KU LEUVEN


# Short Solutions to Nonlinear Systems of Equations

September 2017

**Alan Szepieniec** alan.szepieniec@esat.kuleuven.be
Bart Preneel bart.preneel@esat.kuleuven.be

KU Leuven, ESAT/COSIC

KU LEUVEN

(Digital Signatures From)


# Short Solutions to Nonlinear Systems of Equations

September 2017

**Alan Szepieniec** alan.szepieniec@esat.kuleuven.be
Bart Preneel bart.preneel@esat.kuleuven.be

KU Leuven, ESAT/COSIC

KU LEUVEN

~~(Digital Signatures From)~~


# Short Solutions to Nonlinear Systems of Equations

September 2017

**Alan Szepieniec** alan.szepieniec@esat.kuleuven.be
Bart Preneel bart.preneel@esat.kuleuven.be

KU Leuven, ESAT/COSIC

# Question

880 6 8

## Why do the signature and ciphertext size increase in post-quantum schemes?

Why do the signature and ciphertext size increase in post-quantum schemes? And is there any table or comparison as to how much this increase happens for each scheme type (lattice, multivariate, code-based, etc.)

`encryption` `signature` `post-quantum-cryptography`

asked Jan 5 at

typos
**354**

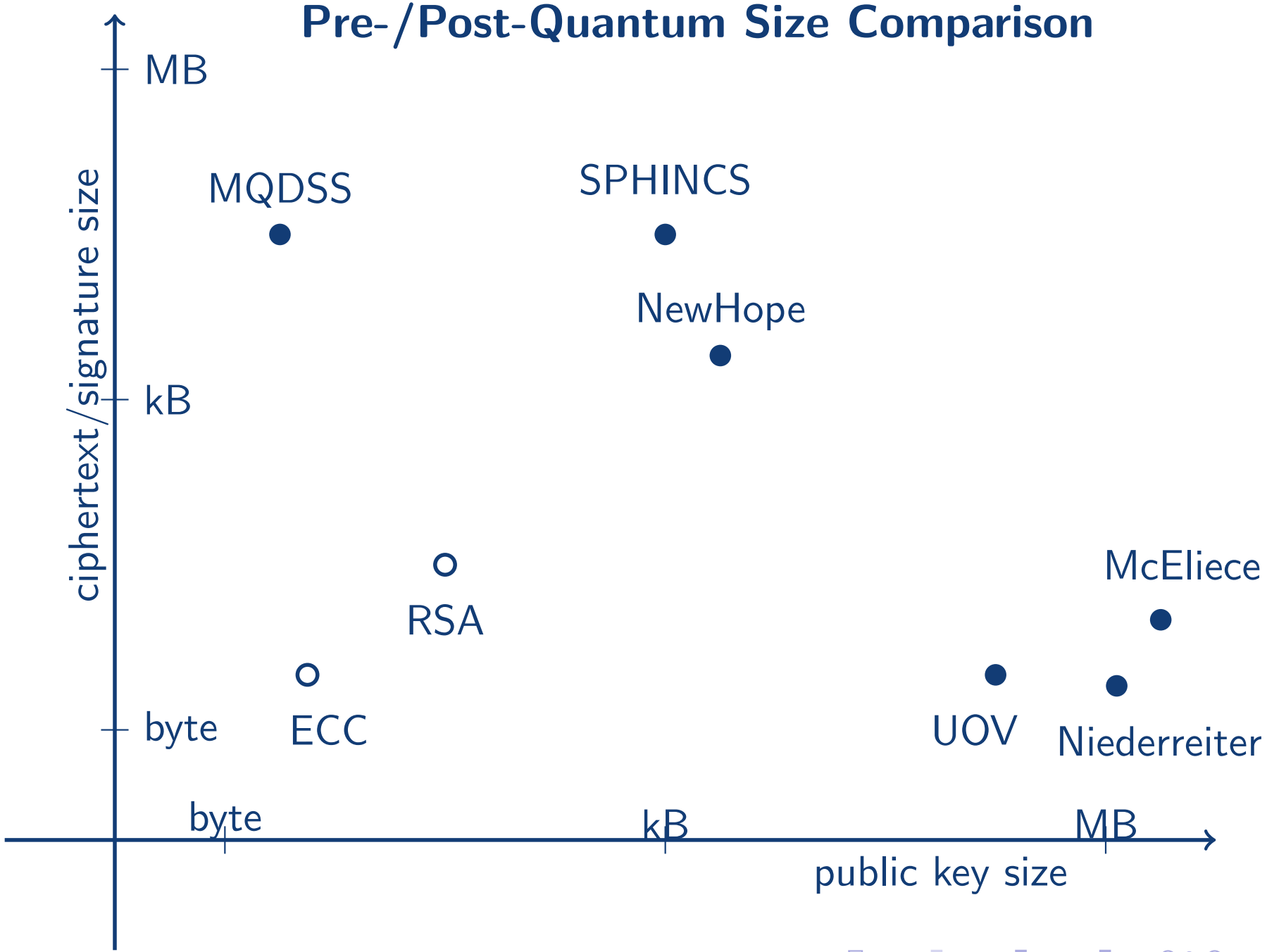
I'm really **not sure at all** about what I'm going to say, but I've heard about the so-called *Grover's quantum algorithm*, which can improve the efficiency of classical algorithms, and therefore improve classical attacks. This is a good forum to read about it: pqcrypto.org/pqc-forum/20161028-152555.txt – Daniel Jan 5 at 21:41

Increase compared to what? The message? Asymmetric schemes? I guess the best we can do is list signature and ciphertext sizes for specific strengths and a minimum amount of message size, say 256 bits for encryption and 512 bits for signature generation (enough for most hybrid systems). – Maarten Bodewes Jan 5 at 22:40

@SolidSnake That's why we can just increase the key size for symmetric cryptosystems. But, for public key crypto, Shor's algorithm breaks them in polynomial time, so people design new alternatives, but they usually have large signature size. – typos Jan 5 at 23:20

I know about that, what I do not know (like Maarten) is what are you comparing the sizes with? MPKC cryptosystems do not have that large ciphertext sizes in general – Daniel Jan 5 at 23:27

# Pre-/Post-Quantum Size Comparison



# Hard Problems

| scheme | type | hard problem | scales with |
|---|---|---|---|
| RSA/ECC | number theory | dlog/factoring | $\log_2 n \sim \kappa^c,\ \log_2 p \sim \kappa$ |
| Niederreiter | coding theory | syndrome decoding | $tm \times n \sim \kappa^2 \log_2 \kappa$ |
| UOV | MQ | MQ problem | $m \times n(n+1)/2 \sim \kappa^3$ |
| MQDSS | zero-knowledge | hiding/binding commit | $\kappa^2$ |
| NewHope | lattice | (R)LWE | $\dim(\mathcal{L}) \times \log_2 q \sim \kappa^c$ |
| SPHINCS | hash | MT-SPR | $\kappa^2$ |

# Hard Problems

| scheme | type | hard problem | scales with |
|---|---|---|---|
| RSA/ECC | number theory | dlog/factoring | $\log_2 n \sim \kappa^c,\ \log_2 p \sim \kappa$ |
| Niederreiter | coding theory | syndrome decoding | $tm \times n \sim \kappa^2 \log_2 \kappa$ |
| UOV | MQ | MQ problem | $m \times n(n+1)/2 \sim \kappa^3$ |
| MQDSS | zero-knowledge | hiding/binding commit | $\kappa^2$ |
| NewHope | lattice | (R)LWE | $\dim(\mathcal{L}) \times \log_2 q \sim \kappa^c$ |
| SPHINCS | hash | MT-SPR | $\kappa^2$ |
| **SSNE** (this talk) | **?** | **SSNE** | $\log_2 q \sim \kappa$ |

# Hard Problems

| scheme | type | hard problem | scales with |
|---|---|---|---|
| RSA/ECC | number theory | dlog/factoring | $\log_2 n \sim \kappa^c,\ \log_2 p \sim \kappa$ |
| Niederreiter | coding theory | syndrome decoding | $tm \times n \sim \kappa^2 \log_2 \kappa$ |
| UOV | MQ | MQ problem | $m \times n(n+1)/2 \sim \kappa^3$ |
| MQDSS | zero-knowledge | hiding/binding commit | $\kappa^2$ |
| NewHope | lattice | (R)LWE | $\dim(\mathcal{L}) \times \log_2 q \sim \kappa^c$ |
| SPHINCS | hash | MT-SPR | $\kappa^2$ |
| ~~**SSNE**~~ (this talk) | **?** | **SSNE** (this talk) | $\log_2 q \sim \kappa$ |

# Short Solutions to Nonlinear Equations

# Short Solutions to Nonlinear Equations

## Definition

**MQ Problem.**

Given: $\mathcal{P}(\mathbf{x}) \in (\mathbb{F}_q[\mathbf{x}^{\leq 2}])^m$

Find: $\mathbf{x} \in \mathbb{F}_q^n$

Such that: $\mathcal{P}(\mathbf{x}) = \mathbf{0}$.

