GCM; The illegal attack

Mathias Hall-Andersen (rot256)

Pwnies @ Copenhagen University

2017

Before we start : Get Sage

Fill your disk

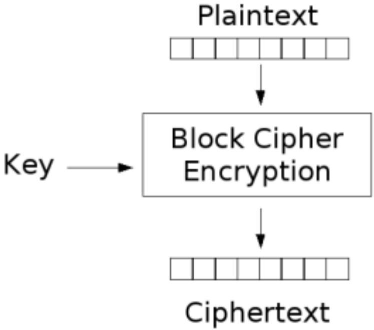
```
docker pull sagemath/sagemath
```

`https://www.sagemath.org/`

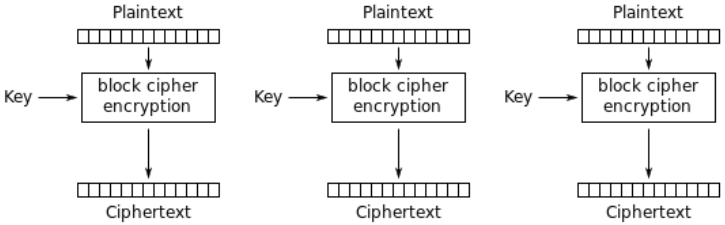
Or, sign up at

`https://cocalc.com/`

Mode of operation

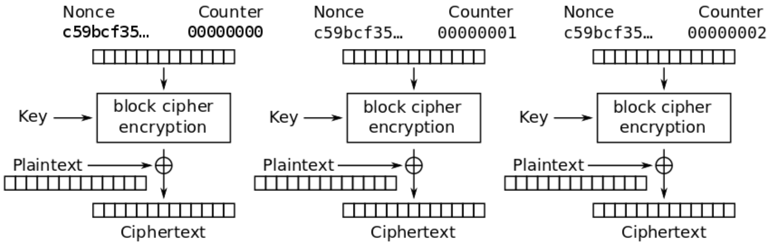


Mode of operation



Electronic Codebook (ECB) mode encryption

Mode of operation

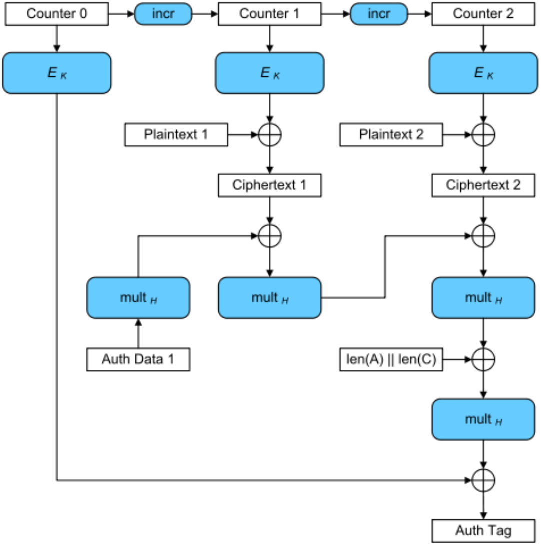


Counter (CTR) mode encryption

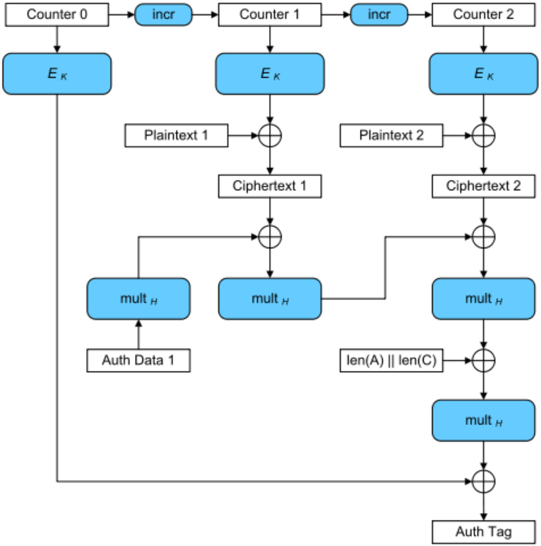
Authenticated encryption

Encryption $\neq$ Authentication

Galois Counter Mode (motivation)



Galois Counter Mode (motivation)



So what is $mult_H(\cdot)$?

Some algebra

$mult_H(\cdot)$ is multiplication by H,
but not quite the way you might think...

