Password Authenticated Key Exchange: From Two Party Methods to Group Schemes

Stephen Melczer, Taras Mychaskiw, and Yi Zhang

Introduction

PART 1
Classical Two Party PAKEs

Password Authenticated Key Exchange (PAKE)

PAKEs allow two parties sharing a *short/weak* password to establish a shared key

Cannot broadcast password directly – would need to be protected (expensive)

Instead, modern PAKEs use *zero-knowledge proof* and/or *hash* of password in protocol

First Protocol: EKE (Bellovin and Merrit 1992)

Pick prim. root $\alpha \in \mathbb{Z}_p$

Alice		Bob
randomly choose $x_a \in \mathbb{Z}_p^*$	$\xrightarrow{[\alpha^{x_a}]_s}$	
	$\xleftarrow{[\alpha^{x_b}]_s}$	randomly choose $x_b \in \mathbb{Z}_p^*$

Alice and Bob share $K = \alpha^{x_a \cdot x_b}$.

First Protocol: EKE (Bellovin and Merrit 1992)

Pick prim. root $\alpha \in \mathbb{Z}_p$

Alice		Bob
randomly choose $x_a \in \mathbb{Z}_p^*$	$\xrightarrow{[\alpha^{x_a}]_s}$	
	$\xleftarrow{[\alpha^{x_b}]_s}$	randomly choose $x_b \in \mathbb{Z}_p^*$

Alice and Bob share $K = \alpha^{x_a \cdot x_b}$.

Uses password directly $\implies$ many insecurities found

First Protocol: EKE (Bellovin and Merrit 1992)

Pick prim. root $\alpha \in \mathbb{Z}_p$

Alice		Bob
randomly choose $x_a \in \mathbb{Z}_p^*$	$\xrightarrow{[\alpha^{x_a}]_s}$	
	$\xleftarrow{[\alpha^{x_b}]_s}$	randomly choose $x_b \in \mathbb{Z}_p^*$

Alice and Bob share $K = \alpha^{x_a \cdot x_b}$.

Uses password directly $\implies$ many insecurities found

Ex: Decypher $[\alpha^{x_a}]_{s'}$ – rule out s' if output in $[p, 2^n - 1]$

Desired Security Properties

Offline dictionary attack resistance

Don't leak info which can be used in brute force search

Forward secrecy for established keys

Past keys secure if password disclosed

Implies passive attacker w/ password cannot compute key

Known session security

All secrets of one session reveals nothing about others

Online dictionary attack resistance

Attacker can only test one password per protocol execution

J-PAKE

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$

Alice picks $x_1 \in_R [0, q-1]$ and $x_2 \in_R [1, q-1]$

Bob picks $x_3 \in_R [0, q-1]$ and $x_4 \in_R [1, q-1]$

J-PAKE

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$
Alice picks $x_1 \in_R [0, q-1]$ and $x_2 \in_R [1, q-1]$
Bob picks $x_3 \in_R [0, q-1]$ and $x_4 \in_R [1, q-1]$

Rd 1 Alice sends g^{x_1}, g^{x_2} and ZKPs of x_1 and x_2 to Bob
Bob sends g^{x_3}, g^{x_4} and ZKPs of x_3 and x_4 to Alice
[Alice and Bob verify ZKPs and $g^{x_2}, g^{x_4} \neq 1$]

J-PAKE

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$
Alice picks $x_1 \in_R [0, q-1]$ and $x_2 \in_R [1, q-1]$
Bob picks $x_3 \in_R [0, q-1]$ and $x_4 \in_R [1, q-1]$

Rd 1 Alice sends g^{x_1}, g^{x_2} and ZKPs of x_1 and x_2 to Bob
Bob sends g^{x_3}, g^{x_4} and ZKPs of x_3 and x_4 to Alice
[Alice and Bob verify ZKPs and $g^{x_2}, g^{x_4} \neq 1$]

Rd 2 Alice sends $A = g^{(x_1+x_3+x_4)x_2 \cdot s}$ and ZKP of $x_2 s$
Bob sends $B = g^{(x_1+x_2+x_3)x_4 \cdot s}$ and ZKP of $x_4 s$

J-PAKE

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$
Alice picks $x_1 \in_R [0, q-1]$ and $x_2 \in_R [1, q-1]$
Bob picks $x_3 \in_R [0, q-1]$ and $x_4 \in_R [1, q-1]$