# Short Solutions to Nonlinear Equations

## Definition

**MQ Problem.**

Given: $\mathcal{P}(\mathbf{x}) \in (\mathbb{F}_q[\mathbf{x}^{\leq 2}])^m$

Find: $\mathbf{x} \in \mathbb{F}_q^n$

Such that: $\mathcal{P}(\mathbf{x}) = \mathbf{0}$.

## Definition

**SIS Problem.**

Given: $A \in \mathbb{F}_q^{m \times n}$

Find: $\mathbf{x} \in \mathbb{Z}^n$

Such that: $A\mathbf{x} = \mathbf{0} \mod q$ and $\|\mathbf{x}\|_2 \leq \beta$.

# Short Solutions to Nonlinear Equations

## Definition

**MQ Problem.**

Given: $\mathcal{P}(\mathbf{x}) \in (\mathbb{F}_q[\mathbf{x}^{\leq 2}])^m$

Find: $\mathbf{x} \in \mathbb{F}_q^n$

Such that: $\mathcal{P}(\mathbf{x}) = \mathbf{0}$.

## Definition

**SIS Problem.**

Given: $A \in \mathbb{F}_q^{m \times n}$

Find: $\mathbf{x} \in \mathbb{Z}^n$

Such that: $A\mathbf{x} = \mathbf{0} \mod q$ and $\|\mathbf{x}\|_2 \leq \beta$.

## Definition

**SSNE Problem.**

Given: $\mathcal{P} \in (\mathbb{F}_q[\mathbf{x}])^m$

Find: $\mathbf{x} \in \mathbb{Z}^n$

Such that: $\mathcal{P}(\mathbf{x}) = \mathbf{0} \mod q$ and $\|\mathbf{x}\|_2 \leq \beta$.

# SSNE — Intuition for Hardness

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis — fails to identify *short* solutions

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis — fails to identify *short* solutions
  - lattice reduction

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis — fails to identify *short* solutions
  - lattice reduction — fails to distinguish *solutions*

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis — fails to identify *short* solutions
  - lattice reduction — fails to distinguish *solutions*
  - best attack —

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis — fails to identify *short* solutions
  - lattice reduction — fails to distinguish *solutions*
  - best attack — *exhaustive search?*

# SSNE — Intuition for Hardness

- MQ
  - hard when $m \approx n$ are large
  - Gröbner basis + guessing
  - Buchberger / F4 / XL
- SIS
  - hard when $m < n$ and $q$ are large
  - lattice reduction + guessing
  - LLL / BKZ
- SSNE
  - choose $m < n$ small but $q$ huge
  - guessing — very unlikely to lead to short solution
  - Gröbner basis — fails to identify *short* solutions
  - lattice reduction — fails to distinguish *solutions*
  - best attack — *exhaustive search?*

$\Longrightarrow$ linear in search space $\sim q^n$

# Design Principle 0a and 0b

$$\dim(\mathsf{var}(\mathcal{P})) > 0$$

# Design Principle 0a and 0b

$$\dim(\mathrm{var}(\mathcal{P})) > 0$$

- kills algebraic attack using Gröbner basis algorithms (Buchberger/F4/XL)
- infeasible to list all solutions

# Design Principle 0a and 0b

$$\dim(\mathrm{var}(\mathcal{P})) > 0$$

- kills algebraic attack using Gröbner basis algorithms (Buchberger/F4/XL)
- infeasible to list all solutions

$$\deg(\mathcal{P}) > 1$$

# Design Principle 0a and 0b

$$\dim(\mathrm{var}(\mathcal{P})) > 0$$

- kills algebraic attack using Gröbner basis algorithms (Buchberger/F4/XL)
- infeasible to list all solutions

$$\deg(\mathcal{P}) > 1$$

- otherwise it is a linear algebra problem with short solutions
- quickly solved with lattice reduction

# Design Principle 1

$$\log_2 \beta \geq \kappa$$

# Design Principle 1

$$\log_2 \beta \geq \kappa$$

- kills Arora-Ge-type algebraic attack
  - add equations $\prod_{j=\lfloor -\beta \rfloor}^{\lceil \beta \rceil} (x_i - j) = 0$

# Design Principle 1

$$\log_2 \beta \geq \kappa$$

- kills Arora-Ge-type algebraic attack
  - add equations $\prod_{j=\lfloor -\beta \rfloor}^{\lceil \beta \rceil} (x_i - j) = 0$
- kills bit-expansion algebraic attack
  - treat bits in bit expansion of $x_0, \ldots, x_{n-1}$ as variables
  - add equations $x_i^{(j)}(1 - x_i^{(j)}) = 0$
  - add equations $x_i^{(j+1)} - 2x_i^{(j)} = 0$
  - add equations $x_i - \sum 2^j x_i^{(j)} = 0$

# Design Principle 2

$$n\log_2(q) - \kappa \geq n\log_2(\beta\sqrt{n})$$

# Design Principle 2

$$n\log_2(q) - \kappa \geq n\log_2(\beta\sqrt{n})$$

- makes guess+algebra infeasible
  - guess small variables until system is determined
  - solve using Gröbner basis algorithm
  - BUT: random guess destroys short solution wop.

# Design Principle 3

$$\log_2 \|\mathbf{x}\|^2/n \geq \log_2 q$$

# Design Principle 3

$$\log_2 \|\mathbf{x}\|^2/n \geq \log_2 q$$

- kills Coppersmith-type lattice attacks

$$\mathcal{P}(x_0, x_1) = 0 \quad \Leftrightarrow \quad \left( \begin{array}{|c|} \hline \\ \\ \hline \end{array} \right) \begin{pmatrix} x_0^2 \\ x_0 x_1 \\ x_1^2 \\ x_0 \\ x_1 \\ 1 \end{pmatrix} = 0$$

# Design Principle 4

$$m \log_2 q \geq \kappa$$

# Design Principle 4

$$m \log_2 q \geq \kappa$$

- makes restricted lattice-reduction infeasible
  - ignore higher order monomials;
  - just reduce submatrix associated with linear monomials
  - hope for the best

# Design Principle 4

$$m \log_2 q \geq \kappa$$

- makes restricted lattice-reduction infeasible
  - ignore higher order monomials;
  - just reduce submatrix associated with linear monomials
  - hope for the best
- and also naïve guessing

# Design Principle 5

- generalized shortness requirement: $\mathbf{x}^{\mathsf{T}} W \mathbf{x} \leq \beta^2$ for some $W \in \mathbb{Z}^{n \times n}$

$$\dim(\mathsf{var}(\mathcal{P})) \leq \mathsf{rank}(W + W^{\mathsf{T}})$$

# Design Principle 5

- generalized shortness requirement: $\mathbf{x}^{\mathsf{T}} W \mathbf{x} \leq \beta^2$ for some $W \in \mathbb{Z}^{n \times n}$

$$\dim(\mathrm{var}(\mathcal{P})) \leq \mathrm{rank}(W + W^{\mathsf{T}})$$

- $\mathrm{rank}(W + W^{\mathsf{T}}) \approx \#$ of variables that must be short
- $\dim(\mathrm{var}(\mathcal{P})) \approx \#$ degrees of freedom in choosing solutions

# Design Principle 6

$$\frac{\kappa}{2n} + \log_2 \beta \leq \frac{n - o + m}{n + 1} \log_2 q$$

# Design Principle 6

$$\frac{\kappa}{2n} + \log_2\beta \leq \frac{n - o + m}{n+1}\log_2 q$$

where

# Design Principle 6

$$\frac{\kappa}{2n} + \log_2 \beta \leq \frac{n - o + m}{n+1} \log_2 q$$

where

$$o = \max_i \{ i \mid \exists S \in \mathsf{GL}_n(\mathbb{F}_q) \,.\, \mathcal{P} \circ S \text{ is linear in } i \text{ variables} \}$$

# Design Principle 6

$$\frac{\kappa}{2n} + \log_2 \beta \leq \frac{n - o + m}{n+1} \log_2 q$$

where

$$o = \max_i \{ i \mid \underline{\exists S \in \mathsf{GL}_n(\mathbb{F}_q) \,.\, \mathcal{P} \circ S \text{ is linear in } i \text{ variables}} \}$$

$\longrightarrow \mathcal{P}$ is *$i$-reconcilable*

# Design Principle 6

$$\frac{\kappa}{2n} + \log_2\beta \leq \frac{n - o + m}{n+1}\log_2 q$$

where

$$o = \max_i \{i \mid \underbrace{\exists S \in \mathsf{GL}_n(\mathbb{F}_q) \,.\, \mathcal{P} \circ S \text{ is linear in } i \text{ variables}}_{\longrightarrow\ \mathcal{P} \text{ is } i\text{-reconcilable}}\}$$

- kills algebraic-lattice hybrid attack

# Algebraic-Lattice Hybrid Attack

- outputs vectors of length $O(q^{(n-o+m)/(n+1)})$
- in polynomial time (assuming $n, m$ are small)

# Algebraic-Lattice Hybrid Attack

- outputs vectors of length $O(q^{(n-o+m)/(n+1)})$
- in polynomial time (assuming $n, m$ are small)
- workflow:

