

Provably Secure Password-Based Authentication in TLS

Michel Abdalla, École normale supérieure

Emmanuel Bresson, CELAR Technology Center

Olivier Chevassut, Lawrence Berkeley National Laboratory

Bodo Möller, University of Calgary

David Pointcheval, École normale supérieure

Provably Secure Password-Based Authentication in TLS

Michel Abdalla, École normale supérieure

Emmanuel Bresson, CELAR Technology Center

Olivier Chevassut, Lawrence Berkeley National Laboratory

Bodo Möller, University of Calgary Ruhr-Universität Bochum

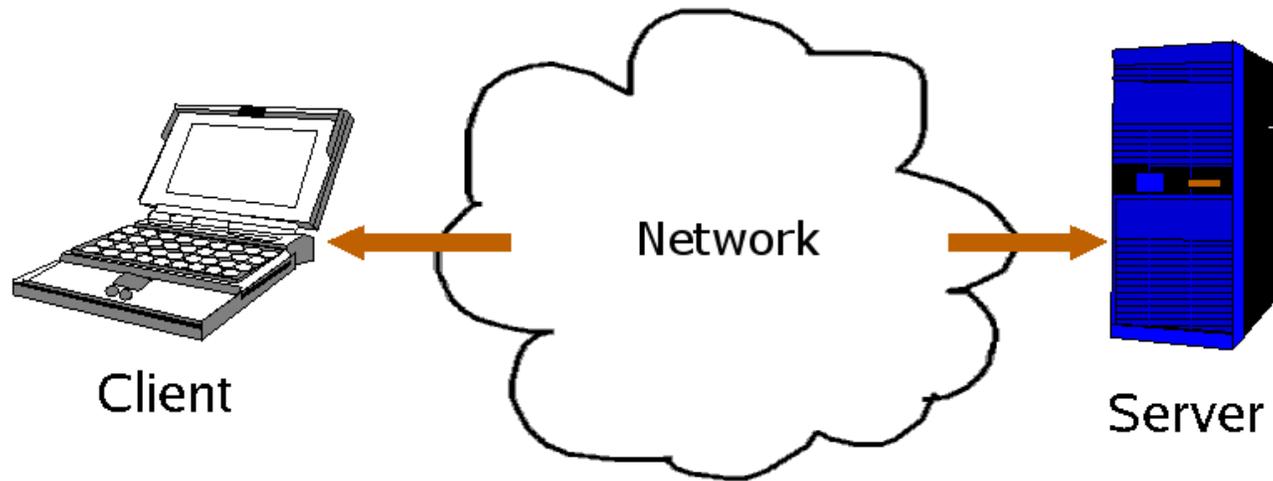
David Pointcheval, École normale supérieure

Overview

- The **scenario**
- Our **protocol**: SOKE-TLS
(**S**imple **O**pen **K**ey **E**xchange for **T**ransport **L**ayer **S**ecurity)
- **Comparison** with other protocols

