CS 465

AES

Last Updated: Sep 7, 2017

Programming Lab #1

- Implement AES
- Use the FIPS 197 spec as your guide
    - Avoid looking at code on the Internet
    - Challenge yourself to implement the algorithm based on sources mentioned in the lab specification
    - The standard provides programming language independent pseudo-code
    - 20 pages at the end of the spec has complete, step-by-step debugging information to check your solution

AES Parameters

- Nb – Number of columns in the State
  - For AES, Nb = 4
- Nk – Number of 32-bit words in the Key
  - For AES, Nk = 4, 6, or 8
- Nr – Number of rounds (function of Nb and Nk)
  - For AES, Nr = 10, 12, or 14

AES methods

- Convert to state array
- Transformations (and their inverses)
  - AddRoundKey
  - SubBytes
  - ShiftRows
  - MixColumns
- Key Expansion

Inner Workings

- See Flash demo URL on course **Lectures** page

Finite Fields

- AES uses the finite field $GF(2^8)$
  - Polynomials of degree 8
  - $b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$
    - $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$

- Byte notation for the element: $x^6 + x^5 + x + 1$
  - $0x^7 + 1x^6 + 1x^5 + 0x^4 + 0x^3 + 0x^2 + 1x + 1$
  - {01100011} – binary
  - {63} – hex

- Has its own arithmetic operations
  - Addition
  - Multiplication

Finite Field Arithmetic

- Addition (XOR)
  - $(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$
  - $\{01010111\} \oplus \{10000011\} = \{11010100\}$
  - $\{57\} \oplus \{83\} = \{d4\}$

- Multiplication is tricky
  - Study section 4.2 in the spec
  - In 4.2.1, a paragraph describes what your implementation will do. Study it. Difficult to interpret.

Finite Field Multiplication (•)

These cancel out

$(x^6 + x^4 + x^2 + x + 1)\,(x^7 + x + 1) =$

$x^{13} + x^{11} + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + x^6 + x^4 + x^2 + x + 1$

$= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$

and

$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$ modulo $(x^8 + x^4 + x^3 + x + 1)$

$= x^7 + x^6 + 1.$

Irreducible Polynomial

Efficient Finite Field Multiply

- There's a better way
  - Patterned after the divide and conquer modular exponentiation algorithm (CS 312)
  - xtime() – very efficiently multiplies its input by {02}
    - This is the same as multiplying a polynomial by x
      - think about what is the binary representation of the polynomial x?
    - Figure out when the mod operation should occur.

- Multiplication by higher powers can be accomplished through repeated applications of xtime()

Efficient Finite Field Multiply

Example: $\{57\} \bullet \{13\}$

$\{57\} \bullet \{02\} = \text{xtime}(\{57\}) = \{ae\}$

$\{57\} \bullet \{04\} = \text{xtime}(\{ae\}) = \{47\}$

$\{57\} \bullet \{08\} = \text{xtime}(\{47\}) = \{8e\}$

$\{57\} \bullet \{10\} = \text{xtime}(\{8e\}) = \{07\}$

$$
\begin{aligned}
\{57\} \bullet \{13\} &= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\
&= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\
&= (\{57\} \bullet \{01\}) \oplus (\{57\} \bullet \{02\}) \oplus (\{57\} \bullet \{10\}) \\
&= \{57\} \oplus \{ae\} \oplus \{07\} \\
&= \{fe\}
\end{aligned}
$$

Efficient Finite Field Multiply

These are hexadecimal numbers {xx}

Example: {57} • {13}

{57} • {02} = xtime({57}) = {ae}

{57} • {04} = xtime({ae}) = {47}

{57} • {08} = xtime({47}) = {8e}

{57} • {10} = xtime({8e}) = {07}

{10} in hex is 16, not decimal 10!

$$
\begin{aligned}
\{57\} \bullet \{13\} &= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\
&= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\
&= (\{57\} \bullet \{01\}) \oplus (\{57\} \bullet \{02\}) \oplus (\{57\} \bullet \{10\}) \\
&= \{57\} \oplus \{ae\} \oplus \{07\} \\
&= \{fe\}
\end{aligned}
$$

See detailed multiplication example on the Lectures web page