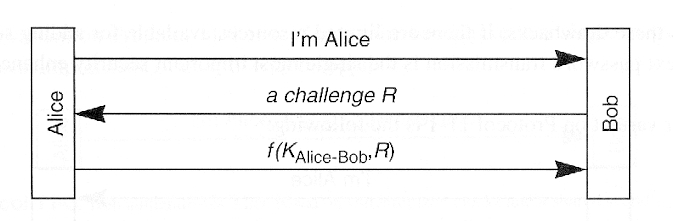
**Presentation Transcript**

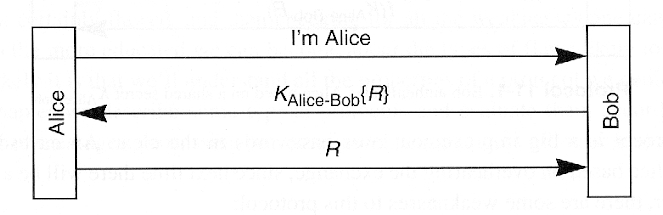
1. [**Security Handshake Pitfalls**](https://image2.slideserve.com/5332869/security-handshake-pitfalls-l.jpg)
2. [**Authentication Handshakes**](https://image2.slideserve.com/5332869/authentication-handshakes-l.jpg) • Secure communication almost always includes an initial authentication handshake: • Authenticate each other • Establish sessions keys • This process may involve many flaws
3. [**Login with shared secret**](https://image2.slideserve.com/5332869/login-with-shared-secret-l.jpg) • Approaches • Using KAlice-Bob as a secret key to encrypt R: KAlice-Bob{R} • Hashing R and KAlice-Bob: h(KAlice-Bob, R) • Different time uses different R • Discussions • Authentication is not mutual: Trudy can convince Alice that she is Bob • Off-line password guessing attack – assuming KAlice-Bob is derived from a password • Trudy can compromise the database at Bob and later impersonate Alice
4. [**Login with shared secret cont’d**](https://image2.slideserve.com/5332869/login-with-shared-secret-cont-d-l.jpg) • All the previous weakness remains • Minor security difference • Alice has to be able to reverse what Bob has done to R • If KAlice-Bob is derived from a password, it is vulnerable to a dictionary attack
5. [**Login with shared secret cont’d**](https://image2.slideserve.com/5332869/login-with-shared-secret-cont-d1-l.jpg) Alice and Bob have reasonably synchronized clocks more efficient Impersonate Alice within the acceptable clock skew If Trudy can set Bob’s clock back – reuse overheard timestamps
6. [**Authentication with One-Way Public Key**](https://image2.slideserve.com/5332869/authentication-with-one-way-public-key-l.jpg) • Advantage • Database at Bob need not be protected from unauthorized disclosure • Weakness • Trudy can trick Alice into signing/decrypting: Trudy can forge Alice’s signature on some quantity
7. [**Mutual Authentication**](https://image2.slideserve.com/5332869/mutual-authentication-l.jpg) • Inefficient
8. [**Mutual Authentication**](https://image2.slideserve.com/5332869/mutual-authentication1-l.jpg) • Reflection Attack
9. [**Reflection Attack**](https://image2.slideserve.com/5332869/reflection-attack-l.jpg) • General Principle • Do not have Alice and Bob do exactly the same thing • Different Keys: have the key used to authenticate Alice be different from the key used to authenticate Bob • Different Challenges: the challenge from the initiator (Alice) looks different from the challenge from the responder
10. [**Mutual Authentication**](https://image2.slideserve.com/5332869/mutual-authentication2-l.jpg) • Password Guessing without eavesdropping: send a message to Bob,… • How to fix?
11. [**Mutual Authentication**](https://image2.slideserve.com/5332869/mutual-authentication3-l.jpg) • Public Keys • Assume Alice and Bob know each other’s public keys
12. [**Mutual Authentication**](https://image2.slideserve.com/5332869/mutual-authentication4-l.jpg) • Timestamps • Require synchronized clocks • Alice and Bob have to encrypt different timestamps
13. [**Integrity/Encryption for Data**](https://image2.slideserve.com/5332869/integrity-encryption-for-data-l.jpg) • In order to provide integrity and/or encryption protection of the data following the authentication exchange, it is necessary for Alice and Bob to encrypt and/or add integrity • Require a session key established during mutual authentication
14. [**Establishment of Session Keys**](https://image2.slideserve.com/5332869/establishment-of-session-keys-l.jpg) • Shared Secret based authentication • After the authentication.. • Use KAlice-Bob{R} as the session key? • Has been used as the third message in the authentication handshake • Use (KAlice-Bob+1){R} as the session key • Use KAlice-Bob{R+1} as the session key? • Trudy impersonates Bob:
15. [**Nonce Types**](https://image2.slideserve.com/5332869/nonce-types-l.jpg) • Timestamps • Require reasonably synchronized clocks • Large random numbers • Tend to make the best nonce • Cannot be guessed/predicted • Sequence number
16. [**Nonce Types**](https://image2.slideserve.com/5332869/nonce-types1-l.jpg) R has to be unpredictable: suppose Eve impersonates Alice
17. [**Nonce Types**](https://image2.slideserve.com/5332869/nonce-types2-l.jpg) • R has to be unpredictable: Eve first impersonates Bob, then impersonates Alice
18. [**Nonce Types**](https://image2.slideserve.com/5332869/nonce-types3-l.jpg)
19. [**Privacy and Integrity**](https://image2.slideserve.com/5332869/privacy-and-integrity-l.jpg) • Replay attack • Use long sequence numbers • Sequence number space rollover • Key rollover: changing keys in the middle of a conversation
20. [**Mediated Authentication**](https://image2.slideserve.com/5332869/mediated-authentication-l.jpg) • Trudy may claim to be Alice and send “I am Alice” • Will not do Trudy any good • It is possible that Alice’s messages get to Bob first, so Bob does not know how to decrypt it • Using a Ticket Must be followed by a mutual authentication exchange – confirm that Alice and Bob have the same key
21. [**Needham-Schroeder Protocol**](https://image2.slideserve.com/5332869/needham-schroeder-protocol-l.jpg) • Classic protocol for authentication with KDC • Many others have been modeled after it (e.g. Kerberos)
22. [**Needham-Schroeder Protocol**](https://image2.slideserve.com/5332869/needham-schroeder-protocol1-l.jpg) • Nonce: a number that is used only once • A sequence number or a large random number • Deal with replay attacks • Reflection Attack • Bob -> Alice: KAB{N2-1, N3} • Assume Ni multiple of encryption blocksize • ECB: • Message splicing: put together own plus revealed • With CBC, no need to decrement N2, N3 • A Vulnerability • When Trudy gets a previous key used by Alice, Trudy may reuse a previous ticket issued to Bob for Alice • Essential Reason • Ticket to Bob stays valid even if Alice changes her key
23. [**Expanded Needham-Schroeder**](https://image2.slideserve.com/5332869/expanded-needham-schroeder-l.jpg) The additional two messages assure Bob that the initiator has talked to KDC since Bob generates NB
24. [**Kerberos V4**](https://image2.slideserve.com/5332869/kerberos-v4-l.jpg) • Based on Needham-Schroeder, but with timestamps • Save exchange of nonce