The scenario

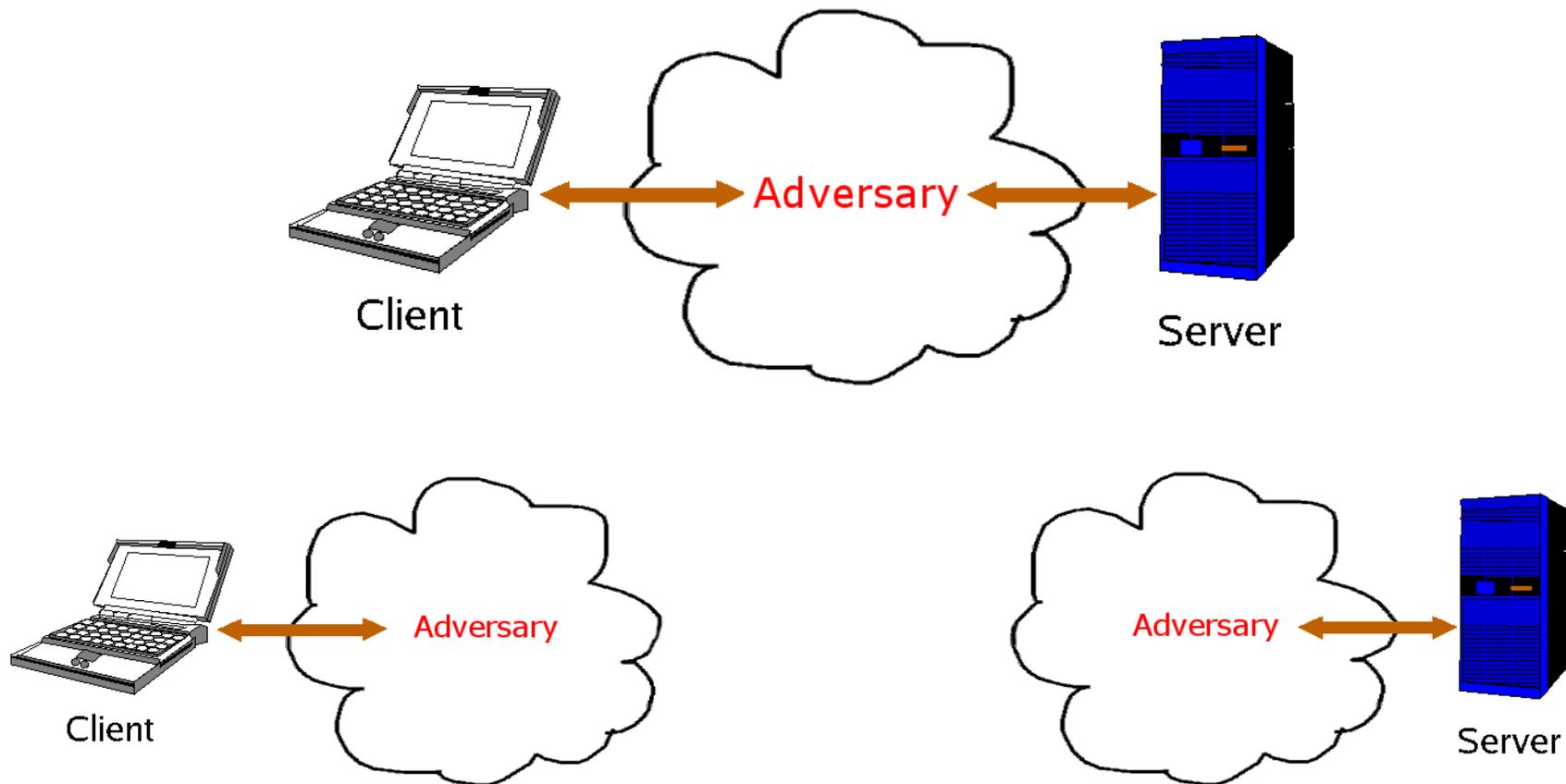
Client wants to talk to server.



Can't trust network . . .

The scenario (cont.)

... the world may look like one of this:



The scenario (cont.)

So, use cryptography: **symmetric** crypto for **authenticated encryption**, and some **key exchange** to get session key for symmetric crypto.

In this paper, we focus on the key exchange.

Want to build on **TLS** (Transport Layer Security) protocol: widely deployed, various open-source implementations available.

Key exchange is **easy** if we can use **long keys**: set up joint secret, or use public-key scheme (such as RSA, DH).

However, keeping a long key is not always always practical.

Want to use a **password** instead, a (short) secret that can be memorized.

The scenario (cont.)

Straightforward **password-based authentication** in TLS:

Client authenticates server, then sends password over secure link.



Bank of America Higher Standards

PERSONAL SMA

Online Banking Sign In

[View demo](#) | [Learn more](#) | [Enroll](#)

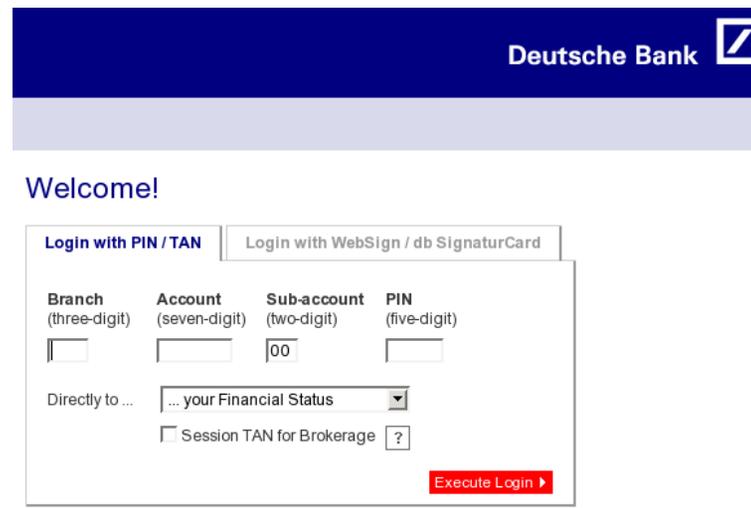
Enter Online ID:

Save this online ID

Enter Passcode:

Account in:

Sign In



Deutsche Bank

Welcome!

Login with PIN / TAN | Login with WebSign / db SignaturCard

Branch (three-digit)	Account (seven-digit)	Sub-account (two-digit)	PIN (five-digit)
<input type="text"/>	<input type="text"/>	00	<input type="text"/>

Directly to ...

Session TAN for Brokerage ?

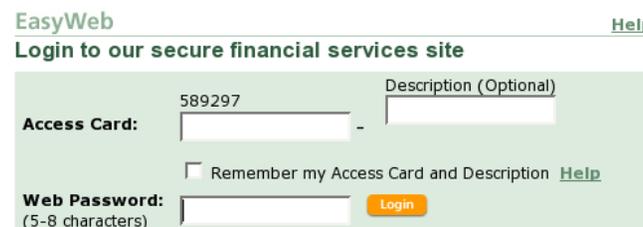
Execute Login ▶



TD Canada Trust EasyWeb

Apply | Search | Contact Us | Login: WebBroker

My Accounts | Customer Service | Products & Services | Markets & Research | Planning



EasyWeb [Help](#)

Login to our secure financial services site

Access Card: 589297 - Description (Optional)

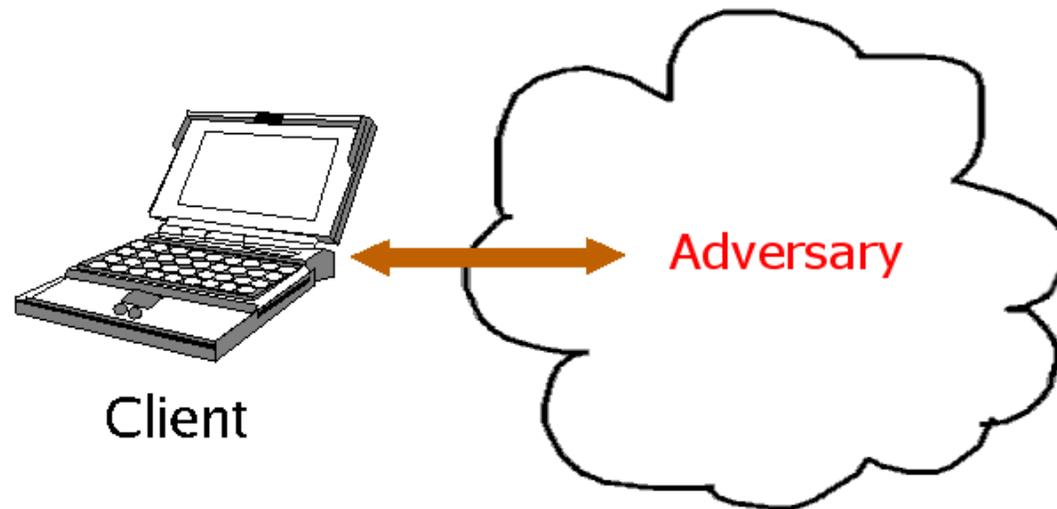
Remember my Access Card and Description [Help](#)

Web Password: (5-8 characters) Login

The scenario (cont.)

Sending the password like this puts a lot of trust in server authentication!

May be handing away the password to a “[phishing](#)” site.



The scenario (cont.)

Can we do better? **Yes!**

Can't prevent adversary from guessing the password.

If guess is right, adversary can act like a legitimate party.

After q protocol runs, adversary may have checked q guesses (success probability q/N for dictionary size N).

But protocol should protect password from exposure:

- Don't actually send password;
- moreover, prevent dictionary attack (off-line brute-force search).

Classic protocol: Bellare/Merritt Encrypted Key Exchange (EKE).

