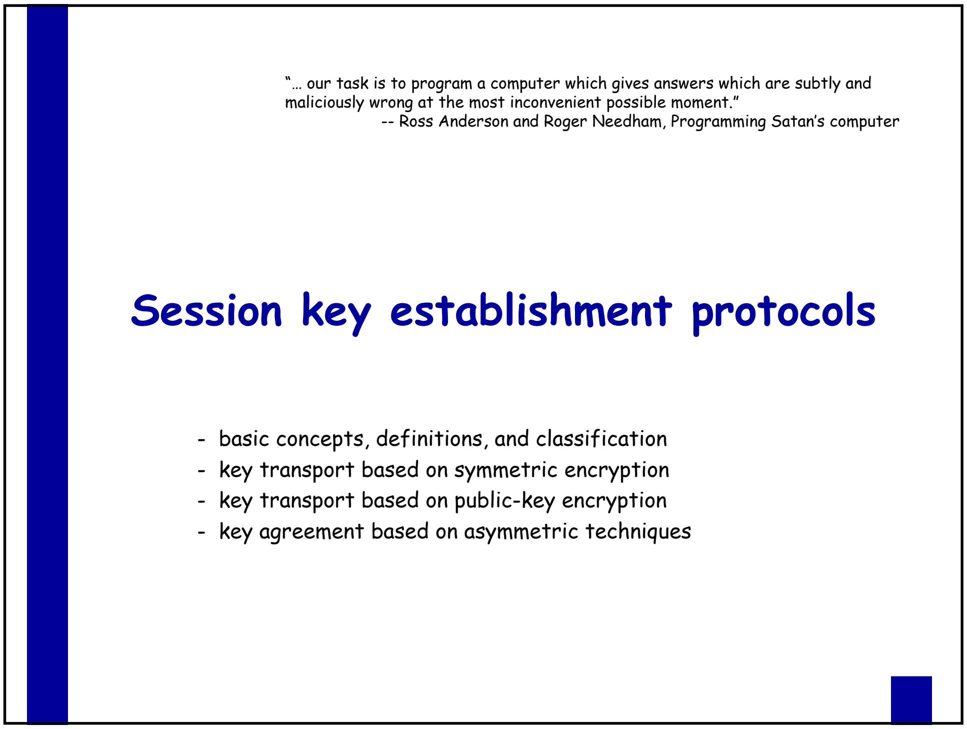
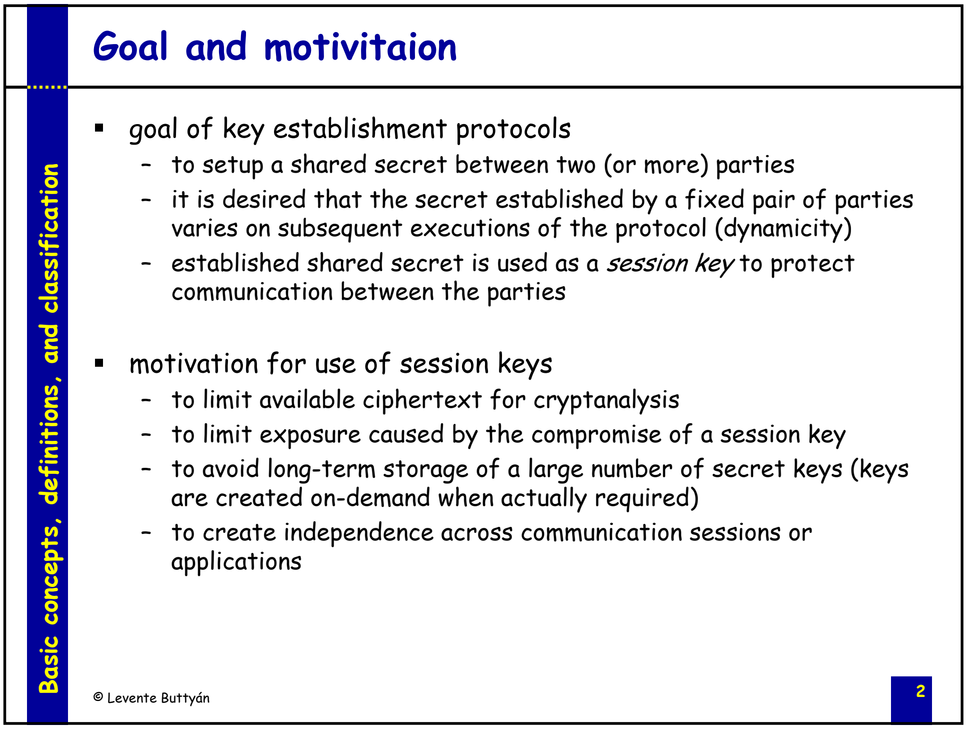


