- Authentication
  - Methods
    - Pairwise
      - messages exchanged all in two clients
      - assumption — Clients A and B have already establish key sharing
      - types
        - 1-way
          - $A \to B$: A
          - $B \to A$: I_b
          - $A \to B$: $\{I_b\}^{K_{AB}}$
          - vulnerability — reflection attack
            - adversary start a new session with sender, repeating the message received
            - causing message direction missing, not synchronized in both sides
        - 2-way
          - $A \to B$: A
          - $B \to A$: I_b
          - $A \to B$: $\{I_a, I_b\}^{K_{AB}}$
          - $B \to A$: $\{I_a\}^{K_{AB}}$
    - Third-party protocol
      - how? — using AS - authentication server — for stroing long-term keys
      - why? — saving key exchanges
      - protocols
        - NS78
          - 1-way auth
            - $A \to AS$: A, B, I_a
            - $AS \to A$: $\{I_a, B, K_{ab}, \{A, K_{ab}\}^{K_{bs}}\}^{K_{as}}$
            - $A \to B$: $\{A, K_{ab}\}^{K_{bs}}$
            - $B \to A$: $\{I_b\}^{K_{ab}}$
            - $A \to B$: $\{I_b - 1\}^{K_{ab}}$
            - should use nonces to avoid old key replay;
            - should determine destination or it will be masquerade by adversary;
          - 2-way auth
            - **A->B** 5'. $\{I_a, I, I_b\}^{K_{as}}$
            - **B->A** 6. $\{I_b - 1\}^{K_{bs}}$
          - vulnerability
            - message 3 repeatedly replay by A to B
              - A knows the plaintext-ciphertext pair, could make off-line guessing
              - should pad a confounder, which is random
            - assume adversary knows Kab — repeatedly replay message 3, make guessing about Kb
        - DS Replay Detection Protocol
          - add timestamp to NS78 to counter replay
          - set valid interval
          - vulnerability
            - tickets for connecting another server cant be reused
            - sensitive to network delay and out-of-sync clock
        - OB87
          - more efficient
          - but servers could not know if the other knows the Key ...
        - Kerberos
          - v4
            - introduces TGS - Ticket Granting Server
              - substituting a K_{server-tgs} for long-term key K_server
              - in order to decrease expose of the time long-term key
              - enable ticket reuse for TGS and application
            - replay detection — sliding window — 2nd chance
              - vulnerability
                - adversary could steal the T and start the replay before the 2nd chance
              - countermeasure? — maybe assert ip verification in the plaintext needed to be encrypted
            - credential cache — containing session keys with version and tickets — key version number
              - for accepting tickets encrypted by old keys
              - maintain the version number of key — having number/time limit
            - network-layer addresses (ticket user's) in tickets — avoid ticket stolen reuse by adversary
            - KDC replication
              - avoid single point failure and performance bottleneck
              - have master KDC and slave KDCs (replicas)
            - cross-realm (aka non-hierarchical) authentication
              - but non-transitive — avoid impersonation of being a third-side local
              - limitation — O(n^2) maintaining all shared keys between every 2 domains in n domains
            - other vulnerability — no pre-authentication — adversary could claim as A to start a conversation with AS, AS will not examine it and will directly generate a response.
          - v5
            - delegation of access
              - forwardable TGT
                - used to request TGT with diff address
                - optional transitive
                - time limited
              - proxyable TGT
                - used to request ticket with diff address
                - no transitive
                - time limited
            - ticket revocation via lifetime control
              - motivation — if compromise is detected, need to revocate the tickets distributed
              - concern — some tickets might be long-lived and could not make revocation to them before their expiration
              - solution — in this mode, the ticket will generally have a short lifetime, but it will constantly be renewed if it is not in a revocation list
            - inter-realm (aka hierarchical) authentication — client should travers a trust path to obtains TGTs (finally he gets the target TGT)
            - pre-authentication
              - motivation — if a client send AS_REQ, the AS will directly reply AS_REP, containing plaintext-ciphertext pair. in this case, an adversary could send multiple AS_REQ to get many plaintext-ciphertext as he wants, then he could do the off-line guessing by decrypt the ciphertext by various guessed passwords (converted to keys) until they find the decryption match the plaintext.
              - solution — PADATA
                - client need to take the timestamp encrypted with the key as part of the request
                - AS should check the decrypted result of the timestamp with the timestamp provided in the block
                - https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-kile/ae60c948-fda8-45c2-b1d1-a71b484dd1f7
              - other promotions
                - separation of entities type (human/server)
                  - avoid human user from obtaining the plaintext-ciphertext pairs
                  - using a flag set
                - separation of different domains
            - double TGT, client-to-client authentication
              - client A and B both obtain tickets (TGT) to TGS from AS (independent process)
              - B send the his TGT to A
              - A send A's ans B's TGT to TGS
              - TGS generate the session key Kab back, passing to A and B in different encryption key
              - * only achieve first order beliefs
  - Design — logic BAN
    - to verify if an authentication protocol could work
    - components
      - assumptions
      - statements

        

      - axioms

        

    - sample — NS78 — see LogicBAN ppt p10-14
    - personal summary for deduction
      - define 2 clients as A and B, and a server AS,
      - usually, first belief of A side will be achieved by message containing a key Kab and a nonce encrypted by Ka-as through: verification of message meaning (A believes AS said X) , freshness rule (A believes #(X)), nonce verification (A believes AS believes X), belief extension (A believes AS believes Kab, A believes AS believes #(Kab)), jurisdiction (A believes Kab, A believes #(Kab));
      - first belief of B side will be achieved by part of the message containing a key Kab and a nonce encrypted by Kb-bs as above (B believes Kab), the second belief of B side will be achieved simultaneously by the other part of the message containing a nonce encrypted by Kab (B believes A believes Kab)
  - Authentication trust
    - **Authentication Trust Axioms:**

      A1: $\forall a, Y > X \Rightarrow a \lhd |Y| \vdash X$

      A2: $\forall a, Y > X \Rightarrow X \lhd |Y| \vdash a$

      A3: $Y > X \ , \ \ldots \ Y \lhd |W| \cdot Z \Rightarrow X \lhd |W| \cdot Z$
    - personal summary
      - according to A1
        for AS1 > A,
        AS2, AS3, B (anyone else) could get authentication from AS1 about A
      - according to A2
        for AS1 > A,
        A could trust AS2 (anyone AS1 trust)
      - the axiom A3 is usually applied when W has a peer link with Z
    - the three axioms allow only the "up to the nearest peer link – across the peer link – and down" policy