VTBPEKE: Verifier-based Two-Basis Password Exponential Key Exchange

IETF 101, London
March, 2018

Guilin Wang
(wang.guilin@Huawei.com)
Content

- **PAKE:** Terminology, Challenges, Existing Solutions
- **Our Proposals:** TBPEKE, VTBPEKE, Comparison, and Implementation
- **Candidate proposal for RFC 8125?**
PAKE: Terminology

- **PAKE**: Password-Authenticated Key Exchange, enable two parties to establish a shared cryptographically strong key over an insecure network using a short common secret as authentication means.

- **Dictionary Attack**: attackers guess users’ passwords from a dictionary.
  - **Online Dictionary Attack**: try dictionary attack on line, which can be countered by limiting the times of trial in a given period.
  - **Offline Dictionary Attack**: Attackers do offline computations to recover users’ passwords, after intercepting messages of some PAKE executions.

- **Forward Security**: Even if the password is later leaked, the privacy of a past communication is still guaranteed.

- **Server Corruption**: In normal PAKE, the compromising of the server may allow immediate leakage of all the passwords.

- **VPAKE**: Verifier PAKE for resisting server corruption, in which the server only stores a verifier of user’s password, like the hash value of a password.
PAKE: Challenges and Our Work

- **Main challenge:** To design PAKE secure against offline dictionary attack.

- **Limitations in most of existing PAKE solutions:**
  - Do not support forward security, but this is essential to guarantee the privacy of a past communication.
  - Security only proved in the multiplicative groups of finite fields, which implies having to use large elements and then conduct huge communications and computations.

- We propose two PAKE protocols, which meet forward security and works in any group. [http://www.di.ens.fr/users/pointche/Documents/Papers/2017_asiaccsB.pdf](http://www.di.ens.fr/users/pointche/Documents/Papers/2017_asiaccsB.pdf)
  - TBPEKE: Two-Basis Password Exponential Key Exchange
  - VTBPEKE: a verifier-based variant of TBPEKE

- Both protocols are proveably secure under standard complexity assumptions.
- **Elliptic curves** can be used to implement the protocols, which leads to better efficiency, for both communication and computation.
PAKE: Existing Solutions

1) EKE (Encrypted Key Exchange) \([BM92]\):

\[
A^{pw} \quad B^{pw}
\]

\[
x \overset{R}{\leftarrow} \mathbb{Z}_p^*, X \leftarrow g^x, X' \leftarrow \mathcal{E}_{pw}(X) \quad A\|X' \rightarrow X \leftarrow \mathcal{D}_{pw}(X')
\]

\[
Y \leftarrow \mathcal{D}_{pw}(X') \rightarrow Y' \quad y \overset{R}{\leftarrow} \mathbb{Z}_p^*, Y \leftarrow g^y, Y' \leftarrow \mathcal{E}_{pw}(Y)
\]

\[
Z \leftarrow Y^x \leftarrow X^y
\]

\[
sk \leftarrow H(A\|B\|X\|Y\|Z)
\]

Comments on EKE:
- The first PAKE protocol, proposed by Bellovin and Merrit.
- **Basic Idea**: Password \(pw\) is used as a symmetric key to improve the security of DH key exchange.
- **Security**: Security under random oracle model [BMP00] and under UC model [ACCP08], by the symmetric encryption is an ideal cipher.

5) SPEKE (Simple Password Exponential Key Exchange) \([Jab96]\):

\[
A^{pw} \quad B^{pw}
\]

\[
G \text{ of prime order } p, G \text{ a hash function onto } G, \text{ and } H \text{ a hash function onto } \{0,1\}^t
\]

\[
g \leftarrow H(pw), x \overset{R}{\leftarrow} \mathbb{Z}_p^*, X \leftarrow g^x
\]

\[
A\|X \rightarrow Y \leftarrow g \leftarrow H(pw), y \overset{R}{\leftarrow} \mathbb{Z}_p^*, Y \leftarrow g^y
\]

\[
\text{accept } \leftarrow \text{true}
\]

\[
Z \leftarrow Y^x \leftarrow X^y
\]

\[
sk \leftarrow H(A\|B\|g\|X\|Y\|Z)
\]

Comments on SPEKE:
- Proposed by David Jablon [Jab96a, Jab96b]
- **Basic Idea**: \(g=H(pw)\), i.e., \(pw\) is used to generate the generator.
- **Security**: Provable security in the BPR model [Mac01] under the CDH assumption. But the proof applies only to a multiplicative sub-group of finite fields \(\mathbb{Z}_p^*\), not for ECC groups.
- **Efficiency**: Due to the above reason, big group size, not efficient in both communication and computation.
Our Proposals: TBPEKE

- **TBPEKE** is an improvement of SPEKE to make it more efficient and secure.
- Security proof applies to ECC and also a group of finite fields.
- **Basic Idea:** $g = H(pw)$ is used in SPEKE, and we here define $g = UV^{pw}$.
- Two bases (U and V) used here, inspired by SPAKE.

