Security Proofs for an Efficient Password-Based Key Exchange

Emmanuel Bresson
CELAR – France

Olivier Chevassut
LBNL – DOE - USA

David Pointcheval
CNRS-ENS – France

ACM CCS
Washington DC
October 29th 2003

Summary

- Authenticated Key Exchange
  - Security Model
  - Example
- Password-Based Authentication
  - EKE and AuthA
  - Security Results
- Conclusion

Summary

- Authenticated Key Exchange
  - Security Model
  - Example
- Password-Based Authentication
  - EKE and AuthA
  - Security Results
- Conclusion

Authenticated Key Exchange

Two parties (Alice and Bob) agree on a ***common*** secret key sk, in order to establish a secret channel

- Intuitive goal: ***implicit authentication***
  - only the intended partners can compute the session key
- Formally: ***semantic security***
  - the session key sk is indistinguishable from a random string r, to anybody else

Further Properties

- ***Mutual authentication***
  - They are both sure to **actually** share the secret with the people they think they do
- ***Forward-secrecy***
  - Even if a long-term secret data is corrupted, previously shared secrets are **still** semantically secure

Semantic Security

- For breaking the semantic security, the adversary asks one **test**-query which is answered, according to a random bit b, by
  - the actual secret data sk (if $b=0$)
  - a random string r (if $b=1$)

$\Rightarrow$ the adversary has to guess this bit b

The Leakage of Information

- The protocol is run over a public network,
  then the transcripts are public:
  - an **execute**-query provides such a transcript to the adversary
- The secret data *sk* may be misused (with a weak encryption scheme, ...):
  - the **reveal**-query is answered by this secret data *sk*

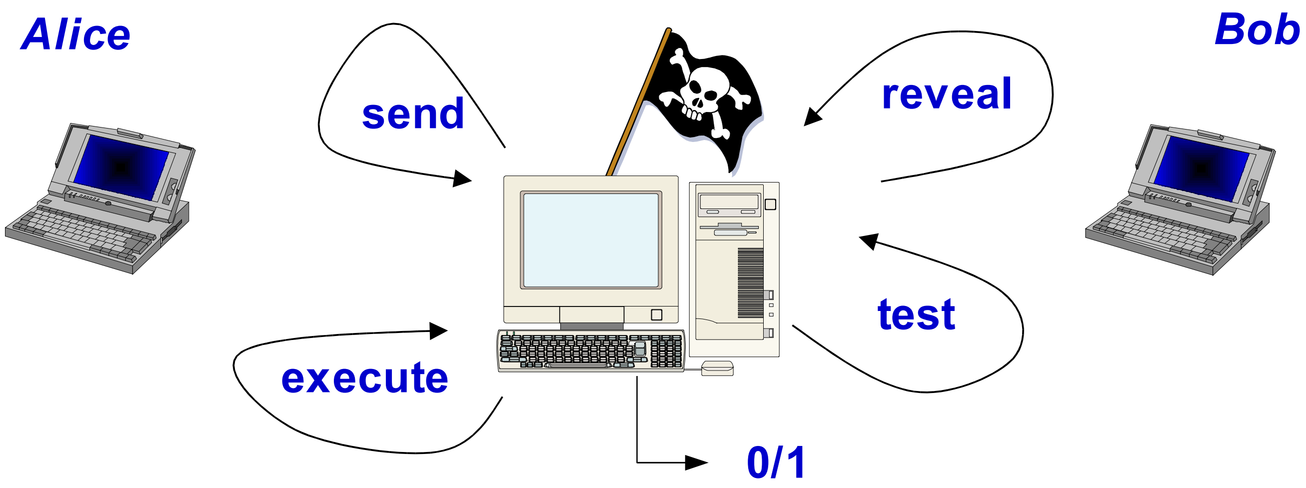
Passive/Active Adversaries

- ***Passive adversary***: history built using
  - the **execute**-queries $\rightarrow$transcripts
  - the **reveal**-queries $\rightarrow$session keys
- ***Active adversary***: entire control of the network
  - the **send**-queries

    active, adaptive adversary on concurrent executions
    - to send message to Alice or Bob
      (in place of Bob or Alice respectively)
    - to intercept, forward and/or modify messages

Security Model

As many **execute**, **send** and **reveal** queries as the adversary wants



But one **test**-query, with b to be guessed...

