# What Price a Provably Secure Cipher?

Ming-Shing Chen

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
ccheng@cc.ee.ntu.edu.tw

February 9, 2010

# Acknowledgement

- This is a joint work with
  - Tien-Ren Chen
  - Chen-Mou Cheng
  - Chun-Hung Hsiao
  - Ruben Niederhagen
  - Dr. Bo-Yin Yang, Academia Sinica, Taiwan

# The Provably-secure $\texttt{QUAD}(q, n, r)$ Stream Cipher

- Proposed by Berbain, Gilbert, and Patarin in Eurocrypt 2006
- $P_i$'s, $Q_j$'s: randomly chosen, public quadratic polynomials

$$
\begin{aligned}
\text{State:}\quad & n\text{-tuple } \mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_q^n \\
\text{Output:}\quad & r\text{-tuple } (P_1(\mathbf{x}), P_2(\mathbf{x}), \ldots, P_r(\mathbf{x})) \\
\text{Update:}\quad & \mathbf{x} \leftarrow (Q_1(\mathbf{x}), Q_2(\mathbf{x}), \ldots, Q_n(\mathbf{x}))
\end{aligned}
$$

# A Graphical Depiction

$$\begin{tikzcd}
\mathbf{x}_0 \arrow[r] \arrow[d] & \mathbf{x}_1 = \mathbf{Q}(\mathbf{x}_0) \arrow[r] \arrow[d] & \mathbf{x}_2 = \mathbf{Q}(\mathbf{x}_1) \arrow[r] \arrow[d] & \mathbf{x}_3 = \mathbf{Q}(\mathbf{x}_2) \arrow[r] \arrow[d] & \cdots \\
\mathbf{y}_0 = \mathbf{P}(\mathbf{x}_0) & \mathbf{y}_1 = \mathbf{P}(\mathbf{x}_1) & \mathbf{y}_2 = \mathbf{P}(\mathbf{x}_2) & \mathbf{y}_3 = \mathbf{P}(\mathbf{x}_3) & \cdots
\end{tikzcd}$$

# Security of QUAD

- Main security theorem of QUAD
  - Breaking QUAD implies the capability to solve $n + r$ random quadratic equations in $n$ variables
- Generic $\mathcal{MQ}$ (Multivariate Quadratics) is NP-hard
  - $\mathcal{MQ}(q, n, n + r)$ = solve for $n$ variables from $n + r$ quadratic equations, all coefficients and variables in $\mathbb{F}_q$
  - All known algorithms have average time complexity $2^{an+o(n)}$ for $r/n$ = constant
    - Most also require exponential space

## Key Observation

- The same reduction carries over to polynomials of *arbitrary* degrees, e.g., cubics, quartics, . . . , *without any modifications*
  - So long as linear terms are dense to keep the same distribution under random linear forms
  - But polynomials with higher degrees have way too many coefficients to be practical!
    - Need to use sparse polynomials
    - Need a new security assumption

## $\mathcal{SMP}(q, d, n, m, (\eta_2, \ldots, \eta_d))$

- An instance **S** in $\mathcal{SMP}(q, d, n, m, (\eta_2, \ldots, \eta_d))$, the class of *sparse multivariate polynomials*, comprises
  - $m$ polynomials $(P_1(\mathbf{x}), P_2(\mathbf{x}), \cdots, P_m(\mathbf{x}))$ in $n$ variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$
  - Each $P_i$ is a degree-$d$ polynomial with exactly $\eta_j = \eta_j(n)$ nonzero degree-$j$ terms for each $2 \leq j \leq d$
  - The affine terms are random
- Obviously $\mathcal{SMP}$ contains $\mathcal{MQ}$
- Furthermore, solving $\mathcal{SMP}$ systems with reasonably many terms appears to be hard
  - Ample empirical evidence to support this conjecture

