Cycling Attacks on GCM, GHASH and Other Polynomial MACs and Hashes

Markku-Juhani O. Saarinen
mjos@reveresecurity.com

REVERE SECURITY
4500 Westgrove Drive, Suite 335
Addison, TX 75001, USA

FSE 2012
Washington D.C.
20 March 2012

Galois / Counter Mode

Let C be a concatenation of optional unencrypted authenticated data, CTR-encrypted ciphertext, and padding. This data is split into m 128-bit blocks C_i:

$$C = C_1 \mid\mid C_2 \mid\mid \cdots \mid\mid C_m.$$

The authentication code GHASH is based on operations in $GF(2^{128})$. Horner's rule is used in this field to evaluate polynomial Y. The authentication key is $H = E_K(0)$.

$$Y_m = \sum_{i=1}^{m} C_i \otimes H^{m-i+1}.$$

The final authentication tag is $T = Y_m \oplus E_K(IV \mid\mid 0^{31}1)$, assuming a 96-bit IV.

Galois / Counter Mode

Let C be a concatenation of optional unencrypted authenticated data, CTR-encrypted ciphertext, and padding. This data is split into m 128-bit blocks C_i:

$$C = C_1 \mid\mid C_2 \mid\mid \cdots \mid\mid C_m.$$

The authentication code GHASH is based on operations in $\mathrm{GF}(2^{128})$. Horner's rule is used in this field to evaluate polynomial Y. The authentication key is $H = E_K(0)$.

$$Y_m = \sum_{i=1}^{m} C_i \otimes H^{m-i+1}.$$

The final authentication tag is $T = Y_m \oplus E_K(IV \mid\mid 0^{31}1)$, assuming a 96-bit IV.

Galois / Counter Mode

Let C be a concatenation of optional unencrypted authenticated data, CTR-encrypted ciphertext, and padding. This data is split into m 128-bit blocks C_i:

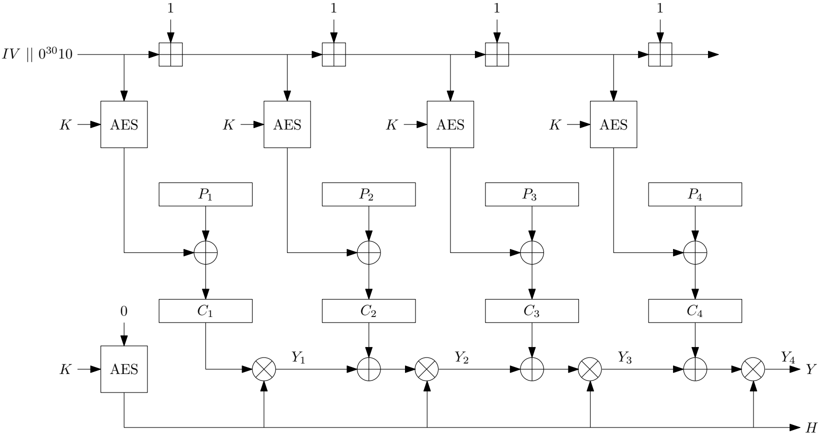
$$C = C_1 \mid\mid C_2 \mid\mid \cdots \mid\mid C_m.$$

The authentication code GHASH is based on operations in GF(2^{128}). Horner's rule is used in this finite field to evaluate polynomial Y. The authentication key is $H = E_K(0)$.

$$Y_m = \sum_{i=1}^{m} C_i \otimes H^{m-i+1}.$$

The final authentication tag is $T = Y_m \oplus E_K(IV \mid\mid 0^{31}1)$, assuming a 96-bit IV.

Four rounds of AES-GCM.



Known Attacks

- Known to be trivially breakable with a repeated IV (Joux's 2004 "Forbidden Attack"). Therefore poorly suited for connectionless protocols.
- Ferguson (2005) showed that an n-bit tag provides only $n - k$ bits of authentication security when messages are 2^k blocks long.
- Hence GCM was already known to be significantly weaker than, say, HMAC-MD5 (which still has the expected 2^{-n} security in "unknown-start-value" mode) prior to its standardization in NIST SP 800-38D.
- Despite these shortcomings and apparently due to industry endorsement and its excellent hardware performance, AES-GCM was adopted as part of NSA's "Suite B" in 2007 and may still be used to secure classified data.

Known Attacks

- Known to be trivially breakable with a repeated IV (Joux's 2004 "Forbidden Attack"). Therefore poorly suited for connectionless protocols.
- Ferguson (2005) showed that an n-bit tag provides only $n - k$ bits of authentication security when messages are 2^k blocks long.
- Hence GCM was already known to be significantly weaker than, say, HMAC-MD5 (which still has the expected 2^{-n} security in "unknown-start-value" mode) prior to its standardization in NIST SP 800-38D.
- Despite these shortcomings and apparently due to industry endorsement and its excellent hardware performance, AES-GCM was adopted as part of NSA's "Suite B" in 2007 and may still be used to secure classified data.

