriors

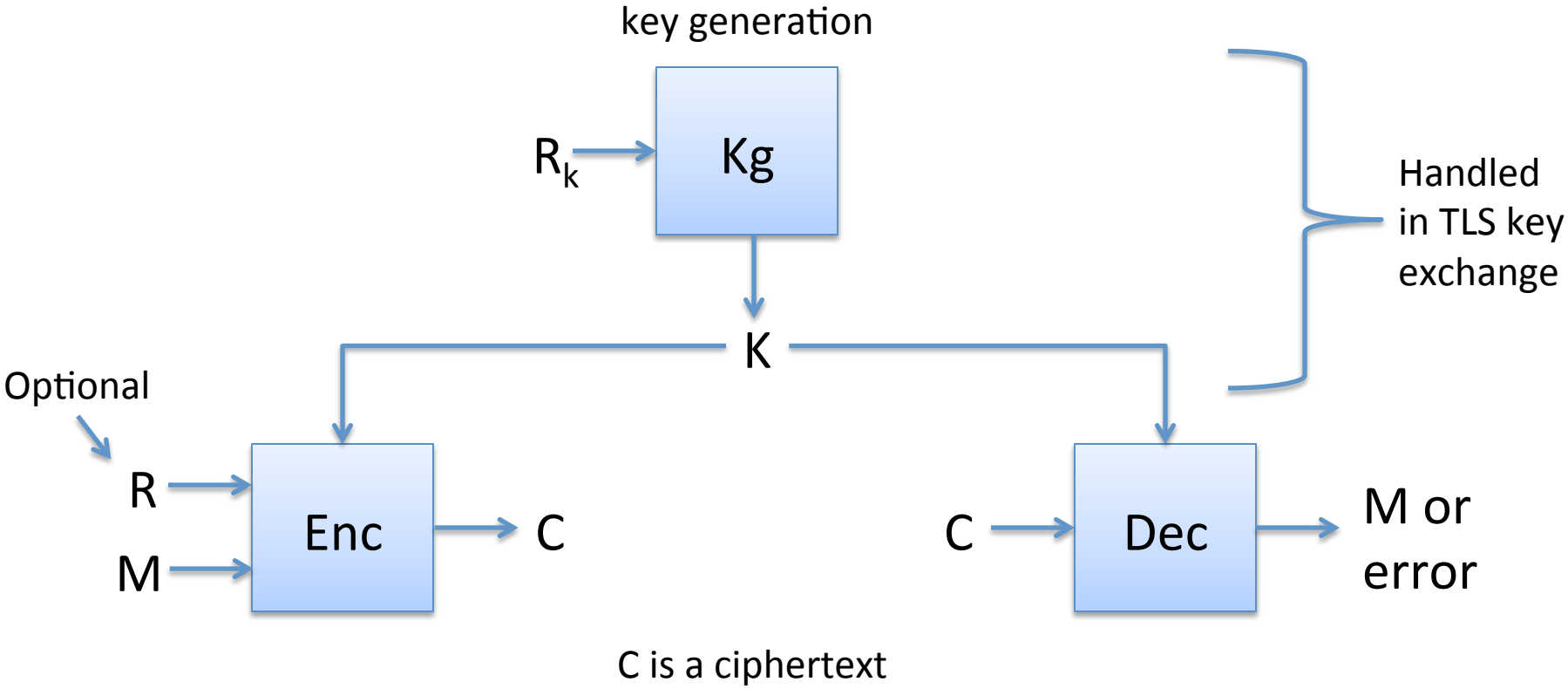
285

samzenpus posted yesterday | from the couch-patrol dept.

HughPickens.com writes

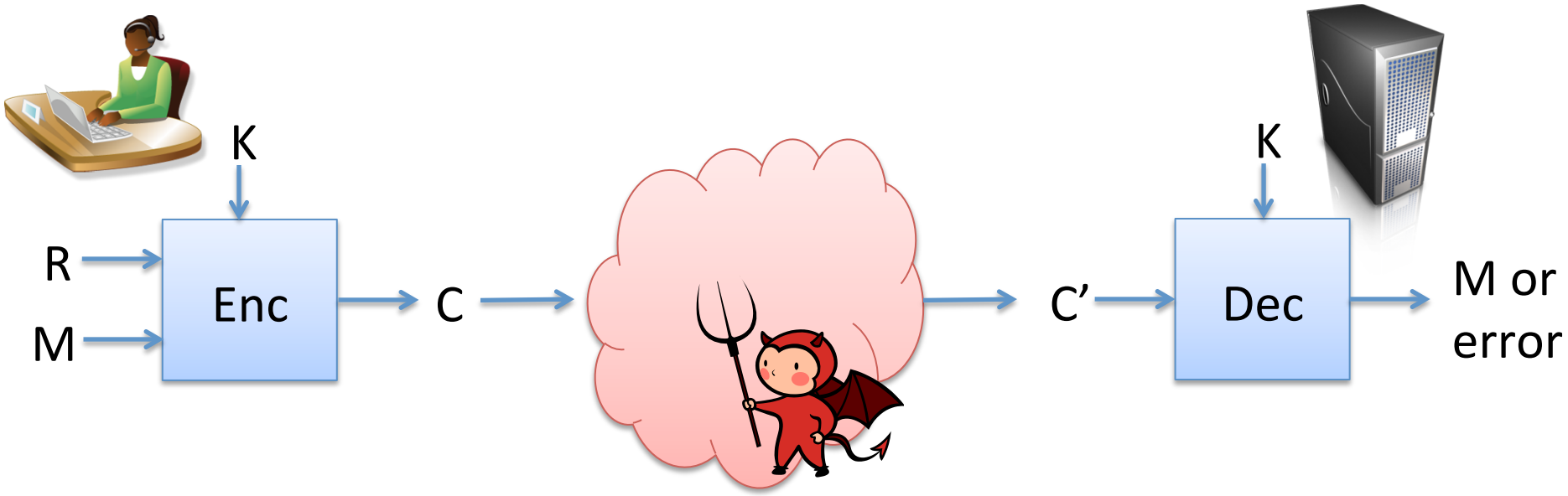
> Clifford Davis reports that only 30% of young people between the ages of 17 and 24 are qualified to become soldiers. This is primarily due to three issues: obesity or health problems; lack of a high school education; and criminal histories. While cognitive and moral disqualifications have held steady, weight issues account for 18% of disqualifications, and the number is rising steadily. It's projected to hit 25% by 2025. The current Army policy is that every recruit, whether enlisting for infantry or graphic design, has to meet the same physical requirements to join — but that requirement may be changing. "Today, we need cyber warriors, so we're starting to recruit for Army Cyber," says Major General Allen Batschelet.

Symmetric encryption



Correctness: D(K , E(K,M,R)) = M with probability 1 over randomness used

In TLS symmetric encryption underlies the Record Layer



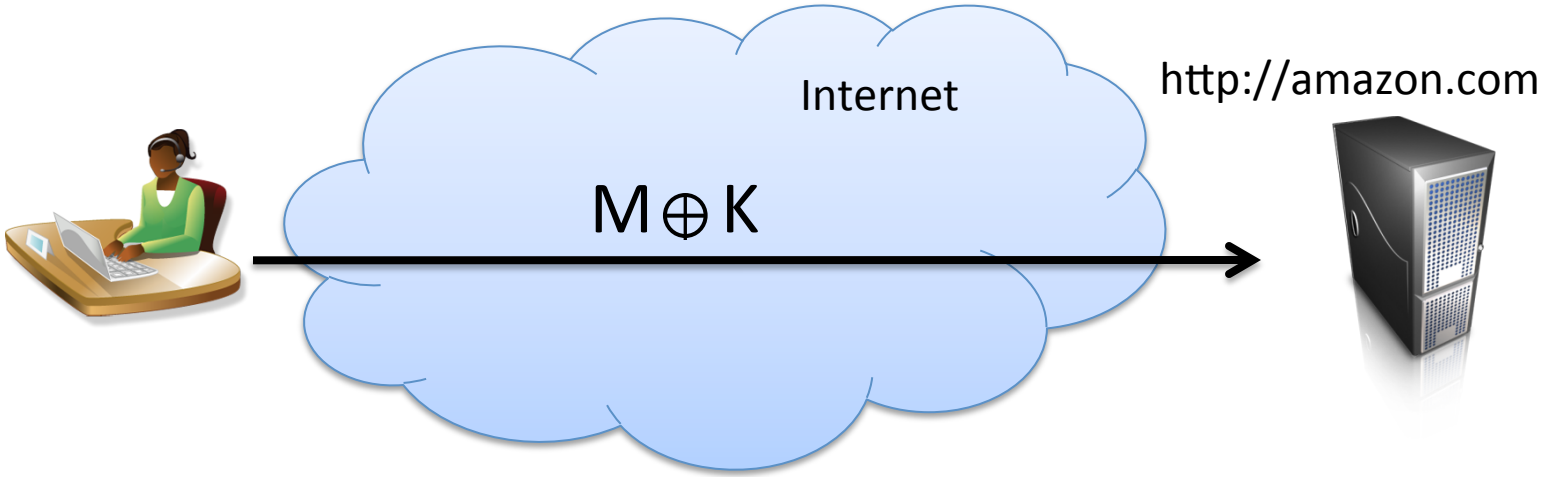
What security properties do we need from symmetric encryption?