Rings

$$(E, \cdot, +, 0, 1)$$

Group + multiplication, e.i.

$$z \cdot (x + y) = z \cdot x + z \cdot y$$

$$\forall x : \exists y : x + y = 0$$

$$\forall x : x + 0 = x$$

$$\forall x : x \cdot 1 = x$$

Examples:

$$\mathbb{Z}$$

Question: How about $\mathbb{N}_{\geq 0}$?

Fields

$$(E, \cdot, +, 0, 1)$$

Ring $+$ multiplicative inverses, e.i.

$$\forall x \neq 0 : \exists y : x \cdot y = 1$$

Usually denote $y = x^{-1}$.
Examples:

$$\mathbb{Q}, \mathbb{R}$$

Question: How about $\mathbb{Z}$?

Finite fields

We will primarily be dealing with the field of two elements: $1, 0$
Where:

$$1 \cdot x = x : \text{ 1 is the multiplicative identity} \tag{1}$$

$$0 + x = x : \text{ 0 is the additive identity} \tag{2}$$

$$1 + 1 = 0 : \text{ the field has characteristic 2} \tag{3}$$

No magic. Question: If considered like bits, what common operations does addition and multiplication in the field correspond to? What implication does it have for bit-slicing techniques?

Polynomials over fields

Given a field. We may consider the polynomials over the field. Examples:

$$\mathbb{Z}$$

Fields

$$(E, \cdot, +, 0, 1)$$

Ring $+$ multiplicative inverses, e.i.

$$\forall x \exists y : x \cdot y = 1$$

Usually denote $y = x^{-1}$. Examples:

$$\mathbb{Q}, \mathbb{R}$$

Finite fields

We will primarily be dealing with the field of two elements: $1, 0$
Where:

$$1 \cdot x = x : \text{ 1 is the multiplicative identity} \tag{4}$$
$$0 + x = x : \text{ 0 is the additive identity} \tag{5}$$
$$1 + 1 = 0 : \text{ the field has characteristic 2} \tag{6}$$

No magic. Question: If considered like bits, what common operations does addition and multiplication in the field correspond to? What implication does it have for bit-slicing techniques?

Polynomials over fields

Given a field. We may consider the polynomials over the field.
E.g. for GF(2):

$$f(x) = x^7 + x^4 + x^1 + 1$$

$$g(x) = x^6 + x^3$$

We write this as $\mathbb{F}[x]$

Multiplication of polynomials over GF(2)

Lets focus on GF(2)[x].

Addition happens coefficient wise and multiplication also happens as you would expect:

$$f(x) = x^7 + x^4 + x^1 + 1$$

$$g(x) = x^6 + x^3$$

$$f(x) \cdot g(x) = x^6 f(x) + x^3 f(x) = (x^{13} + x^{10} + x^7 + x^6) + (x^{10} + x^7 + x^4 + x^3)$$
$$= x^{13} + x^6 + x^4 + x^3$$

Question: Recall the definition! Is this a ring?

Multiplication of polynomials over GF(2)

Lets focus on GF(2)[x].

$$f(x) = x^7 + x^4 + x^1 + 1$$
$$g(x) = x^6 + x^3$$

Question: We can represent the polynomials as bit strings, e.g. $f(x) \sim 10010011$, $g(x) \sim 01001000$ what does xor of bit strings correspond to in the ring? What does left shifting of bit strings correspond to?

From GF(2)[x] to GF(2^{128})

Take my word for it: We can transform the ring of polynomials into a field by reducing modulo a particular class of polynomials in the ring, so called ‘primitive’ polynomials.

For instance $GF(2)[x] \rightarrow GF(2^4)$, by reducing modulo x^2+1:
Since $f = (x^5+x^3+x^2+x+1)(x^2+1)$, $f \cong 0 \mod x^2+1$
$g = (x^4+x^2+x+1)\cdot(x^2+1)+(x+1)$, $g \cong x+1 \mod x^2+1$

From GF(2)[x] to GF(2^{128})

Reduction in GF(2)[x] / p(x). Easy!

