Gong-Harn Digital Signature Algorithm

Susana Sin
October 7, 2003

**GH Public-key Cryptosystem (GH-PKC) Overview**

- GH-PKC is based on the 3-rd order LFSR sequence with particular phase.
- GH-DH / GH-DSA was published in proceeding of the 8th Annual Workshop on Selected Areas in Cryptography, Aug 2001
- Security is based on the difficulty in solving DL problem in GF(q^3) for q = p or q = p^2

**Third-Order Characteristic Sequence**

- An irreducible polynomial \( f(x) \) of degree 3 over \( GF(p) \), where \( p \) is a prime number:
  \[ f(x) = x^3 - ax^2 + bx - 1 \]
- Initial State:
  \[ s_0 = 3, s_1 = a, s_2 = a^2 - 2b \]
- The sequence \( \{s_k\} \) is called a 3rd-order characteristic sequence.
- Period of this sequence is:
  \[ \text{Period} = Q|p^2 + p + 1 \]
- \( k^{th} \)-state of the LFSR is \( s_k = (s_{k, s_{k+1}}, s_{k+2}) \)
Reciprocal Sequence

- Given
  \[ f(x) = x^3 - ax^2 + bx - 1 \]
- The reciprocal polynomial is:
  \[ f^{-1}(x) = x^3 - bx^2 + ax - 1 \]
- Initial State:
  \[ s_0 = 3, s_1 = b, s_2 = b^2 - 2a \]
- Sequence generated by \( f^{-1}(x) \) is also a 3rd-order characteristic sequence

Example, \( p = 5 \)

Algorithms for Computing Sequence Terms

- Three algorithms for computing sequence term in a 3rd order characteristic sequence:
  - Dual-State Fast Evaluation Algorithm (DSEA)
    - To compute the \( s_{k-1} \) state and its dual
  - Computation of a Previous Sequence Term
    - To compute \( s_{z(k-1)} \) using \( (s_k, s_{k+1}) \) and it dual
  - Computation of Mixed term
    - To compute \( s_{zu(k+v)} \)

The DSEA Algorithm

- Binary rep of \( k \): \( k = \sum_{i=0}^{n} k_i 2^{n-i} = k_0 2^n + k_1 2^{n-1} + \cdots + k_n \)
- Let \( T_0 = k_0, T_j = k_j + 2T_{j-1}, T_n = k \) and \( t = T_{j-1}, t' = T_j \)
- For \( k_j = 0 \)
  \[ s_{j+1} = s_{j+2} - 2s_{j-1} \]
  \[ s_j = s_j - 2s_{j-1} \]
  \[ s_{j-1} = s_{j-2} - bs_{j-1} + s_{j-3} \]
- For \( k_j = 1 \)
  \[ s_{j+1} = s_{j+2} - 2s_{j-1} \]
  \[ s_j = s_j - 2s_{j-1} \]
  \[ s_{j-1} = s_{j-2} - bs_{j-1} + s_{j-3} \]
- Interchange \( a \) and \( b \) and find the \( (s_{j+1}, s_j, s_{j-1}) \) terms in sequence generated by the reciprocal polynomial which are the \( (s_{j+1}, s_j, s_{j-1}) \) terms in the original sequence
Computation of Previous Sequence Term

- The 3rd order characteristic sequence has properties of duality and redundancy.
- Three elements in any state of the sequence are not independent.
- If any two consecutive elements are known in the sequence, the third remaining one can be uniquely determined.

\[
\begin{align*}
(s_k, s_{k+1}) & \quad (s_{k-1}, s_k, s_{k+1}) \\
(s_{k-1}, s_k, s_{k+1}) & \quad (s_{k-2}, s_{k-1}, s_k) \\
(s_{k-2}, s_{k-1}, s_k) & \quad (s_{k-3}, s_{k-2}, s_{k-1})
\end{align*}
\]

Computation of Previous Sequence Term

- Given \((s_k, s_{k+1})\) and its dual
- Let \(\delta = s_{k+1}s_{-(k+1)} - s_1s_{-1}\)
- Then \(s_{2(k-1)}\) can be computed by:

\[
\begin{align*}
\delta_{k-1} &= \frac{es_{-(k+1)} + s_{-1}D(e)}{\delta} \\
D(s_k) &= s_{-k} \\
e &= -s_{-1}D(c_1) + c_2 \\
c_1 &= s_is_{k+1} - s_{-1}s_k \\
c_2 &= s_k^2 - 3s_{-k} + (b^2 - a)s_{-(k+1)}
\end{align*}
\]