Rd 1 Alice sends g^{x_1}, g^{x_2} and ZKPs of x_1 and x_2 to Bob
Bob sends g^{x_3}, g^{x_4} and ZKPs of x_3 and x_4 to Alice
[Alice and Bob verify ZKPs and $g^{x_2}, g^{x_4} \neq 1$]

Rd 2 Alice sends $A = g^{(x_1+x_3+x_4)x_2 \cdot s}$ and ZKP of $x_2 s$
Bob sends $B = g^{(x_1+x_2+x_3)x_4 \cdot s}$ and ZKP of $x_4 s$

They share

$$K = \underbrace{(B/g^{x_2 x_4 s})^{x_2}}_{\text{Computable by Alice}} = g^{(x_1+x_3)x_2 x_4 s} = \underbrace{(A/g^{x_2 x_4 s})^{x_4}}_{\text{Computable by Bob}} .$$

J-PAKE

Satisfies all 4 desired properties
(under DDH and CDH assumptions)

Only two rounds of communication

No *explicit* key confirmation (only implicit)

Not patented

Dragonfly

Setup Let Q be a cyclic subgroup of $\mathbb{Z}_p^*$ with prime order q. Both members map the password to an element $\pi \in Q$.

Dragonfly

Setup Let Q be a cyclic subgroup of $\mathbb{Z}_p^*$ with prime order q. Both members map the password to an element $\pi \in Q$.

Alice		Bob
repeat: randomly choose $r_A, m_A \in \mathbb{Z}_q^*$		repeat: randomly choose $r_B, m_B \in \mathbb{Z}_q^*$
$s_A = r_A + m_A \mod q$ until $s_A \geq 2$		$s_B = r_B + m_B \mod q$ until $s_B \geq 2$
$E_A = \pi^{-m_A} \mod p$	$\xrightarrow{s_A, E_A}$	$E_B = \pi^{-m_B} \mod p$
	$\xleftarrow{s_B, E_B}$	
Verify $E_A \neq E_B$ or $s_A \neq s_B$		Verify $E_A \neq E_B$ or $s_A \neq s_B$
$ss = (\pi^{s_B} E_B)^{r_A} = \pi^{r_A r_B}$		$ss = (\pi^{s_A} E_A)^{r_B} = \pi^{r_A r_B}$
$A = H(ss \vert E_A \vert s_A \vert E_B \vert s_B)$	$\xrightarrow{A}$	$B = H(ss \vert E_B \vert s_B \vert E_A \vert s_A)$
	$\xleftarrow{B}$	
Verify B		Verify A
$K = H(ss \vert E_A \times E_B \vert (s_A + s_B) \mod q)$		$K = H(ss \vert E_A \times E_B \vert (s_A + s_B) \mod q)$

Dragonfly

No formal security proofs, but claims resistance to offline dictionary attacks

Only two rounds of communication

Very fast compared to other protocols

PAK/PPK – PAK

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$

Let $p = rq + 1$ where r, q relatively prime.

Let π be the password and H_1, H_{2a}, H_{2b}, H_3 be random, independent hash functions.

PAK/PPK – PAK

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$

Let $p = rq + 1$ where r, q relatively prime.

Let π be the password and H_1, H_{2a}, H_{2b}, H_3 be random, independent hash functions.

PAK

A		B
$x \in_R Z_q$		
$m = g^x \cdot (H_1(A, B, \pi))^r$	$\xrightarrow{\quad m \quad}$	Test $m \overset{?}{\not\equiv} 0 \bmod p$
		$y \in_R Z_q$
		$\mu = g^y$
		$\sigma = (\frac{m}{(H_1(A,B,\pi))^r})^y$
$\sigma = \mu^x$	$\xleftarrow{\quad \mu, k \quad}$	$k = H_{2a}(A, B, m, \mu, \sigma, \pi)$
Test $k \overset{?}{=} H_{2a}(A, B, m, \mu, \sigma, \pi)$		
$k' = H_{2b}(A, B, m, \mu, \sigma, \pi)$		
$K = H_3(A, B, m, \mu, \sigma, \pi)$	$\xrightarrow{\quad k' \quad}$	Test $k' \overset{?}{=} H_{2b}(A, B, m, \mu, \sigma, \pi)$
		$K = H_3(A, B, m, \mu, \sigma, \pi)$

PAK/PPK – PPK

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$

Let $p = rq + 1$ where r, q relatively prime.

