The SPEKE Protocol Revisited

Feng Hao, Siamak Shahandashti

School of Computing Science
Newcastle University, UK

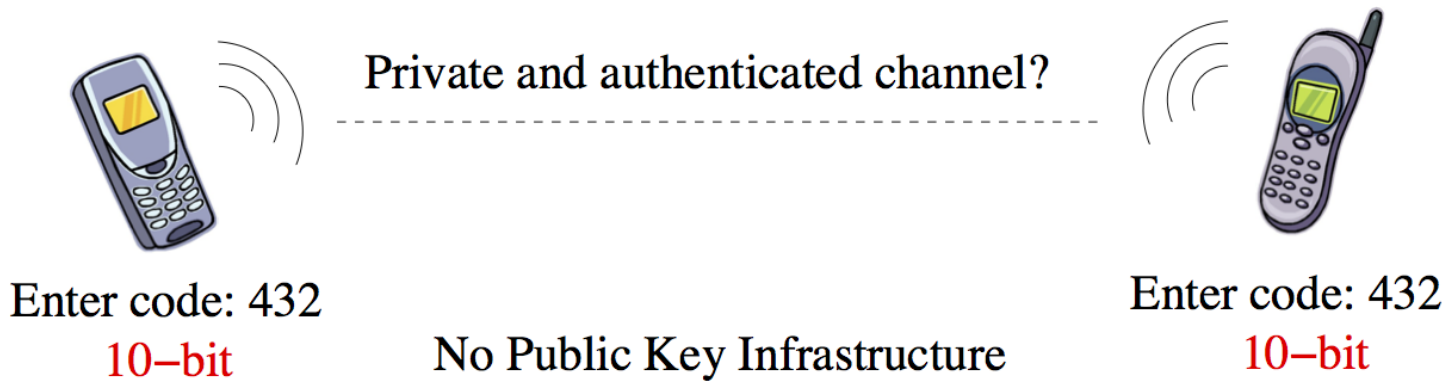
Security Standardisation Research (SSR)
Royal Holloway, 2014

Password Authenticated Key Exchange

Definition

Password Authenticated Key Exchange (PAKE) is a protocol that serves to establish an end-to-end secure channel between two remote parties solely based on a shared low-entropy password.

A concrete example:



Simple Password Encrypted Key Exchange

- First designed in 1996 by David Jablon
- Patented by Phoenix Technologies
- Commonly regarded as one of the classic PAKE protocols
- Standardized by IEEE P1363.2 and ISO/IEC 11770-4
- Deployed in commercial products (e.g., Blackberry)

The original SPEKE specification (Jablon'97)

	Alice		Bob
1.	$x \in_R Z_q$	$\xrightarrow{X = g^x}$	Verify $X \in [2, p-2]$
2.	Verify $Y \in [2, p-2]$	$\xleftarrow{Y = g^y}$	$y \in_R Z_q$
	$\kappa = H(Y^x) = H(g^{xy})$		$\kappa = H(X^y) = H(g^{xy})$
	Explicit key confirmation (optional)		

- Use a safe prime $p = 2q + 1$
- Use a password-derived generator: $g = s^2$

Explicit key confirmation (Jablon'97)

Alice		Bob
1.	$\xrightarrow{H(H(\kappa))}$	Verify $H(H(\kappa))$
2. Verify $H(\kappa)$	$\xleftarrow{H(\kappa)}$	
Secure communication (auth encryption based on κ)		

- Explicit key confirmation is optional
- If one uses κ straightaway, key confirmation is implicit.

A patch to the original SPEKE

- Exponential-equivalence attack (Zhang'05)
  - The use of plaintext password in $g = s^2$ allows an active attacker to test multiple passwords in one go
- The proposed patch
  - Add a hash function: $g = (H(s))^2$
  - Patch adopted by IEEE 1363.2 and ISO/IEC 11770-4

Is the patched SPEKE secure?

