1. Overview of cryptography

* Services provided by cryptography: confidentiality, integrity, non-repudiation, authentication, access control
  + Confidential so no one can steal it (confidentiality)
  + Protected so no one can alter it (data integrity)
  + Authentic so no one can impersonate creator (entity authentication)
  + Safe from intrusion on disk (access control)
  + Safe from denial by instructor or TA (non-repudiation)
    - Assurance that the owner of the signature key pair that could generate an existing signature corresponding to certain data cannot convincingly deny having signed the data
* Notions of cryptographic primitive and cryptographic protocol
  + Cryptographic primitives:
    - Tool that represents a cryptographic security mechanism
    - Well established, low level cryptographic algorithms that are used to build cryptographic protocols
  + Cryptographic protocols:
    - Algorithm (sequence of steps) to be taken by two or more entities to achieve a specific security objective
    - a security protocol is an abstract or concrete protocol that performs a security related function and applies cryptographic methods, often sequences of cryptographic primitives
* Definition and ingredients of a cryptosystem
  + Cryptosystem/cipher: a particular method of encryption, capable of handling arbitrary messages
  + Plaintext
  + Encryption Algorithm
  + Ciphertext
  + Decryption Algorithm
  + Encryption Key
  + Decryption Key
* Notions of active and passive attacks and the corresponding goals of the attacker
  + Passive attacks:
    - Listening, eavesdropping on information without interaction with the system
  + Active attacks: interacting with the system, modifying information (impersonation, replaying messages, changing contents, or denial of service)
* Types of attacks: COA, KPA; CPA, adaptive CCA and their classification of active/passive
  + Passive Attacks:
    - (COA) Ciphertext Only Attack: adversary has only ciphertext, but no plaintext
    - (KPA) Known Plaintext Attack: adversary has some plaintext and corresponding ciphertext
  + Active Attacks:
    - (CPA) Chosen Plaintext Attack: adversary chooses some plaintext (independently of the ciphertext she wishes to decrypt) and obtains the corresponding ciphertext.
    - Adaptive (CPA): adversary’s choice of plaintext may depend on the ciphertext she wishes to decrypt and on ciphertexts received from previous requests
    - Chosen Ciphertext Attack (CCA1): adversary chooses some ciphertext (independently of the ciphertext she wishes to decrypt) and obtains the corresponding plaintext.
    - Adaptive Chosen Ciphertext Attack (CCA2): adversary’s choice of ciphertext may depend on the ciphertext she wishes to decrypt and on plaintexts received from previous requests. She is not allowed to choose the ciphertext she wishes to decrypt.
    - NOTE: A good/secure cryptosystem should be secure against adaptive CCA’s (as strong as possible)
* Kerckhoff’s principle: the security of a cryptosystem should depend entirely upon knowledge of the key, not of the method
  + This implies that a cipher should be completely published and still be secure (against its own designer and everyone else).

2. Classical cryptography

* Classical ciphers: monoalphabetic substitution (with examples: Caesar, shift, one-time pad), polyalphabetic substitution (with example: Vigenere), transposition
  + Two types of Classical ciphers:
    - Substitution cipher: a cipher for which encryption replaces each plaintext symbol by some ciphertext symbol without changing the order of the plaintext symbols
      * Caesar
      * Shift cipher: to encrypt, every plaintext letter is shifter by a fixed position (monoalphabetic: one cipher alphabet)
      * One-Time Pad
      * Vigenere cipher: plaintext letters are shifted by different positions based on a repeated rotating pattern (see handouts) (polyalphabetic: several cipher alphabets)
    - Transposition cipher: a cipher in which the ciphertext is a rearrangement (i.e. permutation) of the plaintext symbols
      * Route cipher: plaintext is arranged in some geometric figure and encrypted by rearranging the plaintext according to some route through the figure
      * Columnar transposition cipher: the plaintext is arranged in a rectangle and the ciphertext consists of a secret permutation of the plaintext columns
* Security of classical ciphers (vulnerability to KPAs or even COAs)
  + Monoalphabetic Substitution Ciphers:
    - Highly vulnerable to KPA’s: each portion of corresponding plaintext and ciphertext reveals some of the cipher
      * E.g. for shift ciphers, one letter pair reveals the key!
    - Each plaintext letter is encrypted to the same ciphertext letter.
      * Frequent ciphertext letters correspond to common plaintext letters
      * Pairs of identical ciphertext letters correspond to such plaintext letter pairs (e.g. “XX” corresponds to “yy”)
  + Polyalphabetic substitution:
    - Vigenere cipher:
      * + Determine the length of rotation patterns (number of cipher alphabets) via guessing, the kappa test or Kasiski’s factoring method
        + Cryptanalyze each subtext as a shift cipher
    - Columnar transposition:
      * + Guess the dimensions of the rectangle
        + Determine the order of the columns via frequency counts (which will be the same as for English text). Place columns adjacent to each other if they produce common latter pairs (e.g. QX is extremely unlikely, but EN is highly likely).