**Unit-II: Security Handshake Pitfalls**

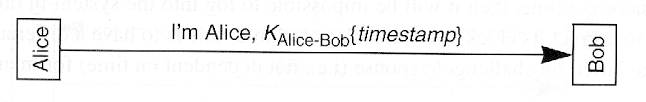
* You should not design their own crypto algorithms (Kerckhoff's Principle)
* But you may need to design a security protocol
* Protocols flaws can be very subtle
* **Topics to be covered**
  + Challenge/response (login only)
  + Mutual authentication
  + Integrity/encryption of data
  + Mediated authentication
  + Nonce types
  + Picking random numbers
  + Performance considerations
  + Authentication protocol checklist

.

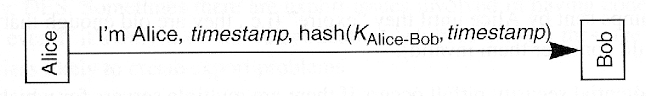
* Challenge/response
  + A bad idea
    - Alice sends name and password in clear (across network) to Bob
    - Bob verifies name and password and communication proceeds
  + Better idea using shared secret  
      
    
    - Implications
      * Authentication is not mutual
      * How to encrypt subsequent conversation?
      * If key derived from a password, offline password guessing is possible
      * Bob knows KAlice-Bob so if Bob's database is compromised, attacker can impersonate Alice