1) Confidentiality: should not learn any information about M

2) Authenticity: should not be able to forge messages

Often referred to as Authenticated Encryption security

Back to our application



Does OTP provide a secure channel?

Integrity easily violated

Reuse of K for messages M,M' leaks M ⊕ M'

Encrypting same message twice under K leaks the message equality

K must be as large as message

Message length revealed

Cryptography as computational science

Use computational intractability as basis for confidence in systems

1. Design a cryptographic scheme
2. Provide proof that no attacker with limited computational resources can break it

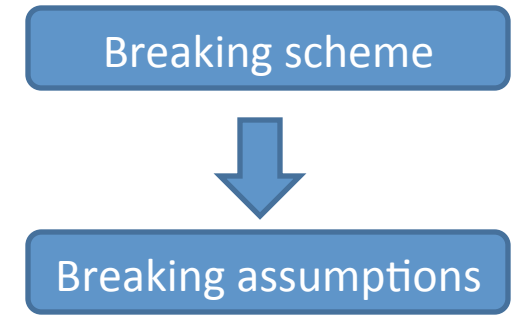
} Goldwasser, Micali and Blum circa 1980's

Formal definitions

Scheme semantics

Security

Security proofs (reductions)



Example:

Attacker can not recover credit card

Can not factor large composite numbers

As long as assumptions holds we believe in security of scheme!

Provable security yields

1) well-defined assumptions and security goals
2) cryptanalysts can focus on assumptions and models

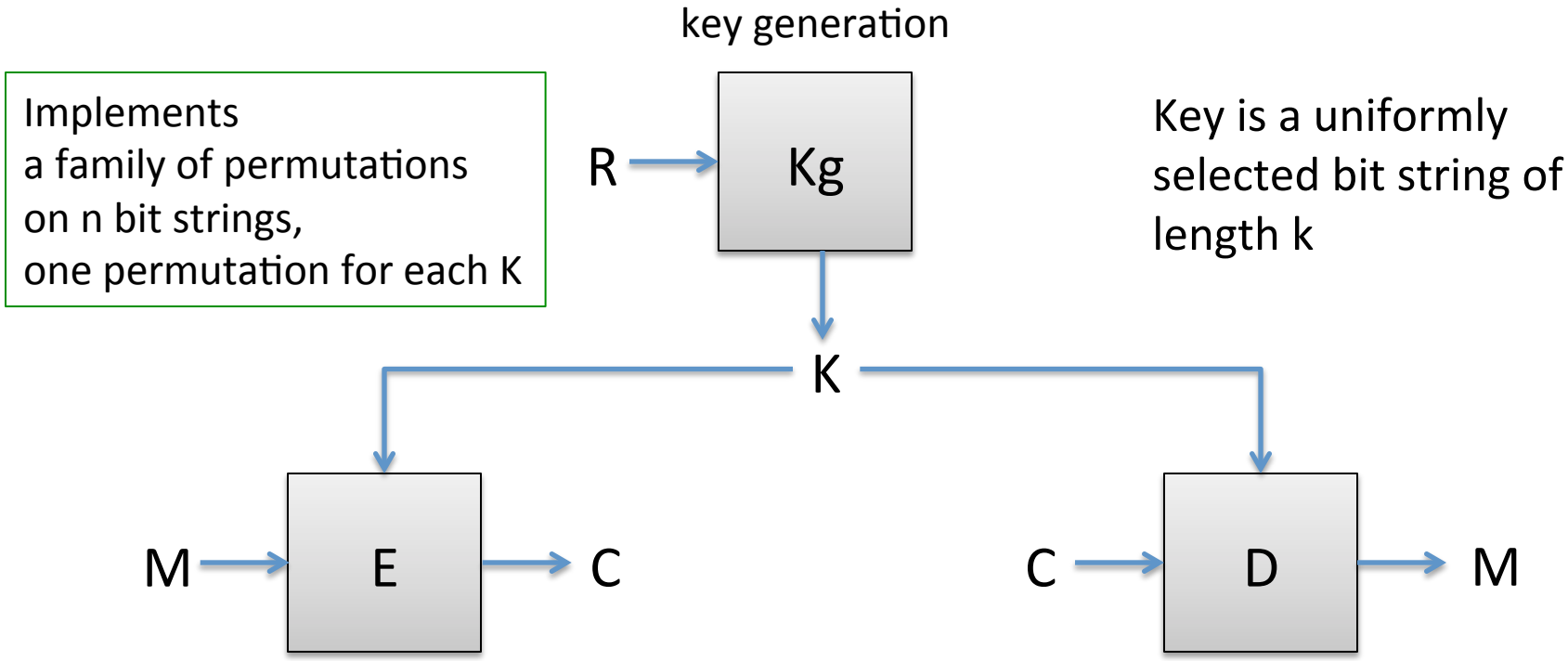
But no one knows how to do this. It's been studied for a very long time!

Typical assumptions

- Basic atomic primitives are hard to break:
  - Factoring of large composites intractable
  - RSA permutation hard-to-invert
  - Block ciphers (AES, DES) are good pseudorandom functions (PRFs)
  - Hash functions are collision resistant

Confidence in atomic primitives is gained by cryptanalysis, public design competitions

Block ciphers



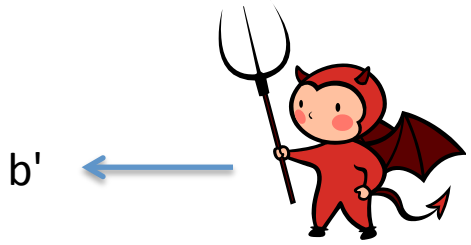
$E: \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$

Security goal: E(K,M) is indistinguishable from random n-bit string for anyone without K

Block cipher security

$E: \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$

Adversary gets to submit ***distinct*** messages to oracle



M_1

C_1

M_2

C_2

⋮

PRF Security Game

Fn(M)
If b == 1 then
    C <- E(K,M)
If b == 0 then
    C <-$ $\{0,1\}^n$
Ret C

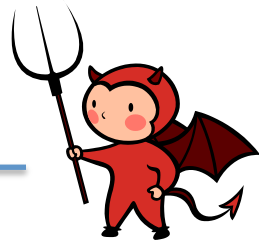
Adversary outputs guess b' of b
Wins if b' = b
Insecure if adversary wins with probability close to 1
Secure if no adversary can get probability more than 1/2

b is a uniformly sampled bit
and K is uniformly sampled key
Both hidden from adversary

Security goal: E(K,M) is indistinguishable from random n-bit string for anyone without K

One-time pad is not a secure PRF

Adversary gets to submit ***distinct*** messages to oracle



b'

M_1

C_1

M_2

C_2

⋮

PRF Security Game

Fn(M)
If b == 1 then
    C <- K ⊕ M
If b == 0 then
    C <-$ $\{0,1\}^n$
Ret C

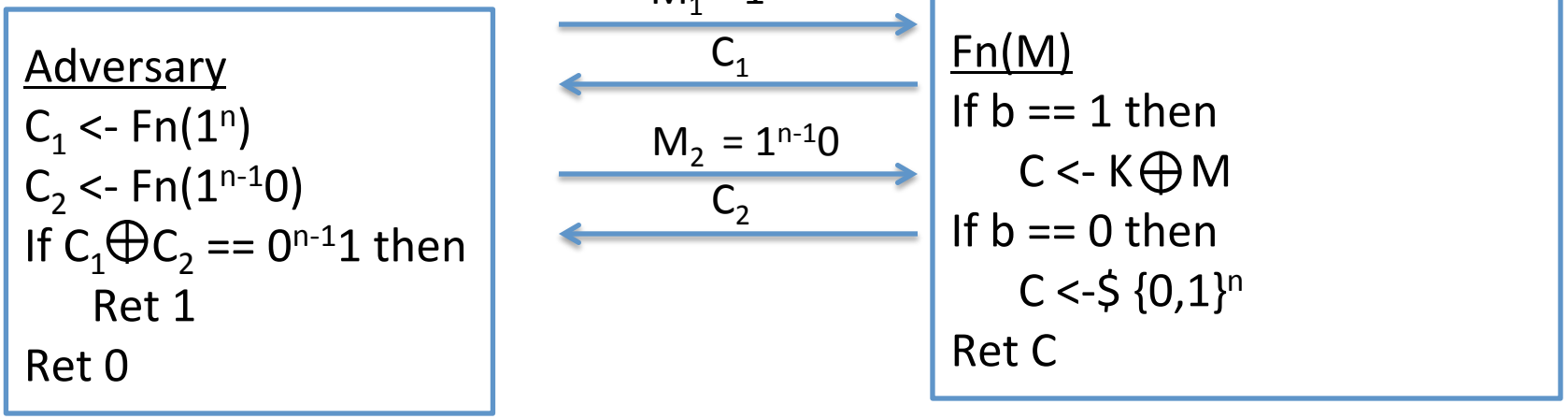
Adversary outputs guess b' of b
Wins if b' = b
Insecure if adversary wins with probability close to 1
Secure if no adversary can get probability more than 1/2

b is a uniformly sampled bit
and K is uniformly sampled key
Both hidden from adversary