Known Attacks

- Known to be trivially breakable with a repeated IV (Joux's 2004 "Forbidden Attack"). Therefore poorly suited for connectionless protocols.
- Ferguson (2005) showed that an n-bit tag provides only $n - k$ bits of authentication security when messages are 2^k blocks long.
- Hence GCM was already known to be significantly weaker than, say, HMAC-MD5 (which still has the expected 2^{-n} security in "unknown-start-value" mode) prior to its standardization in NIST SP 800-38D.
- Despite these shortcomings and apparently due to industry endorsement and its excellent hardware performance, AES-GCM was adopted as part of NSA's "Suite B" in 2007 and may still be used to secure classified data.

Known Attacks

- Known to be trivially breakable with a repeated IV (Joux's 2004 "Forbidden Attack"). Therefore poorly suited for connectionless protocols.
- Ferguson (2005) showed that an n-bit tag provides only $n - k$ bits of authentication security when messages are 2^k blocks long.
- Hence GCM was already known to be significantly weaker than, say, HMAC-MD5 (which still has the expected 2^{-n} security in "unknown-start-value" mode) prior to its standardization in NIST SP 800-38D.
- Despite these shortcomings and apparently due to industry endorsement and its excellent hardware performance, AES-GCM was adopted as part of NSA's "Suite B" in 2007 and may still be used to secure classified data.

Four rounds of AES-GCM.

Horner's iteration:

$$
\begin{aligned}
Y_1 &= C_1 \times H \\
Y_2 &= (Y_1 + C_2) \times H = C_1 \times H^2 + C_2 \times H \\
Y_3 &= (Y_2 + C_3) \times H = C_1 \times H^3 + C_2 \times H^2 + C_3 \times H \\
Y_4 &= (Y_3 + C_4) \times H = C_1 \times H^4 + C_2 \times H^3 + C_3 \times H^2 + C_4 \times H.
\end{aligned}
$$

What if, say, $H = H^4$? Then we may just swap C_1 and C_4 and the Y_4 value will remain unchanged:

$$Y_4 = C_4 \times H^4 + C_2 \times H^2 + C_3 \times H^2 + C_1 \times H.$$

A cycle will lead to a forgery attack.

Four rounds of AES-GCM.

Horner's iteration:

$$
\begin{aligned}
Y_1 &= C_1 \times H \\
Y_2 &= (Y_1 + C_2) \times H = C_1 \times H^2 + C_2 \times H \\
Y_3 &= (Y_2 + C_3) \times H = C_1 \times H^3 + C_2 \times H^2 + C_3 \times H \\
Y_4 &= (Y_3 + C_4) \times H = C_1 \times H^4 + C_2 \times H^3 + C_3 \times H^2 + C_4 \times H.
\end{aligned}
$$

What if, say, $H = H^4$? Then we may just swap C_1 and C_4 and the Y_4 value will remain unchanged:

$$Y_4 = C_4 \times H^4 + C_2 \times H^2 + C_3 \times H^2 + C_1 \times H.$$

A cycle will lead to a forgery attack.

Four rounds of AES-GCM.

Horner's iteration:

$$Y_1 = C_1 \times H$$
$$Y_2 = (Y_1 + C_2) \times H = C_1 \times H^2 + C_2 \times H$$
$$Y_3 = (Y_2 + C_3) \times H = C_1 \times H^3 + C_2 \times H^2 + C_3 \times H$$
$$Y_4 = (Y_3 + C_4) \times H = C_1 \times H^4 + C_2 \times H^3 + C_3 \times H^2 + C_4 \times H.$$

What if, say, $H = H^4$? Then we may just swap C_1 and C_4 and the Y_4 value will remain unchanged:

$$Y_4 = C_4 \times H^4 + C_2 \times H^2 + C_3 \times H^2 + C_1 \times H.$$

A cycle will lead to a forgery attack.

Four rounds of AES-GCM.