# Algebraic-Lattice Hybrid Attack

- outputs vectors of length $O(q^{(n-o+m)/(n+1)})$
- in polynomial time (assuming $n, m$ are small)
- workflow:
  - UOV Reconciliation attack on $\mathcal{P}$
    $\rightarrow$ obtain $\mathcal{F}, S$ such that $\mathcal{P} = \mathcal{F} \circ S$
    $\rightarrow$ such that $\mathcal{F}$ is *linear* in $o$ variables

# Algebraic-Lattice Hybrid Attack

- outputs vectors of length $O(q^{(n-o+m)/(n+1)})$
- in polynomial time (assuming $n, m$ are small)
- workflow:
  - UOV Reconciliation attack on $\mathcal{P}$
    $\rightarrow$ obtain $\mathcal{F}, S$ such that $\mathcal{P} = \mathcal{F} \circ S$
    $\rightarrow$ such that $\mathcal{F}$ is *linear* in $o$ variables
  - choose short vinegar variables

# Algebraic-Lattice Hybrid Attack

- outputs vectors of length $O(q^{(n-o+m)/(n+1)})$
- in polynomial time (assuming $n, m$ are small)
- workflow:
  - UOV Reconciliation attack on $\mathcal{P}$
    $\rightarrow$ obtain $\mathcal{F}, S$ such that $\mathcal{P} = \mathcal{F} \circ S$
    $\rightarrow$ such that $\mathcal{F}$ is *linear* in $o$ variables
  - choose short vinegar variables
  - use LLL to find oil variables such that
    $\mathcal{F}(\mathbf{x}) = \mathbf{0}$ is satisfied, and
    $\sum_{i=n-o}^{n} (S^{-1})_{[:,i]} x_i$ is short ($\sim q^{(n-o+m)/(n+1)}$)

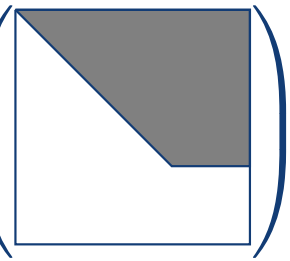
# Algebraic-Lattice Hybrid Attack

- outputs vectors of length $O(q^{(n-o+m)/(n+1)})$
- in polynomial time (assuming $n, m$ are small)
- workflow:
  - UOV Reconciliation attack on $\mathcal{P}$
    $\rightarrow$ obtain $\mathcal{F}, S$ such that $\mathcal{P} = \mathcal{F} \circ S$
    $\rightarrow$ such that $\mathcal{F}$ is *linear* in $o$ variables
  - choose short vinegar variables
  - use LLL to find oil variables such that
    $\mathcal{F}(\mathbf{x}) = \mathbf{0}$ is satisfied, and
    $\sum_{i=n-o}^{n} (S^{-1})_{[:,i]} x_i$ is short ($\sim q^{(n-o+m)/(n+1)}$)
  - solution: $\underbrace{\sum_{i=0}^{n-o-1} (S^{-1})_{[:,i]} x_i}_{\text{vinegar contribution}} + \underbrace{\sum_{i=n-o}^{n} (S^{-1})_{[:,i]} x_i}_{\text{oil contribution}}$

# UOV Signature Scheme

- public key: $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ (multivariate quadratic)
- secret key: $(\mathcal{F}, S)$ such that $\mathcal{P} = \mathcal{F} \circ S$ and such that $\mathcal{F}$ is *linear* in the *oil* variables $x_{n-o}, \ldots, x_{n-1}$

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^\mathsf{T} \left( \right) \mathbf{x}$$



# UOV Signature Scheme

- public key: $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ (multivariate quadratic)
- secret key: $(\mathcal{F}, S)$ such that $\mathcal{P} = \mathcal{F} \circ S$ and such that $\mathcal{F}$ is *linear* in the *oil* variables $x_{n-o}, \ldots, x_{n-1}$

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^\mathsf{T} \left( \begin{array}{cc} \blacksquare & \blacksquare \\ \blacksquare & \square \end{array} \right) \mathbf{x}$$

- verification: $\mathcal{P}(\mathbf{s}) \stackrel{?}{=} \mathsf{H}(d)$

# UOV Signature Scheme

- public key: $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ (multivariate quadratic)
- secret key: $(\mathcal{F}, S)$ such that $\mathcal{P} = \mathcal{F} \circ S$ and such that $\mathcal{F}$ is *linear* in the *oil* variables $x_{n-o}, \ldots, x_{n-1}$

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^\mathsf{T} \left( \begin{array}{cc} \blacksquare & \blacksquare \\ \blacksquare & \square \end{array} \right) \mathbf{x}$$

- verification: $\mathcal{P}(\mathbf{s}) \stackrel{?}{=} \mathsf{H}(d)$
- signature generation:

# UOV Signature Scheme

- public key: $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ (multivariate quadratic)
- secret key: $(\mathcal{F}, S)$ such that $\mathcal{P} = \mathcal{F} \circ S$ and such that $\mathcal{F}$ is *linear* in the *oil* variables $x_{n-o}, \ldots, x_{n-1}$

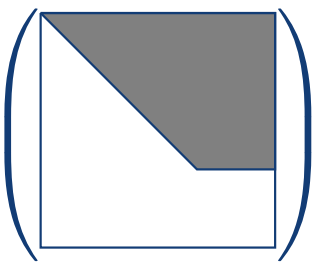
$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^\mathsf{T} \left( \begin{array}{cc} \blacksquare & \blacksquare \\ \blacksquare & \square \end{array} \right) \mathbf{x}$$

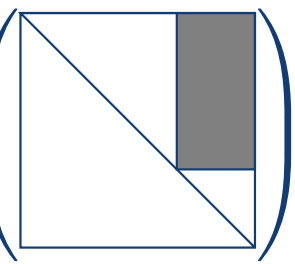
- verification: $\mathcal{P}(\mathbf{s}) \stackrel{?}{=} \mathsf{H}(d)$
- signature generation:
  - choose random *vinegar* variables $x_0, \ldots, x_{n-o-1} \xleftarrow{\$} \mathbb{F}_q$
  - $\mathcal{F}(\mathbf{x}) = \mathsf{H}(d)$ becomes *linear*, so solve
  - $\mathbf{s} \leftarrow S^{-1}\mathbf{x}$

# UOV Reconciliation Attack

- UOV Reconciliation Attack: *find* $S^{-1}$ *from* $\mathcal{P}$
  - treat coefficients of $S^{-1}$ as indeterminates
  - compute $\mathcal{F} = \mathcal{P} \circ S^{-1}$ symbolically
  - equate oil-times-oil coefficients to zero
  - solve system of quadratic equations

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^{\mathsf{T}} \left( \quad \right) \mathbf{x} \qquad S^{-1} = \left( \quad \right)$$

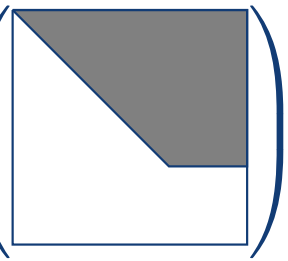


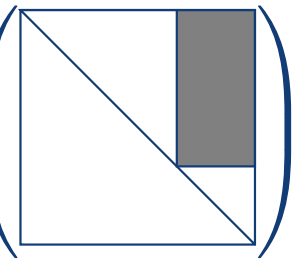


# UOV Reconciliation Attack

- UOV Reconciliation Attack: *find* $S^{-1}$ *from* $\mathcal{P}$
  - treat coefficients of $S^{-1}$ as indeterminates
  - compute $\mathcal{F} = \mathcal{P} \circ S^{-1}$ symbolically
  - equate oil-times-oil coefficients to zero
  - solve system of quadratic equations

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^{\mathsf{T}} \left( \quad \right) \mathbf{x} \qquad S^{-1} = \left( \quad \right)$$



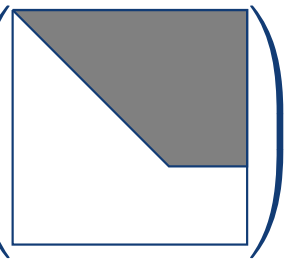


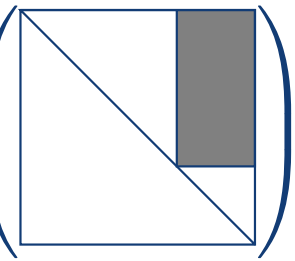
$$\#\text{eqns} = mo(o+1)/2 \qquad \#\text{vars} = (n-o)o$$

# UOV Reconciliation Attack

- UOV Reconciliation Attack: *find* $S^{-1}$ *from* $\mathcal{P}$
  - treat coefficients of $S^{-1}$ as indeterminates
  - compute $\mathcal{F} = \mathcal{P} \circ S^{-1}$ symbolically
  - equate oil-times-oil coefficients to zero
  - solve system of quadratic equations

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^{\mathsf{T}} \left( \quad \right) \mathbf{x} \qquad S^{-1} = \left( \quad \right)$$





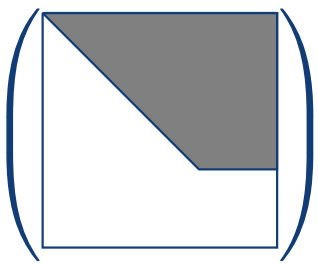
$$\#\text{eqns} = mo(o+1)/2 \qquad \#\text{vars} = (n-o)o$$