Security goal: E(K,M) is indistinguishable from random n-bit string for anyone without K

One-time pad is not a secure PRF

Adversary gets to submit ***distinct*** messages to oracle



Adversary outputs guess b' of b
Wins if b' = b
Insecure if adversary wins with probability close to 1
Secure if no adversary can get probability more than 1/2

b is a uniformly sampled bit
and K is uniformly sampled key
Both hidden from adversary

If b = 1 then:

$$C_1 \oplus C_2 = (1^n \oplus K) \oplus (1^{n-1}0 \oplus K) = 0^{n-1}1$$

If b = 0 then C_1 and C_2 are both random n-bit strings. Their xor equals $0^{n-1}1$ with probability at most $1/2^n$

Data encryption standard (DES)

Originally called Lucifer
- team at IBM
- input from NSA
- standardized by NIST in 1976

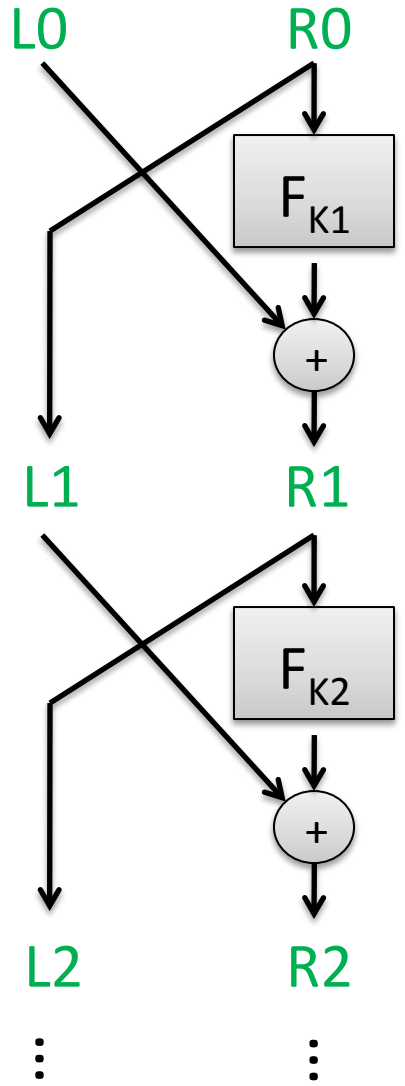
n = 64
k = 56

Number of keys:
72,057,594,037,927,936

Split 64-bit input into L0,R0 of 32 bits each

Repeat Feistel round 16 times

Each round applies function F using separate round key



Best attacks against DES

Attack	Attack type	Complexity	Year
Biham, Shamir	Chosen plaintexts, recovers key	2^{47} plaintext, ciphertext pairs	1992
DESCHALL	Unknown plaintext, recovers key	$2^{56/4}$ DES computations 41 days	1997
EFF Deepcrack	Unknown plaintext, recovers key	~4.5 days	1998
Deepcrack + DESCHALL	Unknown plaintext, recovers key	22 hours	1999

- DES is still used in some places
- 3DES (use DES 3 times in a row with more keys) expands keyspace and still used widely in practice

Advanced Encryption Standard (AES)

Response to 1999 attacks:

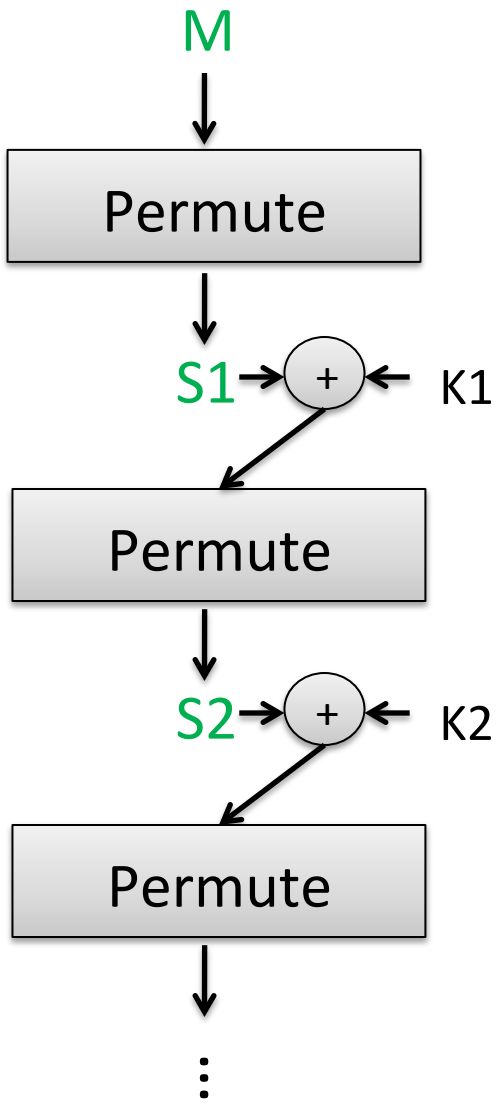
- NIST has design competition for new block cipher standard
- 5 year design competition
- 15 designs, Rijndael design chosen

Advanced Encryption Standard (AES)

Rijndael (Rijmen and Daemen)

n = 128
k = 128, 192, 256

Number of keys for k=128:
340,282,366,920,938,463,463,374,607,431,768,211,456



Substitution-permutation design.
k=128 has 10 rounds of:

1) Permute:
   SubBytes (non-linear S-boxes)
   ShiftRows + MixCols (invertible linear transform)

2) XOR in a round key derived from K

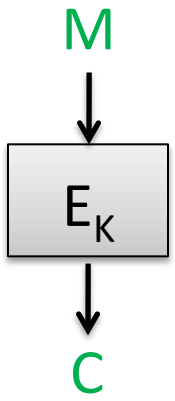
(Actually last round skips MixCols)

Best attacks against AES

Attack	Attack type	Complexity	Year
Bogdanov, Khovratovich, Rechberger	chosen ciphertext, recovers key	$2^{126.1}$ time + some data overheads	2011

- Brute force requires time 2^{128}
- Approximately factor 4 speedup

Are block ciphers good for record layers?



Functional limitations:

- Only encrypt messages that fit in n bits

Security limitations:

- Confidentiality: M = M' => E(K,M) = E(K,M')
- Authenticity: any C of length n is valid ciphertext

Symmetric encryption security

Encryption algorithm denoted Enc

Adversary gets to submit messages to oracle

b'

M_1

C_1

M_2

C_2

⋮

Chosen-plaintext attack security Game

Encrypt(M)
C' <-$ Enc(K,M)
If b == 1 then
    C <- C'
If b == 0 then
    C <-$ $\{0,1\}^{|C'|}$
Ret C

Adversary outputs guess b' of b
Wins if b' = b

b is a uniformly sampled bit
and K is uniformly sampled key
Both hidden from adversary

Security goal: Enc(K,M) looks like random bit string to attackers that can obtain encryptions of chosen plaintexts

Verizon's 'Perma-Cookie' Is a Privacy-Killing Machine

BY ROBERT MCMILLAN 10.27.14 | 6:30 AM | PERMALINK

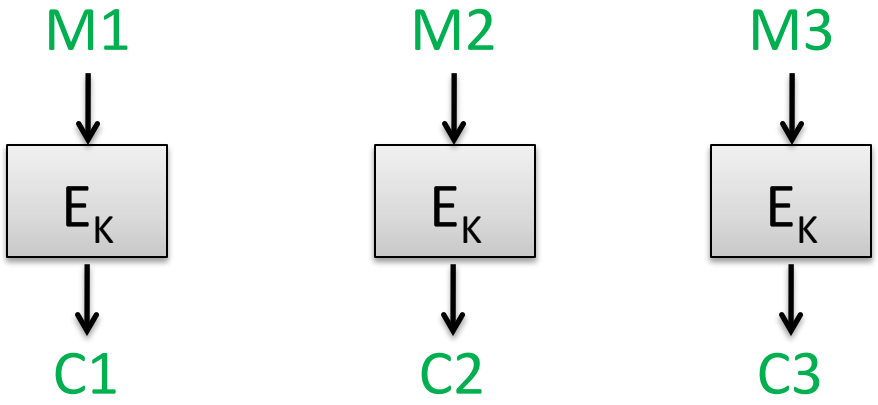
Verizon Wireless has been subtly altering the web traffic of its wireless customers for the past two years, inserting a string of about 50 letters, numbers, and characters into data flowing between these customers and the websites they visit.