Password used as encryption key within key exchange.

Want a protocol that

- works well with TLS,
- is provably secure.

Our protocol: SOKE-TLS

The core of **SOKE** (Simple Open Key Exchange) is a **Diffie-Hellman** exchange with **one share masked**.

Uses DH group with *two* generators, g and U .

Server's DH share: $Y = g^y$ (y secret)

Client's DH share: $X = g^x$ (x secret)

Server sends Y , client sends $X^* = X \cdot U^{pw}$.

Server recovers X from X^* (using pw).

Common DH result: $Z = X^y = Y^x$.

Derive session key from

$$\text{PreMasterSecret} = \text{Hash}(C, S, pw, X^*, Y, Z)$$

where C and S are identifiers (client name, server name).

Our protocol: SOKE-TLS (cont.)

Usual TLS ciphersuites send `Finished` under new keys to confirm handshake:

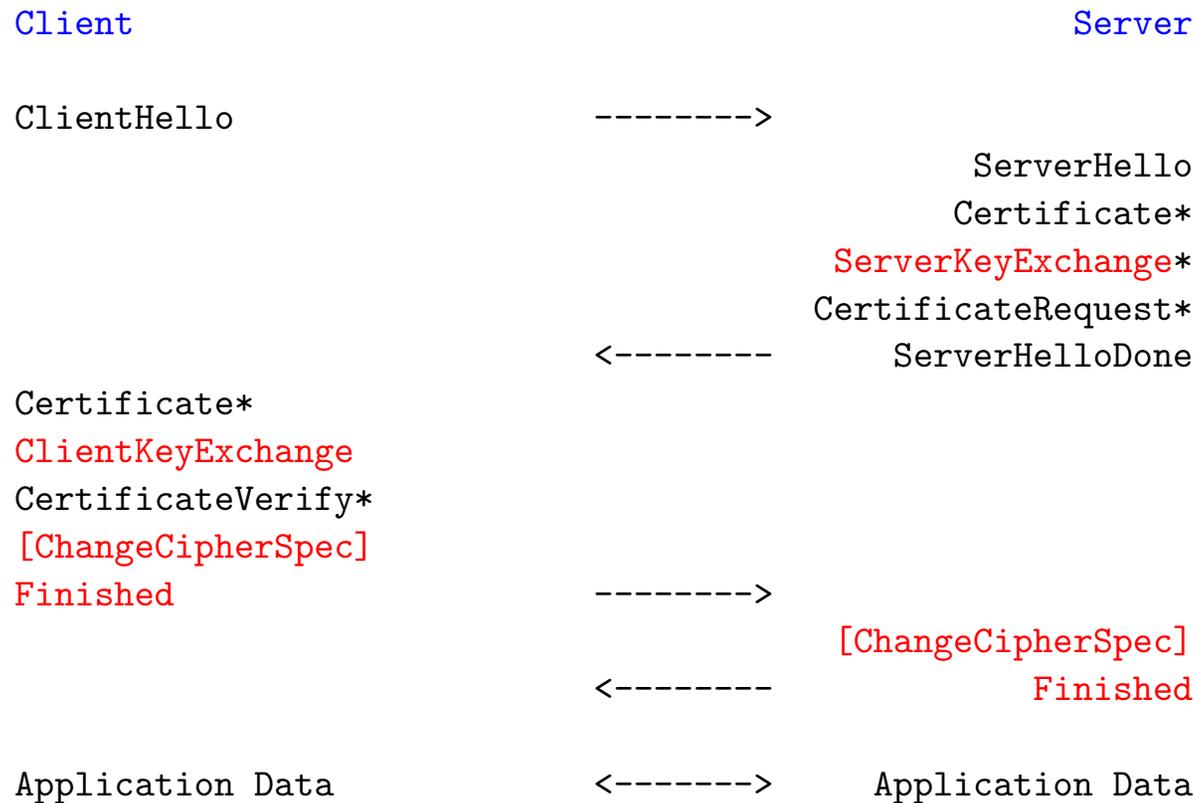


Fig. 1 - Message flow for a full handshake

* Indicates optional or situation-dependent messages that are not always sent.

Can't do this! Would expose client password to rogue server (dictionary attack).

Our protocol: SOKE-TLS (cont.)