\[
\begin{align*}
A^{pw} & \quad B^{pw} \\
G \text{ of prime order } p, \quad U, V & \leftarrow G, \text{ and } H \text{ a hash function onto } \{0, 1\}^t \\
\text{accept} & \leftarrow \text{false} \\
g & \leftarrow U \cdot V^{pw}, \quad x \leftarrow \mathbb{Z}_p^*, \quad X & \leftarrow g^x \\
& \quad \quad \quad A \| X \ \rightarrow \\
Y & \quad g \leftarrow U \cdot V^{pw}, \quad y \leftarrow \mathbb{Z}_p^*, \quad Y & \leftarrow g^y \\
\text{accept} & \leftarrow \text{true} \\
Z & \leftarrow Y^x \\
& \quad \quad \quad \quad \leftarrow Z \leftarrow X^y \\
sk & \leftarrow H(A \| B \| g \| X \| Y \| Z)
\end{align*}
\]
Our Proposals: VTBPEKE

- A client (A) saves $pw$, while the server (B) saves a salt $s$ and the verifier $V^{H(s,pw)}$.
- **Basic Idea:** 1) A needs to prove its knowledge of $H(s, pw)$, for preventing an attack to impersonate A after getting the verifier. 2) The proof response $\rho$ is encrypted for prevent off-line dictionary attack to guess $pw$.

\[
\begin{align*}
A^{pw} & \quad \quad \quad B^{s,Y} \text{ where } \mathcal{P} \leftarrow \mathcal{H}_p(s, pw) \text{ and } \mathcal{V} \leftarrow V^\mathcal{P} \\
\text{Global parameters: } G \text{ of prime order } p, U, V & \leftarrow \mathbb{G}, \mathcal{H}_p \text{ a hash function onto } \mathbb{Z}_p, \text{ and } \mathcal{H} \text{ onto } \{0, 1\}^k \\
\text{accept } & \leftarrow \text{false} \\
\alpha & \leftarrow \mathbb{Z}_p^*, R \leftarrow V^\alpha, R' \leftarrow \mathcal{H}(11||R) \\
\mathcal{P} & \leftarrow \mathcal{H}_p(s, pw), \mathcal{V} \leftarrow V^\mathcal{P} \\
g & \leftarrow U \cdot \mathcal{V}, x \leftarrow \mathbb{Z}_p^*, X \leftarrow g^x, Z \leftarrow Y^x \\
\rho & \leftarrow \alpha + \epsilon \cdot \mathcal{P} \mod p, \sigma \leftarrow \mathcal{E}_{ek}(\rho) \\
\text{accept } & \leftarrow \text{true} \\
\end{align*}
\]
Security Proofs

- **Security Model:**
  - Random oracle+Real-or-Random Game

- **Proofs in several cases:**
  - If Forward Security is required?
  - Password dictionary is considered in 3 cases (Large, Medium, Small)
  - Under generic model

- **Hard problem assumptions:**
  - CDH, Dlin, SDH (new problem introduced)
  - SDH is not easier than DLin [Theorem 1]
  - If Forward Security not required: CDH and SDH
  - If Forward Security required: GCDH and GSDH (G for Gap)

Example: When using bilinear and forward security is required, in this case GDLin and GCDH are actually Dlin and CDH. There is the security result:

Bilinear Settings

Note however that this reduction is really meaningful when the DDH oracle is efficient, which requires a bilinear map. Then, in such a case, the security simply relies on the DLin and the CDH assumptions:

Theorem 6. In the bilinear setting, under the DLin and CDH assumptions, the TBPEKE is a forward-secure PAKE. More precisely, the best advantage an adversary can get in the Real-or-Random security game (see Figure 1) is bounded by

\[
\text{Adv}(t) \leq \frac{q_s}{N} + N^2 \times \text{Adv}^{\text{dlin}}(t) + \text{Succ}^{\text{cdh}}(t) + \frac{q_s q_e}{p^2}, \quad \text{if the dictionary is small, or}
\]
\[
\text{Adv}(t) \leq q_s \times \sqrt{10N_C} \times \text{Adv}^{\text{dlin}}(t) + \text{Succ}^{\text{cdh}}(t) + \frac{q_s q_e}{p^2}, \quad \text{if the dictionary is large.}
\]
## Comparison