3. Perfect secrecy and entropy

* Random variables and probability distributions, Bayes’ Theorem
  + Let and ­ be random variable and
  + Joint probability : probability that and
  + Condition probability : probability that given that
    - aka known as the a posteriori of x given y (“ a posteriori” = after the fact)
  + Joint and conditional probability are related as follows:
  + Bayes’ Theorem: If , then:
    - Used to relate and
  + Two random variables are independent if for all and
  + Corollary 2: and are independent iff for all , with
* Definition of perfect secrecy and equivalent characterization: p(C|M) = p(C) for all M, C with p(M), p(C) > 0
  + Perfect secrecy should reveal no information whatsoever about plaintexts
  + : probability that plaintext occurs
  + : probability that ciphertext occurs (as some encryption)
  + : probability that is the decryption of a given (more formally, that is a possible plaintext, given that ciphertext is encountered, e.g. received in a transmission)
  + : probability thatis the encryption of a given (more formally, the ciphertext was encountered, given that plaintext occurs as a possible plaintext)
    - is the sum of probabilities over all those keys that encipher to
  + : probability that key was chosen
  + Definition: Perfect Secrecy: A cryptosystem provides perfect secrecy if for all and with
    - Formally, perfect secrecy means exactly that the random variables on and are independent. Informally, this implies that knowing the ciphertext gives us no information about .
  + Theorem 3: A cryptosystem provides perfect secrecy iff for all all , withand
    - By Bayes’ Theorem,
* You should be able to work with the formulas for p(C|M) and p(C), but you need not remember them
  + Computing use table

Table

Description automatically generated

* + Description of

Text, letter

Description automatically generated

* + Computing

Text, letter

Description automatically generated

A close-up of a document

Description automatically generated with low confidence

* Perfect secrecy implies |K| ≥ |M|
  + This is a necessary condition for perfect secrecy

Text, letter

Description automatically generated

A screenshot of a computer

Description automatically generated

* Bitwise Exclusive-Or

Fix a string length . Then set is the set of *bit strings* of length .

For , we define

For , we define then

* One-time pad (OTP), strengths and weaknesses of OTP, security of OTP
  + OTP:

(= natural number)

Encryption of under key is bitwise XOR, i.e.

Decryption of under is done the same way, i.e.

Decryption is the inverse of encryption

* + Theorem 1: the OTP provides perfect secrecy if each key is chosen with equal likelihood. Under this assumption, each ciphertext occurs with equal likelihood (regardless of the probability distribution on the plaintext space)
  + Cryptanalysis of the OTP
    - Each key is only used once:
      * Immediately falls to a KPA: if a plaintext/ciphertext pair is known, then the key is
      * Vulnerable to a COA if a key K is used twice:

Note: this encryption with a coherent running key cipher (adding two coherent texts and ), which is like a Vigenere cipher with an extremely long key word (shift rotation pattern) and thus vulnerable to frequency analysis (can find and from ).