Need to modify TLS handshake:

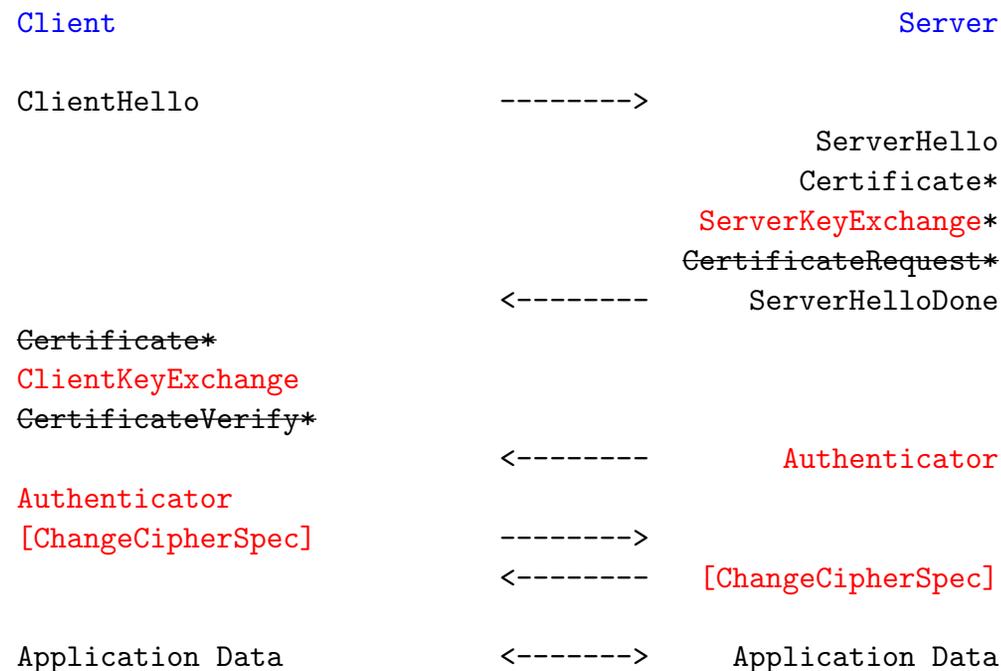


Fig. 1 - Message flow for a full handshake

* Indicates optional or situation-dependent messages that are not always sent.

Authenticator is based on **PreMasterSecret**, just like session key. Client verifies server's **Authenticator** before revealing anything.

Our protocol: SOKE-TLS (cont.)

In the Random Oracle Model, the protocol can be **proven secure**.

Security model for authenticated key exchange provides adversary with oracles that perform protocol operations at the adversary's direction.

Test oracle consistently returns real or random session keys.

The adversary's task is to find out which.

Corrupt oracle reveals a party's password (session key forced to be real afterwards).

Protocol provably provides **forward-security** under **concurrent executions**.

Similarly, can prove **explicit authentication** (adversary cannot impersonate server).

Comparison with other protocols

New protocol: [SOKE-TLS](#)

Earlier protocol proposals:

- [SBEW2001](#): Steiner, Buhler, Eirich, Waidner,
Secure password-based cipher suite for TLS
(ACM Transactions on Information and System Security vol. 4)
- Tom Wu et al.:
Using [SRP](#) [Secure Remote Passwords] for TLS authentication.
Internet-Draft. See <http://srp.stanford.edu/>

Comparison with other protocols (cont.)

Advantages of SOKE-TLS:

- Formal security arguments!
- Efficiency: SOKE-TLS uses prime-order group.
Can use ≈ 160 -bit exponents with 1024-bit DH modulus,
can use elliptic curves or other groups.

