How PGP Works(CMU / PGP Intro)

Source: How PGP works

Basics of Cryptography

- Plaintext / cleartext = readable data.
- Ciphertext = encrypted data (unreadable without decryption).
- Encryption transforms plaintext → ciphertext; decryption reverses it.
- Cryptography = using mathematics to secure data; cryptanalysis = breaking/encrypting data.
- Strong cryptography: defeat adversaries even with substantial resources; PGP aims at strong cryptography (not trivial secrecy).

Conventional / Symmetric Encryption

- Uses one key for both encryption and decryption.
- Very fast performance, good for large data.
- Key distribution problem: securely sharing that secret key is hard over insecure channel.
- Example: Caesar cipher is a simple substitution (shift) cipher.

Public-Key / Asymmetric Encryption

- Uses key pair: public key (shared) + private key (kept secret).
- Anyone can encrypt with public key; only private key holder can decrypt.
- Avoids problem of sharing secret key over insecure channel.
- Examples of public-key algorithms: RSA, ElGamal, Diffie–Hellman, DSA.

PGP: Hybrid System

- PGP combines symmetric and public-key cryptography: use best of both worlds.
- Workflow for encryption (sender side):
- 1. Compress plaintext (removes redundancy → helps security)
- 2. Generate a session key (random) for symmetric encryption
- 3. Encrypt plaintext with session key using symmetric cipher
- 4. Encrypt the session key with recipient's public key (asymmetric)
- 5. Transmit both: encrypted session key + ciphertext.
- Workflow for decryption (receiver side):
- 1. Use private key to decrypt session key
- 2. Use session key to decrypt ciphertext back to plaintext.
- This hybrid approach gives speed (symmetric) + secure key distribution (asymmetric).

Keys & Keyrings

- Public/private keys are large numbers. Larger key → more security (but more cost).
- PGP stores:
- • Public key ring — public keys of others.
- • Private key ring — your private keys, encrypted with a passphrase.

- You must not lose your private key (or passphrase) — without it, you cannot decrypt messages.

Digital Signatures & Integrity

- Digital signatures offer authentication + integrity + non-repudiation.
- Basic signature scheme: encrypt (with private key) a message (or a hash) → decryption with public key verifies origin.
- Use of hash functions:
- • Compute hash (message digest) of the message.
- • Encrypt the digest with private key to form the signature.
- • Receiver recomputes hash, decrypts signature (with sender's public key), compares. If same → integrity + authenticity.

Certificates, Trust, & Revocation

- Digital certificate = public key + identity info + signature(s) binding them.
- Purpose: ensure that a public key really belongs to the claimed user (prevent MITM).
- PGP supports multiple signatures per certificate (multiple people can vouch).
- Certificate distribution: via certificate servers, or in a PKI (with CAs).
- X.509 format vs PGP format:
- • X.509: issued by CA, single-owner, single signature.
- • PGP: user can self-certify, multiple identities/signatures.
- Validity & trust:
- • Validity = certificate correctly binds public key and identity.
- • Trust = whether you accept someone's validation (signature) as authoritative.
- • Trust models: direct trust, hierarchical trust, web of trust (PGP's model)
- Certificate revocation: when key is compromised or no longer valid.
- • PGP: revoke signature or entire certificate; publish revocations on key servers.
- • PKI: revocation via Certificate Revocation Lists (CRLs) published by CA.
- Key splitting: dividing a private key into shares so that multiple pieces are needed to reconstruct it (for shared control).

What Is An SSL/TLS Certificate? (AWS)

Source: AWS — What is an SSL/TLS Certificate?

Definition & Purpose

- SSL/TLS certificate = digital object enabling systems to verify identity & establish encrypted connections via SSL/TLS.
- Operates within Public Key Infrastructure (PKI): trust via a third party (Certificate Authority, CA).
- Acts like a "digital identity card" for websites/servers.

Why They Matter

- They help ensure confidentiality: only client & server see data.
- They build trust: web browsers display padlock icon, “HTTPS,” valid certificate.
- Required for regulatory compliance (e.g., PCI DSS for payment data).
- SEO benefit: major search engines favour HTTPS sites.

Key Principles

- Encryption: ensures scrubbing of messages so only legitimate parties can interpret them. Uses public/private key pairs.
- Authentication: browser verifies the certificate (issued by CA) to ensure the server is who it claims.
- Digital signatures: included in certificates so that recipients can detect tampering.

Who Validates Certificates?

- A Certificate Authority (CA) issues SSL/TLS certificates.
- CA validates domain ownership and identity details before issuing.
- To be a trusted CA, must be recognized by browsers/OS as root CA.

Validity & Lifetime

- Certificates have a validity period; currently max ~13 months (reduced over time for security).
- Upon expiry, certificate must be renewed or replaced.

Certificate Contents

- An SSL/TLS certificate includes:
- Domain name (or names)
- Certificate Authority name
- CA’s digital signature
- Issuance date / expiration date
- Public key of the server
- SSL/TLS version / algorithm info

How It Works (SSL/TLS Handshake)

- Client (browser) requests connection to server (HTTPS).
- Server responds, sending SSL/TLS certificate (includes public key).
- Browser verifies certificate: checks validity, CA signature, domain match.
- Browser generates a session key, encrypts it with server’s public key, sends to server.
- Server uses its private key to decrypt session key.
- Both sides then communicate using symmetric encryption with the shared session key.
- The session key is used because symmetric encryption is more efficient thereafter.

Types / Classifications of Certificates

- By validation level:
- Extended Validation (EV): highest vetting, shows org info in browser.

- Organization Validation (OV): moderate vetting, shows some business info.
- Domain Validation (DV): lowest vetting just proving domain ownership.
- By domain support:
- Single-domain: secures one domain.
- Wildcard: secures domain + subdomains (e.g. *.example.com).
- Multi-domain (SAN / UCC): secures multiple domains under one certificate.

AWS Certificate Manager (ACM)

- AWS offers Certificate Manager to provision, manage, deploy SSL/TLS certificates easily.
- It automates renewal and deployment to AWS services like load balancers, CloudFront, API Gateway.
- It can manage both public and private certificates internally.