The company—one the country's largest wireless carriers, providing cell phone service for about 123 million subscribers—calls this a Unique Identifier Header, or UIDH. It's a kind of short-term serial number that advertisers can use to identify you on the web, and it's the lynchpin of the company's internet advertising program. But critics say that it's also a reckless misuse of Verizon's power as an internet service provider—something that could be used as a trump card to obviate established privacy tools such as private browsing sessions or "do not track" features.

Block cipher modes of operation

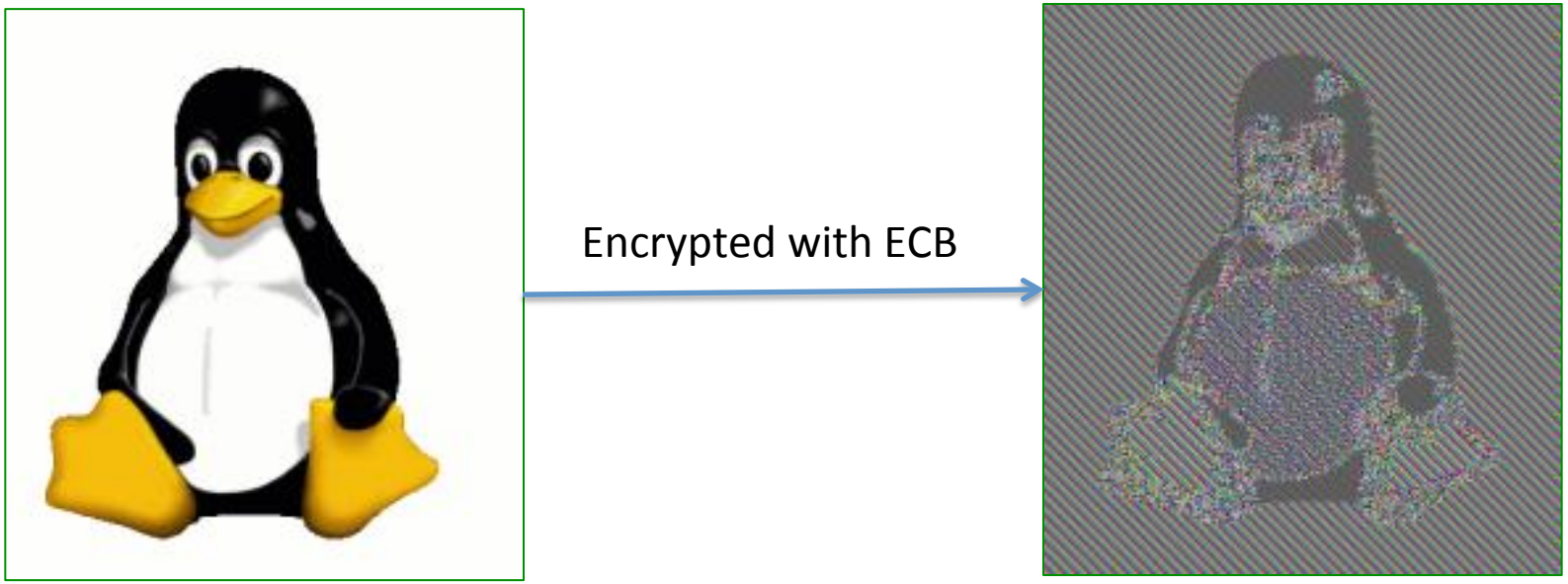
How can we build an encryption scheme for arbitrary message spaces out of a block cipher?

Electronic codebook (ECB) mode
Pad message M to M1,M2,M3,... where each block Mi is n bits
Then:



ECB mode is a more complicated looking substitution cipher

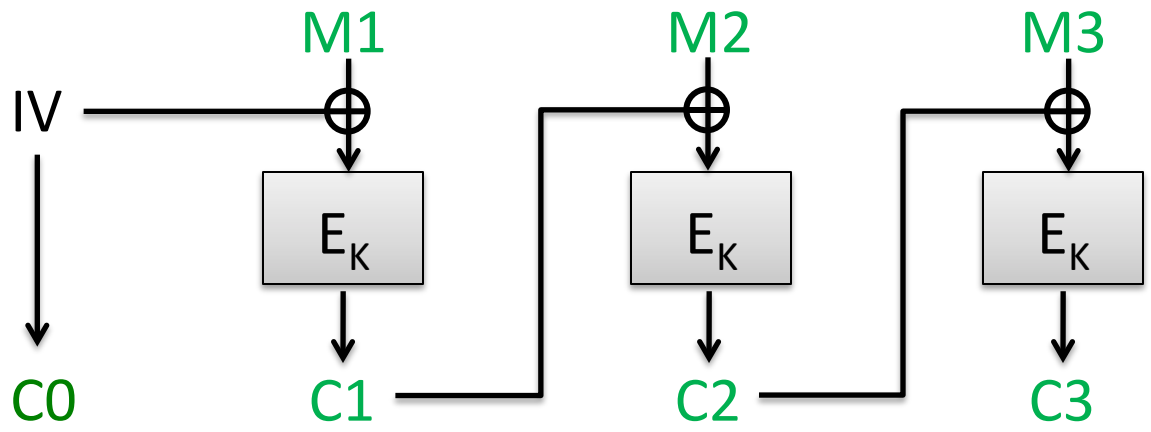
Recall our credit-card number example.
ECB: substitution cipher with alphabet n-bit strings instead of digits



Images courtesy of
http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation

CBC mode

Ciphertext block chaining (CBC)
Pad message M to M1,M2,M3,... where each block Mi is n bits
Choose random n-bit string IV
Then:



How do we decrypt?

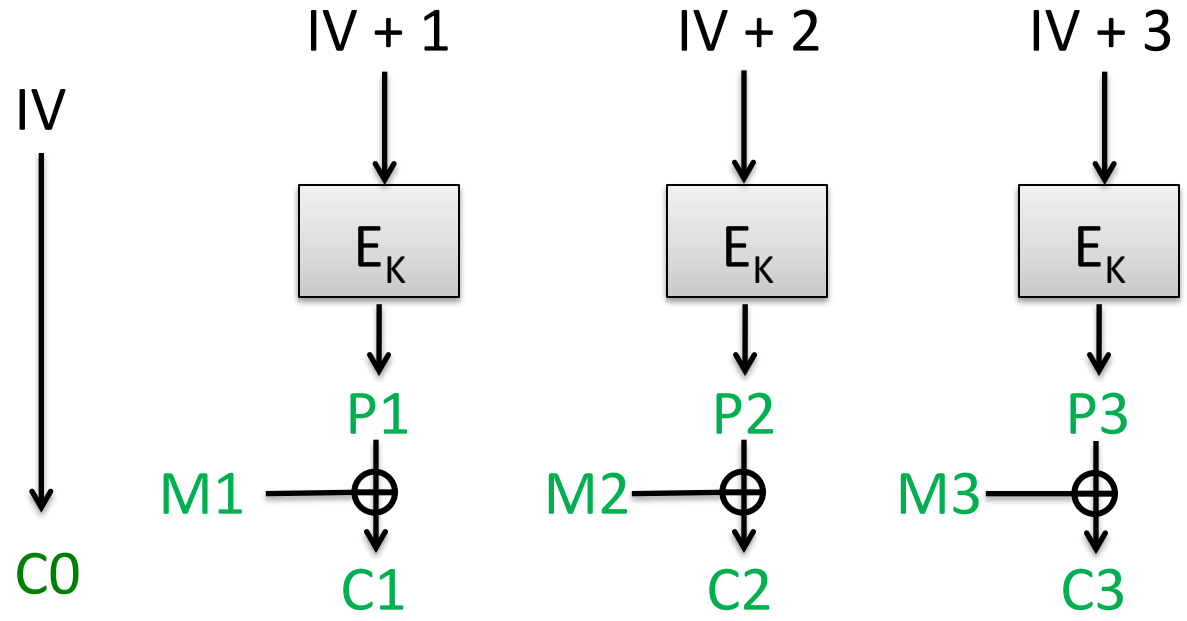
OTP-like encryption using block cipher

Counter mode (CTR)