Horner's iteration:

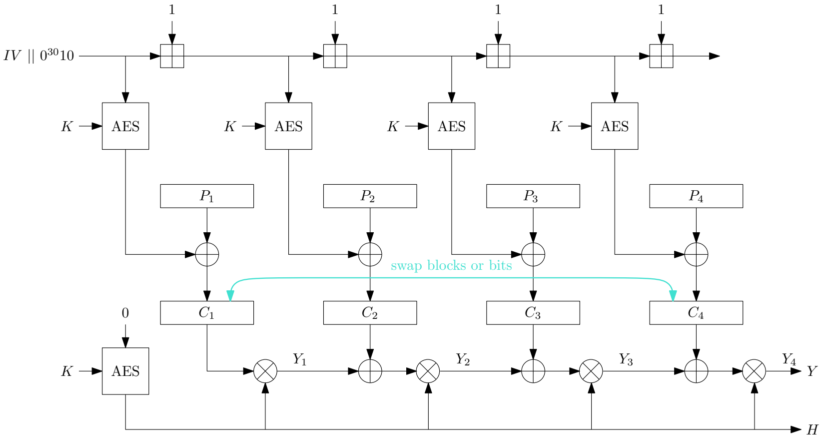
$$
\begin{aligned}
Y_1 &= C_1 \times H \\
Y_2 &= (Y_1 + C_2) \times H = C_1 \times H^2 + C_2 \times H \\
Y_3 &= (Y_2 + C_3) \times H = C_1 \times H^3 + C_2 \times H^2 + C_3 \times H \\
Y_4 &= (Y_3 + C_4) \times H = C_1 \times H^4 + C_2 \times H^3 + C_3 \times H^2 + C_4 \times H.
\end{aligned}
$$

What if, say, $H = H^4$? Then we may just swap C_1 and C_4 and the Y_4 value will remain unchanged:

$$Y_4 = C_4 \times H^4 + C_2 \times H^2 + C_3 \times H^2 + C_1 \times H.$$

A cycle will lead to a forgery attack.

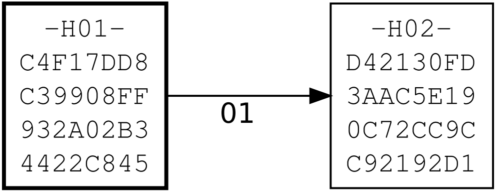
Switching Full Blocks



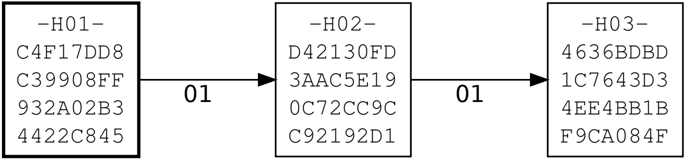
Start With a $H^1 = \mathrm{AES}_k(0)$ for some k.

$\boxed{\begin{array}{c}\texttt{-H01-}\\ \texttt{C4F17DD8}\\ \texttt{C39908FF}\\ \texttt{932A02B3}\\ \texttt{4422C845}\end{array}}$

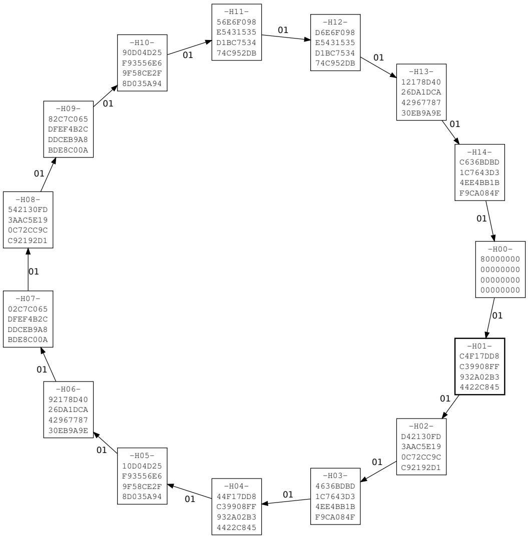
Generate $H^2 = H \times H$ from it



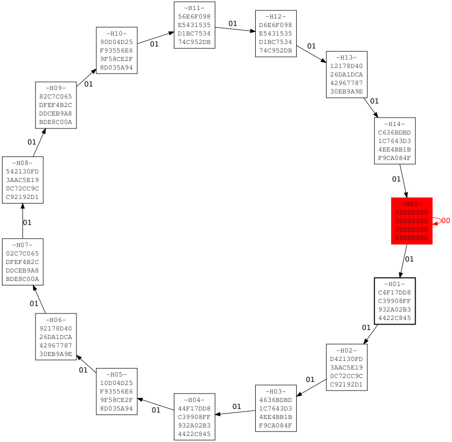
.. and H^3 from $H \times H^2$ ..