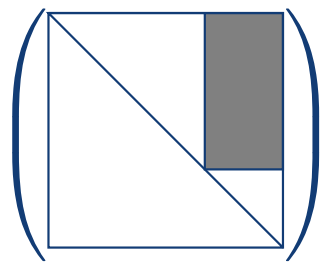
- solving system of quadratic equations is easy for small $m, n$

# UOV Reconciliation Attack

- UOV Reconciliation Attack: *find* $S^{-1}$ *from* $\mathcal{P}$
  - treat coefficients of $S^{-1}$ as indeterminates
  - compute $\mathcal{F} = \mathcal{P} \circ S^{-1}$ symbolically
  - equate oil-times-oil coefficients to zero
  - solve system of quadratic equations

$$\mathcal{F}_i(\mathbf{x}) = \mathbf{x}^\mathsf{T} \left( \quad \right) \mathbf{x} \qquad S^{-1} = \left( \quad \right)$$





$$\#\text{eqns} = mo(o+1)/2 \qquad \#\text{vars} = (n-o)o$$

- solving system of quadratic equations is easy for small $m, n$
- no solution in generic case when $\#\text{eqns} > \#\text{vars}$

# Small Vinegar Variables

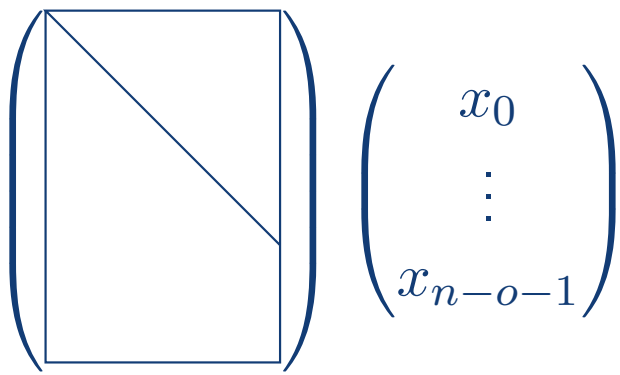
vinegar contribution:

$$\sum_{i=0}^{n-o-1} \left(S^{-1}\right)_{[:,i]} x_i$$

# Small Vinegar Variables

vinegar contribution:

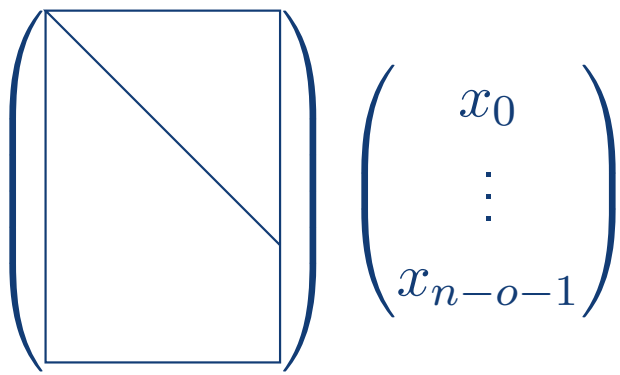
$$\sum_{i=0}^{n-o-1} \left(S^{-1}\right)_{[:,i]} x_i$$



# Small Vinegar Variables

vinegar contribution:

$$\sum_{i=0}^{n-o-1} \left(S^{-1}\right)_{[:,i]} x_i$$



- small vinegar variables guarantee a short vinegar contribution

## Small Oil Variables

- vinegar variables are fixed
- transform $\mathcal{F}(\mathbf{x}) = 0$ into $A\mathbf{x}_o = \mathbf{b}$ with $A \in \mathbb{F}_q^{m \times o}$

## Small Oil Variables

- vinegar variables are fixed
- transform $\mathcal{F}(\mathbf{x}) = 0$ into $A\mathbf{x}_o = \mathbf{b}$ with $A \in \mathbb{F}_q^{m \times o}$
- 1 particular solution $\mathbf{x}^{(p)}$
- $o - m$ kernel vectors $\mathbf{x}_i^{(k)}$

## Small Oil Variables

- vinegar variables are fixed
- transform $\mathcal{F}(\mathbf{x}) = 0$ into $A\mathbf{x}_o = \mathbf{b}$ with $A \in \mathbb{F}_q^{m \times o}$
- 1 particular solution $\mathbf{x}^{(p)}$
- $o - m$ kernel vectors $\mathbf{x}_i^{(k)}$
- find weights $1\mathbf{x}^{(p)} + a\mathbf{x}_0^{(k)} + \cdots + c\mathbf{x}_2^{(k)}$
- such that

$$1(S^{-1})_{[:,-]}\mathbf{x}^{(p)} + a(S^{-1})_{[:,-]}\mathbf{x}_0^{(k)} + \cdots + c(S^{-1})_{[:,-]}\mathbf{x}_2^{(k)}$$

  is short

## Small Oil Variables

- vinegar variables are fixed
- transform $\mathcal{F}(\mathbf{x}) = 0$ into $A\mathbf{x}_o = \mathbf{b}$ with $A \in \mathbb{F}_q^{m \times o}$
- 1 particular solution $\mathbf{x}^{(p)}$
- $o - m$ kernel vectors $\mathbf{x}_i^{(k)}$
- find weights $1\mathbf{x}^{(p)} + a\mathbf{x}_0^{(k)} + \cdots + c\mathbf{x}_2^{(k)}$
- such that

$$1(S^{-1})_{[:,-]}\mathbf{x}^{(p)} + a(S^{-1})_{[:,-]}\mathbf{x}_0^{(k)} + \cdots + c(S^{-1})_{[:,-]}\mathbf{x}_2^{(k)}$$

  is short
- use LLL to find short vector in lattice spanned by

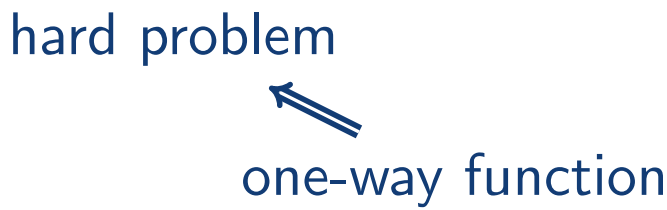
$$\begin{pmatrix} (S^{-}1)_{[:,-]}\mathbf{x}^{(p)} \\ 1 \end{pmatrix}, \begin{pmatrix} (S^{-1}_{[:,-]}\mathbf{x}_0^{(k)} \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} (S^{-1})_{[:,-]}\mathbf{x}_2^{(k)} \\ 0 \end{pmatrix}$$