Additional resources

So $mult_H(\cdot) : GF(2^{128}) \to GF(2^{128})$

Courses

- Algebra 1 @ Department of Mathematical Sciences (UCPH)
- Algebra 2 @ Department of Mathematical Sciences (UCPH)
- Computational Discrete Math @ DTU

Books

- Algebra (2nd Edition) : Micheal Artin
- Abstract Algebra : Dummit & Foote

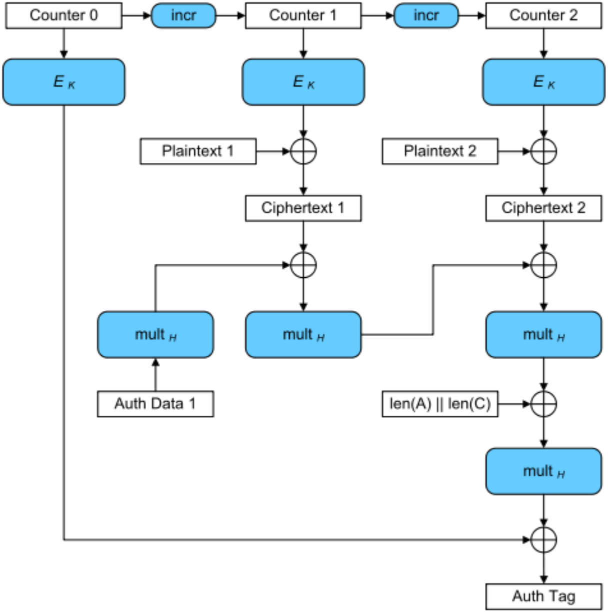
Break?

Galois Counter Mode (MAC calculation)

Authentication key $H \in GF(2^{128})$.
Output blocks: $ct_1, ct_2, \ldots, ct_n$ (last padded with zero)
Make a block: $len(A) || len(C)$ (length in bits)

For every block add, then multiply with the authentication key.
Basically: `foldl (\a x -> (a + x) * H) 0 cs`

Finally add a random (encrypted nonce || 0): E_k


Counter 0
incr
Counter 1
incr
Counter 2
E K
E K
E K
Plaintext 1
Plaintext 2
Ciphertext 1
Ciphertext 2
mult H
mult H
mult H
Auth Data 1
len(A) || len(C)
mult H
Auth Tag

Galois Counter Mode (MAC calculation)

Alternatively we can consider it as evaluation the polynomial:

$$w(y) = c_1 y^n + \ldots + c_n y^2 + c_{n+1} y^1 + E_k \in GF(2^{128})[y]$$

At H, e.i. $T = w(H)$. This will be useful.

Galois Counter Mode (MAC calculation)

Consider the case where, the MAC is computed on two different messages c, c', but $E_k = E'_k$. We know:

$$T = w(H) = c_1 H^n + \ldots + c_n H^2 + c_{n+1} H^1 + E_k$$

$$T' = w'(H) = c'_1 H^m + \ldots + c'_m H^2 + c'_{m+1} H^1 + E_k$$

Move terms over:

$$0 = c_1 H^n + \ldots + c_n H^2 + c_{n+1} H^1 + E_k + w(H)$$

$$0 = c'_1 H^m + \ldots + c'_m H^2 + c'_{m+1} H^1 + E_k + w'(H)$$

Galois Counter Mode (MAC calculation)

$$0 = c_1 H^n + \ldots + c_n H^2 + c_{n+1} H^1 + E_k + w(H)$$
$$0 = c'_1 H^m + \ldots + c'_m H^2 + c'_{m+1} H^1 + E_k + w'(H)$$

Subtract:

$$\begin{aligned}
0 &= (c_1 H^n + \ldots + c_n H^2 + c_{n+1} H^1 + E_k + w(H)) - && (7)\\
&\quad (c'_1 H^m + \ldots + c'_m H^2 + c'_{m+1} H^1 + E_k + w'(H)) && (8)\\
&= (c_1 H^n + \ldots + c_n H^2 + c_{n+1} H^1 + w(H)) + && (9)\\
&\quad (c'_1 H^m + \ldots + c'_m H^2 + c'_{m+1} H^1 + w'(H)) && (10)
\end{aligned}$$

So H is a root of $(w(y) - T) - (w'(y) - T')$. Easy?

SageMath

'SageMath is a free open-source mathematics software system licensed under the GPL. It builds on top of many existing open-source packages: NumPy, SciPy, matplotlib, Sympy, Maxima, GAP, FLINT, R and many more' - https://www.sagemath.org/

Work session

Go to `http://rot256.io:8080` for the challenge.

The algebraic structures we will be needing in Sage:

```
F.<x> = GF(2^128, 'x', x^128 + x^7 + x^2 + x + 1)
G.<y> = PolynomialRing(F)
```

There is a `doit.sage` template at
`https://rot256.io/doit.sage`.
Containing useful helpers if you wish to use these in the attack.