"… our task is to program a computer which gives answers which are subtly and maliciously wrong at the most inconvenient possible moment."
-- Ross Anderson and Roger Needham, Programming Satan's computer

Session key establishment protocols

- basic concepts, definitions, and classification
- key transport based on symmetric encryption
- key transport based on public-key encryption
- key agreement based on asymmetric techniques



Goal and motivitaion

Basic concepts, definitions, and classification

- goal of key establishment protocols
  - to setup a shared secret between two (or more) parties
  - it is desired that the secret established by a fixed pair of parties varies on subsequent executions of the protocol (dynamicity)
  - established shared secret is used as a *session key* to protect communication between the parties

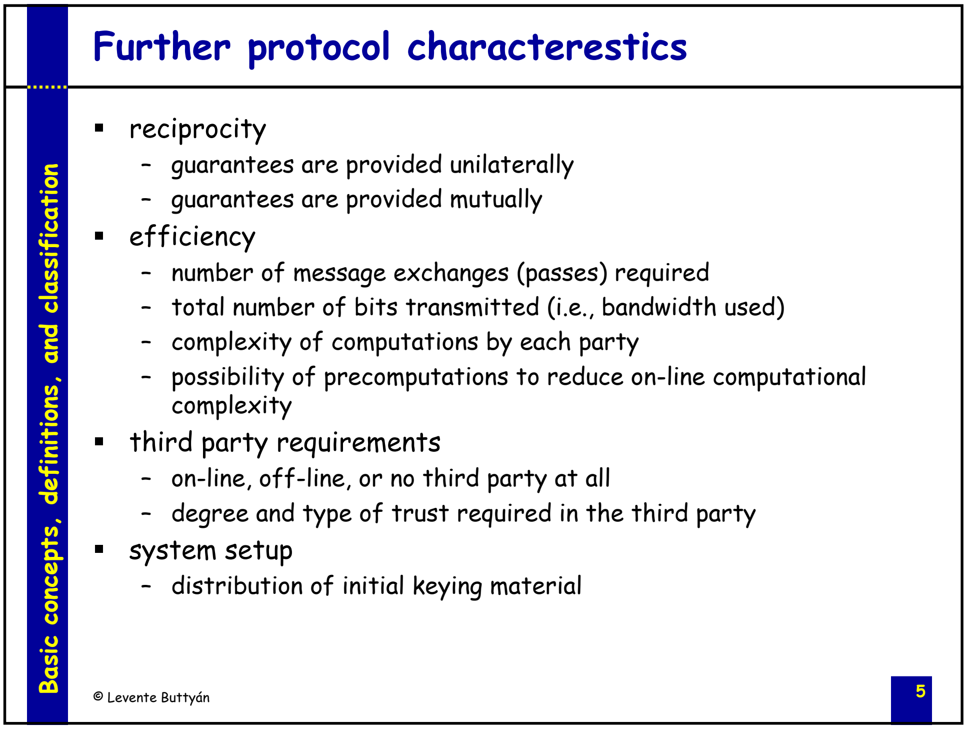
- motivation for use of session keys
  - to limit available ciphertext for cryptanalysis
  - to limit exposure caused by the compromise of a session key
  - to avoid long-term storage of a large number of secret keys (keys are created on-demand when actually required)
  - to create independence across communication sessions or applications