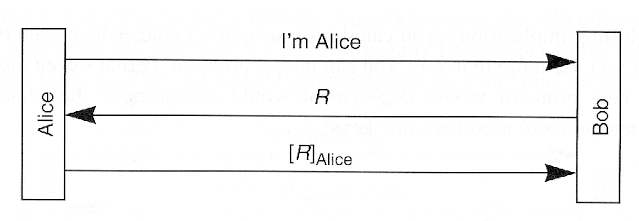


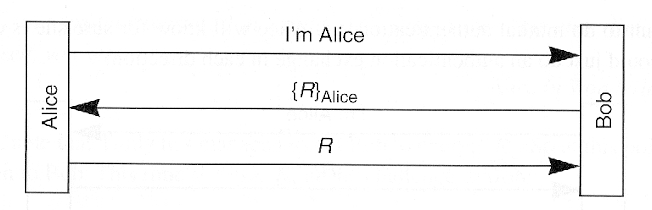
* + - Implications
      * Requires reversible cryptography (hash will not work)
      * If R is known and key derived from password, dictionary attack is possible by simply claiming to be Alice
      * If R has a limited lifetime, Alice can authenticate Bob (mutual authentication)

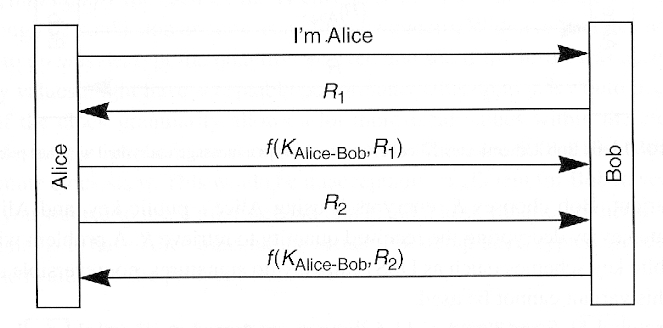


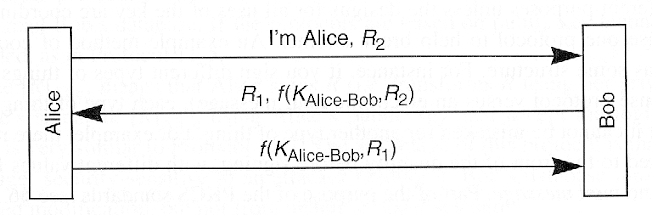
* + - Implications
      * Easy to modify "bad idea" to this form, since no additional messages
      * More efficient
      * Bob does not need to maintain state
      * Eavesdropper can impersonate Alice (within acceptable clock skew); might also be possible to impersonate Alice to another server
      * If Bob sets his clock back, intercepted authentication messages can be replayed
      * Setting time (and agreeing on time) is a security issue



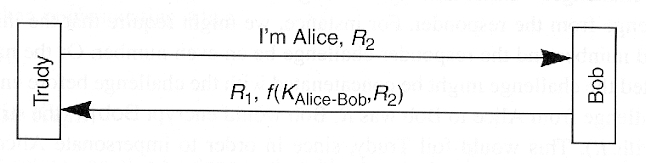
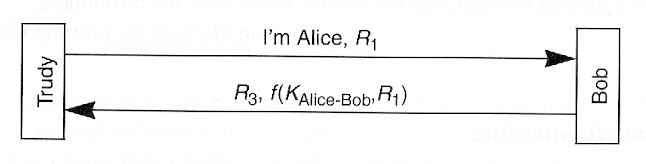
* + - Implications
      * Same as above, but using a hash
      * Why transmit timestamp in the clear?
  + Better idea using public key crypto  
    (Notation: [R]Alice means sign with private key and {R}Alice means encrypt with public key.)  
      
    
    - Implications
      * Compromise of Bob's database will not allow attacker to impersonate Alice
      * Attacker may be able to trick Alice into signing anything