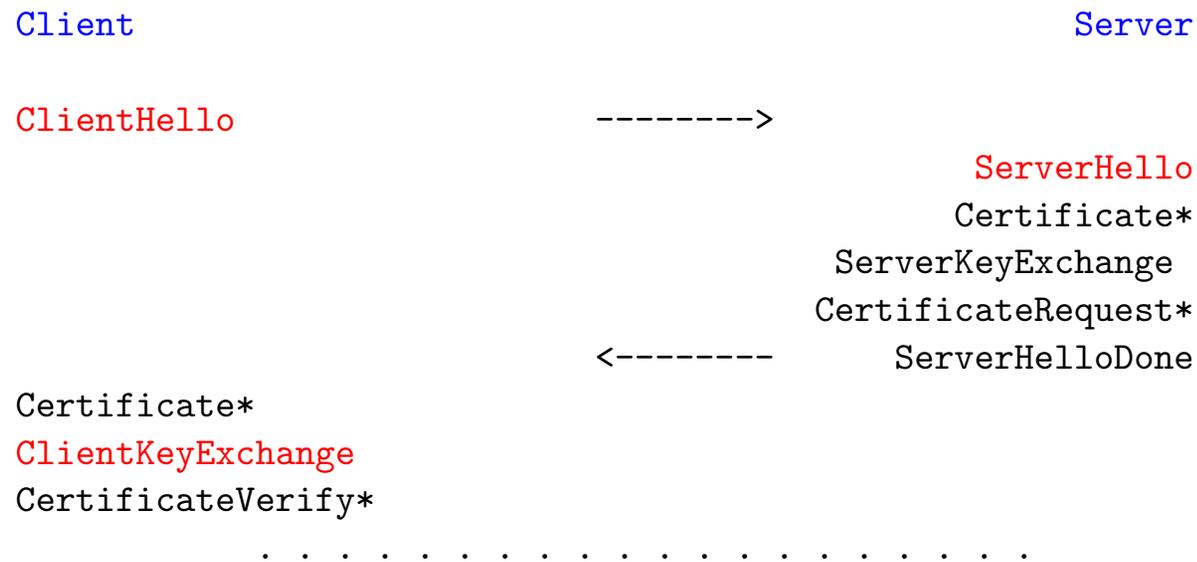
SBEW2001 and SRP handle supergroup elements.

Need larger exponents (≈ 1024 bits), require specific groups.

- Flexible handshake . . . see next slide

Comparison with other protocols (cont.)

- Flexible handshake (like SBEW2001, advantage over SRP)



Send client identity in ClientKeyExchange.
 (SRP sends it earlier, in ClientHello.)

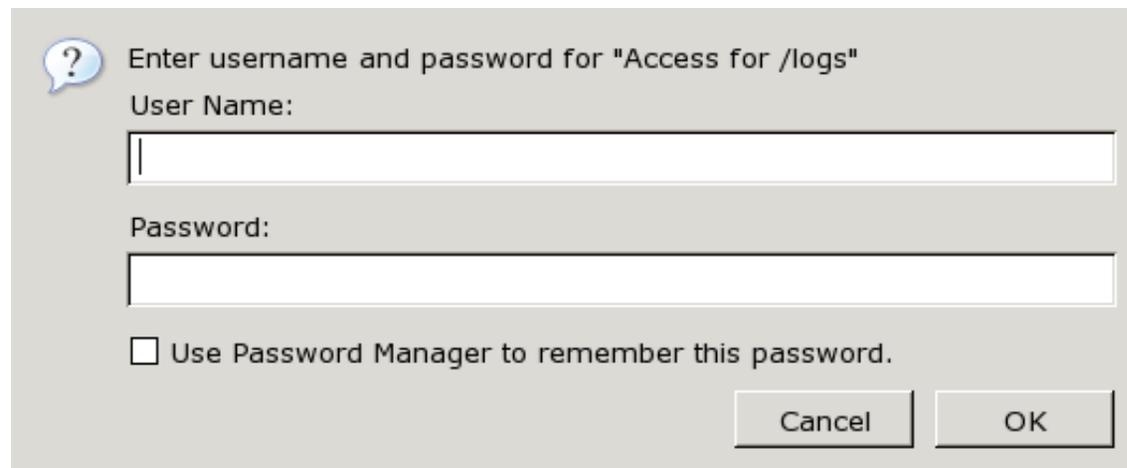
...

Comparison with other protocols (cont.)

- Flexible handshake . . .

Need identity only if server chooses password-based TLS ciphersuite;
can choose client identity after seeing server identity.

Like password authentication in HTTP!



A screenshot of a Windows-style password prompt dialog box. The dialog has a light gray background and a title bar. In the top-left corner, there is a blue speech bubble icon containing a white question mark. To the right of the icon, the text reads "Enter username and password for "/>

Thank you!