

# UNIT V

Email Security, IPSec, Cryptographic Applications & Case Studies

# Agenda — Unit V

- PGP — Pretty Good Privacy
- S/MIME — Secure/Multipurpose Internet Mail Extension
- PGP vs S/MIME Comparison
- IPSec Architecture & Protocols (AH, ESP, IKE)
- Security Association Bundles
- Secure Multiparty Computation (SMC)
- Virtual Elections & Cryptography
- Single Sign-On (SSO)
- Secure Electronic Transactions (SET)
- Cross-Site Scripting (XSS)

# PGP

Pretty Good Privacy — Secure Email Protocol

# What is PGP?

- PGP = Pretty Good Privacy — invented by Phil Zimmermann
- Open-source, freely available software package for email security
- Designed to provide all four aspects of security in email:
  - Privacy (Confidentiality)
  - Integrity
  - Authentication
  - Non-repudiation
- Uses a combination of hashing, secret key encryption and public key encryption
- Digital signature uses: one hash function + one secret key + two private-public key pairs

# PGP — Five Core Services

- **1. Authentication** — Via digital signature (SHA-1 hash + RSA/DSS)
- **2. Confidentiality** — Symmetric block encryption (CAST-128, IDEA or 3DES in CFB mode)
- **3. Compression** — ZIP algorithm applied before encryption, saving space
- **4. Email Compatibility** — Radix-64 encoding for ASCII-only mail systems
- **5. Segmentation** — Large messages split into  $\leq 50,000$  octet segments

# PGP — Steps at Sender Site (A)

- 1 Email is hashed using a hash function to create a digest
- 2 Digest is encrypted with sender's PRIVATE key → creates a signed digest
- 3 Signed digest is appended to the original email message
- 4 Original message + signed digest encrypted using a one-time secret (session) key
- 5 Session key is encrypted using the RECEIVER's public key

# PGP — Steps at Receiver Site (B)

1 Receiver receives: encrypted secret key + encrypted message/digest combination

2 Encrypted secret key is decrypted using receiver's PRIVATE key → recovers session key

3 Session key is used to decrypt the message + digest combination

4 Digest is decrypted using SENDER's public key

5 Original message is hashed independently to create a new digest

# PGP Authentication & Confidentiality (Detail)

- **Authentication**
  - Hash function: SHA-1 → 160-bit message digest
  - Algorithm: RSA or DSS for signing
  - Strength of RSA ensures only the private key holder can sign
- **Confidentiality**
  - Session key generated randomly for each message (used once)
  - Message encrypted: CAST-128, IDEA or 3DES in 64-bit CFB mode
  - Session key encrypted with recipient's public key (RSA) and prepended
  - Receiver uses RSA private key to recover session key, then decrypts

# PGP — Disadvantages

- **Difficult Administration**
  - Different PGP versions complicate administration and key management
- **Compatibility Issues**
  - Both sender and receiver must have compatible PGP versions
- **Complexity**
  - Hybrid approach (symmetric + asymmetric) is harder to understand than pure schemes
- **No Password Recovery**
  - No special recovery program — forgotten passwords result in lost messages permanently

# S/MIME

Secure/Multipurpose Internet Mail Extension

# What is S/MIME?

- S/MIME = Secure/Multipurpose Internet Mail Extension
- Industry standard for email encryption and digital signing
- Widely used by businesses to improve email security
- Supported by most corporate email clients
- Encrypts and digitally signs emails to verify authenticity and integrity
- End-to-end encryption solution for MIME (email) data
- Uses asymmetric cryptography to prevent message tampering by third parties
- Works like wax seals on letters — sender signs, recipient verifies

# S/MIME Certificate Characteristics

- **Authentication** — Verification of user or website identity
- **Message Consistency** — Guarantees contents have not been tampered with
- **Non-Repudiation** — Sender's identity and digital signature are validated
- **Privacy Protection** — Third-party data breach prevention
- **Data Encryption** — Mix of public and private keys (asymmetric cryptography)
- MIME type designated in S/MIME certificate
- Enclosed data fully prepared, encrypted, and packaged inside a digital envelope