Basic classification

- key transport protocols
  - one party creates or otherwise obtains a secret value, and securely transfers it to the other party
- key agreement protocols
  - a shared secret is derived by the parties as a function of information contributed by each, such that no party can predetermine the resulting value

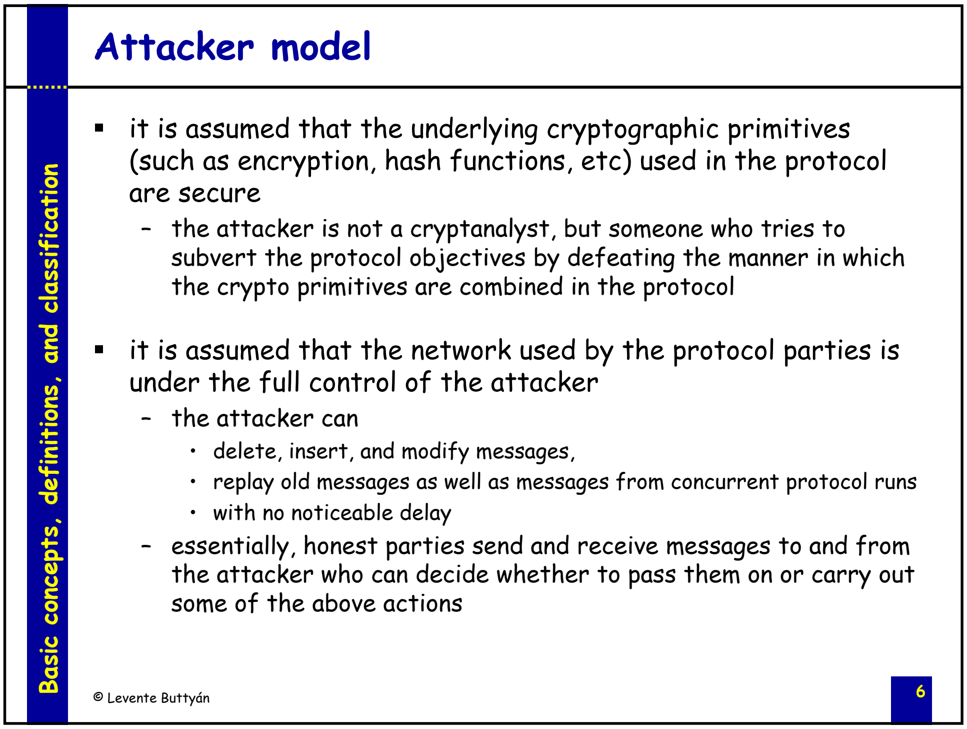
Further protocol characterestics

- provided guarantees
  - entity authentication
    - one party is assured about the identity of a second party with which it is communicating
  - implicit key authentication
    - one party is assured that no other party aside from a specifically identified second party (and possibly some trusted third parties) may gain access to the established session key
  - key confirmation
    - one party is assured that a second (possibly unidentified) party actually possesses the session key
    - possession of a key can be demonstrated by
      - producing a one-way hash value of the key or
      - encryption of known data with the key
  - explicit key authentication
    - implicit key authentication + key confirmation
  - key freshness
    - one party is assured that the key is new (never used before)
  - key control
    - in some protocols, one of the parties control the value of the key (key transport), in others, no party can control (predict) its value (key agreement)



Further protocol characterestics

- reciprocity
  - guarantees are provided unilaterally
  - guarantees are provided mutually
- efficiency
  - number of message exchanges (passes) required
  - total number of bits transmitted (i.e., bandwidth used)
  - complexity of computations by each party
  - possibility of precomputations to reduce on-line computational complexity
- third party requirements
  - on-line, off-line, or no third party at all
  - degree and type of trust required in the third party
- system setup
  - distribution of initial keying material