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Communication (Both Sides)</th>
<th>Computation (Both Sides)</th>
<th>Forward Secrecy</th>
<th>Security Model</th>
<th>Assumptions</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PAKE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EKE [12]</td>
<td>1G / 1G</td>
<td>2E / 2E</td>
<td>Yes</td>
<td>ICM</td>
<td>CDH [19]</td>
<td></td>
</tr>
<tr>
<td>SPAKE [6]</td>
<td>1G / 1G</td>
<td>2E + 2sE / 2E + 2sE</td>
<td>No</td>
<td>ROM</td>
<td>CDH</td>
<td></td>
</tr>
<tr>
<td>SPAKE2 [6]</td>
<td>1G / 1G</td>
<td>2E + 2sE / 2E + 2sE</td>
<td>No</td>
<td>ROM</td>
<td>CDH</td>
<td></td>
</tr>
<tr>
<td>SPEKE [33]</td>
<td>1G / 1G</td>
<td>2E / 2E</td>
<td>No</td>
<td>ROM</td>
<td>CDH</td>
<td>in $\mathbb{Z}_p^*$ only</td>
</tr>
<tr>
<td>SAE [32]</td>
<td>2G / 2G</td>
<td>3E / 3E</td>
<td>No</td>
<td>ROM</td>
<td>CDH [42]</td>
<td>in $\mathbb{Z}_p^*$ only</td>
</tr>
<tr>
<td>SRP [50]</td>
<td>3G / 3G</td>
<td>2E / 3E</td>
<td>No</td>
<td>ROM</td>
<td>CDH</td>
<td>in $\mathbb{Z}_p^*$ only</td>
</tr>
<tr>
<td>TBPEKE</td>
<td>1G / 1G</td>
<td>2E + 1sE / 2E + 1sE</td>
<td>Yes</td>
<td>ROM</td>
<td>GSDH</td>
<td></td>
</tr>
<tr>
<td><strong>Verifier-based PAKE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPAKE2+ [23]</td>
<td>1G / 1G</td>
<td>5E / 5E</td>
<td>No</td>
<td>ROM</td>
<td>CDH</td>
<td></td>
</tr>
<tr>
<td>AugPAKE [45]</td>
<td>1G + k / 1G + k</td>
<td>2E / 3E</td>
<td>No</td>
<td>ROM</td>
<td>Strong DH [48]</td>
<td></td>
</tr>
<tr>
<td>VTBPEKE</td>
<td>1G + k +</td>
<td>p</td>
<td>/ 1G + k</td>
<td>4E / 4E</td>
<td>Yes</td>
<td>ROM</td>
</tr>
</tbody>
</table>
4 security levels are given

- For each security level, 3 case of password size are given.
- For each combination of security level and password size, the required bit length of ECC element is given.

**Fig. 6. Parameters: Bit-length of the order of the groups**

<table>
<thead>
<tr>
<th></th>
<th>64 bits</th>
<th>80 bits</th>
<th>112 bits</th>
<th>128 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4-Digit</td>
<td>8-Char</td>
<td>32-Char</td>
<td>4-Digit</td>
</tr>
<tr>
<td>Without FS</td>
<td>347</td>
<td>401</td>
<td>514</td>
<td>587</td>
</tr>
<tr>
<td>With FS</td>
<td>218*</td>
<td>272*</td>
<td>386*</td>
<td>362*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Without FS</td>
<td>427</td>
<td>441</td>
<td>642</td>
<td>667</td>
</tr>
<tr>
<td>With FS</td>
<td>266*</td>
<td>320*</td>
<td>482*</td>
<td>410*</td>
</tr>
</tbody>
</table>

* in pairing-friendly elliptic curves
Implementation

- OPENSSL Based Implementation

Implementation Environments:

- CPU1: Intel(R) Xeon(R) CPU E5-2690 v2 @ 3.00GHz
- CPU2: Intel(R) Xeon(R) CPU E3-1230 v3 @ 3.30GHz
- NIST P-256 and P-521 curves only
- Each CPU runs the computations by both parties of TBPEKE, and no network communication.

Testing Results: The time used for running the TBPEKE protocol 3000 times

- CPU1, P-256 curve: 6.415319 s
- CPU1, P-521 curve: 29.851816 s
- CPU2, P-256 curve: 4.728513 s
- CPU2, P-521 curve: 21.028553 s

Average per time running time: < 10ms
A Candidate Proposal for RFC 8125?


- This document reviews different types of PAKE schemes.
- It presents requirements and gives recommendations to designers of new schemes.

8 Requirements for PAKE, given by RFC 8125:

- **REQ1:** A PAKE scheme MUST clearly state its features regarding balanced/augmented versions.
- **REQ2:** A PAKE scheme SHOULD come with a security proof and clearly state its assumptions and models.
- **REQ3:** The authors SHOULD show how to protect their PAKE scheme implementation in hostile environments, particularly, how to implement their scheme in constant time to prevent timing attacks.
- **REQ4:** If the PAKE scheme is intended to be used with ECC, the authors SHOULD discuss their requirements for a potential mapping or define a mapping to be used with the scheme.
- **REQ5:** The authors of a PAKE scheme MAY discuss its design choice with regard to performance, i.e., its optimization goals.
- **REQ6:** The authors of a scheme MAY discuss variations of their scheme that allow the use in special application scenarios. In particular, techniques that facilitate long-term (public) key agreement are encouraged.
- **REQ7:** Authors of a scheme MAY discuss special ideas and solutions on privacy protection of its users.
- **REQ8:** The authors MUST follow the IRTF IPR policy https://irtf.org/ipr.
Thank you

www.huawei.com

Copyright©2015 Huawei Technologies Co., Ltd. All Rights Reserved.
The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.