# S/MIME — Supported Email Clients & Delivery

- **Popular S/MIME Supported Clients:**
  - iPhone iOS Mail, Apple Mail, Gmail, IBM Notes
  - Mozilla Thunderbird, Microsoft Outlook / Outlook on the Web
  - MailMate, CipherMail
- **How S/MIME Works:**
  - Sender signs email with private key
  - Email accompanied by sender's public key in transit
  - Recipient uses public key to verify signature
  - Recipient uses own private key to decrypt the message

# PGP vs S/MIME — Comparison (1/2)

#	PGP	S/MIME
1	Designed for plain text processing	Designed for email AND multimedia files
2	Less costly	Comparatively expensive
3	Good for personal & office use	Better suited for industrial use
4	Less efficient than S/MIME	More efficient than PGP
5	Depends on user key exchange	Relies on hierarchically valid certificate

# PGP vs S/MIME — Comparison (2/2)

#	PGP	S/MIME
6	Less convenient	More convenient (secure transformation of all apps)
7	Contains 4096-bit public keys	Contains 1024-bit public keys
8	Standard for strong encryption	Also strong encryption, with some drawbacks
9	Can be used in VPNs	Not used in VPNs; only for email services
10	Uses Diffie-Hellman digital signature	Uses Elgamal digital signature

# IPSec Architecture

Internet Protocol Security — AH, ESP & IKE

# IPSec — Overview

- IPSec = IETF standard suite of protocols for secure IP communication
- Provides: Data Authentication, Integrity, and Confidentiality
- Defines encrypted, decrypted and authenticated packet formats
- **Two core security protocols:**
  - AH — Authentication Header
  - ESP — Encapsulation Security Payload
  - Plus IKE — Internet Key Exchange for key management
- **IPSec Architecture also includes: algorithms, DOI, and key management**

# IPSec — Use Cases

- Encrypt application layer data
- Provide security for routers sending routing data across the public internet
- Provide authentication without encryption (verify data origin)
- Protect network data via IPSec tunneling (VPN connections)
  - All data between two VPN endpoints is encrypted end-to-end
- Prevent replay attacks — unauthorized retransmission of packets
- Provide anti-replay protection for packet sequences

# Authentication Header (AH) Protocol

- Provides: Authentication, Integrity, and Anti-Replay (NO encryption)
- Works on per-packet basis (connectionless services)
- **Two modes:**
  - Transport Mode — Some mutable IP header fields are not protected
  - Tunnel Mode — Entire packet protected; used for mutable fields
- **AH Header Fields:**
  - Next Header (8-bit) — identifies type of payload that follows
  - Sequence Number (32-bit) — monotonically increasing, used for replay protection
  - Security Parameter Index / SPI (32-bit) — identifies the Security Association
  - Authentication Data (variable) — Integrity Check Value (ICV)

# Encapsulating Security Payload (ESP) Protocol

- Provides: Confidentiality, Integrity, Authentication, and Anti-Replay
- Encryption/decryption: only authorized parties can read data
- Operates on payload (NOT the outer IP header)
- **Two implementations:**
  - ESP with optional Authentication
  - ESP with Authentication (recommended)
- **ESP Packet Fields:**
  - SPI (32-bit) — identifies Security Association
  - Sequence Number (32-bit) — anti-replay counter
  - Payload Data (variable) — encrypted actual data
  - Padding (0-255 bytes) — cipher block alignment
  - Next Header (8-bit), Authentication Data (variable, optional)

# Internet Key Exchange (IKE)

- IKE = Hybrid protocol for key management in IPSec networks
- Exchanges encryption/authentication keys over unsecured medium (internet)
- Defined by IETF in November 1998 (RFC 2407, 2408, 2409)
- Roots in: Oakley Protocol + SKEME + ISAKMP
- Default key exchange algorithm: Diffie-Hellman
- **Three Peer Authentication Methods:**
  - Authentication using RSA signatures
  - Authentication using a pre-shared secret
  - Authentication using encrypted RSA nonces