Attacker model

- it is assumed that the underlying cryptographic primitives (such as encryption, hash functions, etc) used in the protocol are secure
  - the attacker is not a cryptanalyst, but someone who tries to subvert the protocol objectives by defeating the manner in which the crypto primitives are combined in the protocol
- it is assumed that the network used by the protocol parties is under the full control of the attacker
  - the attacker can
    - delete, insert, and modify messages,
    - replay old messages as well as messages from concurrent protocol runs
    - with no noticeable delay
  - essentially, honest parties send and receive messages to and from the attacker who can decide whether to pass them on or carry out some of the above actions

Attacker model

- sometimes it is assumed that the attacker has access to additional information beyond what is generally available
  - compromise of past session keys
    - obviously, the corresponding session is compromised
    - can this be used by the attacker to compromise future sessions?
      - if not, then the protocol resists against *known-key attacks*
  - compromise of long-term secrets (symmetric keys or private keys)
    - obviously, owners of compromised keys can be impersonated in future sessions
    - can this be used by the attacker to compromise past session keys?
      - if not, then the protocol provides *break-backward protection* or *perfect forward secrecy*

Attacker model

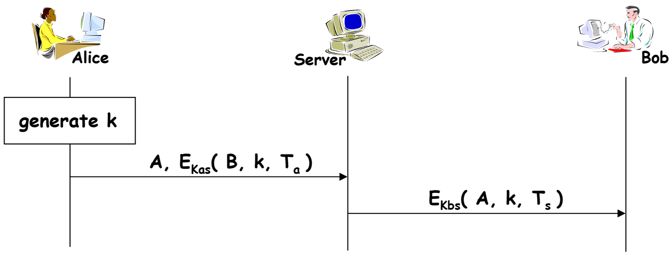
- objectives of the attacker
  - deduce a long-term key
    - the attacker obtains the long-term key of a honest party A
  - deduce a session key
    - the attacker obtains a session key shared between two honest parties A and B
  - masquerade as a honest party B to A
    - A believes that he established a session key with B, but in fact he established the key with the attacker
  - deceive a honest party A regarding the identity of the other party
    - A believes that he established a session key with B, but in fact he established the key with another honest party C
    - the attacker does not know the established session key

- attack types
  - passive eavesdropping
  - man-in-the-middle: participating covertly in the protocol run between two parties and modifying messages unnoticeably
  - interleaving: initiating one or more protocol executions (possibly simultaneously) and *interleave* messages from different executions

Key transport based on symmetric encryption

The Wide-Mouth-Frog protocol

summary: a simple key transport protocol that uses a trusted third party
Alice generates the session key and sends it to Bob via the trusted third party



characteristics: key control for Alice
implicit key authentication for Alice
explicit key authentication for Bob
key freshness for Bob with timestamps (flawed)
unilateral entity authentication of Alice
on-line third party (Server) trusted for secure relaying of keys and
verification of freshness,
in addition A is trusted for generating good keys
initial long-term keys between the parties and the server are required

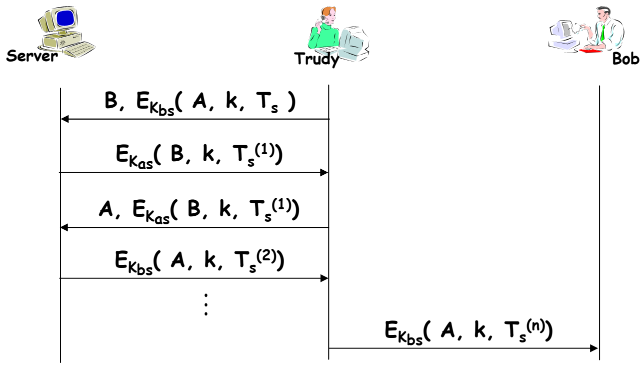
© Levente Buttyán

9

Key transport based on symmetric encryption

A flaw in the Wide-Mouth-Frog protocol

summary: after observing one run of the protocol, Trudy can continuously use the Server
as an oracle until she wants to bring about re-authentication between Alice and Bob