Pad message M to M1,M2,M3,... where each is n bits except last

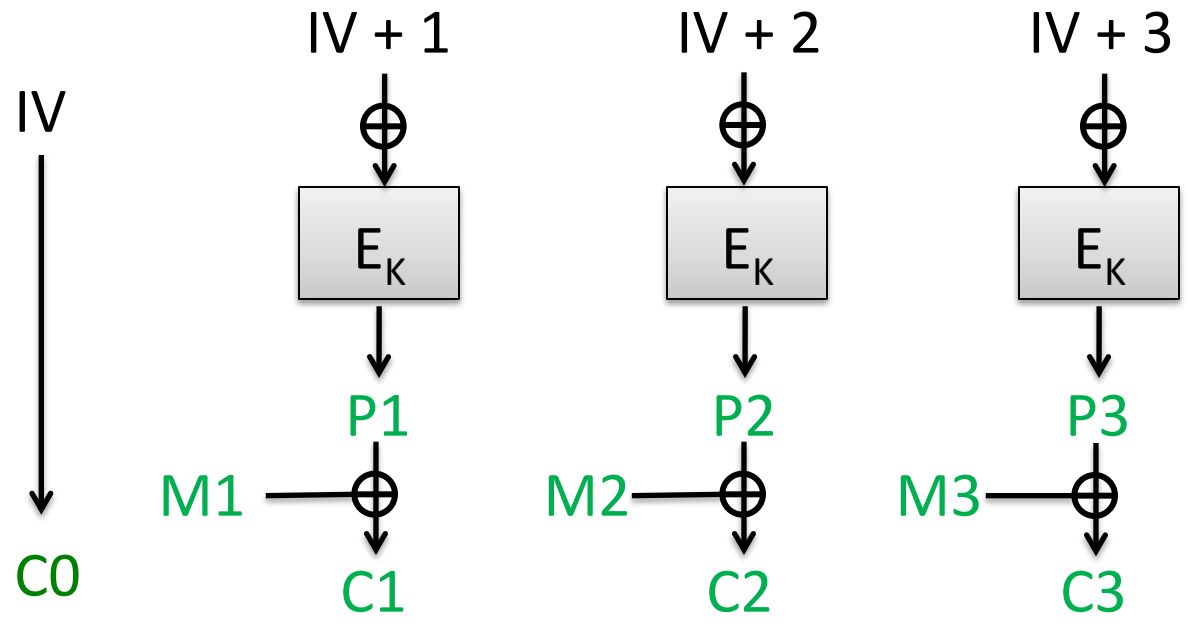
Choose random n-bit string IV

Then:



Maybe use less than full n bits of P3

How do we decrypt?



Can attacker learn K from just C0,C1,C2,C3?

Implies attacker can break E, i.e. recover block cipher key

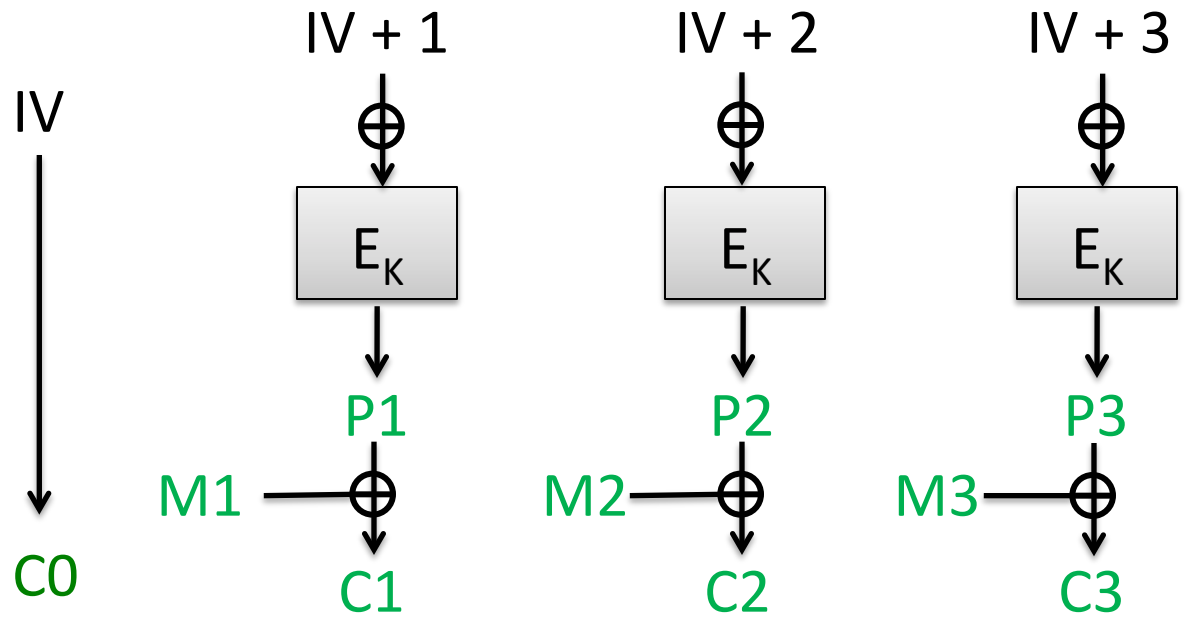
Can attacker learn M = M1,M2,M3 from C0,C1,C2,C3?

Implies attacker can invert the block cipher without knowing K

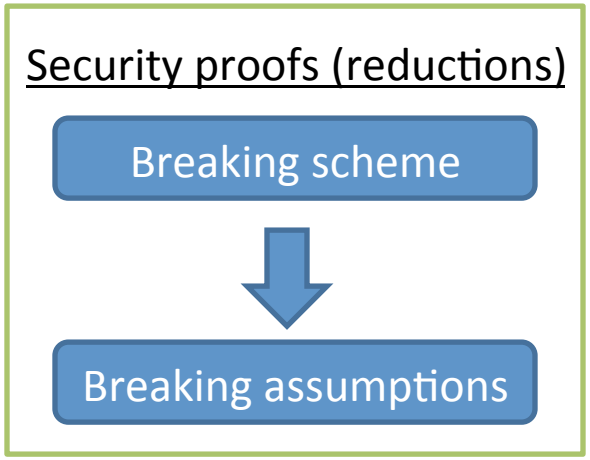
Can attacker learn one bit of M from C0,C1,C2,C3?

Implies attacker can break PRF security of E

Passive adversaries cannot learn anything about messages



<u>Theorem (informal).</u>
Let A be a successful, efficient attacker against security of CBC mode. Then there exists a PRF adversary B against E that is efficient and successful.

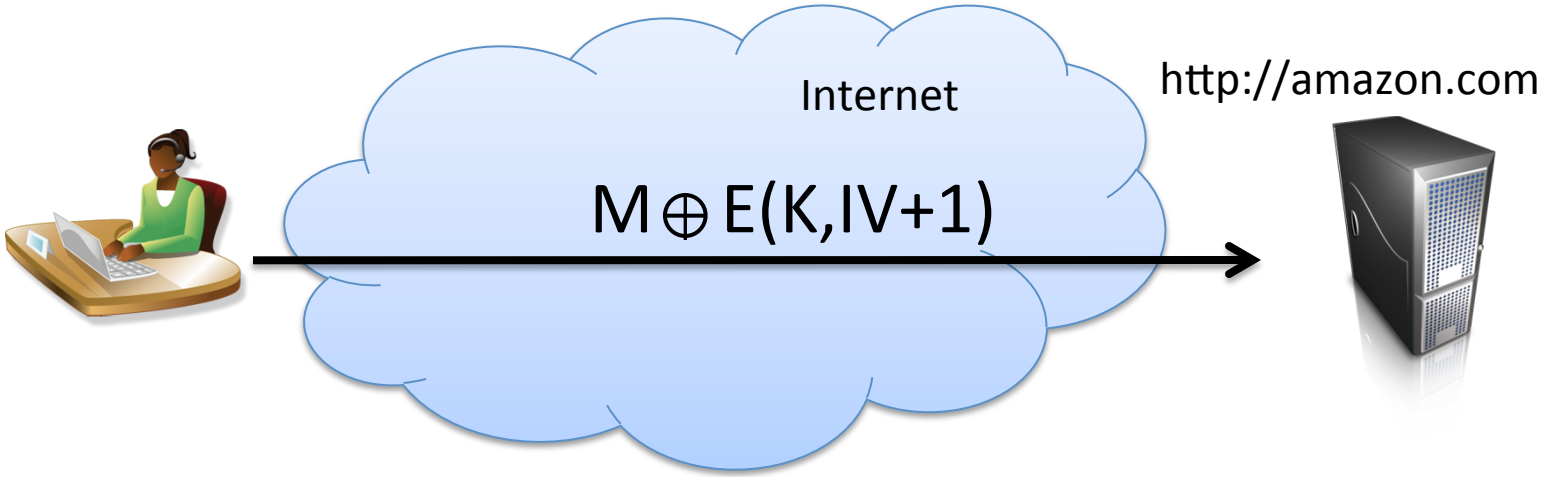


Attacker can not break CBC confidentiality

Can not break E PRF

Reduces analysis now to E and to security definition / model

Back to our application



Does CTR mode provide a secure channel?