# SPELT, Generalization of QUAD

$$
\begin{tikzcd}
\mathbf{x}_0 \arrow[r] \arrow[d] & \mathbf{x}_1 = \mathbf{Q}(\mathbf{x}_0) \arrow[r] \arrow[d] & \mathbf{x}_2 = \mathbf{Q}(\mathbf{x}_1) \arrow[r] \arrow[d] & \mathbf{x}_3 = \mathbf{Q}(\mathbf{x}_2) \arrow[r] \arrow[d] & \cdots \\
\mathbf{y}_0 = \mathbf{P}(\mathbf{x}_0) & \mathbf{y}_1 = \mathbf{P}(\mathbf{x}_1) & \mathbf{y}_2 = \mathbf{P}(\mathbf{x}_2) & \mathbf{y}_3 = \mathbf{P}(\mathbf{x}_3) & \cdots
\end{tikzcd}
$$

1. $\mathbf{P}$,$\mathbf{Q}$ drawn from $\mathcal{SMP}$
2. Need to select good parameters, say for $q = 16$, $n = r$
   - For cubics, need $n = 144$ at least
   - For quartics, need $n = 108$ at least
   - Don't need too many terms
     - 10 cubic terms per equation already makes things hard

# Timing on 3 GHz Intel CPU

| Stream cipher | Cycles/byte | Throughput | Security |
|---|---|---|---|
| AES (Bernstein and Schwabe) | 9.2 | 2.61 Gbps | $\leq 2^?$ |
| $\texttt{SPELT}(16, 4, 32, 32, (10, 8, 5))$ | 1244 | 19.3 Mbps | $\leq 2^{152}$ |
| $\texttt{QUAD}(2, 160, 160)$ (BBG SAC 2006) | 2081 | 11.5 Mbps | $\leq 2^{140}$ |
| $\texttt{SPELT}(16, 4, 108, 108, (20, 15, 10))$ | 5541 | 4.3 Mbps | $\geq 2^{80}$ |
| $\texttt{SPELT}(2, 3, 208, 208, (480, 20))$ | 11744 | 2.0 Mbps | $\geq 2^{82}$ |
| $\texttt{QUAD}(2, 320, 320)$ (BBG SAC 2006) | 13646 | 1.8 Mbps | $\geq 2^{82}$ |

# Latest Development

- We learned how to launch better brute-force attacks
  - $O(2^n)$ rather than $O(2^{n+o(n)})$
  - Bad news for QUAD/SPELT because this means more variables and slower speed
- We learned how to program GPU
  - Can we make QUAD/SPELT usable in practice?

# Preliminary Performance Results

| Stream cipher | Cycles/byte | Throughput | |
|---|---|---|---|
| | | CPU | GPU |
| AES (BS; OBSC, FSE 2010) | 9.2 | 2.61 Gbps | 30.9 Gbps |
| SPELT(64, 4, 32, 32, (10, 8, 5)) | 1244 | 19.3 Mbps | |
| QUAD(2, 160, 160) (BBG SAC 2006) | 2081 | 11.5 Mbps | |
| SPELT(16, 4, 108, 108, (20, 15, 10)) | 5541 | 4.3 Mbps | |
| SPELT(2, 3, 208, 208, (480, 20)) | 11744 | 2.0 Mbps | |
| QUAD(2, 320, 320) (BBG SAC 2006) | 13646 | 1.8 Mbps | |
| SPELT(31, 4, 112, 112, (32, 16, 8)) | 624 | 36.3 Mbps | 784 Mbps |
| SPELT(2, 3, 224, 224, (448, 20)) | 3121 | 7.3 Mbps | 826 Mbps |
| QUAD(2, 320, 320) | 3701 | 6.1 Mbps | 2.6 Mbps |

## Concluding Remarks

- In the case of stream cipher, the cheapest price for provable security seems to be one or two orders of magnitude in terms of speed

## Thanks for Listening!

- Questions or comments?