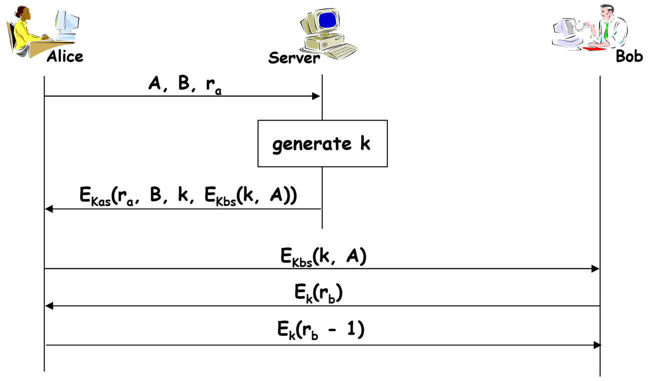

Server
Trudy
Bob
$B,\ E_{Kbs}(\ A,\ k,\ T_s\ )$
$E_{Kas}(\ B,\ k,\ T_s^{(1)})$
$A,\ E_{Kas}(\ B,\ k,\ T_s^{(1)})$
$E_{Kbs}(\ A,\ k,\ T_s^{(2)})$
⋮
$E_{Kbs}(\ A,\ k,\ T_s^{(n)})$

© Levente Buttyán

10

The Needham-Schroeder protocol

summary: Alice requests a session key from the Server; the Server generates the key and sends it to Alice and to Bob via Alice; Alice and Bob performs entity authentication and key confirmation

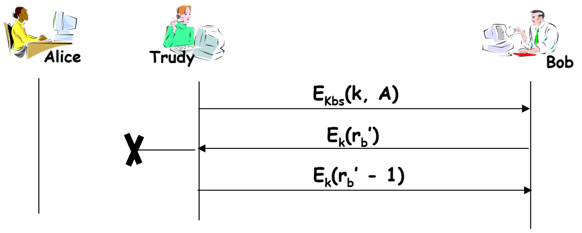


characteristics: mutual entity authentication, mutual explicit key authentication, key freshness with fresh random numbers (flawed), on-line third party trusted for generation of session keys, initial long-term keys between the parties and the server are required

A flaw in the Needham-Schroeder protocol

assumption: Trudy recorded a successful run of the protocol and compromised the session key k; thus, she knows k and $E_{Kbs}(k, A)$

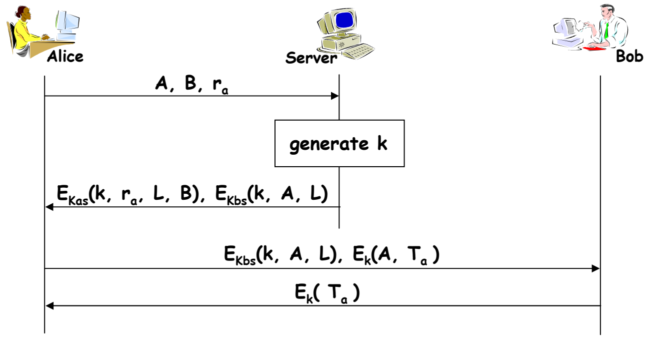
summary: Trudy masquerades as Alice to Bob and makes Bob accept the old and compromised session key k



Key transport based on symmetric encryption

The Kerberos protocol

summary: essentially the Needham-Schroeder protocol with timestamps
the protocol is optimized with respect to the original Needham-Schroeder protocol (fewer messages and no double encryption in the second message)

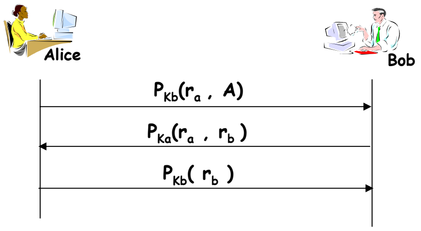

Alice
Server
Bob
A, B, r_a
generate k
$E_{Kas}(k, r_a, L, B), E_{Kbs}(k, A, L)$
$E_{Kbs}(k, A, L), E_k(A, T_a)$
$E_k(T_a)$

characteristics: mutual entity authentication, mutual explicit key authentication, key freshness with timestamp and random number, clock synchronization is required, on-line third party trusted for generation of session keys, initial long-term keys between the parties and the server are required