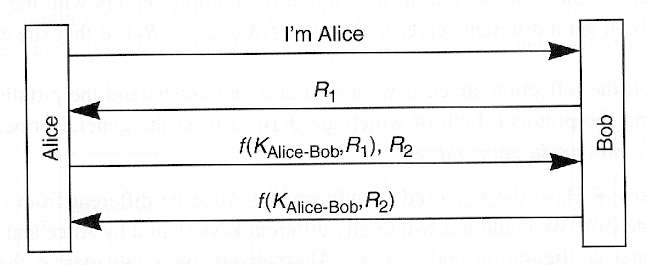
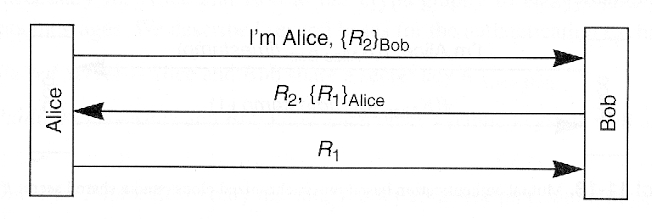
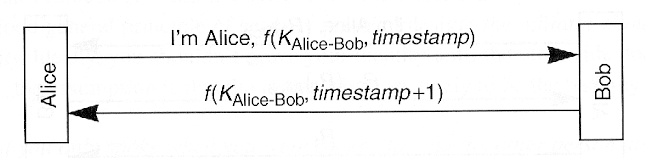
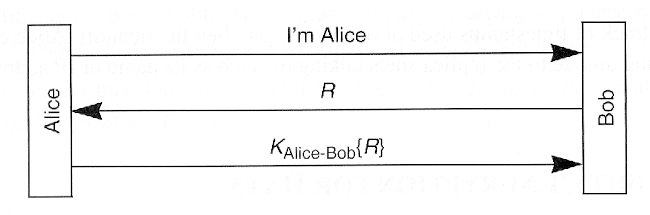


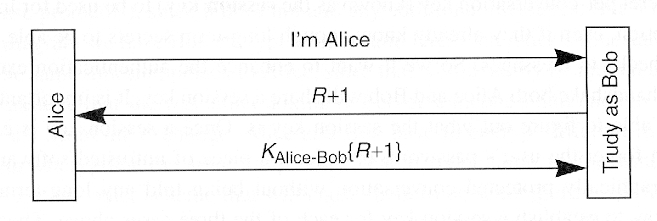
* + - Implications
      * Compromise of Bob's database will not allow attacker to impersonate Alice
      * Attacker may be able to trick Alice into decrypting anything
* Mutual authentication
  + Reflection attack  
      
    
    - Implications
      * Authenticated exchange in each direction
      * Inefficient?

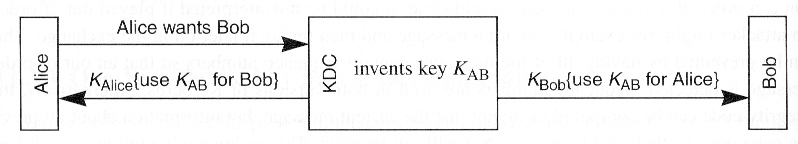
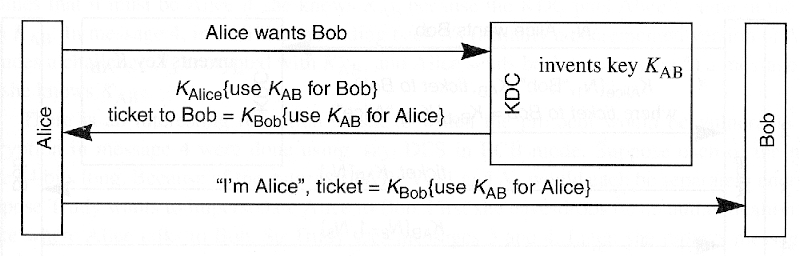
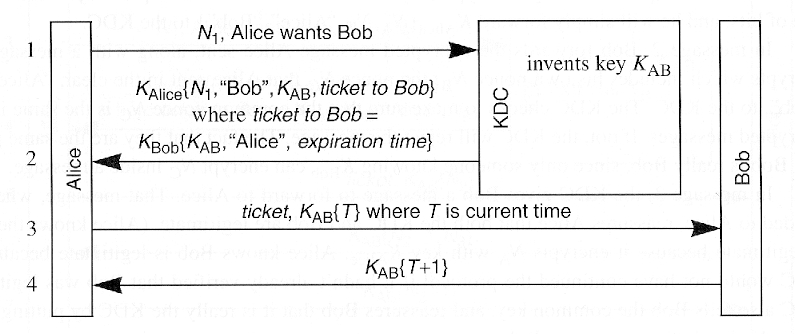
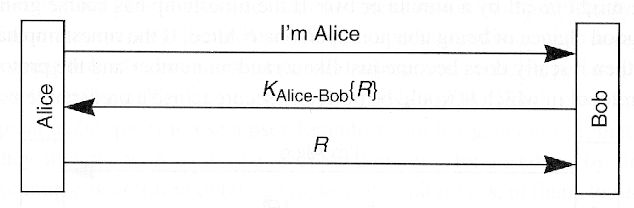
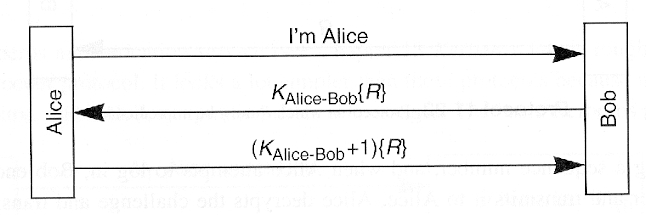