- No major flaws reported in the past 10 years
- Internationally standardized
- Supported by formal analysis with "provable security" (MacKenzie, IACR ePrint 2001/057)

We present two new attacks

1. Impersonation attack
   - A practical weakness
2. Key-malleability attack
   - A theoretical flaw

First attack: impersonation attack

Alice		Impersonator
Select $x \in_R$ [1,q-1]	$\xrightarrow{1.\, g^x, \hat{A}}$	
Compute $\kappa = H(g^{xyz})$	$\xleftarrow{4.\, g^{y \cdot z}, \hat{B}}$	Choose arbitrary z (Session 1)
Start key confirmation	$\xrightarrow{5.\, H(H(\kappa)), \hat{A}}$	
Verify key confirmation	$\xleftarrow{8.\, H(\kappa), \hat{B}}$	
		$\{g^{xz}, H(H(\kappa))\} \downarrow\uparrow \{g^y, H(\kappa)\}$
Select $y \in_R$ [1,q-1]	$\xleftarrow{2.\, g^{x \cdot z}, \hat{B}}$	
Compute $\kappa = H(g^{xyz})$	$\xrightarrow{3.\, g^y, \hat{A}}$	(Session 2)
Verify key confirmation	$\xleftarrow{6.\, H(H(\kappa)), \hat{B}}$	
Reply key confirmation	$\xrightarrow{7.\, H(\kappa), \hat{A}}$	

Implication of the impersonation attack

- Without knowing the password, the attacker is able to impersonate someone who knows the password.
- Alice thinks she is sharing a session key with “Bob” (someone who knows the password), but she is actually sharing the session key with herself.

Confusion in identity can be problematic - an example

- After key exchange, Alice uses the derived session key κ to encrypt messages in the authenticated mode
- Alice $\longrightarrow$ "Bob": E_κ("Send 5 Bitcoins to Charlie"). (Intercepted by the attacker)
- The attacker replays the encrypted message to Alice in the second session
- The message is verified to be authentic from "Bob".
- Alice follows the instruction and pays Charles 5 Bitcoins.
- In practice, Alice may be a computer program.

Second attack: key-malleability attack

Alice		MITM		Bob
Select $x \in_R [1, q-1]$	$\xrightarrow{g^x, \hat{A}}$		$\xleftarrow{g^y, \hat{B}}$	Select $y \in_R [1, q-1]$
		Select $z \in [2, q-2]$		
Check $(g^y)^z \in [2, p-2]$	$\xleftarrow{(g^y)^z, \hat{B}}$	Raise to power z	$\xrightarrow{(g^x)^z, \hat{A}}$	Check $(g^x)^z \in [2, p-2]$
Compute $\kappa = H(g^{xyz})$				Compute $\kappa = H(g^{xyz})$

Implications of this Key-malleability attack

- No practical harm identified
- But has an unfavorable theoretical implication – a clean reduction to CDH/DDH is impossible

Applicability of the attacks

	Impersonation attack	Key-malleability attack
Original (patched) SPEKE with KC	Yes	Yes
Original (patched) SPEKE without KC	Yes	Yes
SPEKE in IEEE P1363.2 with KC	Yes	No
SPEKE in IEEE P1363.2 without KC	Yes	Yes
SPEKE in ISO/IEC 11770-4 with KC	Maybe	No
SPEKE in ISO/IEC 11770-4 without KC	Maybe	Yes
SPEKE in IETF I-D with KC	Yes	No
SPEKE in IETF I-D without KC	Yes	Yes

- The explicit key confirmation is optional in all standards

Definition of the shared secret π

IEEE P1363.2	ISO/IEC 11770-4
A password-based octet string, used for authentication. π is generally derived from a password or a hashed password, and may incorporate a salt value, identifiers for one or more parties, and/or other shared data.	A password-based octet string which is generally derived from a password or a hashed password, identifiers for one or more entities, an identifier of a communication session if more than one session might execute concurrently, and optionally includes a salt value and/or other data.

