Fuzzy Cryptanalysis: Applying Fuzzy Logic to Cryptanalysis

John Washburn

Phone: (262) 238-8940

Fax: none

cryto@WashburnResearch.org http://www.washburnresearch.org/cryptography

This research was self funded, but benefits from the remarkable energy and clarity of Bart Kosko; one of the great evangelists of fuzzy and multi-state logic.

Abstract

This paper describes the extension of fuzzy logic to the discipline of cryptanalysis and cryptography. Fuzzy logic¹ is the branch of logic in which the truth value of a logical proposition (or variable) is represented as a real value on unit interval [0,1]. Most cryptographic primitives (e.g. bit, byte, bitwise operators, logical operations, substitution, permutation, addition, etc) are ready adapted the concept of a "fuzzy bit". In this paper, the truth value of a fuzzy bit is the Baysian probability² the bit is set (equal to 1).

Building up from a single fuzzy bit the author has created all the cryptographic primitives needed to implement the DES and AES algorithms using fuzzy bits throughout. This gives rise to a powerful new approach to cryptanalysis because quantitative measures of confusion, diffusion, and avalanche can be obtained. With such measurements it is now possible to quantitatively compare the cryptographic features of various algorithms.

Keywords: Fuzzy Logic, Cryptanalysis, Cryptography, Fuzzy Cryptanalysis, and Fuzzy Cryptography.

Introduction

The initial impetus for the author to apply the concepts of fuzzy logic to cryptography was to use the techniques to recover cryptographic keys from a <u>known-plaintext attack</u>³ in O(N^2) time instead of the O(2^{N-1}) time required for a brute force enumeration of the keys.

The general scheme was to carefully track the flow of information as it was transformed by various cryptographic primitives. In a known text attack the input and output texts are known with perfect certainty. By setting all of the bits of the key to perfect uncertainty (truth value is equal $\frac{1}{2}$), one could set and clear each bit of the key in succession. For each of the 2N keys where only 1 bit is known with certainty, the cryptographic algorithm is applied to the perfectly known input to produce an output where each output bit has a value in the real interval [0,1]. The set of fuzzy bits could then be compared to the expected output. The value of this comparison is the probability the vector of fuzzy bits is equal to the expected output bits. Form these comparisons, it was hoped that based on the fuzzy output which was "closest" to the expected output, a single bit of the key could then be set or cleared. Once one bit is clear the process would be repeated where each possible second key bit is set or cleared in succession.

The author's visualization of this was that such fuzzy cryptanalysis was similar to picking a physical lock. Cryptographic attacks designed to recover keys, operate on the whole key. The thought was to attack the cryptographic key one bit at a time much as lock picking attacks a physical lock one tumbler at a time.

The cryptographic algorithm settled on was the DES as there is much research and several successful attacks known. By applying fuzzy cryptanalysis to the DES it would be possible to compare and contrast the fuzzy cryptanalysis to other more traditional breaks in the DES algorithm

As is often the case, The best-laid schemes o' mice an' men gang aft agley⁴. It turns out the a perfectly known input is transformed into a perfectly unknown

output if the DES keys has even one bit of perfect uncertainty. The bit-picking scheme will not work.

But, the research was useful because the above statement regarding the 16-round DES is provable and quantifiable. Fuzzy Cryptanalysis makes it possible to precisely measure the degree and spread of uncertainty as the uncertainty propagates through a cryptographic system.