Diffie-Hellman Key Exchange

The most classical key exchange scheme has been proposed by Diffie and Hellman:

G $= \langle g \rangle$, cyclic group of prime order q

- Alice chooses a random $x \in \mathbf{Z}_q$, computes and sends $X = g^x$
- Bob chooses a random $y \in \mathbf{Z}_q$, computes and sends $Y = g^y$
- They can both compute the value

$$K = Y^x = X^y$$

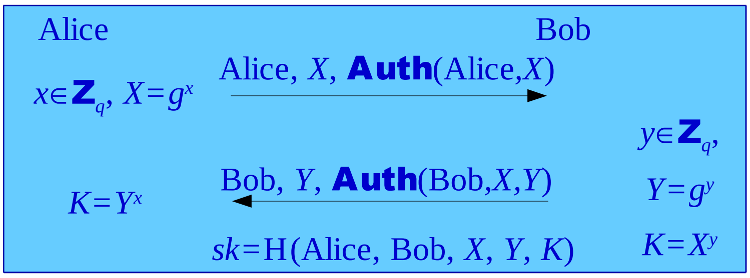
Properties

- Without any authentication, no security is possible: man-in-the-middle attack

⇒ some authentication is required

- If flows are **authenticated** (MAC or Signature), it provides the semantic security of the session key under the **DDH Problem**
- If one derives the session key as $sk = \mathrm{H}(K)$, in the random oracle model, semantic security is relative to the **CDH Problem**

Replay Attack



No explicit authentication ⇒ replay attacks

- The adversary intercepts "Alice, X, **Auth**(Alice,X)"
- He can initiate a new session with it

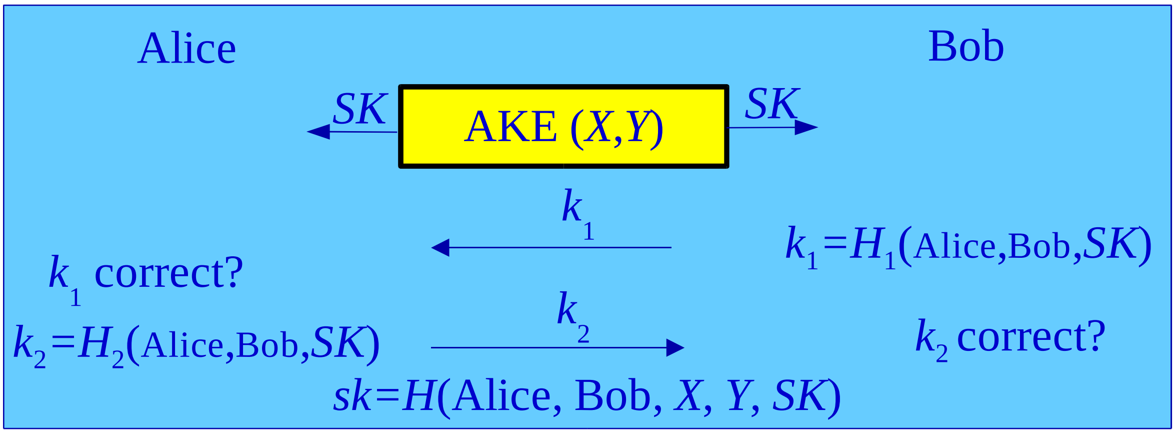
Bob believes it comes from Alice

- Bob accepts the key, but does not share it with Alice ⇒ **no mutual authentication**
- The adversary does not know the key either ⇒ **still semantic security**

Mutual Authentication

Adding key confirmation rounds:
mutual authentication

[Bellare-Pointcheval-Rogaway Eurocrypt '00]



Summary

- Authenticated Key Exchange
  - Security Model
  - Example
- Password-Based Authentication
  - EKE and AuthA
  - Security Results
- Conclusion

Authentication

- **Asymmetric**: (sk_A,pk_A) and possibly (sk_B,pk_B)
  - they authentify to each other using the knowledge of the private key associated to the certified public key
- **Symmetric**: common (long – high-entropy) secret
  - they use the long term secret to derive a secure and authenticated ephemeral key sk
- **Password**: common (short - low-entropy) secret

  let us assume a **20-bit** password

Password-based Authentication

Password (short – low-entropy secret – say 20 bits)