# Map of Public Key Crypto

# Map of Public Key Crypto

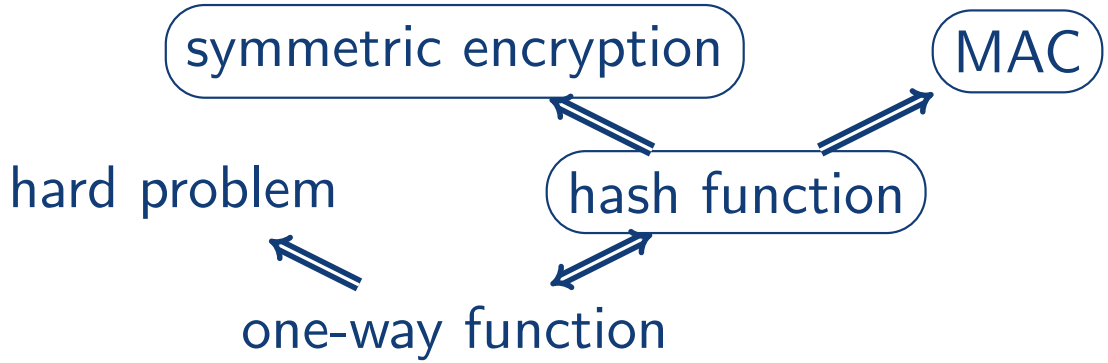
hard problem

# Map of Public Key Crypto

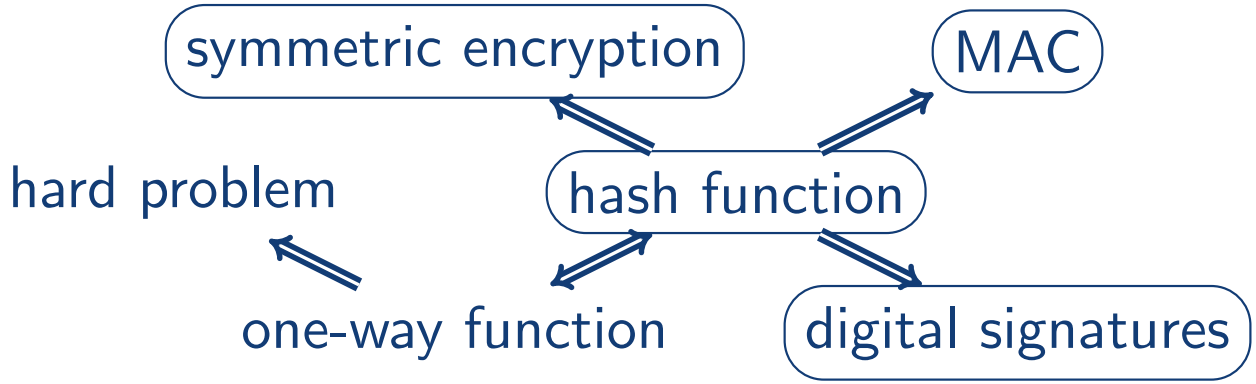


# Map of Public Key Crypto

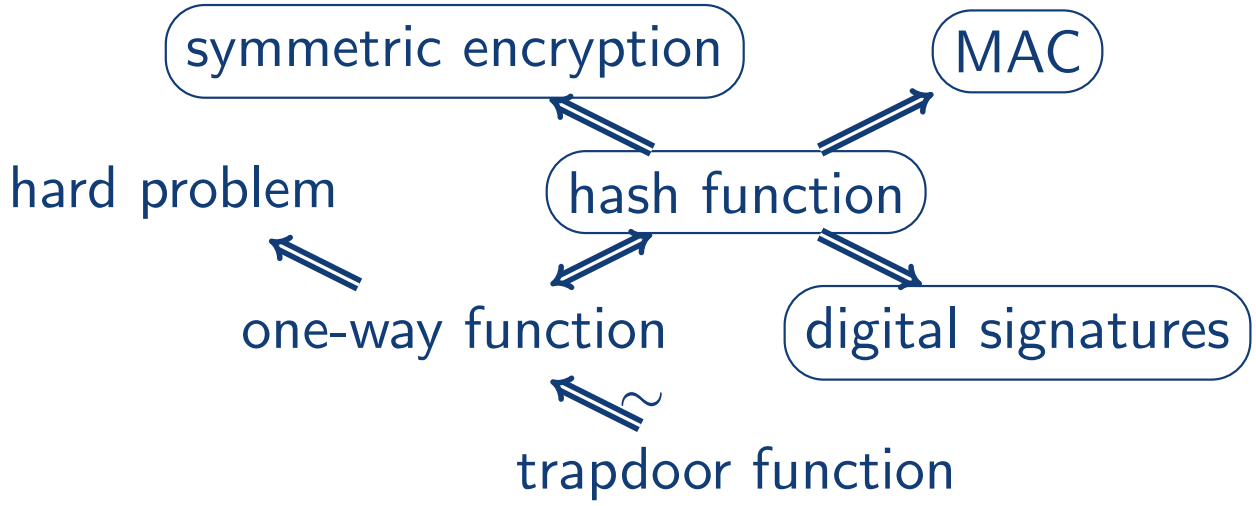