* + For the same reason, we can’t use shorter keys and “re-use” portions of them. Keys must be randomly chosen and at least as long as messages. This makes the one-time pad impractical
  + Main disadvantage of OTP:
    - Requires a random key which is as long as the message
    - Each key can only be used once
  + OTP schemes are used when perfect secrecy is crucial, and practicality is less of a concern
  + Major problem with OTP is the cost. These ciphers succumb to exhaustive search, because there is a unique “distinguished” decipherment. The computational difficulty of finding this solution foils the cryptanalyst. The proof of security does not exist for any proposed computationally secure system (just a reduction, subject to certain assumptions, to presumably computationally intractable problem)
* Meaning of and formula for entropy, maximal and minimal entropy values and their characterizations
  + Entropy

Let be a random variable with outcomes and a probability distribution

where

The **entropy** of ­­ is defined by the weighted average

* An event with probability 1/2n can be optimally encoded with bits
* An event occurring with probability can be optianlly encoded with bits.
* The weighted sum is the expected number of bits (i.e. the amount of A information) in an optimal encoding of (i.e. one that minimizes the number of bits required).
* If are outcomes (e.g. plaintexts, ciphertexts, keys) occurring with respective probabilities , then is the average amount of information required to represent an outcome.

A screenshot of a computer

Description automatically generated with low confidence

* + For random variables on a plaintext space , its entropy measures the uncertainty of plaintexts.
  + For random variable on a key space measures the amount of partial information that must be learned about a key to actually uncover it (e.g. the number of bits that must be guessed correctly to recover the whole key).
  + For k-bit key, the best scenario is that all bits must be guessed correctly to know the whole key (i.e. no amount of partial information reveals the key, only full information does)
    - Entropy of the random variable on the key space should be **maximal**
    - Happens when each key is equally likely
    - Best strategy is to select keys in order to give away as little as possible is to choose them with equal likelihood (uniformly at random)

Text

Description automatically generated

* + Extremal Entropy: Recall that the entropy of equally likely outcomes (i.e. each occurring with probability is . This is indeed the maximum:
    - is **maximized** iff all outcomes are equally likely. That is, for any , is maximal iff for .

Text

Description automatically generated

* + - is **minimized** iff for exactly one and for all

Text, letter

Description automatically generated

4. Modern symmetric cryptography

* Concepts of confusion and diffusion and which cipher building blocks provide them
  + Shannon suggested applying two simple(substitution) ciphers with a fixed mixing transformation (transposition) in between to
    - **Diffuse** language redundancy into long-term statistics: dissipate the statistical properties of the plaintext across the ciphertext (accomplished by applying transpositions or P-boxes)
    - **Confuse** the cryptanalyst by obscuring the relationship between the ciphertext and the key: Make the relationship between the key and ciphertext as complex as possible (accomplished by applying substitutions or S-boxes)
      * IBM’s lucifer system uses permutations (transpositions) on large blocks for the mixing transformation, and substitutions on small blocks for confusion

Diagram, schematic

Description automatically generated

* + Error Propagation: the degree to which a change in the input leads to changes in the output
  + Avalanche Effect: changing one input bit leads to significant changes in the output (e.g. half the output bits flip)
  + Block Cipher: Encrypts plaintext blocks of some fixed length to ciphertext blocks of some fixed (possible different) length.
* Notion and design idea of a product cipher
  + The **product**of two ciphers is the result of applying one cipher followed by another AKA superencipherment
    - NOTE: all modern symmetric key ciphers in use are product ciphers
  + If different ciphers are used in a product cipher, ciphertexts of one cipher need to have the correct format to be plaintexts for the next cipher to be applied
    - This is composition of encryption maps
  + Applying a product cipher potentially increase security. E.g. ­-fold encryption with one cipher and keys potentially corresponds to a cipher that has times longer keys.
  + This results in a loss of speed by a factor of , but this might be worth it for added security
  + NOTE: the product of two substitution ciphers is a substitution cipher. The product of two transposition ciphers is a transposition cipher. Such ciphers are *closed* under encryption, so multiple encryption under different keys provides no extra security.
* Basic specifications of DES (format of plaintexts, ciphertexts and keys) and the fact that is a product cipher with a Feistel network architecture
  + DES is obsolete (key space too small) and is still used in legacy code as triple encipherment (3DES)
  + Developed by IBM
  + Block cipher that encrypts 64-bit plaintext blocks to 64-bit ciphertext blocks using 64-bit keys
    - Note that 8 of the key bits are parity bits, resulting in 56 actual bits of the key
  + and
  + Algorithm consists of 16 rounds of permutations and substitutions
  + Campbell and Wiener proved that DES is *not* closed, so multiple DES encryptions/decryptions could potentially provide additional security
  + 3DES

