Pairing-Friendly Curves
draft-yonezawa-pairing-friendly-curves

Shoko YONEZAWA, Tsunekazu SAITO, Tetsutaro KOBAYASHI

July 25, 2019
CFRG Meeting
IETF105 Montreal
Brief Overview

• Problem statement
  • Pairing-based cryptography is getting widely used
  • The security evaluation of pairing-friendly curves, which realize pairing-based cryptography, has been changed due to the attack proposed in 2016
  • Introducing secure pairing-friendly curves are required

• Goal
  • Show the latest security evaluation of well-known pairing-friendly curves
  • Show the parameters of pairing-friendly curves with each security level
    • According to their security evaluations in several papers and implementation status in several libraries
Related RG Items

• BLS Signature Schemes (draft-boneh-bls-signature, to appear as draft-irtf-bls-signature)
  • Pairing-based schemes that enable signature aggregation
  • Pairing-friendly curves are necessary for construction

• Hashing to Elliptic Curves (draft-irtf-cfrg-hash-to-curve)
  • Most pairing-based schemes (including BLS signatures) require hashing to pairing-friendly curves
Pairing-Based Cryptography

- A kind of elliptic curve cryptography which utilizes “pairing”
- Thanks to the property of pairing, cryptographic algorithms and protocols with more functionalities are getting widely used

Standards
- Identity-based cryptography (IBCS) [RFC5091]
- Sakai-Kasahara Key Encryption (SAKKE) [RFC6508]
- Identity-based authenticated key exchange (IBAKE) [RFC6539]
- (Identity-based) key agreement (ISO/IEC)
- Elliptic Curve Direct Anonymous Attestation (ECDAA) (TCG, FIDO, W3C)
- MIKEY-SAKKE (3GPP) – key encryption

Implementations
- M-Pin (MIRACL) – multi-factor authentication protocol
- Intel SGX EPID (Intel) – remote anonymous attestation protocol
- Geo Key Manager (Cloudflare) – attribute-based encryption
- zk-SNARKs (Zcash) – zero-knowledge proof for blockchain
- Decentralized Random Beacon (DFINITY) – threshold signature
- BLS signature (Algorand) – aggregate signature
Pairing-Based Cryptography (cont.)

- Like standard elliptic curve cryptography, pairing-based cryptography requires underlying elliptic curves
- Such elliptic curves are called **pairing-friendly curves**
- The security of pairing-based cryptography relies on the security of underlying pairing-friendly curves
Pairing

• Pairing (a.k.a. bilinear map) is a map from $G_1$ and $G_2$ onto $G_T$

$$e : G_1 \times G_2 \rightarrow G_T$$

satisfying

$$e([a]S, [b]T) = e(S, T)^{ab}.$$  

• In general, $G_1$, $G_2$ and $G_T$ are chosen as follows.
  • $G_1$ : a subgroup of the group defined over an elliptic curve $E$
  • $G_2$ : a subgroup of the group defined over a twisted curve of $E$
  • $G_T$ : a multiplicative group of finite field

• Various pairings
  • Weil pairing
  • Tate pairing
  • Optimal Ate pairing $\rightarrow$ most efficient and popular
Pairing-Friendly Curves

• A special kind of elliptic curves where pairing is efficiently computable
• Examples curves
  • Barreto-Naehrig (BN) Curve
  • Barreto-Lynn-Scott (BLS) Curve
    • BLS12 (embedded degree = 12)
    • BLS24 (embedded degree = 24)
    • BLS48 (embedded degree = 48), etc.
  • Kachisa-Schaefer-Scott (KSS) Curve
  • Miyaji-Nakabayashi-Takano (MNT) Curve
  • etc.
• Pairing-friendly curves vary in parameters (key length), which determine the security strength
  • ex. BN254, BN256, BLS12-381, ...
Security of Pairing-Friendly Curves

- Since the security of most pairing-based cryptography is reduced to the difficulty of these problems, we can only consider these DLPs.
- We should evaluate FFDLP in $G_T$ as well as ECDLP in $G_1$ and $G_2$. 

$$e : G_1 \times G_2 \rightarrow G_T$$

Elliptic Curve Discrete Logarithm Problem (ECDLP)  Elliptic Curve Discrete Logarithm Problem (ECDLP)  Finite Field Discrete Logarithm Problem (FFDLP)
Impact of Attack to Pairing-friendly Curves