Key transport based on public-key encryption

The Needham-Schroeder public-key protocol

summary: originally a challenge-response type mutual authentication protocol based on public-key encryption only (no signatures); however, since the random numbers exchanged never appear in clear, it was suggested to derive a session key from them



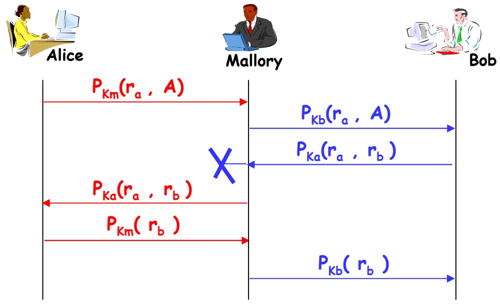
key derivation: both party computes $k = f(r_a , r_b)$

characteristics: mutual entity authentication, mutual implicit key authentication (flawed), no key confirmation, key freshness with random numbers, no party can control the key, off-line third party for issuing public key certificates may be required, initial exchange of public keys between the parties may be required

Lowe's attack

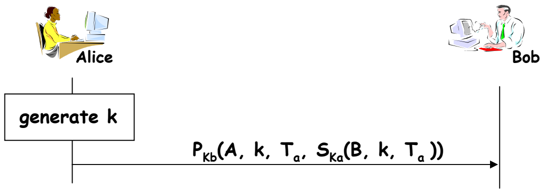
assumption: Mallory is a malicious user, his public key is K_m

summary: when Alice starts the protocol with Mallory, he can masquerade as Alice to Bob;
Mallory uses Alice as an oracle to decrypt a message received from Bob;
if the protocol is used for key establishment, then Bob falsely believes that he shares a secret key with Alice, but indeed he shares it with Mallory



Encrypting signed keys

summary: Alice generates a session key, sings it, then encrypts it with Bob's public key, and sends it to Bob

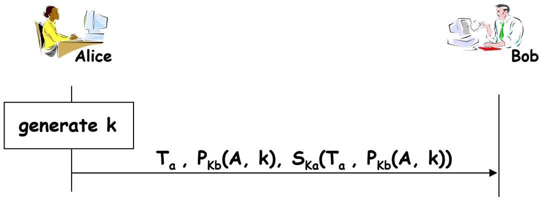


characteristics: unilateral entity authentication (of Alice), mutual implicit key authentication, key confirmation for Bob, key freshness with timestamp, clock synchronization needed, off-line third party for issuing public key certificates may be required, initial exchange of public keys between the parties may be required, Alice is trusted to generate keys, non-repudiation guarantee for Bob

notes: the ID of Bob in the signature prevents Bob from sending the signed key on to another party and impersonating Alice;
the ID of Alice in the encrypted message is a hint for Bob that helps him to choose the right key for verification of the signature.

Signing encrypted keys

summary: Alice generates a session key, encrypts it with Bob's public key, then sings it, and sends it to Bob



characteristics: unilateral entity authentication (of Alice), mutual implicit key authentication, no key confirmation, key freshness with timestamp, clock synchronization, off-line third party for issuing public key certificates may be required, initial exchange of public keys between the parties may be required, Alice is trusted to generate keys, non-repudiation guarantee for Bob

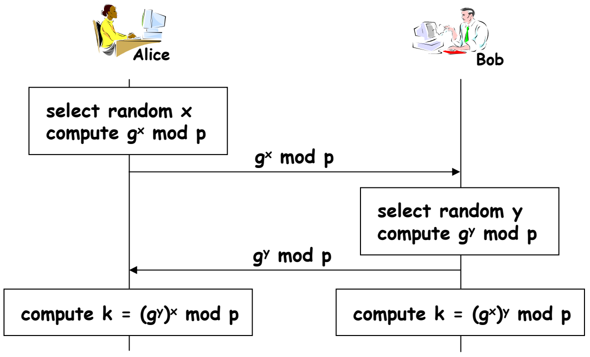
notes: the ID of Alice in the encrypted message is a hint for Bob that helps him to choose the right key for verification of the signature

an advantage of this protocol over the "encrypting signed keys" protocol is that here less data is encrypted (almost surely fits in the block size)