Text, letter

Description automatically generated

* Basic specifications and high-level design of AES: format of plaintexts, ciphertexts and keys, notion of a state, the fact that it is a product cipher consisting of three layers (linear mixing, non-linear substitutions, key X-OR’ing)
  + Rijndael algorithm uses two different types of arithmetic:
    - Arithmetic on bytes (8 bit vectors – actually, elements of the finite field GF (28) of 256 elements)
    - 4-byte vectors (actually polynomial operations over GF(28))
  + Algorithm uses addition, multiplication, and inversion on bytes as well as addition and multiplication on 4 byte vectors
  + Rijndael is a **product cipher**, but NOT a Feistel cipher like DES. It has **three layers** per round:
    - A linear mixing layer (SHIFTROWS, transposition, and MIXCOLUMNS, a linear transformation; for diffusion over multiple rounds)
    - A non-linear layer (SUBBYTES, substitution, done with an S-box)
    - A key addition layer (ADDROUNDKEY, X-OR with key)
  + Rijndael Properties
    - Designed for block sizes and key lengths to be any multiple of 32
    - Iterated cipher: number of rounds Nr depends on the key length.
      * 10 rounds for 128 bit keys, 12 for 192, 14 for 256
    - Algorithm operates on a 4 x 4 array of bytes (8 bit vectors) called the state:

A picture containing text, group, dark, close

Description automatically generated

* + Strengths of Rijndael
    - Secure against all known attacks at the time; some newer attacks seem to pose no real threat
    - Thwarts linear and differential cryptanalysis
    - SHIFTROWS and MIXCOLUMNS ensures that after a few rounds, all output bits depend on all input bits (great diffusion)
    - Knowledge of part of the cipher key or round key does not enable calculation of many other round key bits
    - Each key bit affects many round key bits
    - Very low memory requirements
    - Very fast hardware and software
  + Weaknesses of Rijndael
    - Decryption is slower than encryption
    - Decryption algorithm is different from encryption (requires separate circuits and/or tables)
* Exhaustive attacks on block ciphers: basic key search, basic idea of a time-memory trade-off, meet-in-the-middle attack on double encryption and its run time requirement,
  + Exhaustive Search:
    - Brute force search for the key is the simplest attack on a block cipher

Text, letter

Description automatically generated

* + Hellman’s Time-Memory Tradeoff
    - KPA that shortens search time by using a lot of memory

Text

Description automatically generated

* + Meet in the Middle Attack on Double Encryption
    - Naïve exhaustive search requires up to N2 encryptions (N2 key pairs)
    - Much faster KPA, but more memory intense
    - Setup:
      * Adversary has two known plaintext/ciphertexts pairs (m1,c1), (m2,c2) (one for key search, the other for checking correct guess)
      * Assume double-encryption: for where are two unknown keys
      * Note: for

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

Text, letter

Description automatically generated

* Familiarity with the terms differential and linear cryptanalysis and their aim (you need not know the other advanced attacks)
  + Linear cryptanalysis: A cryptosystem is *affine* (linear) if encryptions are affine (linear) functions relating plaintexts to ciphertexts
    - Affine equation
    - Linear Equation (i.e. )
    - Where A and B are matrices of appropriate dimensions
    - A and B reveal information about the key used to encrypt M to C

Graphical user interface, text, application

Description automatically generated

* + Differential Cryptanalysis: Compares input XORs to output XORs, and traces these differences through the cipher
  + Both linear and differential cryptanalysis work on DES with fewer than 16 rounds
* Block ciphers and stream ciphers: basic idea, notions of SSC and self-SSC, purpose of stream ciphers
  + Stream Ciphers: don’t treat incoming character independently like block ciphers
    - Encryption of plaintext character depends on internal state of device
    - After encryption, the device changes state according to some rule
    - Result: two occurrences of the same plaintext character will usually not result in the same cipher text character
    - Incorporate a *key* *stream* into encryption and decryption that is generated form the key. (pseudorandom sequence of bits) Blocks of key bits are XORed with plaintext blocks for encryption and XORED with ciphertext blocks for decryption
  + SSC Synchronous Stream Cipher
    - State depends only on previous state, not on input
    - Output depends only on and , not on
    - Implemented by Boolean logic that should produce a pseudo-random sequence synchronized by the key (e.g. a shift register)
    - Example: the one time pad is SSC