# Map of Public Key Crypto



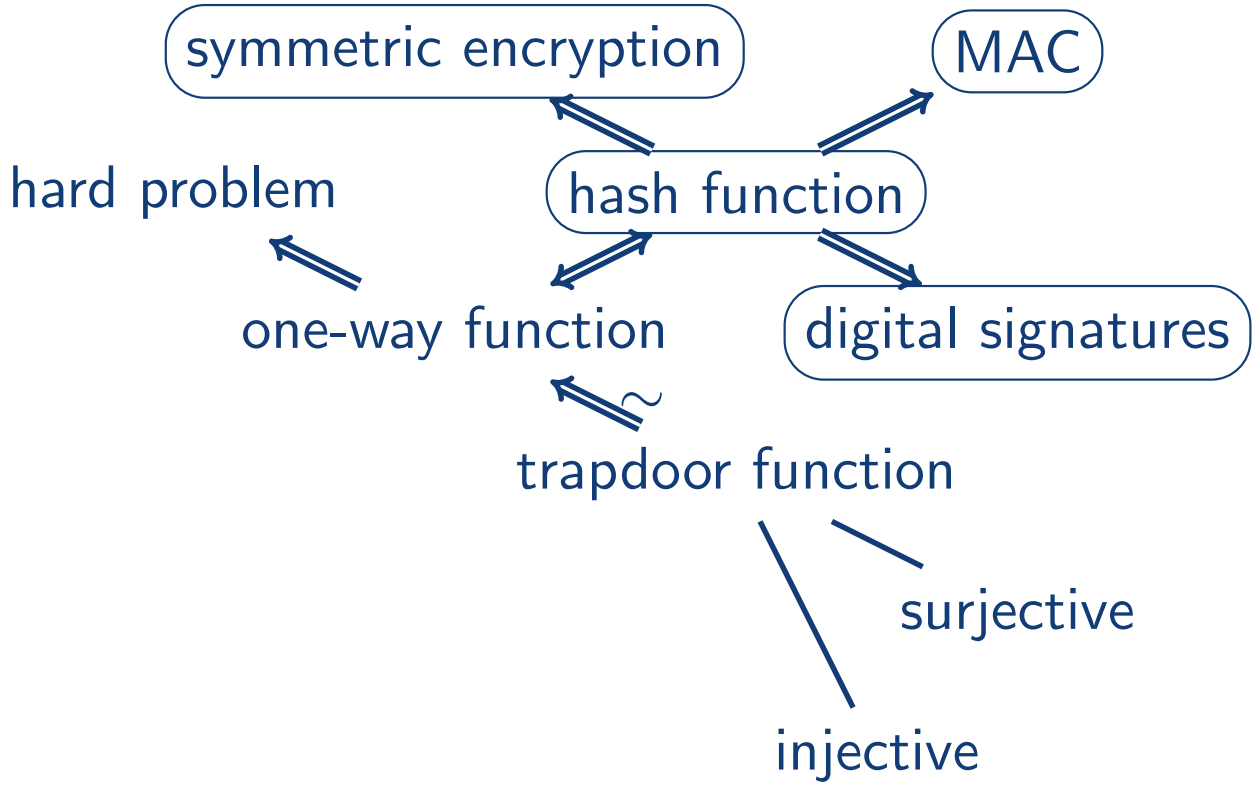
# Map of Public Key Crypto



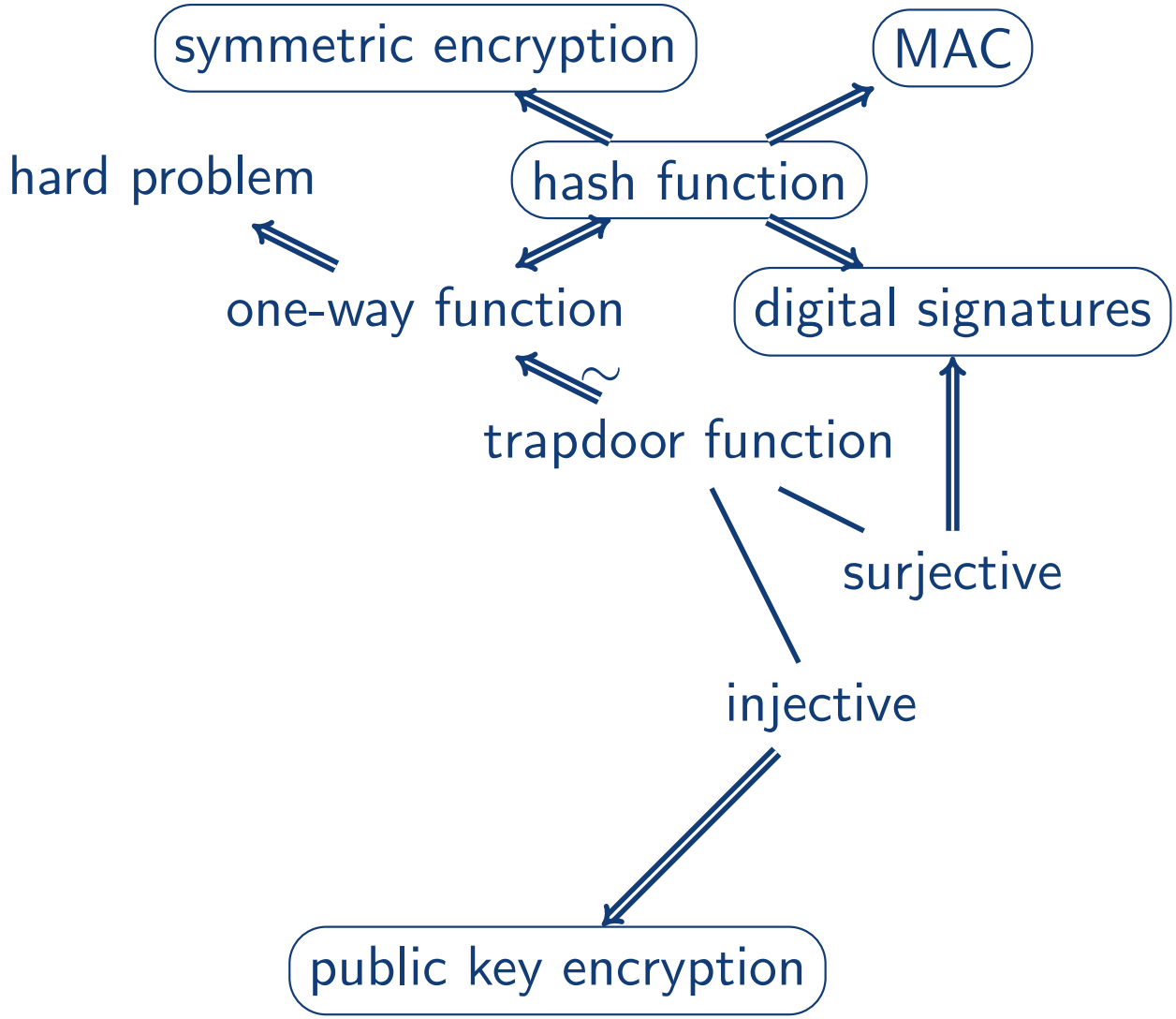
# Map of Public Key Crypto



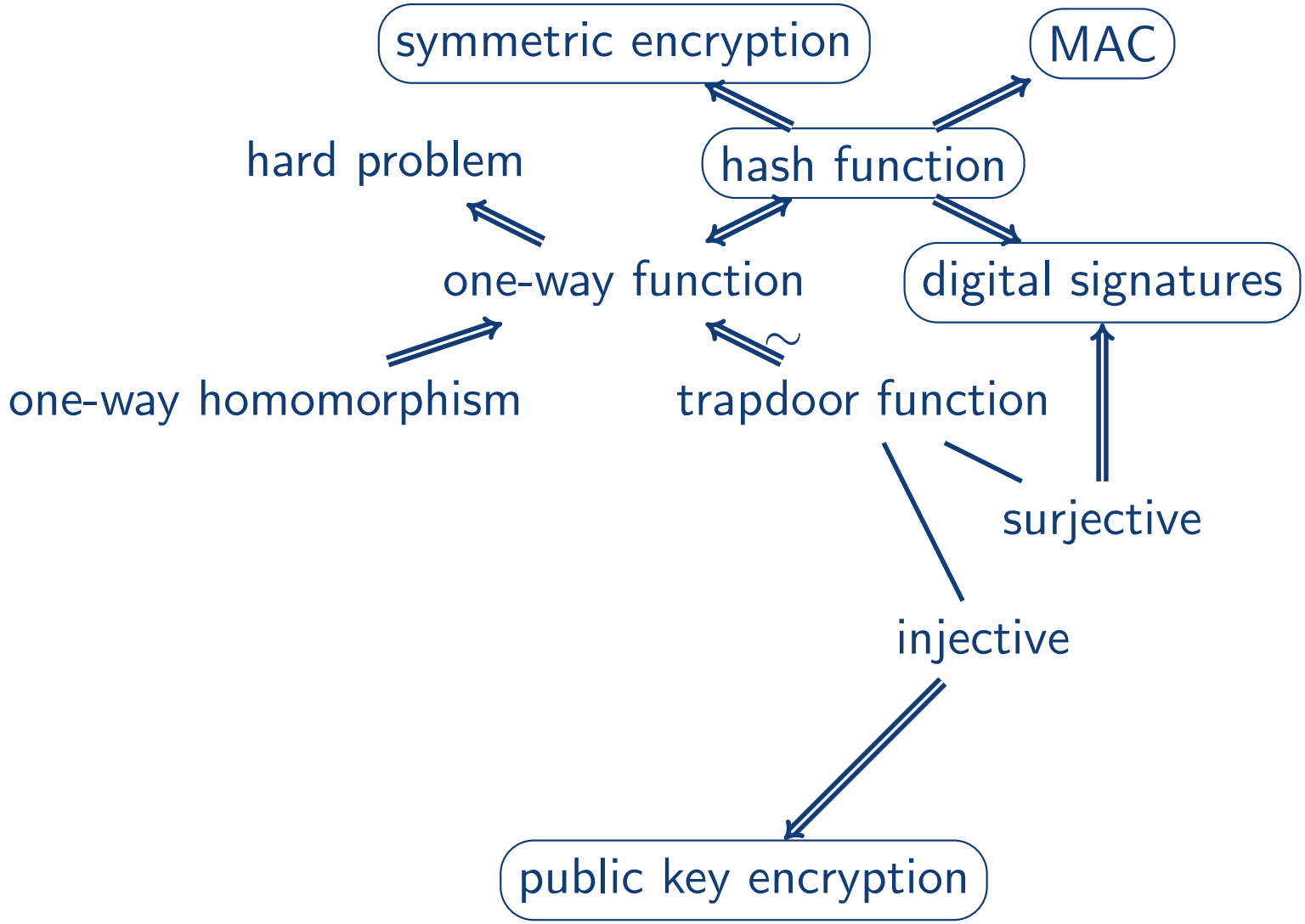