Computation of Previous Sequence Term

- It is found that for any prime \(p\), \(s_p = a = s_{-(p+1)}\)
- \(s_{p+1} = b = s_{-p}\)
- ie, if \(k\) equals to either \(p-1\) or \(p\), the product \(s_{k+1}s_{-(k+1)}\) equals to \(a^2b\) for any prime \(p\). This restriction is added to the signing key selection.
- If either \(a\) or \(b\) is zero, \(k\) will be more likely yield a zero delta. Both \(a\) and \(b\) should be chosen as non-zero integers.
- As \(p\) increases, the percentage of delta that equals zero decreases.
- With the additional restrictions on \(a\) and \(b\) and the additional rules on \(k\), it is reasonable to reduce bandwidth usage as a way of minimizing cost.
Computation of Mixed Term $s\pm u(k+v)$ with Known $s_{k-1}$ State

- This algorithm is used to compute mixed terms $s\pm u(k+v)$ with known $(s_{k-1}, s_k, s_{k+1})$ and its dual.
- The procedure is as follows:
  1. Compute the sequence terms $s\pm u(k+v)$
  2. Construct another irreducible polynomial as:
    \[ g(x) = x^3 - s_{(k+v)}x^2 + s_{(k+v)}x - 1 \]
  3. Compute the $\pm u$th term of sequence generate by $g(x)$

Computation of $s\pm(k+v)$

- $s\pm(k+v)$ terms cannot be computed using the DSEA algorithm
- Instead, use a general result for LFSR sequence:

\[
\begin{align*}
\text{Transitional Matrix } & = A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & -b \\ 0 & 1 & a \end{bmatrix}, \quad \text{State Matrix } M_v = \begin{bmatrix} s_{v-2} & s_{v-1} & s_v \\ s_{v-1} & s_v & s_{v+1} \\ s_v & s_{v+1} & s_{v+2} \end{bmatrix} \\
\Sigma_{v-1} \cdot A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & -b \\ 0 & 1 & a \end{bmatrix} = (s_{v+1}, s_{v+2}, s_{v+3}) = \Sigma_{v+1} \\
\end{align*}
\]

Computation of $s\pm(k+v)$

- Procedure for finding $s_{(k+v)}$ term using $s_{k-1}$ state
  1. Construct Matrix $M_0$ and compute $M_0^{-1}$ — Can be done offline
  2. Compute $(s_{v-1}, s_v, s_{v+1})$ terms generated by $f(x)$ using DSEA algorithm
  3. Compute $s_{v-2}$ and $s_{v+2}$ terms and construct Matrix $M_v$
  4. Compute $s_{(k+v)}$ term by:

\[
\Sigma_{k-1} = \Sigma_{k-1} \cdot A^v = \Sigma_{k-1} \cdot (M_0^{-1} \cdot M_v)
\]

In particular, the $s_{(k+v)}$ term is equal to $s_{k-1}$ multiplied by the middle column of $(M_0^{-1} \cdot M_v)$
Computation of $s_{\pm(k+v)}$

- Consider $M_0$:
  $$M_0 = \begin{bmatrix} s_{-2} & s_{-1} & s_0 \\ s_{-1} & s_0 & s_1 \\ s_0 & s_1 & s_2 \end{bmatrix} = \begin{bmatrix} b^2 - 2a & b & 3 \\ b & 3 & a \\ 3 & a & a^2 - 2b \end{bmatrix}$$

- Determinant of $M_0$ is:
  $$\det(M_0) = (b^2 - 2a)\left[3(a^2 - 2b) - a^2\right] - b[3(a^2 - 2b) - 3a] + 3(ab - 3 \cdot 3) \neq 0$$

- It depends only on $a$ and $b$. By choosing $a$ and $b$ correspondingly, $\det(M_0)$ can be guaranteed to be non-zero.

GH-DSA

- ElGamal-like signature algorithm

- System Public Parameters:
  Irreducible polynomial $f(x) = x^3 - ax^2 + bx - 1$ over $GF(p)$, where $p$ is a prime number, with period $Q$

- Alice:
  Private Key: Choose $x$, with $0 < x < Q$ and $\gcd(x, Q) = 1$
  Public Key: The $s_{\pm x}$ terms generate by $f(x)$

GH-DSA: Signing Process

- Signing Process:
  1. Randomly choose $k$, with $0 < k < Q$ and $\gcd(k, Q) = 1$. Use DSEA algorithm to compute $(s_{k-1}, s_k, s_{k+1})$ and its dual such that $\gcd(s_k, Q) = 1$. $r = s_k$
  2. Compute $h = h(m)$, where $h()$ is a hash function
  3. Solve for $t$ in the signing equation: $h \equiv xr + kt \mod Q$  
    $\Rightarrow t \equiv k^{-1}(h - rx) \mod Q$

- $(r, t)$ is a digital signature of the message $m$. Alice sends Bob $(m, r, t)$ together with $(s_k, s_{k+1})$ and its dual.