Fuzzy Bits

The basis unit of fuzzy cryptanalysis is the fuzzy bit. A traditional binary bit is the electronic implementation of Boolean logic; yeas/no, true/false, 1/0. Multistate logic admits more than 2 states for the truth value of a proposition. Most cryptographers are familiar with 3-state logic. Tri-state bits are used to study and describe avalanche criteria for cryptographic systems. These bits have the values of Zero, One and Unknown (0,1,?). For example, the first substitution box of the DES algorithm would substitute 0011?0 for either 1011 or 1000. This propagation of uncertainty is typically represented by 10??. It is certain the 2 high order bits are 10 but the 2 low order bit could be each either 1 or 0. What if the 6bit input has the 2 unknown bits, 0011??. Then the possible outputs are: 1011, 1101, 1000 or 0001. The approach of cryptanalysts in the past would be to list the 4-bit output as [????]. All 4 output bits are unknown. But the bits do not have

equal uncertainty. The true odds for the 4 output bits are: $\left[\frac{3}{4}, \frac{1}{4}, \frac{1}{4}, \frac{3}{4}\right]$.

Returning to the first example, if the 6-bit input is instead: $[0,0,1,1, \frac{1}{4},0]$ the expected values of the output bits are: $[1,0, \frac{3}{4}, \frac{3}{4}]$. Because the 6-bit input, 001100, is more likely, the 4-bit of 1011 is more likely.

It quickly becomes clear that tri-state logic is inadequate if one wishes to track the introduction and propagation of uncertainty with this kind of precision. In fact, no multi-state logic of any fixed precision will do. What is needed is fuzzy logic.

Fuzzy Logic

Fuzzy logic is the system of logical reasoning where the options are not binary but instead the truth value is part of a continuum on the interval from Certainly False (Zero) and Certainly True (One). Fuzzy logic is different than the binary logic of Aristotle. For example the Aristotle's Law of the Excluded Middle is not true under fuzzy logic. An item can simultaneously be both somewhat Tall and somewhat Not Tall. If I am 2 meters tall, my Tallness varies depending on whether I am among horse jockeys or NBA players.

Within the discipline of fuzzy logic there is some dispute as to on the nature of the truth value. Some contend the truth value of fuzzy variable is strictly the probability the proposition is true or not. This is uncertainty. Others contend concepts such as vagueness and ambiguity can be model by fuzzy logic but are separate and distinct from simple probabilistic uncertainty. These practitioners the truth value of a fuzzy variable is different than the value derived using Baysian probability theory. For more information on this fascinating aspect of fuzzy logic I would refer you to the book, *Fuzzy Thinking*⁵ by Bart Kosco⁶.

Cryptography admits only one form of fuzzy; probabilistic uncertainty. A bit in a cryptographic system is either One or Zero. We are uncertain which value is the correct value of the bit, but there is one and only one correct value. Uncertainty of this form is correctly modeled by Baysian probability theory. A fuzzy bit for the purposes of cryptanalysis is a binary bit where the value of the bit may be uncertain. But, while the author has selected probability bits for his first implementation of fuzzy cryptanalysis, there is no reason purely fuzzy logic could not be applied to cryptanalysis. This is an area for future research because the mathematic for the predicate logic of pure fuzzy logic is simpler than that of probability. It is possible the same results can be had from fuzzy cryptanalysis with a significantly lower computational cost.

The Mathematics of a Probability Bit

Thus, a fuzzy bit for the purposes of this paper is defined as follows: *A fuzzy bit is a real value,* ρ *, on the closed interval,* [0,1] *where* ρ *represents the probability the bit is set (has a value of one).* If a fuzzy bit has a value of 0, then the value of the bit is certainly 0. The bit is certainly Cleared. If a fuzzy bit has a value of 1, then the bit is certainly Set. If a fuzzy bit has a value of $\frac{1}{2}$ then it is equally likely the bit is Set or Cleared. The bit is perfectly uncertain. If the fuzzy bit has a value of $\frac{17}{52}$, then bit is probably zero since there only a 1 in 3 chance the bit has a value of 1.

The only operations used to calculate probabilities from other probabilities are addition, subtraction, multiplication and division. These mathematical operations are all closed on the set of rational numbers. Thus, if the initial values of the fuzzy bits are constrained to the set of rational numbers, then all subsequent values for the fuzzy bits will also be rational numbers. This initial condition is reasonably easy to meet as all cryptography starts with initial values of one, zero, or unknown

 $\left(\rho = \frac{1}{2}\right)$. The author's implementation of the Java Class, FuzzyBit, uses the Java

class, BigRational, to track the values of a fuzzy bit. See Appendix A for more details on the Java code written to implement the ideas of this paper.