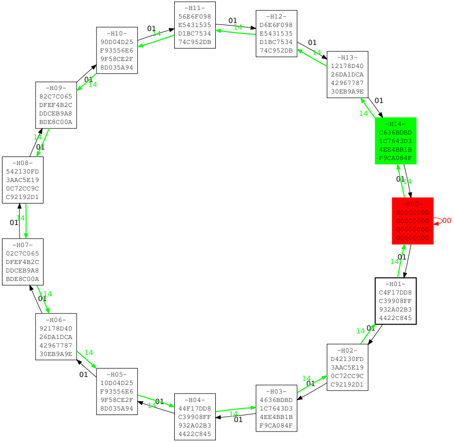
Wow! $H^{16} = h^1$ again.



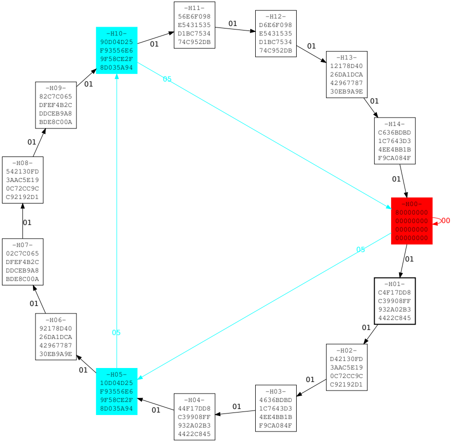
Hence $H^0 = H^{15}$. It's the unique identity element with cycle length 1.



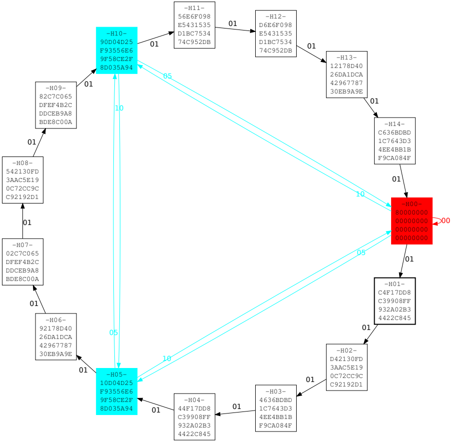
This subgroup is isomorphic to addition in $\mathbb{Z}_{15}$. $H' = H^{14}$ will generate the same cycle backwards.



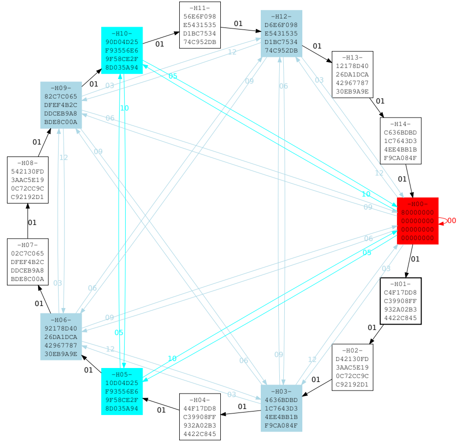
If we skip over 4 (add 5 mod 15), we will get back in 3 steps.



This can also be generated backwards with $H' = H^{10}$.



Since $15 = 3 \times 5$, there's also an unique subgroup of size 5.



Elementary Number Theory & Abstract Algebra 101

- The (full) multiplicative group of $GF(2^{128})$ is isomorphic to the additive group $\mathbb{Z}_{2^{128}-1}$ (all elements except 0).
- There are subgroups of size n for any $n \mid 2^{128} - 1$.
- $2^{128} - 1 = 3 * 5 * 17 * 257 * 641 * 65537 * 274177 * 6700417 * 67280421310721$ – nine prime factors.
- Hence there are $2^9 = 512$ different-sized subgroups, almost log-uniformly distributed in the range.

Theorem.
Let n be a number satisfying $\gcd(2^{128} - 1, n) = n$. Blindly swapping blocks C_i and C_j, where $i \equiv j \pmod{n}$ will result in a successful forgery with probability of at least $\frac{n+1}{2^{128}}$ if H is random.

Elementary Number Theory & Abstract Algebra 101

- The (full) multiplicative group of $GF(2^{128})$ is isomorphic to the additive group $\mathbb{Z}_{2^{128}-1}$ (all elements except 0).
- There are subgroups of size n for any $n \mid 2^{128} - 1$.
- $2^{128} - 1 = 3 * 5 * 17 * 257 * 641 * 65537 * 274177 * 6700417 * 67280421310721$ – nine prime factors.
- Hence there are $2^9 = 512$ different-sized subgroups, almost log-uniformly distributed in the range.

Theorem.
Let n be a number satisfying $\gcd(2^{128} - 1, n) = n$. Blindly swapping blocks C_i and C_j, where $i \equiv j \pmod{n}$ will result in a successful forgery with probability of at least $\frac{n+1}{2^{128}}$ if H is random.