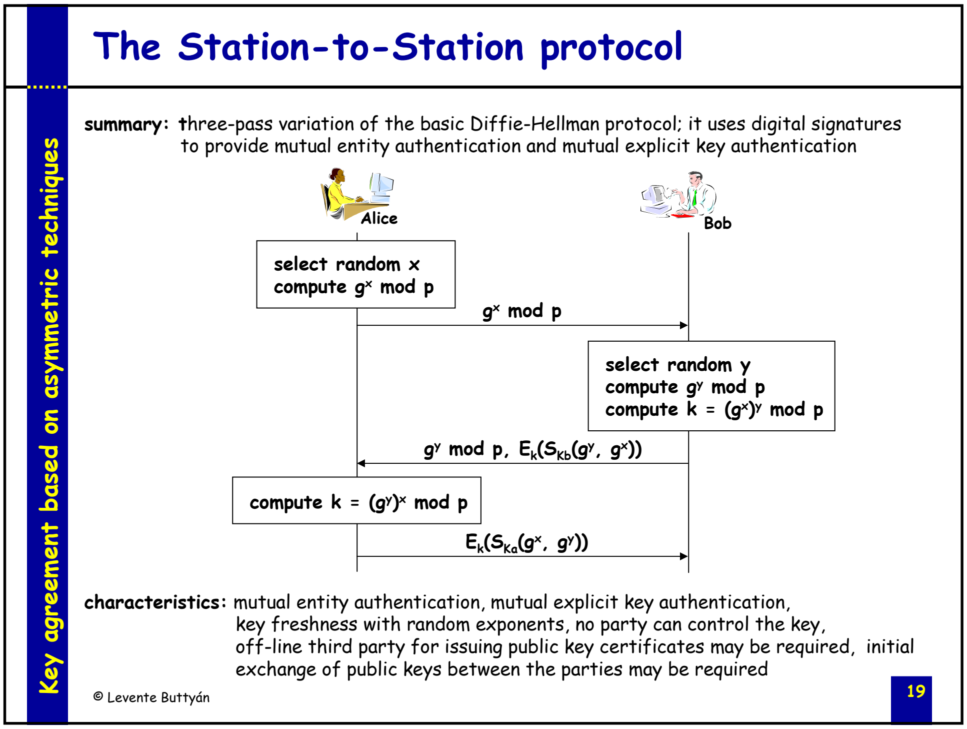
The Diffie-Hellman protocol

summary: a key agreement protocol based on one-way functions; in particular, security of the protocol is based on the hardness of the discrete logarithm problem and that of the Diffie-Hellman problem

assumptions: p is a large prime, g is a generator of Z_p^*, both are publicly known system parameters



characteristics: NO AUTHENTICATION, key freshness with randomly selected exponents, no party can control the key, no need for a trusted third party



Key agreement based on asymmetric techniques

The Station-to-Station protocol

summary: three-pass variation of the basic Diffie-Hellman protocol; it uses digital signatures to provide mutual entity authentication and mutual explicit key authentication

Alice
Bob
select random x
compute g^x mod p
g^x mod p
select random y
compute g^y mod p
compute k = $(g^x)^y$ mod p
g^y mod p, $E_k(S_{Kb}(g^y, g^x))$
compute k = $(g^y)^x$ mod p
$E_k(S_{Ka}(g^x, g^y))$

characteristics: mutual entity authentication, mutual explicit key authentication, key freshness with random exponents, no party can control the key, off-line third party for issuing public key certificates may be required, initial exchange of public keys between the parties may be required