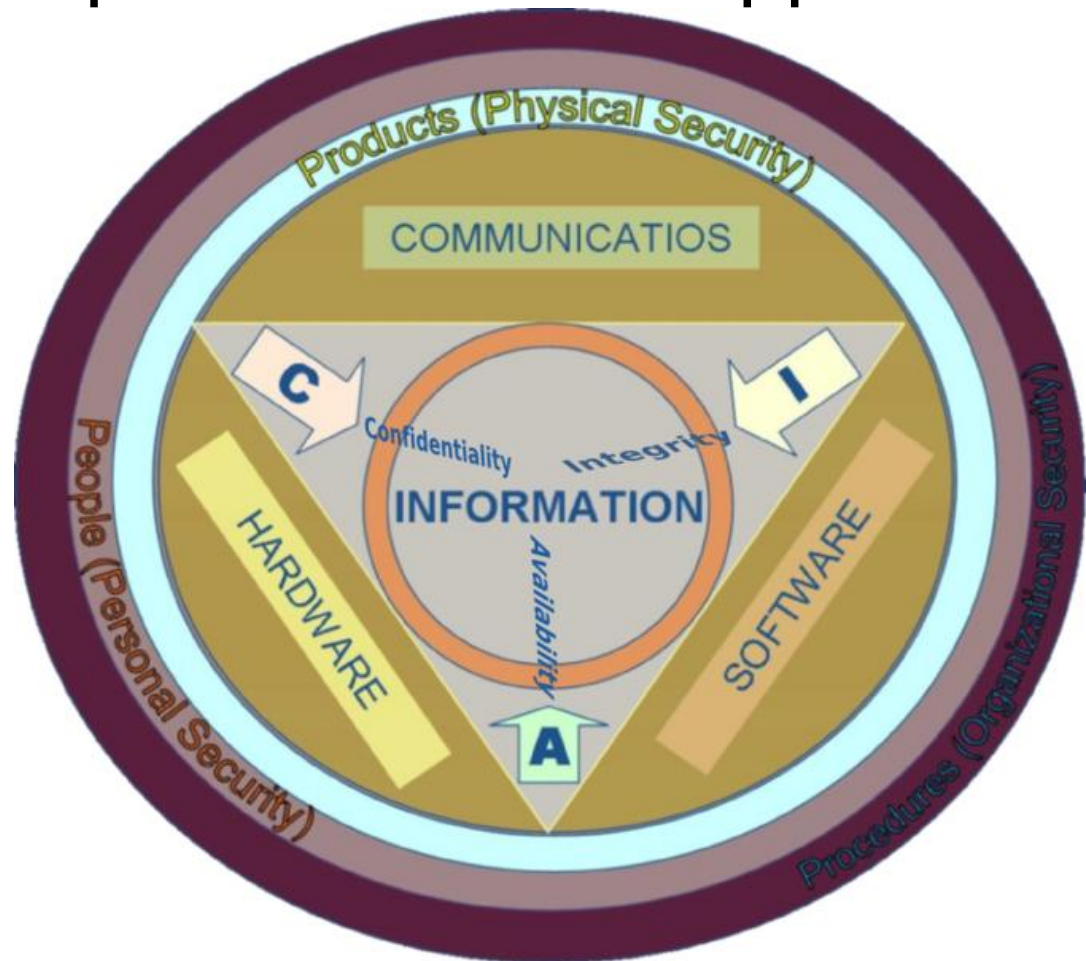


Security Fundamentals



Application-Driven Design

- Relate to application requirements
- What security properties does an application require?
 - Confidentiality
 - Integrity
 - Availability