Elementary Number Theory & Abstract Algebra 101

- The (full) multiplicative group of $GF(2^{128})$ is isomorphic to the additive group $\mathbb{Z}_{2^{128}-1}$ (all elements except 0).
- There are subgroups of size n for any $n \mid 2^{128}-1$.
- $2^{128}-1 = 3*5*17*257*641*65537*274177*6700417*67280421310721$ – nine prime factors.
- Hence there are $2^9 = 512$ different-sized subgroups, almost log-uniformly distributed in the range.

Theorem.
Let n be a number satisfying $\gcd(2^{128}-1, n) = n$. *Blindly swapping blocks* C_i *and* C_j, *where* $i \equiv j \pmod{n}$ *will result in a successful forgery with probability of at least* $\frac{n+1}{2^{128}}$ *if H is random.*

Elementary Number Theory & Abstract Algebra 101

- The (full) multiplicative group of $GF(2^{128})$ is isomorphic to the additive group $\mathbb{Z}_{2^{128}-1}$ (all elements except 0).
- There are subgroups of size n for any $n \mid 2^{128} - 1$.
- $2^{128} - 1 = 3 * 5 * 17 * 257 * 641 * 65537 * 274177 * 6700417 * 67280421310721$ – nine prime factors.
- Hence there are $2^9 = 512$ different-sized subgroups, almost log-uniformly distributed in the range.

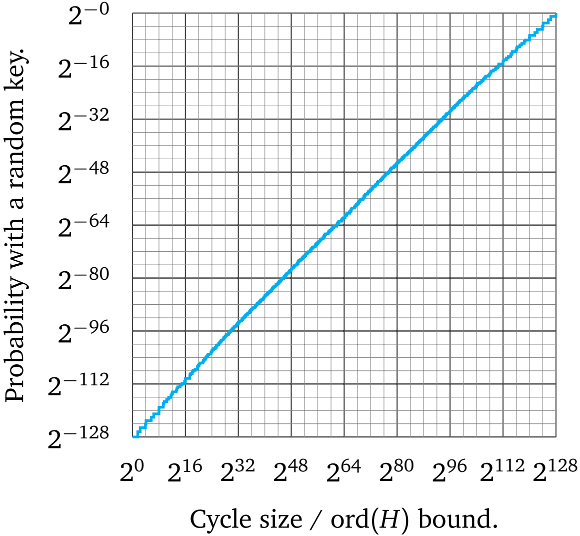
Theorem.
Let n be a number satisfying $\gcd(2^{128} - 1, n) = n$. Blindly swapping blocks C_i and C_j, where $i \equiv j \pmod{n}$ will result in a successful forgery with probability of at least $\frac{n+1}{2^{128}}$ if H is random.

Elementary Number Theory & Abstract Algebra 101

- The (full) multiplicative group of $GF(2^{128})$ is isomorphic to the additive group $\mathbb{Z}_{2^{128}-1}$ (all elements except 0).
- There are subgroups of size n for any $n \mid 2^{128} - 1$.
- $2^{128} - 1 = 3 * 5 * 17 * 257 * 641 * 65537 * 274177 * 6700417 * 67280421310721$ – nine prime factors.
- Hence there are $2^9 = 512$ different-sized subgroups, almost log-uniformly distributed in the range.

Theorem.
Let n be a number satisfying $\gcd(2^{128} - 1, n) = n$. Blindly swapping blocks C_i and C_j, where $i \equiv j \pmod{n}$ will result in a successful forgery with probability of at least $\frac{n+1}{2^{128}}$ if H is random.

Probability vs Length is Almost Log-Linear



Multiforgery Attack

- The H value depends solely on the AES key, which may be a fixed key or something from a key exchange algorithm.
- If a cycle of n is detected, **any number** of subsequent forgeries can be performed with probability $P = 1$.
- The *average* complexity of an individual forgery can be made **arbitrarily small** (compare to *multicollision attacks*) if we assume an attack model **FRK** where the advisory can force rekeying until a successful forgery occurs.
- Note that **FRK** is a reasonably realistic model in real-world VPN protocols which disconnect and rekey immediately on a MAC mismatch. Under this model the security bound of the proof is broken (in the average case).

Multiforgery Attack

- The H value depends solely on the AES key, which may be a fixed key or something from a key exchange algorithm.
- If a cycle of n is detected, **any number** of subsequent forgeries can be performed with probability $P = 1$.
- The *average* complexity of an individual forgery can be made **arbitrarily small** (compare to *multicollision attacks*) if we assume an attack model **FRK** where the advisory can force rekeying until a successful forgery occurs.
- Note that **FRK** is a reasonably realistic model in real-world VPN protocols which disconnect and rekey immediately on a MAC mismatch. Under this model the security bound of the proof is broken (in the average case).