- Not sufficiently clear how π is defined in ISO/IEC 11770-4
- Neither standard provides any concrete formula.
- In practice, Blackberry doesn't include any identifiers in π.

Key confirmation in the original SPEKE paper [Jab97]

	One party		The other party
1.		$\xrightarrow{H(H(\kappa))}$	Verify $H(H(\kappa))$
2.	Verify $H(\kappa)$	$\xleftarrow{H(\kappa)}$	

- Which party should initiate the key confirmation?
  - It doesn't matter; either party is fine [Jab97]
- But what if both parties initiate the key confirmation?
  - They may enter a deadlock.

Key confirmation in ISO/IEC 11770-4 (and IEEE P1363.2)

	One party		The other party
1.		$H(\text{“0x03”} \| g^x \| g^y \| g^{xy} \| g) \rightarrow$	Verify
2.	Verify	$\leftarrow H(\text{“0x04”} \| g^x \| g^y \| g^{xy} \| g)$	

- Which party should initiate the key confirmation?
  - It doesn't matter; either party is fine [ISO/IEC 11770-4]
- Exactly the same limitation at in [Jab97]
- The key confirmation cannot be completed in one round.

We propose two changes to both standards

1. Redefine the session key computation in SPEKE
   - To address the identified attacks
2. Redefine the key confirmation method in SPEKE
   - To improve round efficiency

Proposed change 1: session key computation

Alice ($\hat{A}$)		Bob ($\hat{B}$)
Select $x \in_R [1, q-1]$. Compute $M = g^x$	$\hat{A}, M = g^x \longrightarrow$	Check $M \in [2, p-2]$
Check $N \in [2, p-2]$	$\hat{B}, N = g^y \longleftarrow$	Select $y \in [1, q-1]$. Compute $N = g^y$
Alice Computes: $\kappa_a = H\left(\min(\hat{A}, \hat{B}), \max(\hat{A}, \hat{B}), \min(M, N), \max(M, N), g^{xy}\right)$		
Bob Computes: $\kappa_b = H\left(\min(\hat{A}, \hat{B}), \max(\hat{A}, \hat{B}), \min(M, N), \max(M, N), g^{xy}\right)$		

- This change addresses both the impersonation and key-malleability attacks

Proposed change 2: key confirmation

$$\begin{array}{ccccc} \text{Alice} & \rightarrow & \text{Bob} & : & HMAC(\kappa, \text{“}KC\_1\_U\text{”} \| \hat{A} \| \hat{B} \| g^x \| g^y) \\ \text{Bob} & \rightarrow & \text{Alice} & : & HMAC(\kappa, \text{“}KC\_1\_U\text{”} \| \hat{B} \| \hat{A} \| g^y \| g^x) \end{array}$$

- Based on key confirmation in NIST SP 800-56A Revision 1
- The two steps can be performed simultaneously within one round

Summary of results

	Round efficiency	Impersonation attack	Key-malleability attack
Original (patched) SPEKE with KC	3	Yes	Yes
Original (patched) SPEKE without KC	1	Yes	Yes
SPEKE in IEEE P1363.2 with KC	3	Yes	No
SPEKE in IEEE P1363.2 without KC	1	Yes	Yes
SPEKE in ISO/IEC 11770-4 with KC	≥ 3	Maybe	No
SPEKE in ISO/IEC 11770-4 without KC	≥ 1	Maybe	Yes
SPEKE in IETF I-D with KC	3	Yes	No
SPEKE in IETF I-D without KC	1	Yes	Yes
Patched SPEKE with KC	***2***	***No***	***No***
Patched SPEKE without KC	***1***	***No***	***No***

Status update

- The attacks and proposed changes were discussed in ISO/IEC SC 27/WG2 meeting in Mexico, October, 2014
- It was agreed that the SPEKE protocol in ISO/IEC 11770-4 should be revised to address the identified weaknesses.
- The revision is currently in progress.

Q & A

Thank you!

For more technical details, see
`https://eprint.iacr.org/2014/585.pdf`