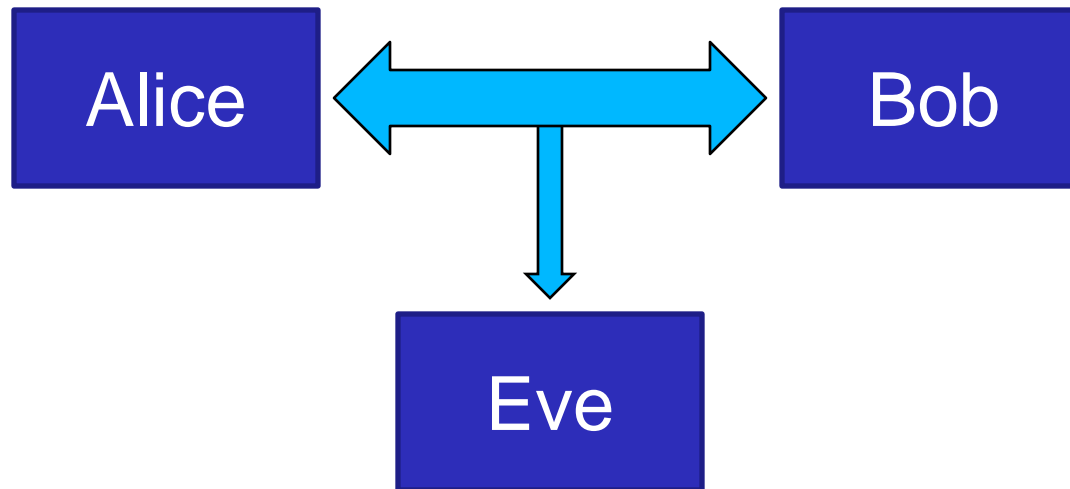
Types of Adversaries

- Adversary Location
 - On-Path: able to see all traffic between communicating parties
 - Off-Path: not able to see traffic
- Adversary Capability
 - Passive: only observe, but not transmit
 - Active: able to transmit
- Off-Path Passive not a threat
- On-Path Active most dangerous
 - Change, inject, delete packets



Confidentiality

- Protect sensitive information
- Eavesdropper threat
 - Alice and Bob talking
 - Eve listening
- Need to protect information in transit
- Basic solution: encrypt it



Encryption

- Two basic approaches
 - Symmetric Key Cryptography
 - Public Key Cryptography

P = Plaintext

C = Ciphertext

K = Key

- Symmetric Key

- Single key to encrypt and decrypt, known to A, B

$$C = f(K, P) \quad P = f^{-1}(K, C)$$

- Public Key

- Everyone knows public key, B knows private key
- A encrypts with public key, only B can decrypt

$$C = f(K_{pub}, P) \quad P = f^{-1}(K_{priv}, C)$$



Simple Examples

- Symmetric Key

$$f(K, P) = P \otimes K \quad f^{-1}(K, C) = C \otimes K$$

- *Perfect secrecy* if $\text{len}(K) \geq \text{len}(P)$ and $H(K) = \text{len}(K)$

- Keys as long as your message impractical

- Imagine creating a unique, random 2000-character password for every packet your computer sends

- Need way for receiver to know random password

- Want shorter passwords

- Use key-stream generator (pseudo-random number generator)

- Example: RC4

$$f(K, P, IV) = P \otimes g(K + IV)$$

$$f^{-1}(K, C, IV) = C \otimes g(K + IV)$$



Simple Examples

- Block Ciphers
 - Pseudorandom function: $h(K, msg)$
 - Input: key, plaintext
 - Output: ciphertext, statistically independent of inputs
 - Encrypt multiple blocks: Encryption “Modes”
 - Cipher-Block Chaining (CBC)
 - Break message into blocks B_1, \dots, B_N

$$f(K, IV, B_1, \dots, B_N) = g(K, IV, B_1) \parallel g(K, IV, B_2) \parallel \dots \parallel g(K, IV, B_N)$$

$$g(K, IV, B_i) = h(K, B_i \otimes h(K, B_{i-1} \otimes h(K, \dots h(K, B_1 \times IV) \dots)))$$