Cryptographic Primitives

Once the definition of bit is expanded to use fuzzy logic and becomes the definition of a fuzzy bit, it is time to make all the cryptographic primitives fuzzy. These primitives the author has implemented as fuzzy cryptographic primitives are the bitwise operators, bytes, shifts, permutation, substitution boxes, addition, multiplication in the Galois field, $GF(2^8)$ and logical comparisons. For the following definitions $A(\rho)$ represents that the fuzzy bit, A, is set with a probability of ρ .

Bitwise Operators

Operator	Boolean Formula	Probability
Bitwise Not	$B = \overline{A}$	$B(\alpha) \ \alpha = 1 - \rho$
Bitwise AND	C = AB	$C(\beta) = A(\rho)B(\alpha) \ \beta = \alpha\rho$
Bitwise OR	C = A + B	$C(\beta) = A(\rho) + B(\alpha)$
		$\beta = \alpha + \rho - \alpha \rho$
Bitwise XOR	$C = A \oplus B$	$C(\beta) = A(\rho) \oplus B(\alpha)$
		$\beta = \alpha + \rho - 2\alpha\rho$
		$= \alpha (1 - \rho) + (1 - \alpha) \rho$
Fuzzy Bit Vector	Is an ordered vector of N fuzzy bits	
Fuzzy Byte	Is an ordered vector of 8 fuzzy bits	
Fuzzy Short	Is an ordered vector of 16 fuzzy bits	
Fuzzy Long	Is an ordered vector of 32 fuzzy bits	
Logical Equal One Bit	Does A=B?	$\beta = \left[A(\rho) = B(\alpha) \right]$
		$\beta = 1 - \alpha - \rho + 2\alpha\rho$
		$= \alpha \rho + (1 - \alpha)(1 - \rho)$
Logical Equal Vector of Bits	Is on bit vectors equal	$\beta = \prod_{n=1}^{N-1} \left[A(\alpha) = B(\alpha) \right]$
	to a second bit vector	$\boldsymbol{\rho} - \prod_{i=0} \left[A_i (\boldsymbol{\rho}_i) - \boldsymbol{B}_i (\boldsymbol{\alpha}_i) \right]$
	of equal length?	$\beta = \prod_{n=1}^{N-1} \left[1 - \alpha - \alpha + 2\alpha \alpha \right]$
	$A_i = B_i$	$ \begin{array}{c} \mathcal{P} \prod_{i=0} \left[1 \mathcal{U}_i \mathcal{P}_i \mathcal{U}_i \right] \end{array} $
Scaling	Scale a Bit Vector	$\boldsymbol{B}_{i}(\boldsymbol{\beta}_{i}) = \overline{\alpha \boldsymbol{A}_{i}(\boldsymbol{\rho}_{i})}$
	$B = \alpha A$	$\beta_i = \alpha \rho_i$

The resulting probabilities for the resulting fuzzy bit(s) for the basic bitwise and logical operators common to cryptography are given below

Permutation, Shifts, Rotation, Expansion and Contraction

Seven other operations common to cryptography are Permutation, Shift Left, Shift Right, Rotation Left, Rotate Right, Expansions and Contractions. All of these operations are a <u>mapping</u> of N bits to M bits. The number of output bits, M, may or may not be equal to the number of input bits, N. Also, as is the case with the 2 shift operators and contractions, there is also a loss of information. Permutation, Rotation, and Expansions do not lose information. Permutations and rotations map N bits to N bits. Shift operators map N bits to N-1 bits with the single bit replaced by either a One or Zero. Expansions map N bits to N+k bits and Contractions map N bits to N-k bits; where k > 0.