• In 2016, Kim and Barbulescu presented a new number field sieve algorithm, exTNFS, at CRYPTO 2016 [KB16]
• Attacking by exTNFS affected the difficulty of FFDLP
• Due to the attack, the security level of ALL pairing-friendly curves has fallen
  • ex. BN256: 128-bit secure $\rightarrow$ 100-bit secure

Security Evaluation of Pairing-Friendly Curves

• After exTNFS, BN256 (regarded as 128-bit secure so far) achieves at most 100 bits of security now

• Introducing new parameters for each security level is required
  • 128 bits of security
  • 192 bits of security
  • 256 bits of security

• We select the parameters in terms of
  • Security
  • Efficiency
  • Wide use
128 / 256 Bits of Security

• 128 bits
  • BN462
    • Evaluated as approx. 133.49 bits of security [BD18] – conservative
    • Implementation available
  • BLS12-381
    • Evaluated as approx. 117 - 120 bits of security [NCCG] – optimistic
    • Implementation available and widely used

• 256 bits
  • BLS48-581
    • Evaluated as approx. 256 bits of security [KIK+17]
    • Implementation available

[NCCG] NCC Group, “Zcash Overwinter Consensus and Sapling Cryptography Review,”
Open Issue: 192 Bits of Security

- Candidate curve: BLS24
- Several papers for 192-bit-secure pairing-friendly curves
- NO implementation published
  - RELIC – preparing BLS24-477 but no executable code
  - AMCL – implementing BLS24 curve but not published

- QUESTION: How can we treat 192-bit parameters?
Fact: 192 Bits of Security

- **US CNSA Suite**
  - In order to protect up to TOP SECRET, the security parameters for asymmetric cryptography are set to satisfy 192 bits of security.

- **SSL Pulse Trends (June 2019)**
  - As for the key length of ECDH(E) in TLS servers, 5.23% of the servers support 521bit, 4.89% supports 571bit while 4.63% supports 381bit.

---

**Transition Algorithms**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Function</th>
<th>Specification</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced Encryption Standard (AES)</td>
<td>Symmetric block cipher used for information protection</td>
<td>FIPS Pub 197</td>
<td>Use 256 bit keys to protect up to TOP SECRET</td>
</tr>
<tr>
<td>Elliptic Curve Diffie-Hellman (ECDH) Key Exchange</td>
<td>Asymmetric algorithm used for key establishment</td>
<td>NIST SP 800-56A</td>
<td>Use Curve P-384 to protect up to TOP SECRET.</td>
</tr>
<tr>
<td>Elliptic Curve Digital Signature Algorithm (ECDSA)</td>
<td>Asymmetric algorithm used for digital signatures</td>
<td>FIPS Pub 188-4</td>
<td>Use Curve P-384 to protect up to TOP SECRET.</td>
</tr>
<tr>
<td>Secure Hash Algorithm (SHA)</td>
<td>Algorithm used for computing a condensed representation of information</td>
<td>FIPS Pub 180-4</td>
<td>Use SHA-384 to protect up to TOP SECRET.</td>
</tr>
<tr>
<td>Diffe-Hellman (DH) Key Exchange</td>
<td>Asymmetric algorithm used for key establishment</td>
<td>IEEE Std 3526</td>
<td>Minimum 3072-bit modulus to protect up to TOP SECRET.</td>
</tr>
<tr>
<td>RSA</td>
<td>Asymmetric algorithm used for key establishment</td>
<td>NIST SP 800-56B rev 1</td>
<td>Minimum 3072-bit modulus to protect up to TOP SECRET.</td>
</tr>
<tr>
<td>RSA</td>
<td>Asymmetric algorithm used for digital signatures</td>
<td>FIPS PUB 180-4</td>
<td>Minimum 3072 bit-modulus to protect up to TOP SECRET.</td>
</tr>
</tbody>
</table>

---


https://kjur.github.io/www/sslpulsetrend/index.html#kxecdh
History and Next Steps

• 00 version
  • Initial submission

• 01 version
  • Added pseudo-codes for pairing computation (from Kenny)
  • Added example parameters and test vectors of each curve (from Kenny)

• 02 version
  • Added 192 bits of security (no parameter provided yet) (from John)
  • Resolved comments from ML (from Mike, David, Marek and John)
  • Updated the status on applications and libraries

• Next Steps
  • Adoption call if interested