GH-DSA: Verification Process

- Signing Equation $h \equiv xr + kt \mod Q$
- But, Bob can verify Sequence Terms
- Verification Process:
  1. Compute $s_{\pm(k-1)}$ using $(s_k, s_{k+1})$ and its dual
     If $\gcd(t, Q) = 1 \Rightarrow$ Case 1, else $\Rightarrow$ Case 2
  2. **Case 1**: $s_{\pm x} = s_{\pm x - 1} - (k-t) = s_{\pm x - 1} - (k - t - v)$
     1. Compute $u = -r^{-1}t \mod Q$, $v = -ht^{-1} \mod Q$
     2. Compute mixed terms $s_{u(k+v)}$
     3. Verify sequence terms $s_{\pm x} = s_{u(k+v)}$

- $x$ and $k$ are unknown to Bob
GH-DSA: Verification Process

- Case 2: \( s_{\pm k} = s_{\pm (h-rt)} = s_{\pm (hr^{-1}+x)} = s_{\pm u(x+v)} \)
  1. Compute \( u = -r \mod Q, v = -hr^{-1} \mod Q \)
  2. Compute mixed terms \( s_{\pm u(x+v)} \)
  3. Compute \( s_{\pm k} \)
  4. Verify sequence terms \( s_{\pm k} = s_{\pm u(k+v)} \)

 Software Design

- Signing Process:
  1. Input private key \( x \). Compute \( (s_x, s_{-x}) \).
  2. Input signing key \( k \). Compute \( (s_{k-1}, s_k, s_{k+1}) \) and its dual. If gcd\((s_k, Q) \neq 1\), prompt user to choose another \( k \).
  3. If delta = 0, prompt user to pick another \( k \).
  4. Input message \( m \) to be signed. Assume \( h = m \).
  5. \( r = s_k \). Compute \( t \).
  6. If \( t = 0 \), prompt user to choose another \( k \).
  7. Output \( (s_x, s_{-x}), (m, r, t), (s_k, s_{k+1}) \) and its dual

- Verification Process:
  1. Input \( (m, r, t), (s_x, s_{-x}), (s_k, s_{k+1}) \) and its dual.
  2. Compute \( s_{\pm (k-1)} \).
  3. If gcd\((t, Q) = 1\), then
     i. \( u = -r^{-1} t \mod Q, v = -ht^{-1} \mod Q \)
     ii. Compute \( s_{\pm u(k+v)} \)
     iii. Verify if \( s_{\pm u(k+v)} = s_{\pm x} \)
  4. If gcd\((t, Q) \neq 1\), then
     i. \( u = -r \mod Q, v = -hr^{-1} \mod Q \)
     ii. Compute \( s_{\pm u(x+v)} \)
     iii. Compute \( s_{\pm k} \)
     iv. Verify if \( s_{\pm u(x+v)} = s_{\pm k} \)
  5. Accept signature if sequence terms match

Design Issues

- Storage requirement for recursive iteration in DSEA algorithm
  - Only six sequence terms need to be kept after each iteration.
- Delta cannot be zero
  - \( k \) cannot be either \( p - 1 \) or \( p \)
  - Both \( a \) and \( b \) should be non-zero integers
- Invertible Matrix \( M_0 \)
  - This can be guaranteed by choosing \( a \) and \( b \) correspondingly such that det\((M_0) \neq 0\).
An irreducible polynomial $f(x)$ over a small field is used to make it possible to verify all the sequence term computations including the ones in the intermediate steps.

Irreducible polynomial $f(x)$ over $\text{GF}(5)$:

$$f(x) = x^3 + x - 1$$

Period of this sequence, $Q$:

$$Q = 5^2 + 5 + 1 = 31$$

All sequence terms are compared to the sequence terms generated by the recursive formula:

$$s_{j+3} = as_{j+2} - bs_{j+1} + s_j, \text{ for } i=0,1,2,\ldots$$

Implement GH-PKC over $\text{GF}(p)$

For 1024-bit security level, a 341-bit $p$ is used. Parameters are chosen as:

$$p = 2524100142802065091319986475346620439442782528122381$$
$$640812816384384364195892628818440024729407595209291$$
$$a = 1009678462466634534373236165995478977791322864153207$$
$$1493304907762091482797330771799383971091115148708951$$
$$b = 2062160226441847598150245499542278481087087236598545$$
$$481740882935002939062370689540637392192938836162683$$
$$Q = 1647052193950202913767588849369624124585134956111$$
Testing – Real System

- NTL library is used in the implementation to handle the large numbers in C++.
- It is not practical to verify all the generated sequence terms as they are too big!
- Instead, two copies of the program are executed at the same time to simulate two users, Alice and Bob, to make sure the generated shared key pair is the same and the message can be successfully signed and verified.

Remarks

- The GH-DH key agreement protocol and the signature generation in the GH-DSA scheme are successfully implemented in C++ and are tested thoroughly.
- The implementation of signature verification is still in progress.
- Several restrictions on system parameters and signing key selection are added to the system to increase system efficiency.
- Implement GH-PKC on constrained devices, such as blackberry, to analyze performance.

References

- Papers on GH-PKC:
- NTL Library is available for download at: www.shoup.net/ntl