**IS ZA Authentication**

Identification + Authentication + Authorization + Accounting = Access Control

Threee Different Authentication Factors:

* What you know
* … have
* … are

Two-factor authentication:

* Originally introduced because:
  + Risk that an attacker could learn a password was considered too high
  + Passwords are sometimes easy to guess, told to others, sent in emails etc
* 1st generation possession factor:
  + TAN
  + One-time password tokens
  + Advantages:
    - Security is increased
    - Simple to use, usable with any client device
  + Limitations:
    - Vulnerable to classic e-mail phishing attacks
      * Victims using fake website and giving out their credentials
* 2nd generations: prevent E-mail Phishing
  + Idea: User can only produce a valid one-time password during a session with the website
  + Is typically implemented using a challenge-response based approach:
    - iTan lists
    - challenge-response token
  + Limitations:
    - Vulnerable to MITM attacks
      * Lure victim with a phishing e-mail to the fake banking website
      * In the background, directly interact with the real banking website to authenticate with the credentials of the victim
* 3rd Generation: Mobile Phone-based Approaches
  + Security-wise, nothing is improved
    - MITM attacks still possible
  + MTAN

User authentication protocols:

* Define procedures and messages needed to authenticate users
* NTLM, Kerberos, Shibboleth
  + Most of these protocols are designed to be efficiently used in large environments
  + Most of them do not only perform authentication, but also establish a session key between the two endpoints => to protect the data exchange

Direct user authentication

* User contacts a host/server and sends credentials
* Host/server has all information to authenticate the user

Indirect user authentication

* User contacts a host/server and sends credentials
* Host/server does not directly authenticate the user
* To authenticate the user, an additional authentication server is used
* Examples: RADIUS, NTLM, Kerberos, Shibboleth
* Allows to centralize authentication and management of credentials
  + Scales much better with larger environments and allows Single Sign-On

NTLM:

* Windows domain : collection of users and services (emails, fileshares, printers,…)
  + Access is controlled by a Domain controller (DC)
  + Within a windows domain, we have centralized administration & SSO:
  + Flexibility:
    - Assignement of users to groups
    - Multiple domains possible (trust relationships between DCs)
  + All users and servers must trust the domain controller

Positive:

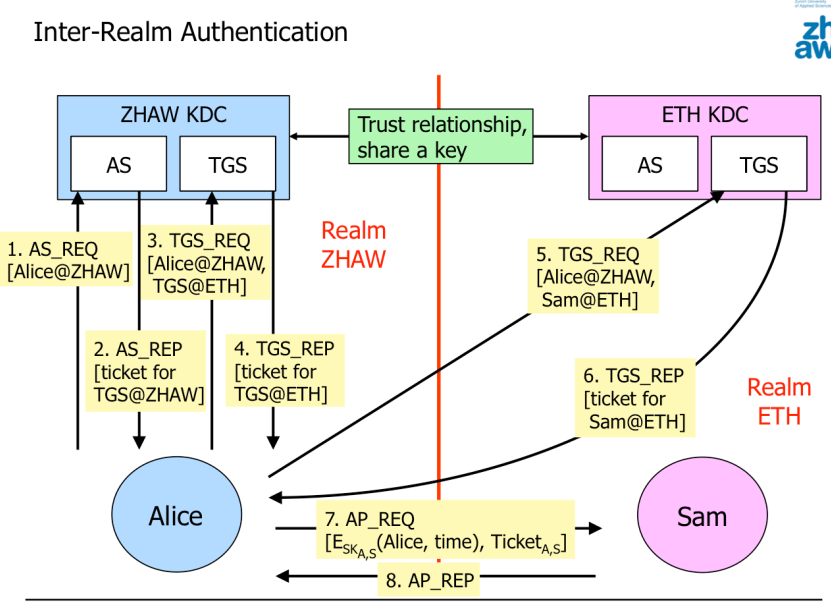
* Overall complexity is low
* Password is never transmitted in the clear
* Using a fresh challenge (nonce) each time prevents replay attacks
* Server learns the user’s key, which can be used to create session keys to protect the subsequent communication between client and server

Negative:

* The authentication process must be repeated for every use of a server
  + DC can become a bottleneck
* Weak or short NTLM passwords can be cracked offline
* The server is not authenticated => can be impersonated
* The server learns the user’s key

Kerberos

* User authentication in IP networks
* Use of secret key (no public key) cryptography
  + Duty: distribute a shared secret between alice and sam
* A Kerberos participant is called Principal
  + Each principal shares a common secret => principal’s master key
    - With a centralized server => Key Distribution Center (KDC)
  + All principals trust the KDC
  + Principals and the KDC form a realm
  + Communication between two principals:
    - Authentication and session key distribution is done via the KDC
* Ticket-based approach
  + A principal A that wants to access principal B asks the KDC for a ticket
  + This ticket can then be used by A to get access to B
* In windows:
  + KDC = domain controller
  + Realm = domain
* KDC
  + Splitting into two services
    - Reasons:
      * Load balancing
      * Separation of known credentials
  + Authentication Service (AS) authenticates users
    - User receives a Ticket-Granting Ticket (TGT)
    - Ticket is valid for a specified lifetime
    - Usually only needed once per workday just after logging in
  + Ticket Granting Service (TGS) issues tickets to access services
    - To access the TGS, the user needs a TGT from the AS
    - Tickets have a lifetime and can be reused if a service is accessed repeatedly
    - Therefore, the TGS must usually be contacted once per workday for reach service used
* Timestamps are used in both directions between client and AS, TGS and servers
  + This enables authentication and replay detection in both directions
* Much more efficient than NTLM
  + Reasonably large ticket lifetimes avoid contacting the AS/TGS frequently
* Kerberos requires a “reasonably synchronidsed” time among the participants
  + Because replay attacks are detected based on timestamps



* KDCs in the different realms => trust relationship and share a common key
* Initial authentication always happens in the own realm
* The only fundamental difference compared to the intra-realm authentication is the involment of two TGSs
* Only the TGS from the other realm distributes the credentials to access Sam
* Trust relationship is necessary for the KDCs to distribute and TGTs from and the other KDC

Shibboleth

* a system for federated identity management
  + Identity information is managed and used across multiple security domains
    - Security domains: individual organizations, universities eg.
    - Federation:
      * Collection of security domains that agree to interoperate
  + HTTPS
  + Public key crypto