Integrity easily violated

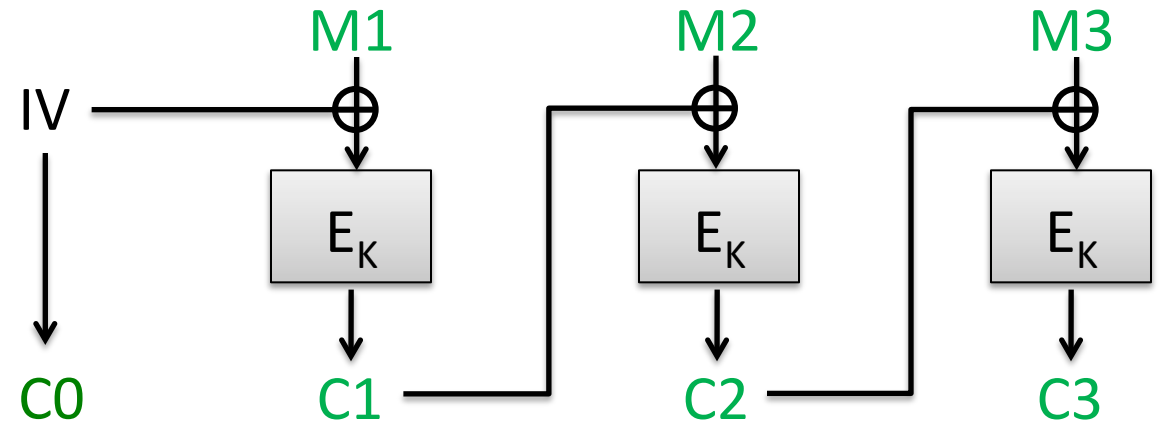
~~Reuse of K for messages M,M' leaks M ⊕ M'~~

~~Encrypting same message twice under K leaks the message equality~~

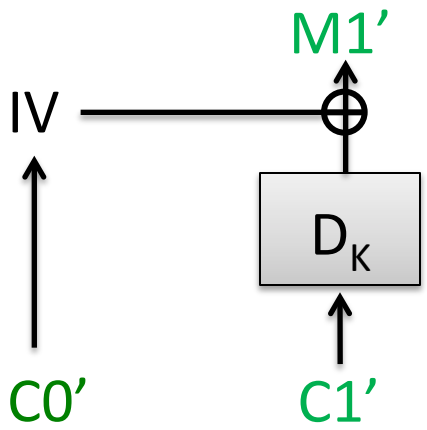
~~K must be as large as message~~

Message length revealed

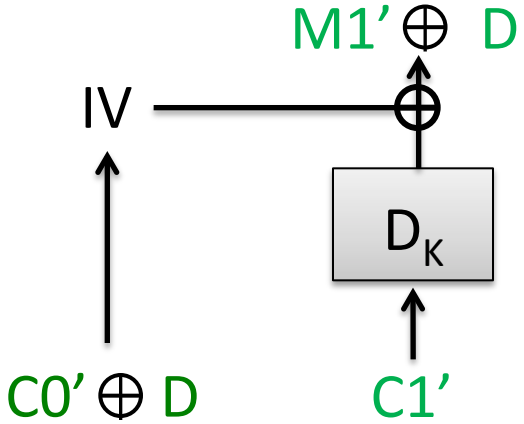
Active security of CBC mode



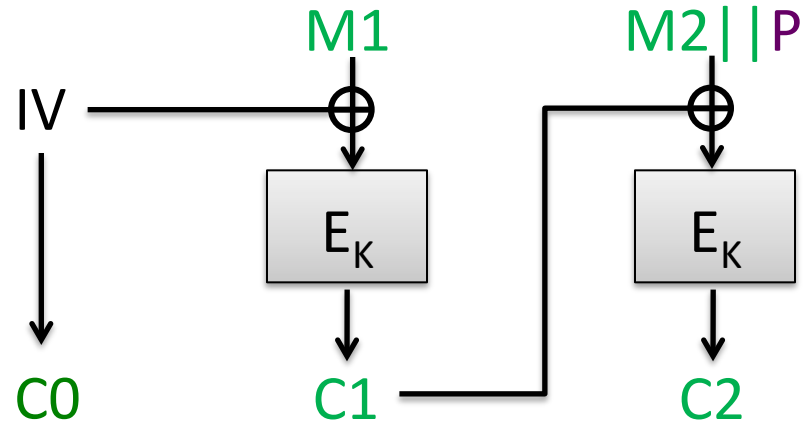
What about forging a message? Pick any C0', C1' …



Better yet for any D:



Padding oracle attack



Assume that
M1||M2 has length
2n-8 bits

P is one byte of padding
that must equal 0x00

Adversary
obtains
Ciphertext
C0,C1,C2

C0 , C1 , C2

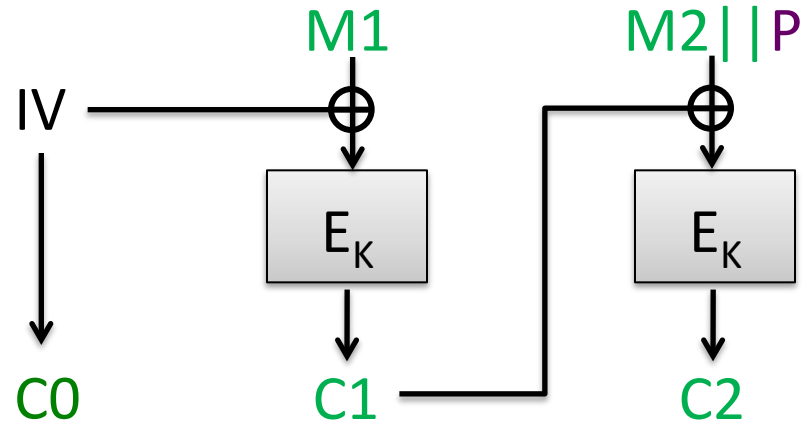
ok

C0, C1⊕1 , C2

error

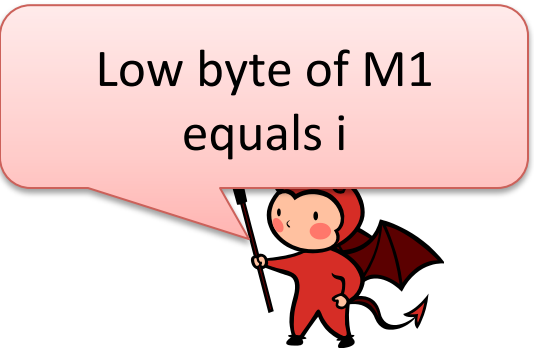
Dec(K, C')
M1'||M2'||P' = CBC-Dec(K,C')
If P' ≠ 0x00 then
    Return error
Else
    Return ok

Padding oracle attack



Assume that M1||M2 has length 2n-8 bits

P is one byte of padding that must equal 0x00



R, C0 , C1

error

R , C0 ⊕ 1 , C1

error

R , C0 ⊕ 2 , C1

error

…

R , C0 ⊕ i , C1

ok

Adversary obtains ciphertext C = C0,C1,C2

Let R be arbitrary n bits

Dec(K, C')

M1'||M2'||P' = CBC-Dec(K,C')

If P' ≠ 0x00 then

  Return error

Else

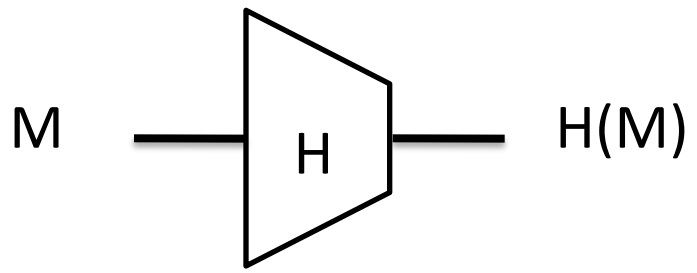
  Return ok

Chosen ciphertext attacks against CBC

Attack	Description	Year
Vaudenay	10's of chosen ciphertexts, recovers message bits from a ciphertext. Called "padding oracle attack"	2001
Canvel et al.	Shows how to use Vaudenay's ideas against TLS	2003
Degabriele, Paterson	Breaks IPsec encryption-only mode	2006
Albrecht et al.	Plaintext recovery against SSH	2009
Duong, Rizzo	Breaking ASP.net encryption	2011
Jager, Somorovsky	XML encryption standard	2011
Duong, Rizzo	"Beast" attacks against TLS	2011

Hash functions and message authentication

Hash function H maps arbitrary bit string to fixed length string of size m



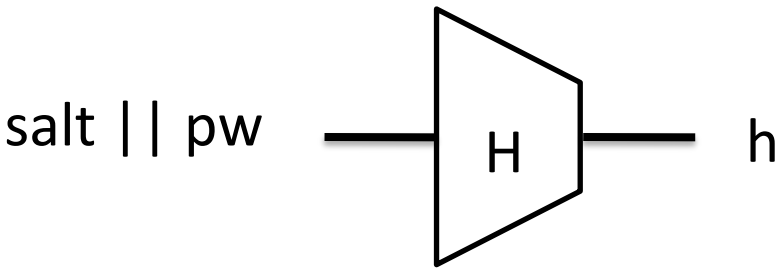
MD5: m = 128 bits
SHA-1: m = 160 bits
SHA-256: m = 256 bits

Some security goals:

- collision resistance: can't find M != M' such that H(M) = H(M')
- preimage resistance: given H(M), can't find M
- second-preimage resistance: given H(M), can't find M' s.t. H(M') = H(M)

Hash function application example

Password hashing. Choose random salt and store (salt,h) where:



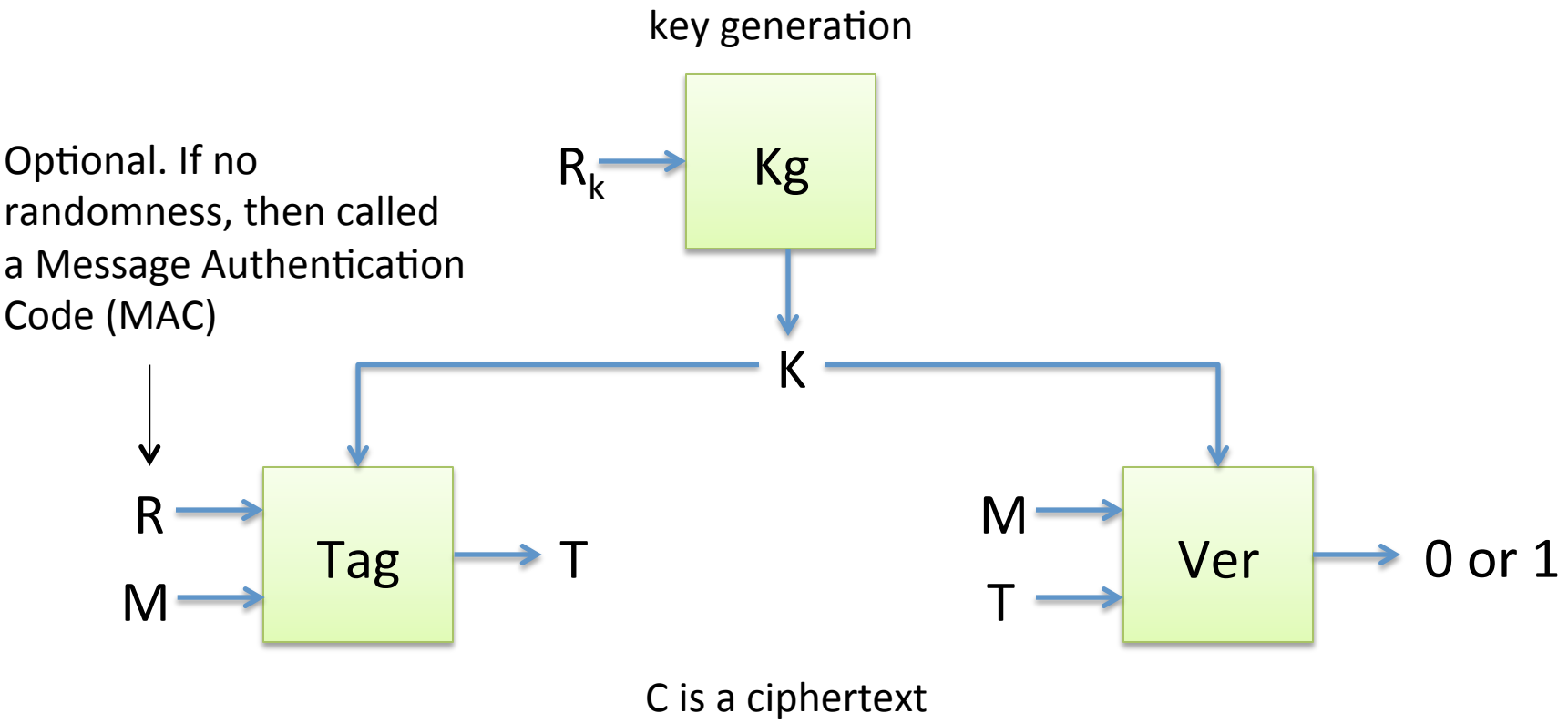
The idea: Attacker, given (salt,h), should not be able to recover pw

Or can they?

For each guess pw':
    If H(salt||pw') = h then
        Ret pw'

Rainbow tables speed this up in practice by way of precomputation. Large salts make rainbow tables impractical

Message authentication



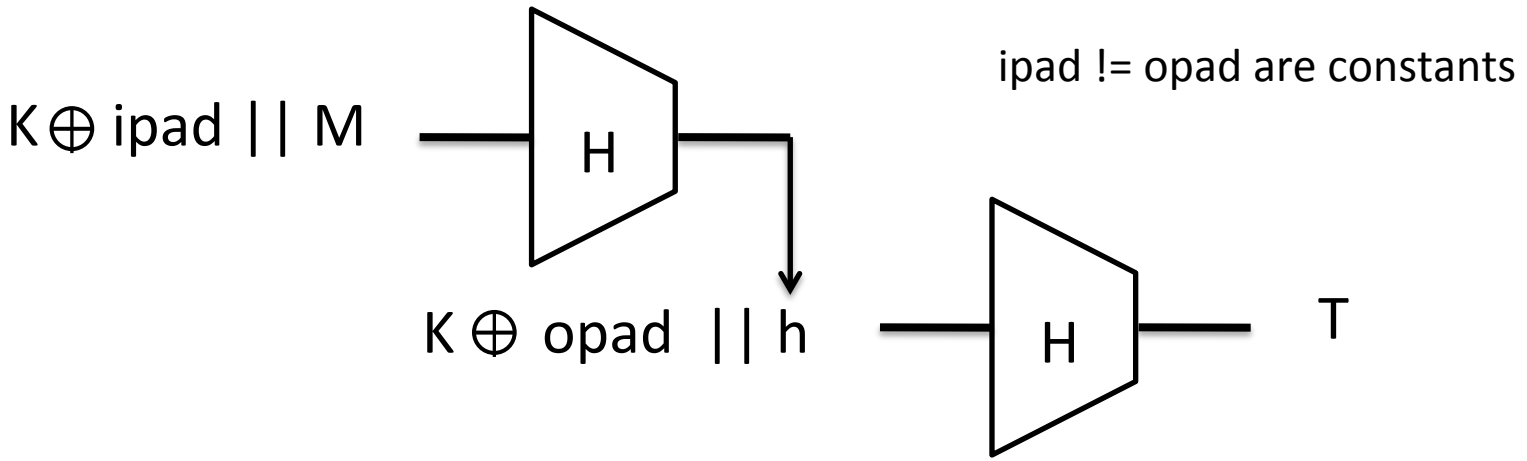
Correctness: Ver(K , Tag(K,M,R)) = 1 with probability 1 over randomness used

Unforgeability: Attacker can't find M',T such that V(K,M',T) = 1

Message authentication with HMAC

Use a hash function H to build MAC.
Kg outputs uniform bit string K

Tag(K,M) = HMAC(K,M) defined by:

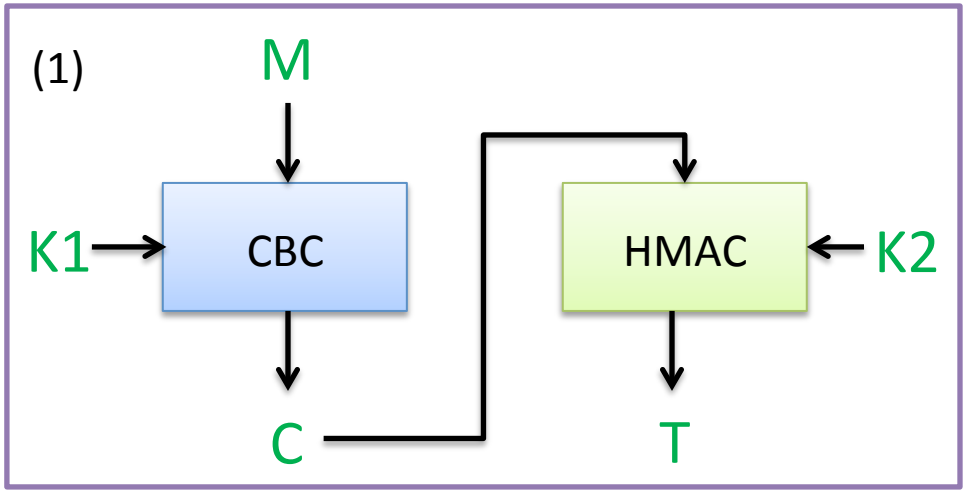


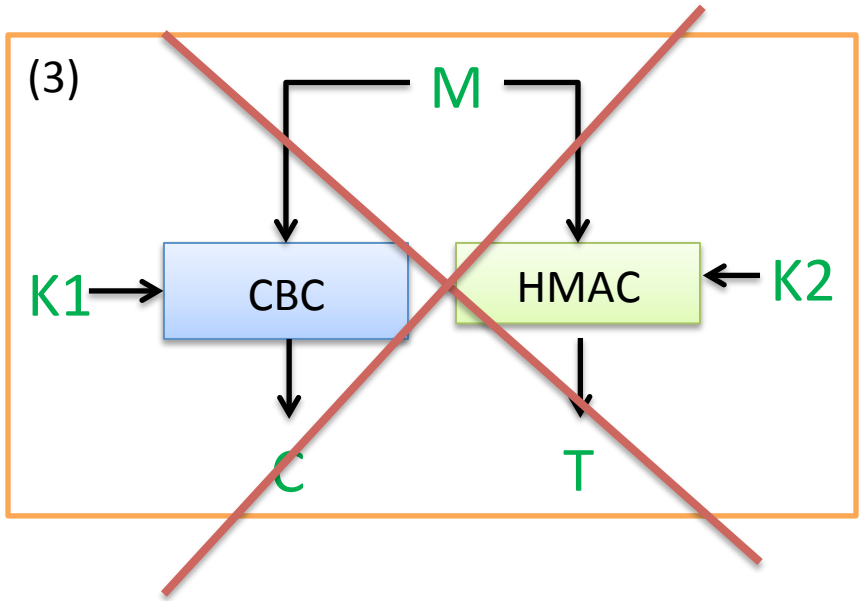
To verify a M,T pair, check if HMAC(K,M) = T