Let π be the password and H_1, H_3 be random, independent hash functions.

PAK/PPK – PPK

Setup Let $G = <g>$ be an order q subgroup of $\mathbb{Z}_p^*$

Let $p = rq + 1$ where r, q relatively prime.

Let π be the password and H_1, H_3 be random, independent hash functions.

PPK

A		B
$x \in_R Z_q$		
$m = g^x \cdot (H_1(A, B, \pi))^r$	$\xrightarrow{\quad m \quad}$	Test $m \overset{?}{\not\equiv} 0 \bmod p$
		$y \in_R Z_q$
		$\mu = g^y \cdot (H_1(A, B, \pi))^r$
		$\sigma = (\frac{m}{(H_1(A,B,\pi))^r})^y$
Test $\mu \overset{?}{\not\equiv} 0 \bmod p$	$\xleftarrow{\quad \mu \quad}$	$K = H_3(A, B, m, \mu, \sigma, \pi)$
$\sigma = (\frac{\mu}{(H_1(A,B,\pi))^r})^x$		
$K = H_3(A, B, m, \mu, \sigma, \pi)$		

PAK/PPK

Proposes a new formal security model
based on the random oracle model

Satisfies all 4 desired properties
(under DDH assumptions)

Only two/three rounds of communication

PAK has explicit key confirmation
PPK has implicit key confirmation

PART 2
Extension to Group Setting

The Fairy-Ring Dance

Desciption of general procedure

How pairwise + group keys are constructed

Dragonfly+

Setup Let Q be a cyclic subgroup of $\mathbb{Z}_p^*$ with prime order q with generator g. Each member maps the password to an element $\pi \in Q$.

Every member P_i chooses $r_{ij}, m_{ij} \in_R \mathbb{Z}_q^*$ for all $j \in \{1, \ldots, n\} \setminus \{i\}$

Each member also chooses $y_i \in_R \mathbb{Z}_q$ and computes $g^{y_i} \bmod p$

Dragonfly+

Setup Let Q be a cyclic subgroup of $\mathbb{Z}_p^*$ with prime order q with generator g. Each member maps the password to an element $\pi \in Q$.

Every member P_i chooses $r_{ij}, m_{ij} \in_R \mathbb{Z}_q^*$ for all $j \in \{1, \ldots, n\} \setminus \{i\}$

Each member also chooses $y_i \in_R \mathbb{Z}_q$ and computes $g^{y_i} \mod p$

Rd 1 Each member broadcasts $s_{ij} = r_{ij} + m_{ij} \mod q$ and $E_{ij} = \pi^{-m_{ij}} \mod p$

They also broadcast $g_{y_i} \mod p$ along with $\mathrm{ZKP}\{y_i\}$

[All members verify the ZKP, verify $g_{z_i} \neq 1$ and check for reflection attacks]

Dragonfly+

Rd 2 Each member computes their pairwise shared secrets:
$ss_{ij} = (\pi^{s_{ji}} E_{ji})^{r_{ij}}$
Each member broadcasts $H(ss_{ij}||E_{ij}||s_{ij}||E_{ji}||s_{ji})$
[All members verify the pairwise hash values]

Dragonfly+

Rd 2 Each member computes their pairwise shared secrets:
$ss_{ij} = (\pi^{s_{ji}} E_{ji})^{r_{ij}}$
Each member broadcasts $H(ss_{ij}||E_{ij}||s_{ij}||E_{ji}||s_{ji})$
[All members verify the pairwise hash values]

Rd 3 Every member broadcasts $(g^{z_i})^{y_i}$ and ZKP$\{\tilde{y}_i\}$.
Let K_{ij} be the pairwise Dragonfly key between members i and j. Members compute
$\kappa_{ij}^{MAC} = H(K_{ij}||"MAC"), \kappa_{ij}^{KC} = H(K_{ij}||"KC")$
Members broadcast
$t_{ij}^{MAC} = HMAC(\kappa_{ij}^{MAC}, g^{y_i}||\text{ZKP}\{y_i\}||(g^{z_i})^{y_i}||\text{ZKP}\{\tilde{y}_i\})$
and $t_{ij}^{KC} = HMAC(\kappa_{ij}^{KC}, "KC"||i||j||E_{ij}||E_{ji})$
[All members verify ZKP$\{\tilde{y}_i\}$, t_{ji}^{MAC} and t_{ij}^{KC} are correct]