* + - Implications
      * More efficient
      * Easy to get chosen plaintext
      * Subject to a reflection attack

* + - Attack
      * Trudy opens 1st session to Bob
      * Trudy opens 2nd session to Bob in order to get information needed to complete 1st session
    - Solution?
      * Alice and Bob should not do exactly the same thing
      * Have Bob encrypt with KAlice-Bob and Alice encrypt with KAlice-Bob+1 or
      * Initiator sends odd R, responder sends even R, etc.
  + Password guessing (chosen plaintext)  
      
    
    - Implications
      * One "extra" message and Alice cannot obtain chosen plaintext
  + Public keys  
      
    
    - Implications
      * How to obtain public keys?
      * How can workstation obtain private key from password? (Easy with symmetric key crypto, not so easy with public key crypto.)
      * Identity-based encryption is an active research area
  + Timestamps  
      
    
    - Implications
      * Only 2 messages
      * Alice and Bob must encrypt different things
      * Everyone must agree on the time
      * Time is now security-critical
* How to establish a session key?
  + Shared secret  
      
    
    - Implications
      * Why not use KAlice-Bob as session key?
      * Why not use KAlice-Bob+1 as session key?
      * Why not use KAlice-Bob+1 to encrypt R to obtain session key?
      * Why not use KAlice-Bob to encrypt R+1 to obtain session key?



* + Two-way public key authentication
    - Alice sends {K}Bob to Bob  
        
      **Issues**: Trudy can hijack the conversation
    - Alice sends [{K}Bob]Alice to Bob  
        
      **Issues**: If Trudy records conversation and later overruns Bob, she can recover K
    - Alice sends {K1}Bob and Bob sends {K2}Alice to Bob. The session key is K1 ⊕ K2  
        
      **Issues**: Trudy can cause confusion, but cannot recover K
    - Alice and Bob do a Diffie-Hellman key exchange, and sign the quantities: Alice sends [ga mod p]Alice to Bob and Bob sends [gb mod p]Bob to Alice  
        
      **Issues**: Even if Trudy overruns both Alice and Bob, she cannot recover K
  + One-way public key authentication --- similar to two-way public key authentication
  + Privacy and integrity --- key rollover
* Mediated Authentication with key distribution center (KDC)
  + In priciple  
      
    
    - Implications
      * KDC does not authenticate Alice but only Alice can decrypt KAB sent to "Alice"
      * Message from Alice to Bob could arrive before Bob gets his key
  + In practice  
      
    
    - Implications
      * Alice and Bob must still mutually authenticate
  + Kerberos  
      
    
    - Implications
      * Everyone must agree on time
* Nonce types
  + Unpredictable nonce required  
      
    
  + Predictable nonce is OK  
      
    
* Picking random numbers
  + Different application require different types of "random" numbers
  + For pseudo-random number generator, seed value is critical
  + Interesting mistakes listed in book
* Performance considerations
  + Number of crypto operations (private key, public key, symmetric key, hash)
  + Number of messages transmitted
  + Cache previous state? (more speed, less security)
* Authentication protocol checklist --- play role of Trudy
  + Eavesdrop
    - data?
    - replay?
    - off-line password (or other secret) guessing?
  + Pretend to be Alice
    - convince Bob?
    - off-line password guessing?
    - replay (e.g., impersonate Alice, impersonate Bob)?
    - get Bob to decrypt or sign something?
  + Pretend to be Bob (accept connection from Alice)
    - convince Alice?
    - off-line password guessing?
    - replay?
    - get Alice to decrypt or sign something?
  + Read Alice's database (can impersonate Alice)
    - impersonate Bob to Alice?
    - decrypt old conversations?
  + Read Bob's database (can impersonate Bob)
    - impersonate Alice?
    - decrypt old conversations?
  + Examine and/or modify data sent between Alice and Bob
    - off-line password guessing?
    - read data?
    - hijack conversation?
    - modify messages without detection?
  + Any combination of the above
    - For example, even after overrunning both Alice and Bob's databases, it should not be possible to decrypt recorded conversations