Multiforgery Attack

- The H value depends solely on the AES key, which may be a fixed key or something from a key exchange algorithm.
- If a cycle of n is detected, **any number** of subsequent forgeries can be performed with probability $P = 1$.
- The *average* complexity of an individual forgery can be made **arbitrarily small** (compare to *multicollision attacks*) if we assume an attack model **FRK** where the advisory can force rekeying until a successful forgery occurs.
- Note that **FRK** is a reasonably realistic model in real-world VPN protocols which disconnect and rekey immediately on a MAC mismatch. Under this model the security bound of the proof is broken (in the average case).

Multiforgery Attack

- The H value depends solely on the AES key, which may be a fixed key or something from a key exchange algorithm.
- If a cycle of n is detected, **any number** of subsequent forgeries can be performed with probability $P = 1$.
- The *average* complexity of an individual forgery can be made **arbitrarily small** (compare to *multicollision attacks*) if we assume an attack model **FRK** where the advisory can force rekeying until a successful forgery occurs.
- Note that **FRK** is a reasonably realistic model in real-world VPN protocols which disconnect and rekey immediately on a MAC mismatch. Under this model the security bound of the proof is broken (in the average case).

Any Number of Targeted Bit Forgeries

Counter mode behaves **like a stream cipher**; flipping a ciphertext bit will result in the corresponding plaintext bit being flipped after decryption.

If $\mathrm{ord}(H) \mid (i - j)$ the authentication tag will remain valid as long as the following equation holds (for some c):

$$C_i \times H^{m-i+1} + C_j \times H^{m-j+1} = c.$$

Writing $H^{m-i+1} = H^{m-j+1} = H_c$, this can be simplified to

$$C_i + C_j = c \times H_c^{-1}.$$

The tag will be valid if the XOR sum of ciphertext blocks on the left side remains constant. We may manipulate **any number of specific target bits** by appropriately compensating them.

Any Number of Targeted Bit Forgeries

Counter mode behaves **like a stream cipher**; flipping a ciphertext bit will result in the corresponding plaintext bit being flipped after decryption.

If $\mathrm{ord}(H) \mid (i-j)$ the authentication tag will remain valid as long as the following equation holds (for some c):

$$C_i \times H^{m-i+1} + C_j \times H^{m-j+1} = c.$$

Writing $H^{m-i+1} = H^{m-j+1} = H_c$, this can be simplified to

$$C_i + C_j = c \times H_c^{-1}.$$

The tag will be valid if the XOR sum of ciphertext blocks on the left side remains constant. We may manipulate **any number of specific target bits** by appropriately compensating them.

Any Number of Targeted Bit Forgeries

Counter mode behaves **like a stream cipher**; flipping a ciphertext bit will result in the corresponding plaintext bit being flipped after decryption.

If $\mathrm{ord}(H) \mid (i - j)$ the authentication tag will remain valid as long as the following equation holds (for some c):

$$C_i \times H^{m-i+1} + C_j \times H^{m-j+1} = c.$$

Writing $H^{m-i+1} = H^{m-j+1} = H_c$, this can be simplified to

$$C_i + C_j = c \times H_c^{-1}.$$

The tag will be valid if the XOR sum of ciphertext blocks on the left side remains constant. We may manipulate **any number of specific target bits** by appropriately compensating them.

Any Number of Targeted Bit Forgeries

Counter mode behaves **like a stream cipher**; flipping a ciphertext bit will result in the corresponding plaintext bit being flipped after decryption.

If $\mathrm{ord}(H) \mid (i-j)$ the authentication tag will remain valid as long as the following equation holds (for some c):

$$C_i \times H^{m-i+1} + C_j \times H^{m-j+1} = c.$$

Writing $H^{m-i+1} = H^{m-j+1} = H_c$, this can be simplified to

$$C_i + C_j = c \times H_c^{-1}.$$

The tag will be valid if the XOR sum of ciphertext blocks on the left side remains constant. We may manipulate **any number of specific target bits** by appropriately compensating them.

Secure Fields

- For polynomial authentication, use either:
  1. $GF(p)$ prime fields with $(p-1)/2$ also a prime. These are called **Sophie Germain** prime fields. If $H \notin \{0, 1, p-1\}$ the cycle is $(p-1)$ or $(p-1)/2$, depending on the quadratic residuosity (Legendre symbol) of H.

     .. or ..
  2. $GF(2^p)$ binary fields with $2^p - 1$ a prime. These may be called **Mersenne** binary fields. If $H \notin \{0, 1\}$, the cycle is $2^p - 1$.
- However, an n-bit MAC **can** and **should** have 2^{-n} security against forgery. Polynomial MACs do not have that.
- Remember: A good MAC should also be able to resist repeated-IV attacks. These polynomial MACs do not resist them.