Dragonfly+

Rd 2 Each member computes their pairwise shared secrets:
$ss_{ij} = (\pi^{s_{ji}} E_{ji})^{r_{ij}}$
Each member broadcasts $H(ss_{ij}||E_{ij}||s_{ij}||E_{ji}||s_{ji})$
[All members verify the pairwise hash values]

Rd 3 Every member broadcasts $(g^{z_i})^{y_i}$ and ZKP$\{\tilde{y_i}\}$.
Let K_{ij} be the pairwise Dragonfly key between members i and j. Members compute
$\kappa_{ij}^{MAC} = H(K_{ij}||"MAC"), \kappa_{ij}^{KC} = H(K_{ij}||"KC")$
Members broadcast
$t_{ij}^{MAC} = HMAC(\kappa_{ij}^{MAC}, g^{y_i}||\text{ZKP}\{y_i\}||(g^{z_i})^{y_i}||\text{ZKP}\{\tilde{y_i}\})$
and $t_{ij}^{KC} = HMAC(\kappa_{ij}^{KC}, "KC"||i||j||E_{ij}||E_{ji})$
[All members verify ZKP$\{\tilde{y_i}\}$, t_{ji}^{MAC} and t_{ij}^{KC} are correct]

All members share:

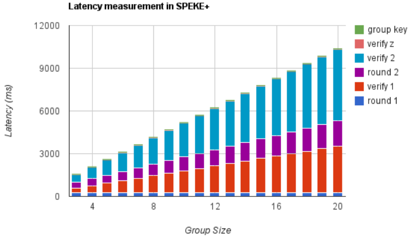
$$K = g^{y_1 \cdot y_2 + y_2 \cdot y_3 + \cdots + y_n \cdot y_1}$$

PART 3
Timings

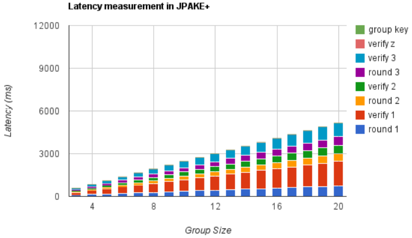
Specifications

- All protocol benchmarks were implemented in Java 1.6 and run on a server (3GHz AMD processor, 6GB of RAM) running Ubuntu 12.04.
- Benchmarks measured latency, the amount of work each device would have to do in the group excluding communication.

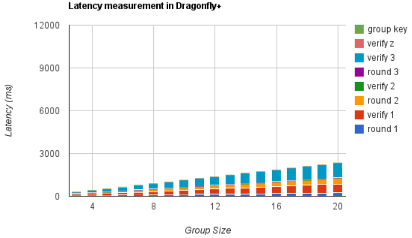
Results



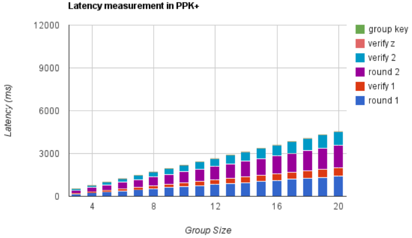
Latency measurement in SPEKE+
Latency (ms)
Group Size
group key
verify z
verify 2
round 2
verify 1
round 1



Latency measurement in JPAKE+
Latency (ms)
Group Size
group key
verify z
verify 3
round 3
verify 2
round 2
verify 1
round 1



Latency measurement in Dragonfly+
Latency (ms)
Group Size
group key
verify z
verify 3
round 3
verify 2
round 2
verify 1
round 1



Latency measurement in PPK+
Latency (ms)
Group Size
group key
verify z
verify 2
round 2
verify 1
round 1

Conclusion

Conclusion

It is possible to transfer PAKEs into GPAKEs while preserving round efficiency

SPEKE+ is very slow

J-PAKE+ is a bit slow, but proven secure (under CDH)

PPK is faster but weaker security proof

Dragonfly is fastest, but no security proof
(despite IEEE 802.11-2012 standard)

THANK YOU