Regardless of the precise details of the mapping, all mapping involves the rearrangement of bits without changing the value of the bits transported. Because of this all of these cryptographic primitives can be implemented regardless of whether the bits in question are fuzzy bits or binary bits.

Within the Java implementation of fuzzy cryptanalysis all of the mappings discussed above are covered by slightly abusing the term permutation to include any N-bit to M-bit Mapping. See Appendix A for more details on the Java implementation of the ides in this paper.

Addition

What does it mean to add on vector of fuzzy bits to a second vector of fuzzy bits? Brute enumeration would seem to be an obvious answer, but it is computationally infeasible. It turns out though that by blindly applying the algorithm of addition to fuzzy bits the resulting answer matches the value derived from complete enumeration and can be done in a computationally efficient manner:

Multiplication

The approach used for addition did not work for multiplication of any form. The brute application of the computation algorithms yielded answers significantly different than those derived by enumeration all the possible multiplicands and multipliers. This too is an area where more research is required as module multiplication and multiplication in <u>Galois fields</u>⁷ is common in cryptography.

Substitution (S-Boxes)

A substitution box (S-Box) selects and output of M bits based on an input of Nbit. Given a input vector of N fuzzy bits it is possible to determine the probability the input to the S-Box is a particular constellation of N binary bits. This probability can then be applied to scale the output of M binary bits. By enumerating all 2^{N} possible inputs, calculating the probability of that particular input, the resulting 2^{N} possible outputs (suitably scaled) can be summed to produce an M-bit fuzzy result of the S-Box.

Design Considerations

Each of the formulas or algorithm used to defined the fuzzy bit analogs of common cryptographic primitives were:

- 1. If all the fuzzy bits are crystal bits (fuzzy bits exactly equal 1 or exactly equal to 0), then the fuzzy result of the operation should also be composed of crystal bits.
- 2. If all the fuzzy bits are crystal bits, then the result of the operation is not only crystal bits, but the result is identical to the results common binary bits been used throughout.

The term crystal bit is used to distinguish between common binary bits and fuzzy bits where the truth value of the fuzzy bit is an integer (0 or 1). This term arose from several diagrams where the binary values possible are the vertices of a crystal lattice and the space of truth values is the space enclosed and bounded by this crystal lattice. Fuzzy bits which happen to be equal to zero or one occur frequently, but must be distinguished from binary bit as the term is customarily used

Conclusions

The bit picking scheme for recovering the cryptographic key of a known text pair was not successful. The research proved useful even with this failure. It is illuminating to observe and document the introduction and spread of uncertainty in a cryptographic system. This gives rise to possibility to more precisely quantify the information theory concepts, confusion and diffusion. Such precision provide the opportunity to use fuzzy cryptanalysis to construct better cryptograph better cryptographic primitives such as S-Boxes. It also allows for demonstration and measurement of the strength or weakness of various cryptographic systems.

Also, the focus of this paper and is research were on encryption, there seems to be no restriction to applying fuzzy cryptanalysis to other areas of cryptographic research such as transfer protocols or blinding protocols.

¹ <u>http://en.wikipedia.org/wiki/Fuzzy_logic</u>

² <u>http://en.wikipedia.org/wiki/Baysian</u>

³ <u>http://en.wikipedia.org/wiki/Known-plaintext_attack</u>

⁴ "To a Mouse, on Turning Her Up in Her Nest, With The Plough", Robert Burns,

http://www.cs.rice.edu/~ssiyer/minstrels/poems/776.html

⁵ <u>http://www.amazon.com/Fuzzy-Thinking-New-Science-Logic/dp/078688021X/sr=8-</u>

^{1/}qid=1170519560/ref=pd_bbs_1/102-4481497-2248112?ie=UTF8&s=books

⁶ <u>http://sipi.usc.edu/~kosko/</u>

⁷ http://en.wikipedia.org/wiki/Galois_field