Week 1

CIA Security Properties:

* Confidentiality: authorised parties only
  + cryptography, encryption
* Integrity: unmodifiable by unauthorised parties
  + data (content), origin (authentication)
* Availability: cannot be accessed by authorised parties
* Authenticity: origin is guaranteed
* Accuracy: Error-free

Main Categories of Threats

* disclosure (unauthorised access) (C)
* deception (acceptance of false data) (I, A)
* disruption (interruption of operation) (A)
* usurpation (unauthorised control of some system) (A)

Attacks

* MITM – intercept connection and pose as legit user
* DDoS – overload with traffic, more devastating with internet of things (IoT)
* Ransomware – encrypt data and hold it for ransom

Week 2

Encryption/Decryption:

* Symmetric key encryption – same encryption and decryption keys
* Asymmetric key encryption – different encryption and decryption keys

Classical Ciphers

* Principles: substitution/replace (confusion), permutation/reorder (diffusion)
* Caesar / Vigenere
* Cannot be made secure against statistical analysis
* Monoalphabetic
  + number of possible keys for alphabet of size N is N! = sqrt(2\*pi\*N) \* (N/e)N
* Unicity Distance
  + Least number of ciphertext characters needed to uniquely determine key
    - (log2 E) / d, where E is the number of keys and d is the redundancy
* Redundancy: d = R – r bits
  + Absolute rate R = log2A is the number if bits needed to represent each character in alphabet of size A, in English R = 4.7 bits
  + True rate r of a language is the average number of bits required to represent characters in a language, in English: r = 1-1.5 bits.
  + All natural languages are redundant, e.g. “Bb invitd Alc fr dinr” can still be read.
  + A completely random source has no redundancy
* DES: Key = 56 bits, Block = 64 bits, Key space = 256, Feistel structure
  + Step 0: Split 64-bit text into two: L0 and R0 , introduce key K1 - KN
  + Step 1: L1 = R0, R1 = L0 XOR f(K1, R0)
  + Step 2 - 15: Rk = Lk-1 xor f(Kk, Rk-1),
  + Step 16: R16 = L15 xor f(K16, R15), L16 = R15

Distributed Computing

client ⬄ service = common interface  
middleware platform = different interface  
OS A ⬄ OS B = cannot directly communicate – needs middleware

Remote Procedure Call (RPC)

Kerberos

Threats to distributed networks

User impersonation – pretend to be another user from the same workstation  
Network address impersonation – change the network address to impersonate another workstation  
Eavesdropping, replay attacks, etc

Basic Ideas

Authentication Server – centralised system distributing long lifetime tickets for the whole system  
Ticket Granting Servers – short lifetime tickets  
Service Servers – different services

Architecture

* Client => AS – request
* AS => Client – TGS ticket
* Client => TGS – TGS ticket + authenticator
* TGS => Client – Server ticket
* Client => Server – Server ticket + Authenticator
* Server => Client – Service

Phase 1

1. C => AS: IDc, IDtgs, TS (client sends the **client** ID, **ticket granting server** ID and **timestamp**)
2. AS => C: if client is authenticated, Kc[Kc, tgs, IDtgs, TS2, LT2, Tickettgs] (AS returns encrypted key – client key, TGS, TGS ID, timestamp when correctly authenticated, how long ticket is valid, ticket for client: Tickettgs Ktgs[Kc, tgs, IDc, ADc (client address), IDtgs, TS2, LT2])

Phase 2

1. C => TGS: IDs, Tickettgs, Authc = Kc, tgs[IDc, ADc, TS3] (client sends **server** ID, ticket received from TGS and encrypted key containing client ID, client address and 3rd timestamp)
2. TGS => C: Kc, tgs[Kc,s(one-time key), IDs, TS4, LT4, Tickets => Ks[Kc,s, IDc, ADc, IDs, TS4, LT4]]

Phase 3

1. C => S: Tickets, Auth’c => Kc,s[IDc, ADc, TS5]
2. S => C: kc,s [TS5 + 1]

Inter-realm auth

Client is in Kerberos A, server is in Kerberos B  
Client must be authenticated but NOT registered by Kerberos B TGS  
TGS in both realms must share secret