- Examples: DES, AES



Public-Key Cryptography

- Based on Abstract Algebra and Number Theory
- RSA: most common, easiest to understand
 - Pick primes p, q , compute $n=pq$
 - Compute $\varphi(n) = (p-1)(q-1)$
 - Select e , such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$
 - Compute d , such that $de \equiv 1 \pmod{\varphi(n)}$
 - d is e^{-1} modulo $\varphi(n)$
 - Public key: n, e ; Private key: d
 - Encryption: $c = f(m) = m^e \pmod{n}$
 - Decryption: $m = f^{-1}(c) = c^d \pmod{n}$
 - Works because $c^d \equiv (m^e)^d \equiv m^{ed} \equiv m^1 \equiv m \pmod{n}$
 - ... since $x^{\varphi(n)} \equiv 1 \pmod{n}$



Public-Key Cryptography

- Implications
 - Alice distributes public key to everyone
 - Anyone can send a secure message to Alice by encrypting it with her public key
 - Only Alice knows the private key; only Alice can decrypt it
- Security of Public Key crypto
 - Relies on integer factorization problem
 - If adversary can factor n into p and q he can determine $\varphi(n)$ and consequently private key d
 - Integer factorization still exponentially difficult
 - Quantum computers can factor in polynomial time
 - If quantum computers are realized, PKC broken



Integrity

- Confidentiality allows two people to communicate in secret
- How do you know an adversary hasn't change the message?

$$C = f(K, P) = P \otimes K$$

$$C' = C \otimes \alpha$$

$$\begin{aligned} f^{-1}(K, C') &= C' \otimes K \\ &= (C \otimes \alpha) \otimes K \\ &= ((P \otimes K) \otimes \alpha) \otimes K \\ &= P \otimes \alpha \end{aligned}$$

- Need integrity protection
- Cryptographically-secure checksum



Integrity

- Need checksum that can be validated with a key
- Only those with the key can create it
- Symmetric-Key
 - Sender and receiver share a key
 - Same key for creating and validating checksum
 - Example: Hash function with key as additional input
- Public-Key
 - Sender has private key, everyone has public key
 - Sender “digitally signs” with private key
 - Everyone can validate with public key
 - RSA can be adapted – simply encrypt with private key



Integrity: Authentication

- Side effect of integrity: Authentication
- Know **WHO** you are talking to
- Identity associated with keys used for integrity
- Anything protected with that key implies it originated from the person with that key

- Separate issue: authorization
 - Is the authenticated user *allowed* to send the message that they sent?



Availability

- Probability of a system being available when users need to access it

- Official definition:
$$A = \frac{E[Uptime]}{E[Uptime] + E[Downtime]}$$

- Alternate derivation:
 - Let $X(t) = 1$ if system available at time t

$$A(t) = \Pr[X(t) = 1] \quad A = \lim_{c \rightarrow \infty} \frac{1}{c} \int_0^c A(t) dt$$

- Adversary seeks to minimize A to deny service to valid users



Denial of Service Attacks

- Interference Attacks
 - Interfere with link to prevent communications
 - Physically break link
 - Add noise to link
 - Manipulate protocol to prevent communications
 - TCP-FIN attack
 - Advertise faulty router information
- Resource Exhaustion Attacks
 - Consume resources of hosts
 - TCP-SYN attack to prevent new connections to server
 - Consume resource of network
 - Flood network with traffic



Denial of Service Attacks

- Mitigation
 - Build in protections to specific attacks in protocols and applications
 - Network-based filtering to block sources of DoS attacks
- Bigger Problem
 - Distributed Denial of Service (DDoS)
 - Many hosts acting as one to disrupt service
 - More difficult to mitigate



Case Study: PKI

- Public Key Infrastructure (PKI)
- Everyone generates public/private key pair
- Public keys sent to certificate authority (CA) with proof of identity
- CA signs public keys with their private key, returns to user, called “certificate”, which contains identity of the user
- To send secure message to user, ask for their certificate, validate certificate with public key of CA, then encrypt with the public key within