Diagram

Description automatically generated

* + Block Cipher SSCs

Diagram

Description automatically generated

Text

Description automatically generated

* + Self Synchronizing Stream Cipher (Self-SSC)
    - Asynchronous stream cipher
    - Similar to SSC, except the counter is replaced by a register containing the previous k ciphertexts
    - Self synchronizing after k steps
    - Implemented with a block cipher as above
    - Limited error propagation (k steps)

Diagram

Description automatically generated

* Modes of operation and their purpose. Block ciphers can be used in a number of different modes of operations.
  + Electronic code book (ECB) BLOCK CIPHER
    - Blocks are encrypted sequentially, one at a time:
    - This is a substitution cipher
  + Cipher Block Chaining (CBC) BLOCK CIPHER
    - Sends initial *random* block (e.g. a simple plaintext encrypted in ECB mode such as
    - Encryption: “Pre-whitening”
    - Decryption:
    - Not a stream cipher (XOR with plaintext happens inside encryption

Diagram

Description automatically generated

* + Block cipher to stream cipher by generating a pseudorandom key using the encryption function
  + Counter (CTR) BLOCK CIPHER TO STREAM CIPHER (Synchronous)
    - a counter value
    - SCC with stream key where is a counter of the same size as the plaintext block size.
      * Subsequent values of the counter are computed via an iterating function – the FIPS recommendation is simply assuming an n-bit counter
    - Counter must be unique for each plaintext block that is ever encrypted under a give key, across all messages
      * Keep count of # of plaintext blocks encrypted under a given counter sequence
      * Uses a new block cipher key before exceeding blocks (n-bit blocks)
  + Cipher Feedback (CFB) BLOCK CIPHER TO STREAM CIPHER (self-synchronizing)
    - previous ciphertext bits
    - Requires a different IV for each cryptographic session
    - Self-SSC
    - Simplest form, one register:
    - In general, r cipher bits are fed back (for DES, r = 8 and IV is at least 48 random bits, right justified, padded with 0’s
  + Output Feedback (OFB) BLOCK CIPHER TO STREAM CIPHER (Synchronous)
    - previous key stream bits
    - Requires a different IV for each cryptographic session
    - SSC
    - Simplest form, one register:
    - In general r keystream bits are fed back

5. Key Agreement

* One way functions
  + A function that satisfies the following two properties:
    - Ease of Computation: is easy to evaluate for a given .
    - Pre-image Resistance: Given , it is computationally infeasible to find
    - It is not known whether one way functions exist, but several that are believed to be one-way are used in cryptography

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

* Diffie-Hellman protocol
  + Alice and Bob wish establish a common key for encryption over a public channel in such a way that an eavesdropper cannot determine the key
  + It must be a large prime and an integer with (these two quantities can be public)

Graphical user interface, text, application

Description automatically generated

* Primitive roots: definition, properties, primitive root test
  + Primitive Root: for a prime , a primitive root of is an integer ( is a generator of the multiplicative group ) such that the smallest positive exponent with is
  + Properties: Every element of is a unique power of a primitive root of with exponent between 0 and :

Graphical user interface, text

Description automatically generated

* + Primitive Root Test

Graphical user interface, text, application

Description automatically generated

Text, letter

Description automatically generated

* Security of Diffie-Hellman, discrete logarithms and the DLP
  + Security of DH

Graphical user interface, application, Teams

Description automatically generated

* + Discrete Logarithms: the integer is the discrete logarithm (or index) of y (to base g)

Graphical user interface, text

Description automatically generatedText, letter

Description automatically generated

* + Discrete Logarithm Problem (DHP):
    - Given , find
    - If an adversary can solve an instance of the DLP, she can solve the DHP
    - It is unknown if there are ways of solving the DHP, and hence breaking DH key agreement, other than extracting discrete logs
* Man-in-the-middle attacks on “textbook” Diffie-Hellman