# Map of Public Key Crypto



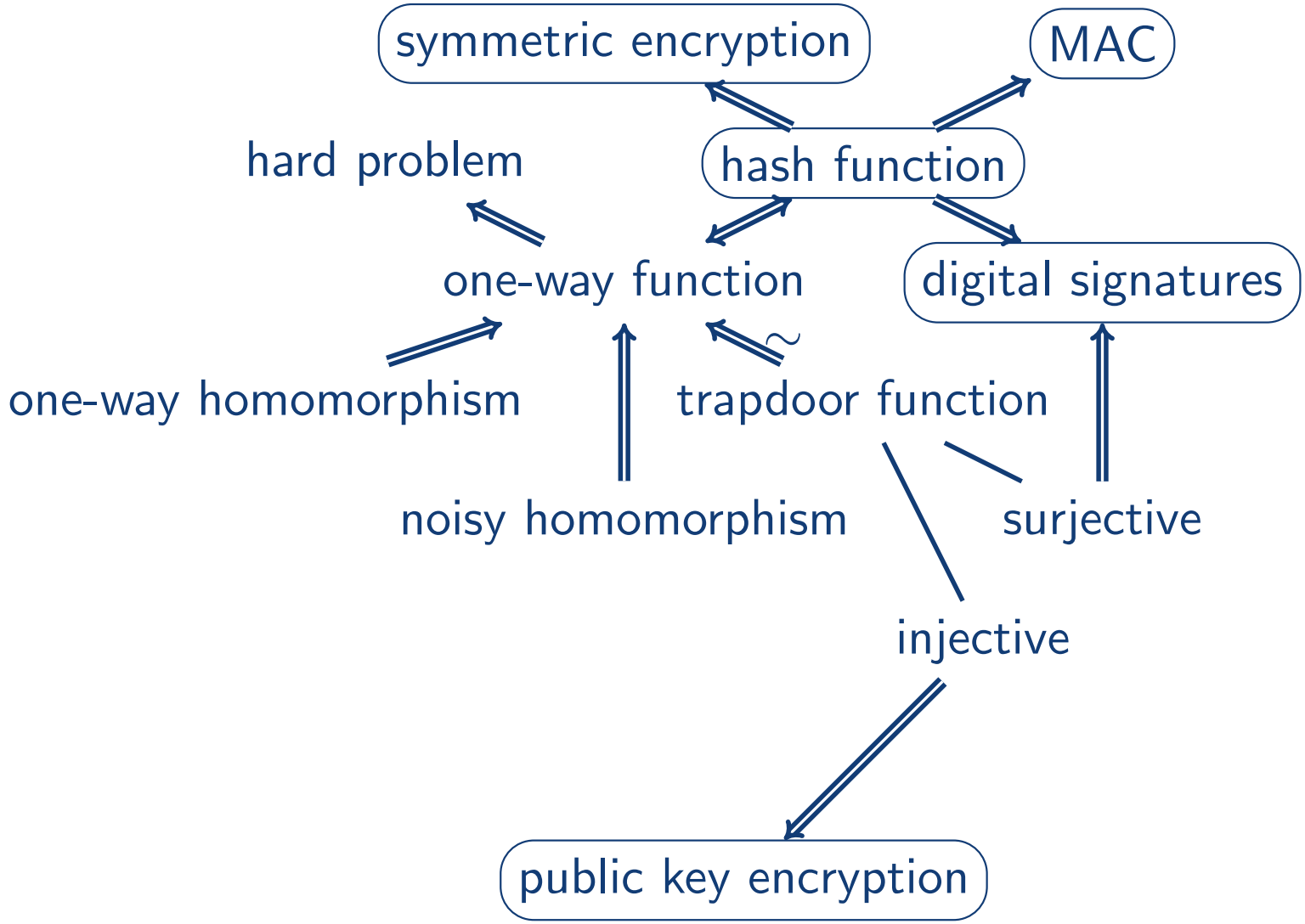
# Map of Public Key Crypto



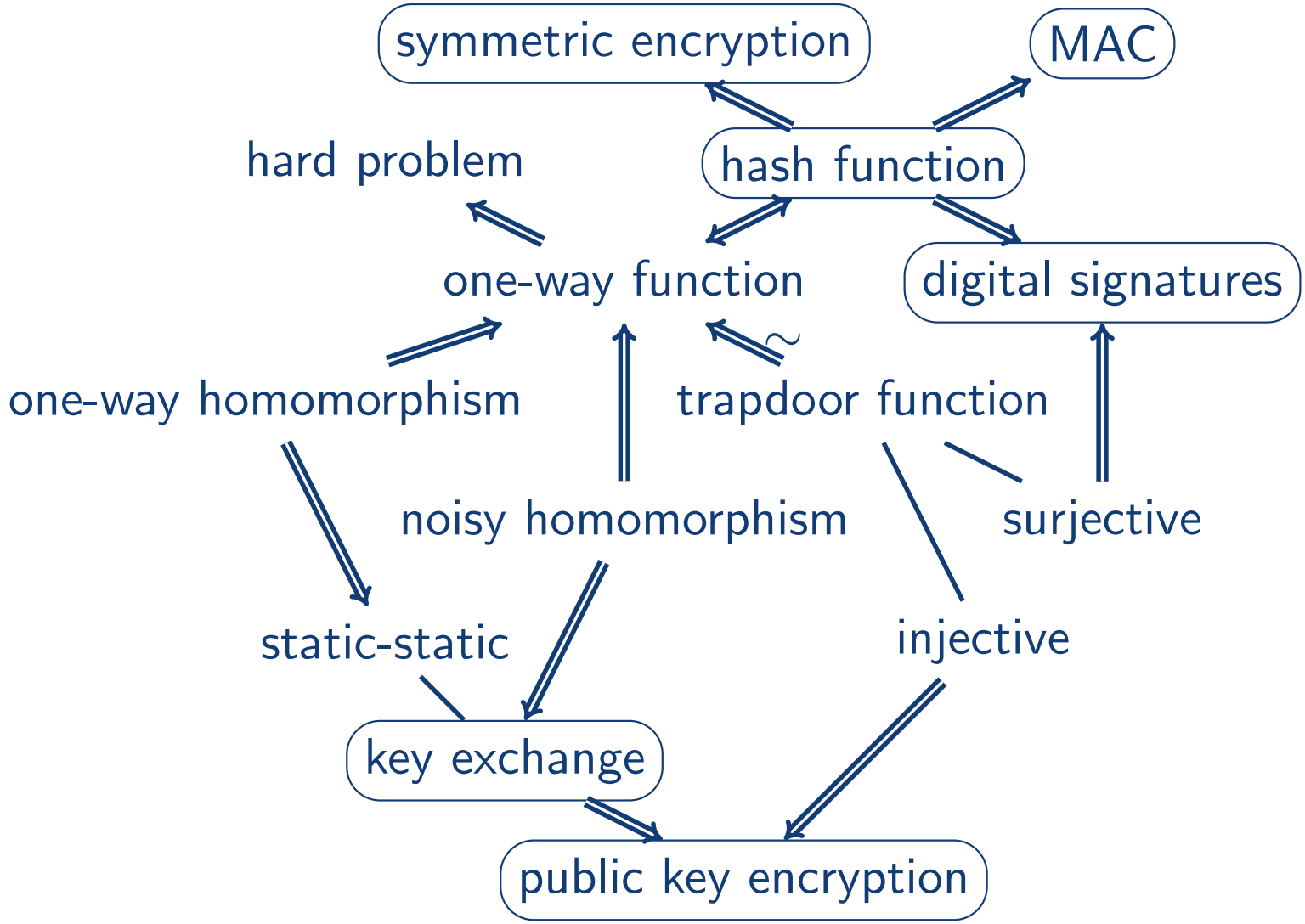
# Map of Public Key Crypto



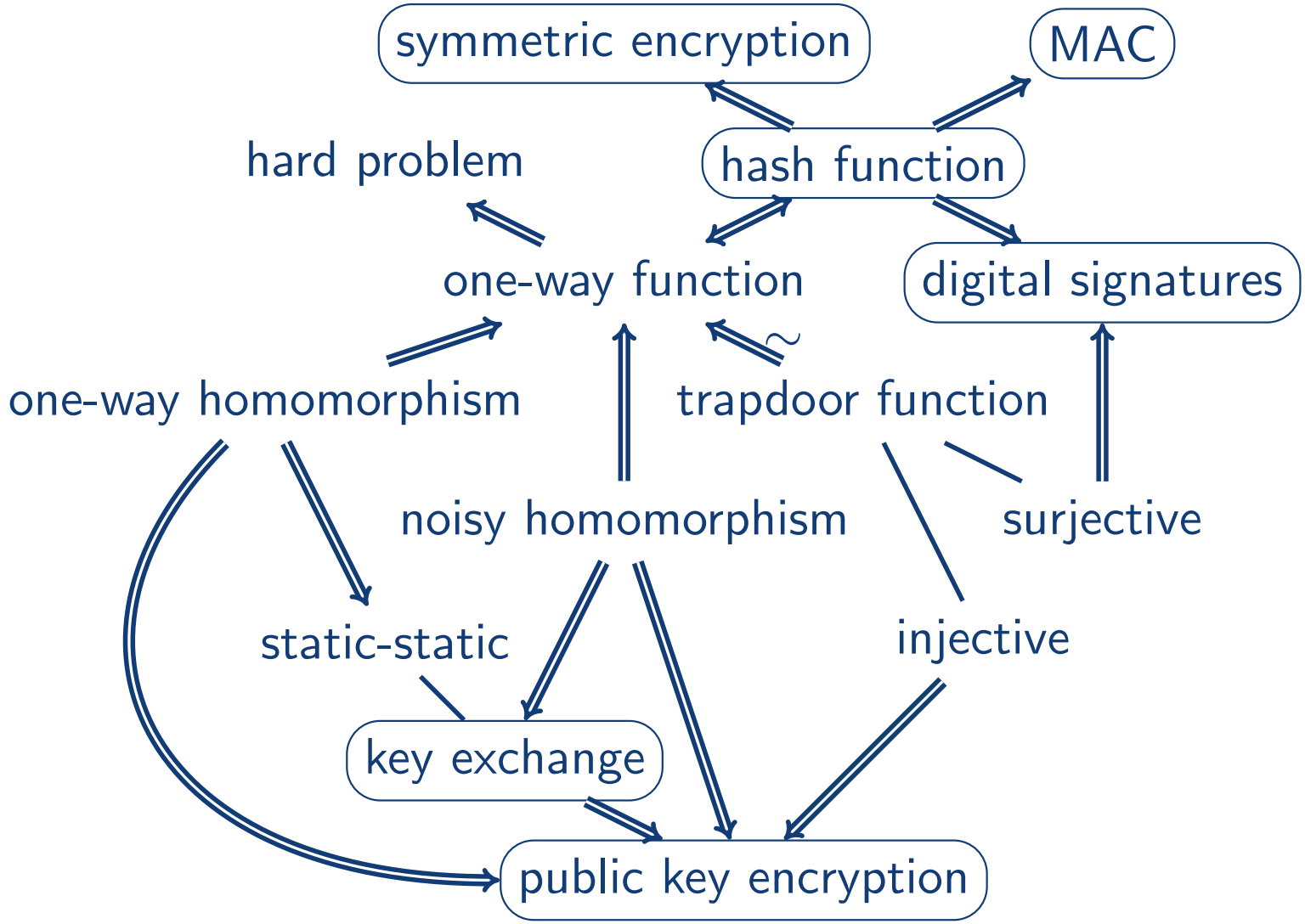