Secure Fields

- For polynomial authentication, use either:
  1. $GF(p)$ prime fields with $(p-1)/2$ also a prime. These are called **Sophie Germain** prime fields. If $H \notin \{0, 1, p-1\}$ the cycle is $(p-1)$ or $(p-1)/2$, depending on the quadratic residuosity (Legendre symbol) of H.

     .. *or* ..

  2. $GF(2^p)$ binary fields with $2^p - 1$ a prime. These may be called **Mersenne** binary fields. If $H \notin \{0, 1\}$, the cycle is $2^p - 1$.
- However, an n-bit MAC **can** and **should** have 2^{-n} security against forgery. Polynomial MACs do not have that.
- Remember: A good MAC should also be able to resist repeated-IV attacks. These polynomial MACs do not resist them.

Secure Fields

- For polynomial authentication, use either:
  1. $GF(p)$ prime fields with $(p-1)/2$ also a prime. These are called **Sophie Germain** prime fields. If $H \notin \{0, 1, p-1\}$ the cycle is $(p-1)$ or $(p-1)/2$, depending on the quadratic residuosity (Legendre symbol) of H.

     .. *or* ..
  2. $GF(2^p)$ binary fields with $2^p - 1$ a prime. These may be called **Mersenne** binary fields. If $H \notin \{0, 1\}$, the cycle is $2^p - 1$.
- However, an n-bit MAC **can** and **should** have 2^{-n} security against forgery. Polynomial MACs do not have that.
- Remember: A good MAC should also be able to resist repeated-IV attacks. These polynomial MACs do not resist them.

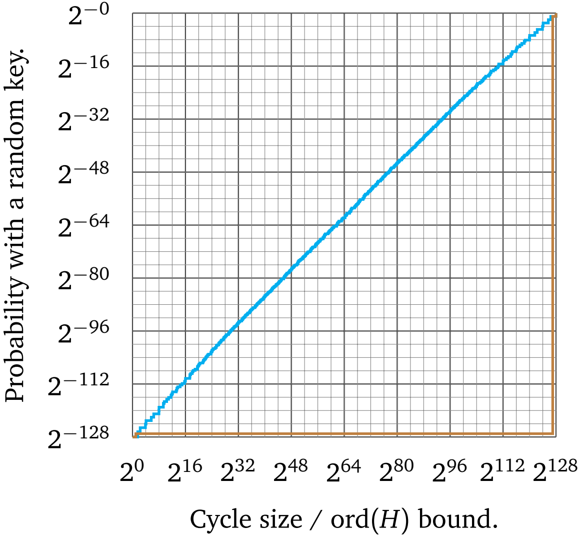
Secure Fields

- For polynomial authentication, use either:
  1. $GF(p)$ prime fields with $(p-1)/2$ also a prime. These are called **Sophie Germain** prime fields. If $H \notin \{0, 1, p-1\}$ the cycle is $(p-1)$ or $(p-1)/2$, depending on the quadratic residuosity (Legendre symbol) of H.

     .. or ..

  2. $GF(2^p)$ binary fields with $2^p - 1$ a prime. These may be called **Mersenne** binary fields. If $H \notin \{0, 1\}$, the cycle is $2^p - 1$.
- However, an n-bit MAC **can** and **should** have 2^{-n} security against forgery. Polynomial MACs do not have that.
- Remember: A good MAC should also be able to resist repeated-IV attacks. These polynomial MACs do not resist them.

Some Fields Are Much Better! $GF(2^{128})$ vs $GF(2^{127})$



Testing for AES-GCM Weak Keys

- Finding weak H values is easy, so a natural question arises on how to determine weak AES keys K that produce these weak H roots.
- To determine group order, we use a simple algorithm which is related to the Silver-Pohlig-Hellman algorithm for discrete logarithms [PoHe78].
- The algorithm can be made especially fast due to the linear nature of binary field squaring.
- Raising to "Fermat exponents" $2^n + 1$ (as $2^{128} - 1$ factors into Fermat numbers) involves repeated squarings and a single multiplication. The X^{2^n} tables do not depend on the particular H value.

Testing for AES-GCM Weak Keys

- Finding weak H values is easy, so a natural question arises on how to determine weak AES keys K that produce these weak H roots.
- To determine group order, we use a simple algorithm which is related to the Silver-Pohlig-Hellman algorithm for discrete logarithms [PoHe78].
- The algorithm can be made especially fast due to the linear nature of binary field squaring.
- Raising to "Fermat exponents" $2^n + 1$ (as $2^{128} - 1$ factors into Fermat numbers) involves repeated squarings and a single multiplication. The X^{2^n} tables do not depend on the particular H value.