# IKE — Benefits & Advantages

- Allows encryption key changes during active IPsec sessions
- Eliminates need for manual specification of all IPsec security parameters
- Supports certification authority — additional layer of security
- Allows setting specific lifetimes for IPsec Security Associations
- Permits dynamic authentication of peers
- **Key IKE Features:**
  - Automatic negotiation and authentication
  - Anti-replay services
  - Fast connection speeds using NAT traversal
  - Reconnection attempts on dropped connections
  - Prevents DoS and replay attacks

# IPSec — AH vs ESP at a Glance

## Authentication Header (AH)

- Provides: Authentication + Integrity
- NO encryption/confidentiality
- Protects: IP header + payload
- Anti-replay protection
- Protocol Number: 51
- Mode: Transport or Tunnel

## Encapsulating Security Payload (ESP)

- Provides: Confidentiality + Auth + Integrity
- FULL encryption of payload
- Protects: payload only (not IP header)
- Anti-replay protection
- Protocol Number: 50
- Mode: Transport or Tunnel

# Security Association (SA) Bundles

- Individual SA can implement EITHER AH or ESP — not both
- For flows needing both, multiple SAs form a "bundle"
- SA Bundle = sequence of SAs through which traffic must be processed
- **Two ways to combine SAs into bundles:**
  - Transport Adjacency — Multiple security protocols on same IP packet, no tunneling
  - Iterated Tunneling — Multiple layers of security through IP tunneling, multiple nesting levels
- **ESP with Authentication Option — Auth + encryption, auth covers ciphertext**
- **Transport-Tunnel Bundle — Inner AH transport SA + outer ESP tunnel SA**

# IPSec — 4 Basic SA Combination Cases

- **Case 1 — End-to-End (Host-to-Host) IPSec**
  - All security between end systems; can use AH, ESP, or ESP+AH in transport/tunnel
- **Case 2 — Gateway-to-Gateway (VPN)**
  - Security only between gateways (firewalls/routers); single tunnel SA required
- **Case 3 — Gateway + End-to-End Security**
  - Builds on Case 2 with additional host-level security for specific applications
- **Case 4 — Remote Host to Firewall**
  - Remote host accesses org firewall via Internet; tunnel mode between remote host and firewall

# Secure Multiparty Computation

Privacy-Preserving Distributed Computation

# Secure Multiparty Computation (SMC)

- Cryptographic technique for  $n$  parties to jointly compute a function securely
- Computes on distributed inputs without any party revealing their data
- Also called SFE — Secure Function Evaluation
- History: originated in 1970s; formalized as two-party computation in 1982
- Enabled by: blockchain, mobile computing, IoT, cloud computing
  - Example: Compute maximum salary among co-workers without revealing individual salaries
- Data is kept encrypted, broken up, and distributed — no quantum attack risk
- Hot area of research due to blockchain and distributed systems demand

# SMC — Key Techniques

- **Shamir Secret Sharing** — Secret  $s$  shared among  $n$  parties;  $t+1$  needed to reconstruct
- **Honest Majority MPC** — Uses finite field arithmetic; circuit is Turing complete
- **Input Sharing** — Each party shares input with random noise; noise removed at output
- **Circuit Evaluation** — Gates evaluated serially; addition and multiplication gates
- **Private Set Intersection** — Two parties find intersection without revealing input sets
- **Threshold Cryptography** — Operations split across parties; no single party holds secret
- **Dishonest Majority MPC** — GMW, garbled circuits, Tiny OZ protocols for corrupted majority

# SMC — Benefits vs Limitations

## Benefits

- No trusted third party needed
- Data privacy preserved during computation
- High accuracy in cryptographic results
- Quantum-safe: data broken and encrypted
- Enables private analytics across orgs
- Supports distributed system designs

## Limitations

- Computational overhead (random number generation slows runtime)
- High communication costs (data distributed across network)
- Complex implementation
- Not practical for very large-scale real-time systems
- Semi-honest vs malicious adversary handling is complex

# Virtual Elections

Can Cryptography Secure Electronic Voting?

