Week 7 - Supplementary Materials

Markdown | PDF | MS Word DOCX | Libre ODT | HTML

Introduction to Encryption

This week we'll be scrambling secret messages and learning the basics of encryption. We walk through the RSA algorithm and generate our own PGP keys to encrypt and sign files.

As promised, we don't make you crunch complex math for this class. The following notes are provided for context and will give you better footing for our conversation around RSA.

Basic Arithmetic Concepts

- **Factor:** x is a factor of y whenever x divides y evenly (i.e., leaves no remainder)
    - Ex: 3 and 6 are factors of 12 because 12/6 = 2 and 12/3 = 4 -5 and 7 are not factors of 12 because 12/5 and 12/7 have remainders
- **Common factor:** x and y share a common factor z when z divides both x and y evenly
    - Ex: 3 and 8 share no common factor other than 1
    - 3 and 6 share a common factor other than 1 (i.e., 3)
- **"x mod y":** mean "the remainder of x divided by y"
    - Ex: 12 mod 6 = 0 (12 divided by 6 leaves no remainder)
    - 12 mod 5 = 2 (12 divided by 5 leaves 2 as a remainder)
    - 5 mod 12 = 5
    - 30 mod 29 = 1
    - 30 mod 31 = 30
    - **Note:** if y is a factor of x, x mod y = 0
    - If z is a common factor of x and y, x mod z = 0 and y mod z = 0
- **Modulus:** the "y" in "x mod y"
    - Ex: 6 is the modulus in "12 mod 6"
    - 12 is the modulus in "6 mod 12"
- **Prime number:** A number whose only factors are itself and 1.
    - Ex: 3, 5, 7, 11, 13, 17 are prime numbers
    - 4, 6, 9, 10, 12, 15 are not prime numbers (they are composite numbers)
- **Co-primes:** x is the co-prime of y if their only common factor is 1
    - Ex: 5 and 12 are co-primes because they share no factor other than 1
    - 4 and 10 are not co-primes because they share a common factor (i.e., 2)

PGP/GPG key generation

PGP is "Pretty Good Encryption", which uses asymmetric encryption (aka a public+private keypair). GPG is GNU Privacy Guard, which is the software that commonly generates PGP keys.

Generate a public/private keypair: `gpg --full-generate-key`

Export the public key from the keyring: `gpg --output ~/kermit.pub --armor --export kermit@lawfareblog.com`

Create a message to encrypt: `echo "Kermit drinking tea" > ~/message.txt`

Encrypt the message with your public key: `gpg --encrypt --sign --armor -r kermit@lawfareblog.com message.txt`

View the encrypted message: `cat ~/message.txt.asc`

Decrypt the message with your private key: `gpg --decrypt message.txt.asc > message2.txt`