Graphical user interface, text, application

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

Text, letter

Description automatically generated

* Binary exponentiation

Text

Description automatically generated

Text

Description automatically generated

Text, letter

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

6. Hash functions and message authentication codes

* Definition and properties of a cryptographic hash function
  + Hash Function: a function that is easy to compute. An image is referred to as a message digest or a digital fingerprint or a checksum or simply a hash
    - Hash functions satisfy two properties:
    - Compression: H maps an input M of arbitrary bit length to an output of fixed bit length.
    - Ease of computation: for any input M, H(M) is easy to compute
  + A hash function is cryptographic (ally secure) if it is pre-image resistant and collision resistant
* Idea of iterated hash functions
  + Composed of rounds (like DES or AES)
  + Repeated use of compression function-takes m-bits input from the previous step (chaining variable) and an r-bit block from M; produces m-but output
  + Input to H: message M consisting of r-bit blocks (padded if necessary, so total length is multiple of r)
  + Can be setup in a way so that if f is collision-resistant, so it H
* Basic idea of the sponge design
  + Used to build various cryptographic primitives (stream ciphers, hash functions, message authentication codes)
  + Ingredients:
    - A width b (an integer)
    - A bit rate r (an integer < b)
    - An input S (a bit string of length b)
    - A fixed-length permutation f that operates on S
    - A padding rule “pad” that pads blocks of length r to blocks of length b
    - Capacity of sponge is the padding amount
  + The input to the absorption phase is the message M- padded so total length is multiple of r
  + Output is a string S of length b
  + Squeezing phase outputs on input S a hash of the message M whose bit length is a user-supplied value m
* Basic overview of Keccak: high-level design, format of states
  + Input: bit string S of length b
  + Output: bit string S of length b
    - Convert S to state A where
    - For to do : each of these are maps
    - Convert A to a string S of length b
    - Output S

Text, letter

Description automatically generated

* + More on the maps in slides07
* Attacks on hash functions and their complexity: brute force and birthday attacks
  + Objectives of attacking hash functions
    - Find a pre-image: given any hash, create corresponding message with that hash
    - Find a weak collision: given a message, modify it to another message with the same hash
    - Find a collision: find two messages with the same hash
  + Brute-force attacks:
    - Like block ciphers, brute force should be the best attack

Text, letter

Description automatically generated

* + Birthday attack:

Text, letter

Description automatically generated

* Definition and properties of a MAC, services provided by a MAC
  + Message Authentication code (MAC): a small, fixed-size, key-dependent block that is appended to a message to check data integrity- AKA keyed hash function or tag
    - A single parameter family of many-to-one functions satisfying:
      * Ease of computation with knowledge of K: For any and is easy to compute
      * Computing resistance: for any , given zero or more message/MAC pairs (, it is comutationall yinfeasible to compute any new message/MAC pair ( for all , without knowledge of .
  + Provides limited sender authentication in a similar manner to encryption
    - Only sender or receiver (who both know K) can generate the MAC
  + Computation resistance implies data integrity
    - Unlike collision resistance, computation resistance does not ask for two different messages with the same MAC
    - Unlike pre-image resistance, computation resistance does not ask for a pre-image of a give MAC tag.
* Basic idea of deriving MACs from block ciphers or hash functions
  + Block cipher can be used to generate MACs (two methods)
    - CBC-MAC:
      * Encrypt the message (IV of all zeros, last block padded with 0s) using CBC mode
      * The last cipher block (whose bits are dependent on all the key bits and all message bits) is the MAC)
    - CFB-MAC: same idea as CBC-MAC

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated

* + Hash based MACs: where H is cryptographically secure hash function and K is a secret key.
    - Faster than block ciphers

A screenshot of a computer

Description automatically generated with medium confidenceText

Description automatically generated

* Attacks on MACs and their complexity: MAC space, key space attacks
  + Objective of adversary is to defeat computation resistance:
    - Given zero or more pairs ( compute a new message/MAC pair (for some message for all I, without knowledge of K.
    - Known-message, chosen-message, and adaptive-chosen-message variations are possible
  + A more ambition adversary goal is to find the MAC key K
  + MAC Space:

A screenshot of a computer

Description automatically generated with medium confidence

* + Exhaustive search attack on Key Space:

Text

Description automatically generated

7. Other miscellaneous mathematics

* Basic proof techniques
* Zn and Z∗n, Euler phi function
* Congruences and integer modular arithmetic
* Modular arithmetic on polynomials with binary coefficients
* Base 2 logarithms