Introduction to GnuPG Presentation

- Who's This Guy?
- Overview
- History
  - Phil Zimmerman
    - Long-Time Anti-Nuke Activist
    - Created PGP for activism
    - US criminal investigation due to [unintentional] encryption export
      - MIT Press published book of source code
  - PGP – Pretty Good Privacy
    - Freeware: No license required for non-commercial use; source code available
    - Commercial versions available
    - A time when email was replacing paper mail
  - OpenPGP Standard
    - Created due to corporation patent issues and code export restrictions
    - Many people worldwide wanted to write their own software compatible w/ PGP
  - GnuPG
    - GPL-licensed implementation of OpenPGP backed by the GNU Project
- Protection Against
  - ISP, well-funded government organization, boss, or business competitor reading your message
  - Adversary changing “pay $200 to Alice” to “pay $200 to Bob”
  - Me pretending to be President Obama
- No Protection Against
  - Coercion: private key password, message contents
  - Metadata: Who/when/how you're talking to
  - Loss of private key: “Unpickable look on front door, but someone pickpockets the key”
  - Data leaks (email client saving draft to disk unencrypted)
- How Does GnuPG Work?
  - Two separate mechanisms
    - Data Encryption: Difficult for anyone other than defined recipient to decode
    - Data Authentication (signing): Ensure data is unaltered (emails, software downloads)
  - Theory
    - Without first exchanging a secret...
      - Julius Caesar's Cipher: Shift x letters down the alphabet
      - Steganography: Ancient Greece, tattoo head under hair
    - Public Key Crypto
      - Symmetric: Both parties need access to secret
      - 1997 NSA Deputy Directory: “If all the personal computers in the world, 260 million, were put to work on a single PGP-encrypted message, it would still take an estimated 12 million times the age of the universe, on average, to break a single message.”
      - 2009-concluded study: Hundreds of systems 2+ years to break 232 digit number (RSA-768)
    - Requires users to create a keypair
      - Don't rely on password to protect private key (the math is huge compared to a password)
      - Security hinges on confidentiality of private key

        - Bob sends a message to Alice
            - Message encrypted by Bob with Alice's **public** key
            - Message decrypted by Alice with Alice's **private** key
        - Web of Trust
            - Decentralized; Prevents impersonation
            - How far to satisfy trust? Blog post? Verbal? Photo ID? DNA?
            - Keysigning parties – BigLumber.com
            - Diagram
                - I trust Alice; Alice trusts Bob; thus I trust Bob
                - Chris and Doug trust Ed; Ed does **not** trust me; thus Chris and Doug do **not** trust me
    - Practice
        - Can't just paste ciphertext into email client (including webmail)
            - Word wrapping
            - HTML formatting
            - Email headers
        - Generate Key
            - Defaults are sane
            - Use a passphrase, not password!
        - Every program/app is different; general terms are the same
    - Beyond the scope of this presentation...
        - Four keys: encryption & signing (public and private)
        - See the internet for suggested defaults (beware of outdated info)
            - Search engine: https://duckduckgo.com
            - https://help.riseup.net/en/security/message-security/openpgp/best-practices
            - http://keyring.debian.org/creating-key.html