# Effective Voting System Requirements

- **Security** — Prevent attacks and tampering from adversaries
- **Accuracy** — System must return results that accurately reflect votes cast
- **Verifiability** — Results must be verifiable to maintain public trust
- **Anonymity** — Votes must remain secret to prevent coercion or bribery
- **Usability** — System must be easy enough that all voters can use it
- **Speed** — Election results needed within a reasonable timeframe
- Key tension: Anonymity vs Verifiability are fundamentally in conflict
- Public vote list enables verification but destroys anonymity

# Cryptographic Properties for Secure Voting

- **Confidentiality**
  - Keeping vote information secret from unauthorized parties — achieved via encryption
- **Authentication**
  - Validating voter identity and that data transfers genuinely come from them
  - Implemented using digital signatures
- **Integrity**
  - Ensuring vote data has not been tampered with
  - Also proven via digital signatures
- Cryptography addresses Security, Accuracy, Verifiability, and Anonymity
- BUT: organization, processes, and implementations are equally critical

# Single Sign-On (SSO)

One Credential — Multiple Applications

# Single Sign-On (SSO) — Overview

- Authentication scheme: one set of credentials → access multiple apps
- Example: Google login gives access to Gmail, Google Docs, Drive, etc.
- **Without SSO:**
  - Each website maintains its own database of usernames and passwords
- **With SSO:**
  - Shared cluster of authentication servers
  - Users enter credentials once; SSO authenticates for all connected apps
  - Identity provider (e.g. Active Directory) verifies identity
- Highly recommended to combine with MFA (smart tokens, OTP)

# How SSO Works — Step by Step

1 User enters login credentials on a website

2 Website checks if user is already authenticated by SSO solution

3 If not, SSO login page is presented to the user

4 User submits username and password to the SSO solution

5 SSO seeks authentication from the Identity Provider (e.g. Active Directory)

# SSO — Advantages vs Disadvantages

## Advantages

- No need for separate accounts/passwords per app
- Risk of 3rd party credential storage mitigated
- Prevents keylogging (fewer password entries)
- Increased user convenience
- Reduces IT costs for credential management
- Improves customer experience and retention

## Disadvantages

- Single point of failure: if credentials stolen, ALL apps are compromised
- Auth system must have high availability
- Loss of availability = denial of service across all connected apps
- Requires strong MFA to compensate for single-credential risk

# Secure Electronic Transaction (SET)

Cryptographic Protocol for Online Payments

# SET — Steps 1 to 6: Order & Verification

1 Customer opens a credit card account with a bank (issuer) supporting SET

2 Customer receives a digital certificate from CA (name, public key, expiry)

3 Merchant receives a digital certificate to prove trustworthiness

4 Customer places order — merchant sends order details back to customer

5 Merchant sends digital certificate to customer to verify merchant identity

# SET — Steps 7 to 11: Payment & Fulfillment

1 Merchant requests payment authorization from payment gateway via acquirer

2 Payment gateway cross-verifies credit card with issuer — authorizes or rejects

3 Merchant confirms order to customer after gateway authorization

4 Merchant provides goods and services per the customer's order

5 Merchant requests payment — gateway interacts with issuer, acquirer, clearinghouse to transfer funds

# Cross-Site Scripting (XSS) — Types & Prevention

- XSS: application sends untrusted data to browser without validation → executes malicious scripts
- **Stored XSS (Persistent)**
  - Malicious input stored on server (database/forum/comment field); retrieved by victims later
- **Reflected XSS (Non-Persistent)**
  - User input immediately returned by web app in error/search results without storage
- **DOM-Based XSS**
  - Data source and sink both in DOM; data flow never leaves the browser
- **Prevention:**
  - Escape all untrusted data based on HTML context (body, attribute, JS, CSS, URL)
  - Robust validation mechanisms before accepting special characters as input