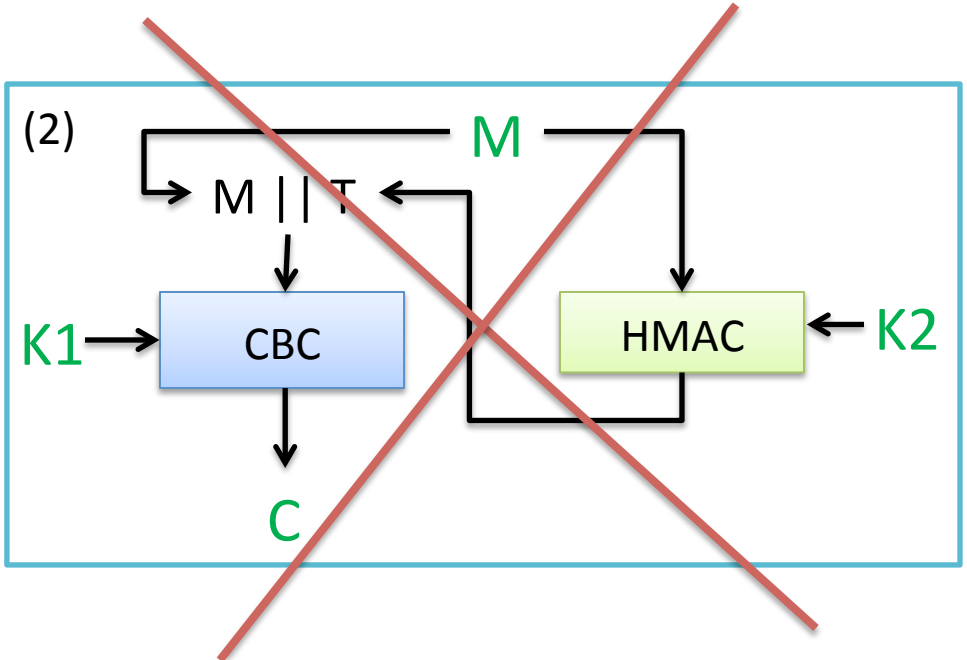
Unforgeability holds if H is a secure PRF when so-keyed

Build a new scheme from CBC and HMAC
Kg outputs CBC key K1 and HMAC key K2

Several ways to combine:
(1) encrypt-then-mac
(2) mac-then-encrypt
(3) encrypt-and-mac

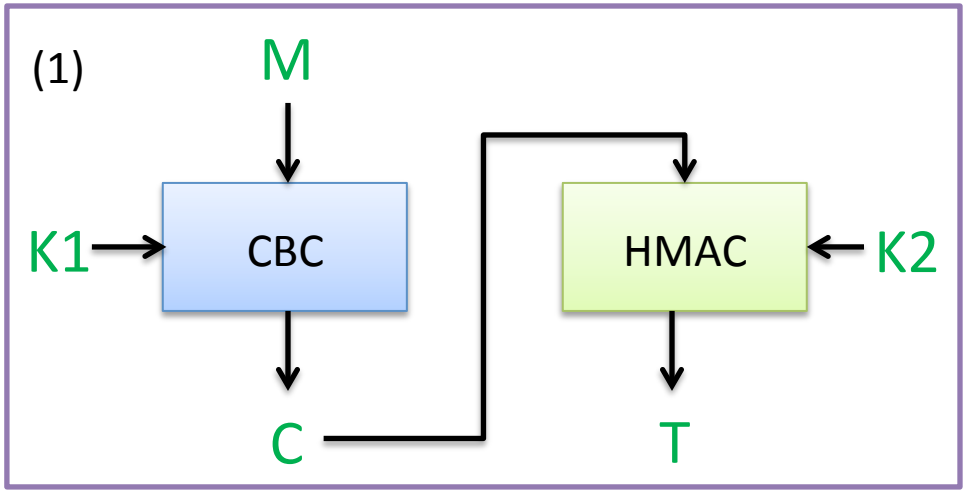






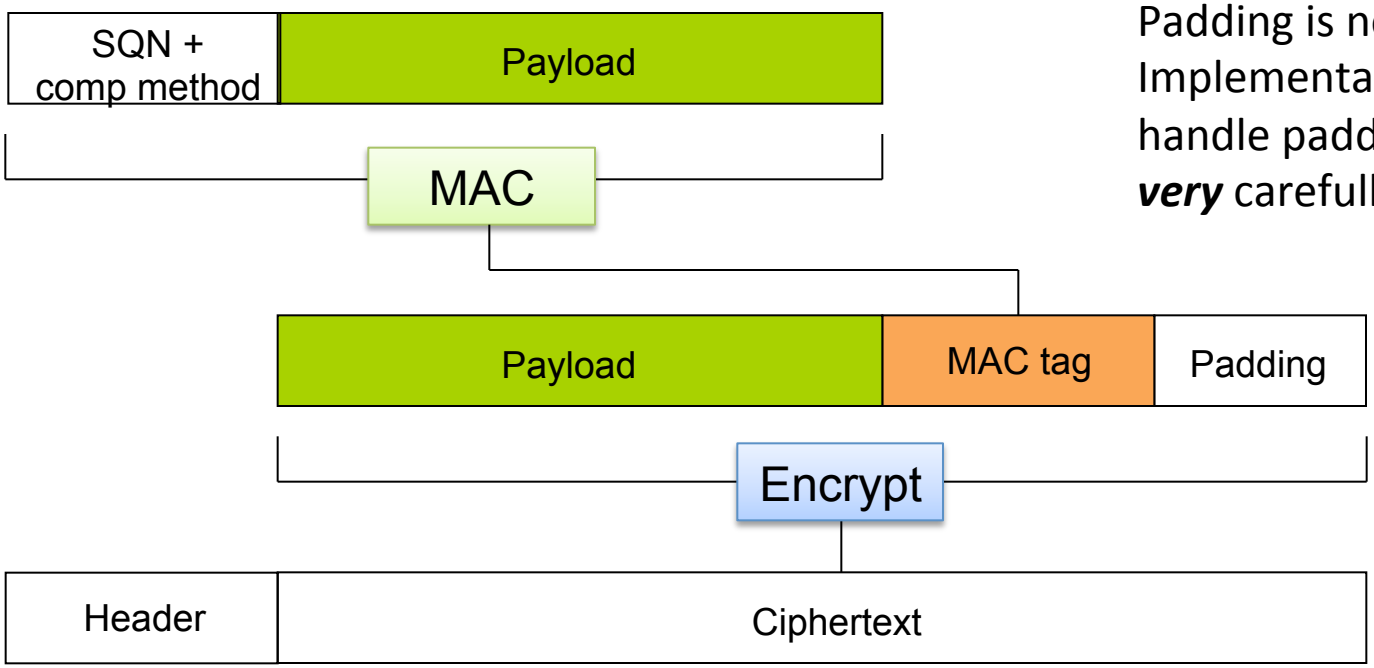
Build a new scheme from CBC and HMAC
Kg outputs CBC key K1 and HMAC key K2

Several ways to combine:
(1) encrypt-then-mac
(2) mac-then-encrypt
(3) encrypt-and-mac



Thm. If encryption scheme provides confidentiality against passive attackers and MAC provides unforgeability, then Encrypt-then-MAC provides secure authenticated encryption

TLS record protocol: MAC-Encode-Encrypt (MEE)



Padding is not MAC'd. Implementations must handle padding checks ***very*** carefully.

MAC	HMAC-MD5, HMAC-SHA1, HMAC-SHA256
Encrypt	CBC-AES128, CBC-AES256, CBC-3DES, RC4-128

Dedicated authenticated encryption schemes

Attack	Inventors	Notes
OCB (Offset Codebook)	Rogaway	One-pass
GCM (Galios Counter Mode)	McGrew, Viega	CTR mode plus specialized MAC
CWC	Kohno, Viega, Whiting	CTR mode plus Carter-Wegman MAC
CCM	Housley, Ferguson, Whiting	CTR mode plus CBC-MAC
EAX	Wagner, Bellare, Rogaway	CTR mode plus OMAC

Symmetric Encryption Advice

Never use CTR mode or CBC mode by themselves

Passive security is almost never good enough!!

Encrypt-then-MAC better than MAC-then-Encrypt, Encrypt and MAC

Dedicated modes that have been analyzed thoroughly are also good