- exhaustive search is possible
- basic attack: **on-line exhaustive search**
  - the adversary guesses a password
  - tries to play the protocol with this guess
  - failure ⇒ it erases the password from the list
  - and restarts…

after 2^{20} attempts, the adversary wins

Dictionary Attack

- The on-line exhaustive search
  - cannot be prevented
  - can be made less serious (delay, limitations, …)

We want it to be the best attack…

- The **off-line exhaustive search**
  - a few passive or active attacks
  - failure $\Rightarrow$ erasure of MANY passwords from the list

this is called dictionary attack

Security

One wants to prevent dictionary attacks:

- a passive trial (**execute** + **reveal**)
  - does not reveal any information about the password
- an active trial (**send**)
  - allows to erase **at most one** password from the list of possible passwords

    (or maybe 2 or 3 for technical reasons in the proof)

Example: EKE

The most famous scheme EKE:
Encrypted Key Exchange

Flows are encrypted with the password.

Must be done carefully: **no redundancy**

- From X' ,
  for any password π
  - decrypt X'
  - check whether
    it begins with "Alice"

bad one

Alice	Password π	Bob
$x \in \mathbf{Z}_q$, $X = g^x$	$X' = \mathbf{E}_\pi(\text{Alice}, X)$ →	
		$X \leftarrow \mathbf{D}_\pi(X')$, $y \in \mathbf{Z}_q$, $Y = g^y$, $K = X^y$
$Y \leftarrow \mathbf{D}_\pi(Y')$, $K = Y^x$	← $Y' = \mathbf{E}_\pi(\text{Bob}, Y)$	
	$sk = H(\text{Alice}, \text{Bob}, X, Y, K)$	

EKE - AuthA

Alice	Password π	Bob
$x \in \mathbf{Z}_q$, $X = g^x$	Alice, $X' = \mathbf{E}_\pi(X)$ →	$X = \mathbf{D}_\pi(X')$
$Y = \mathbf{D}_\pi(Y')$, $K = Y^x$	← Bob, $Y' = \mathbf{E}_\pi(Y)$	$y \in \mathbf{Z}_q$, $Y = g^y$, $K = X^y$
	$sk = H(\text{Alice}, \text{Bob}, X, Y, K)$	

EKE

Bellovin-Merritt 1992

Two-flow Encrypted Key Exchange

AuthA

Bellare-Rogaway 2000

OEKE = One-flow Encrypted Key Exchange

Alice	Password π	Bob
$x \in \mathbf{Z}_q$, $X = g^x$	Alice, X →	
$Y = \mathbf{D}_\pi(Y')$, $K = Y^x$	← Bob, $Y' = \mathbf{E}_\pi(Y)$	$y \in \mathbf{Z}_q$, $Y = g^y$, $K = X^y$
$k_1 = H_1(\text{Alice}, \text{Bob}, K)$	k_1 →	k_1 correct ?
	$sk = H(\text{Alice}, \text{Bob}, X, Y, K)$	

- **EKE**: security claimed, but never fully proved
- **OEKE** and **AuthA**: security = open problem

OEKE: New Security Result

- Assumptions
  - the ideal-cipher model – for (**E**,**D**)
  - the random-oracle model – for H and H_1
- Notations
  - q_s, the number of **send**-queries (active and adaptive)
  - q_h, the number of **hash**-queries to H and H_1
  - N, the number of passwords

Semantic security of OEKE :

advantage $\geq 3\, q_s/N + \varepsilon$,

$\Rightarrow$**CDH problem** : probability $\geq \varepsilon/8q_h$

(within almost the same time)

Further Security Results

- Forward-secrecy is considered:
  - provably secure but with a worse reduction
- Verifier-based (included in some version of **AuthA**):
  - Alice knows a password π,
  - Bob just knows a verifier of the password = $f(\pi)$,
    - it is enough to check whether Alice really knows π
    - it does not immediately lead to π (off-line exhaustive search)

Summary

- Authenticated Key Exchange
  - Security Model
  - Example
- Password-Based Authentication
  - EKE and AuthA
  - Security Results
- Conclusion

Conclusion

OEKE and other **AuthA** variants are

- provably secure
  - semantic security
  - unilateral or mutual authentication
- more efficient than EKE
  - only one flow is encrypted
- more suitable for client-server schemes
  - the server can first send a generic flow not encrypted, and thus independent of the client