Testing for AES-GCM Weak Keys

- Finding weak H values is easy, so a natural question arises on how to determine weak AES keys K that produce these weak H roots.
- To determine group order, we use a simple algorithm which is related to the Silver-Pohlig-Hellman algorithm for discrete logarithms [PoHe78].
- The algorithm can be made especially fast due to the linear nature of binary field squaring.
- Raising to "Fermat exponents" $2^n + 1$ (as $2^{128} - 1$ factors into Fermat numbers) involves repeated squarings and a single multiplication. The X^{2^n} tables do not depend on the particular H value.

Testing for AES-GCM Weak Keys

- Finding weak H values is easy, so a natural question arises on how to determine weak AES keys K that produce these weak H roots.
- To determine group order, we use a simple algorithm which is related to the Silver-Pohlig-Hellman algorithm for discrete logarithms [PoHe78].
- The algorithm can be made especially fast due to the linear nature of binary field squaring.
- Raising to "Fermat exponents" $2^n + 1$ (as $2^{128} - 1$ factors into Fermat numbers) involves repeated squarings and a single multiplication. The X^{2^n} tables do not depend on the particular H value.

Experimental Results

Over couple of days I tested 2^{32} AES-128 keys on my laptop and found progressively smaller subgroups:

$n \approx 2^{126.4}$ $K =$ `00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02`

$n \approx 2^{125.6}$ $K =$ `00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03`

$\dots$

$n \approx 2^{96.52}$ $K =$ `00 00 00 00 00 00 00 00 00 00 00 00 24 3E 8B 40`

$n \approx 2^{96.00}$ $K =$ `00 00 00 00 00 00 00 00 00 00 00 00 37 48 CF CE`

$n \approx 2^{93.93}$ $K =$ `00 00 00 00 00 00 00 00 00 00 00 00 42 87 3C C8`

$n \approx 2^{93.41}$ $K =$ `00 00 00 00 00 00 00 00 00 00 00 00 EC 69 7A A8`

Here $n = \mathrm{ord}(\mathrm{AES}_K(0))$. The groups size shrinks slightly faster than the keyspace is exhausted (as expected).

Concluding

- Since the authenticator H is derived as $H = AES_k(0)$ and there are plenty of low-order roots of unity in $GF(2^{128})$, there are large classes of weak AES-GCM keys.
- In a forced-rekeying attack model the **average** cost of a single forgery is less than what is indicated by the security proof (the cost can be made arbitrarily low, à la multicollision attacks on hash functions).
- Don't use GCM with something like **SSH**. However, there may be rational grounds for using it with extremely high-speed VPN (IPSec) links if the risks are understood (and parallelism is required).
- If you absolutely want to do polynomial message authentication, use a secure field rather than $GF(2^{128})$.

Concluding

- Since the authenticator H is derived as $H = AES_k(0)$ and there are plenty of low-order roots of unity in $GF(2^{128})$, there are large classes of weak AES-GCM keys.
- In a forced-rekeying attack model the **average** cost of a single forgery is less than what is indicated by the security proof (the cost can be made arbitrarily low, à la multicollision attacks on hash functions).
- Don't use GCM with something like **SSH**. However, there may be rational grounds for using it with extremely high-speed VPN (IPSec) links if the risks are understood (and parallelism is required).
- If you absolutely want to do polynomial message authentication, use a secure field rather than $GF(2^{128})$.

Concluding

- Since the authenticator H is derived as $H = AES_k(0)$ and there are plenty of low-order roots of unity in $GF(2^{128})$, there are large classes of weak AES-GCM keys.
- In a forced-rekeying attack model the **average** cost of a single forgery is less than what is indicated by the security proof (the cost can be made arbitrarily low, à la multicollision attacks on hash functions).
- Don't use GCM with something like **SSH**. However, there may be rational grounds for using it with extremely high-speed VPN (IPSec) links if the risks are understood (and parallelism is required).
- If you absolutely want to do polynomial message authentication, use a secure field rather than $GF(2^{128})$.

Concluding – Thank You

- Since the authenticator H is derived as $H = AES_k(0)$ and there are plenty of low-order roots of unity in $GF(2^{128})$, there are large classes of weak AES-GCM keys.
- In a forced-rekeying attack model the **average** cost of a single forgery is less than what is indicated by the security proof (the cost can be made arbitrarily low, à la multicollision attacks on hash functions).
- Don't use GCM with something like **SSH**. However, there may be rational grounds for using it with extremely high-speed VPN (IPSec) links if the risks are understood (and parallelism is required).
- If you absolutely want to do polynomial message authentication, use a secure field rather than $GF(2^{128})$.