# Map of Public Key Crypto



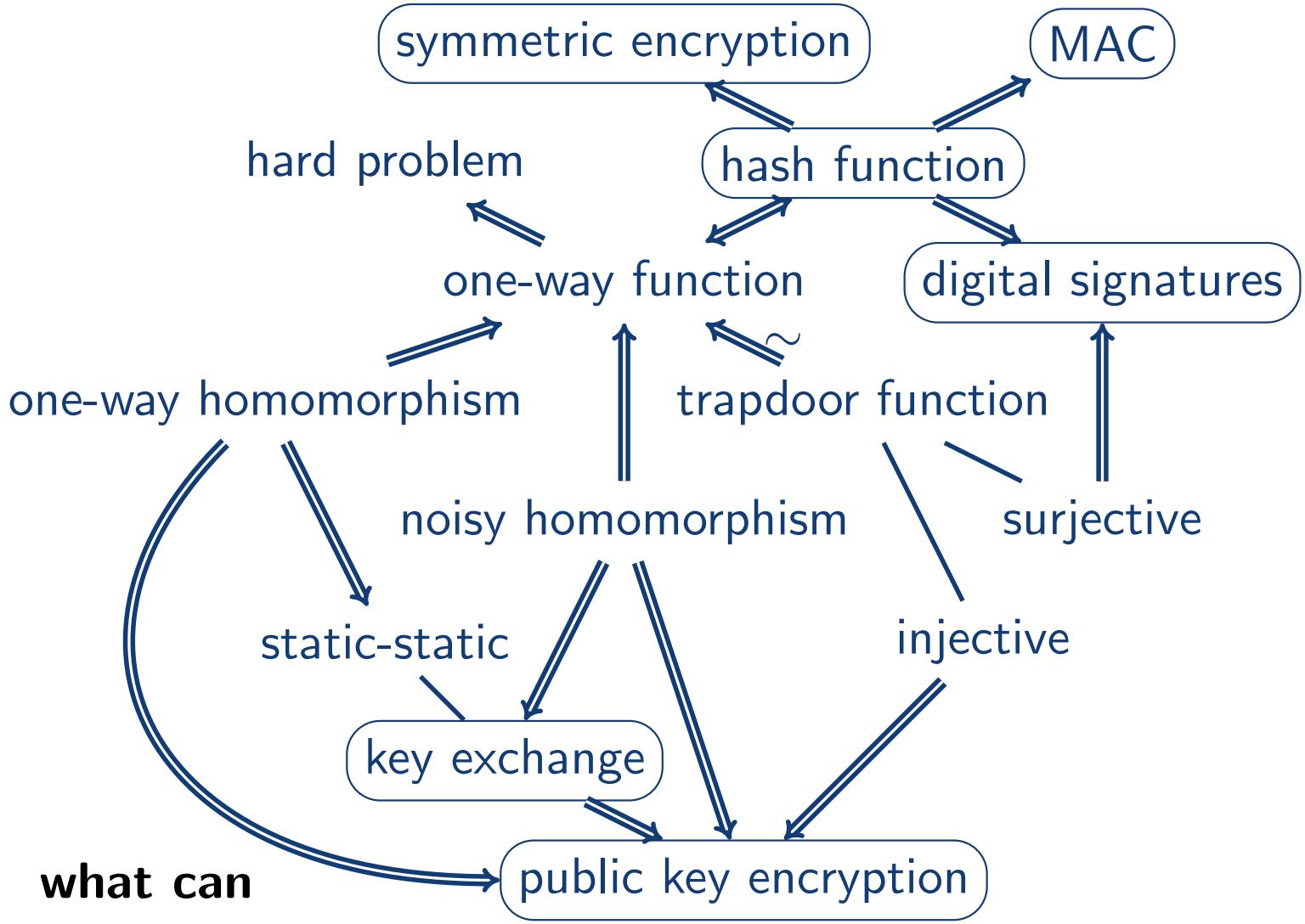
# Map of Public Key Crypto



# Map of Public Key Crypto

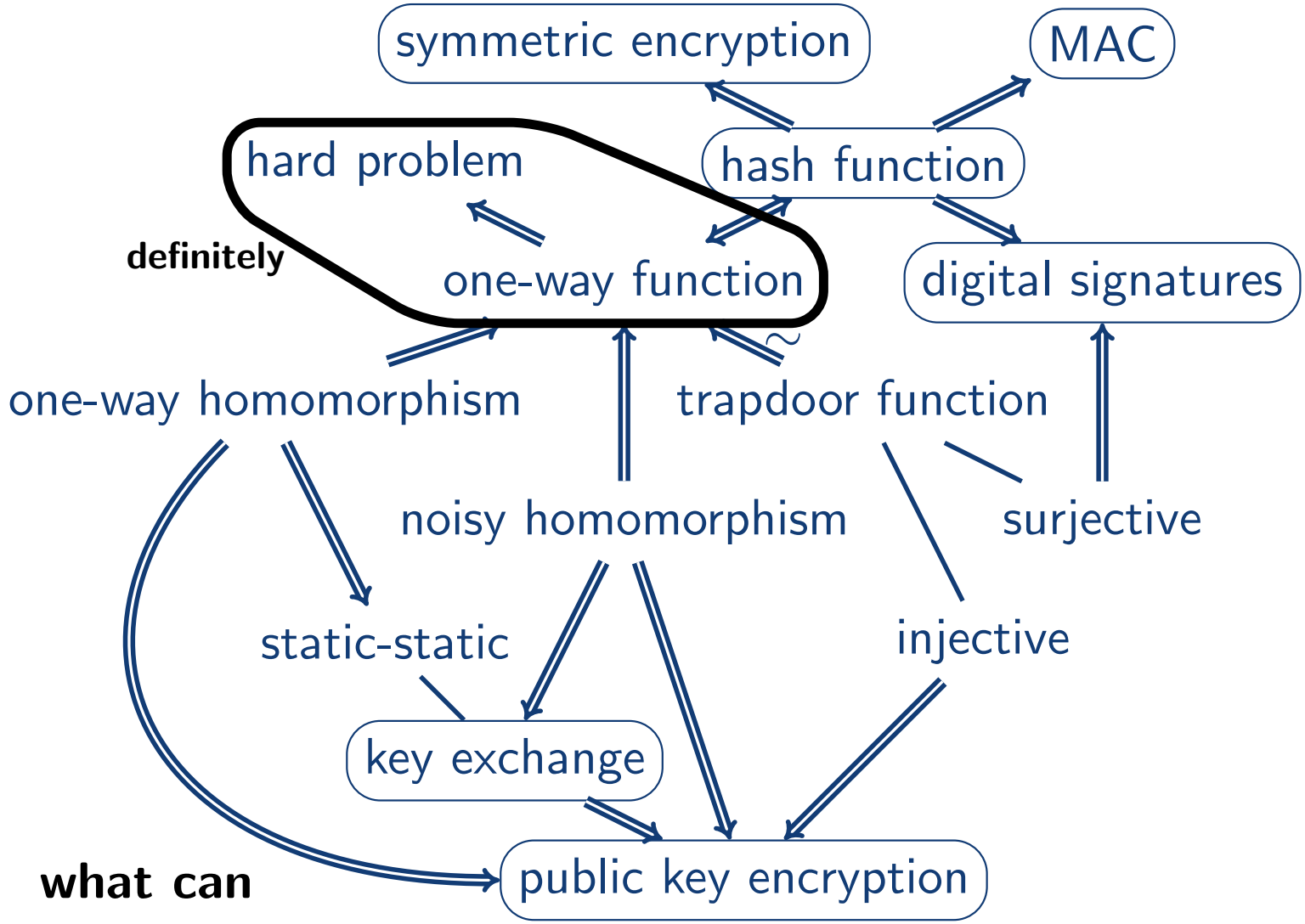


# Map of Public Key Crypto

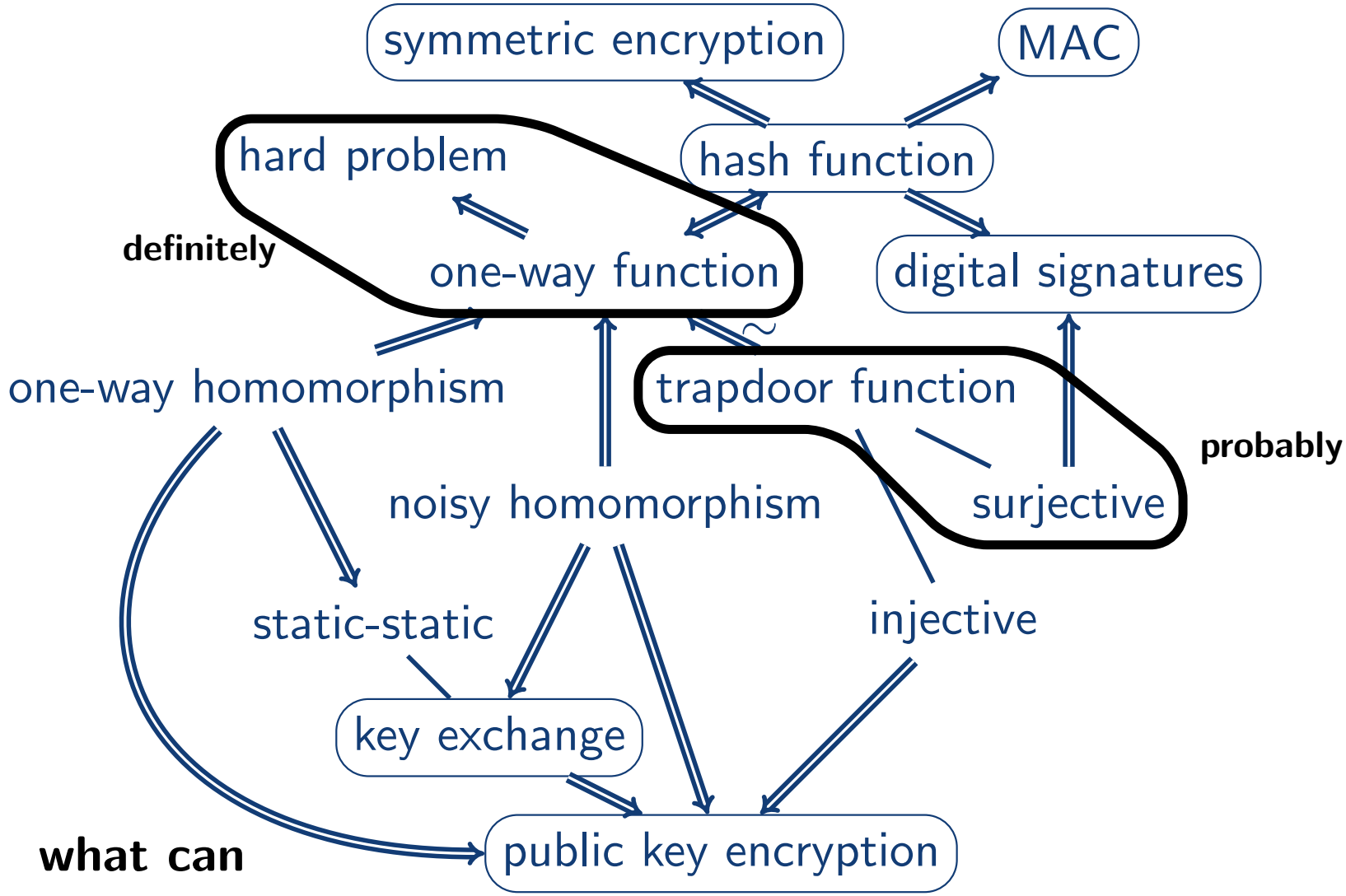



**what can SSNE do?**

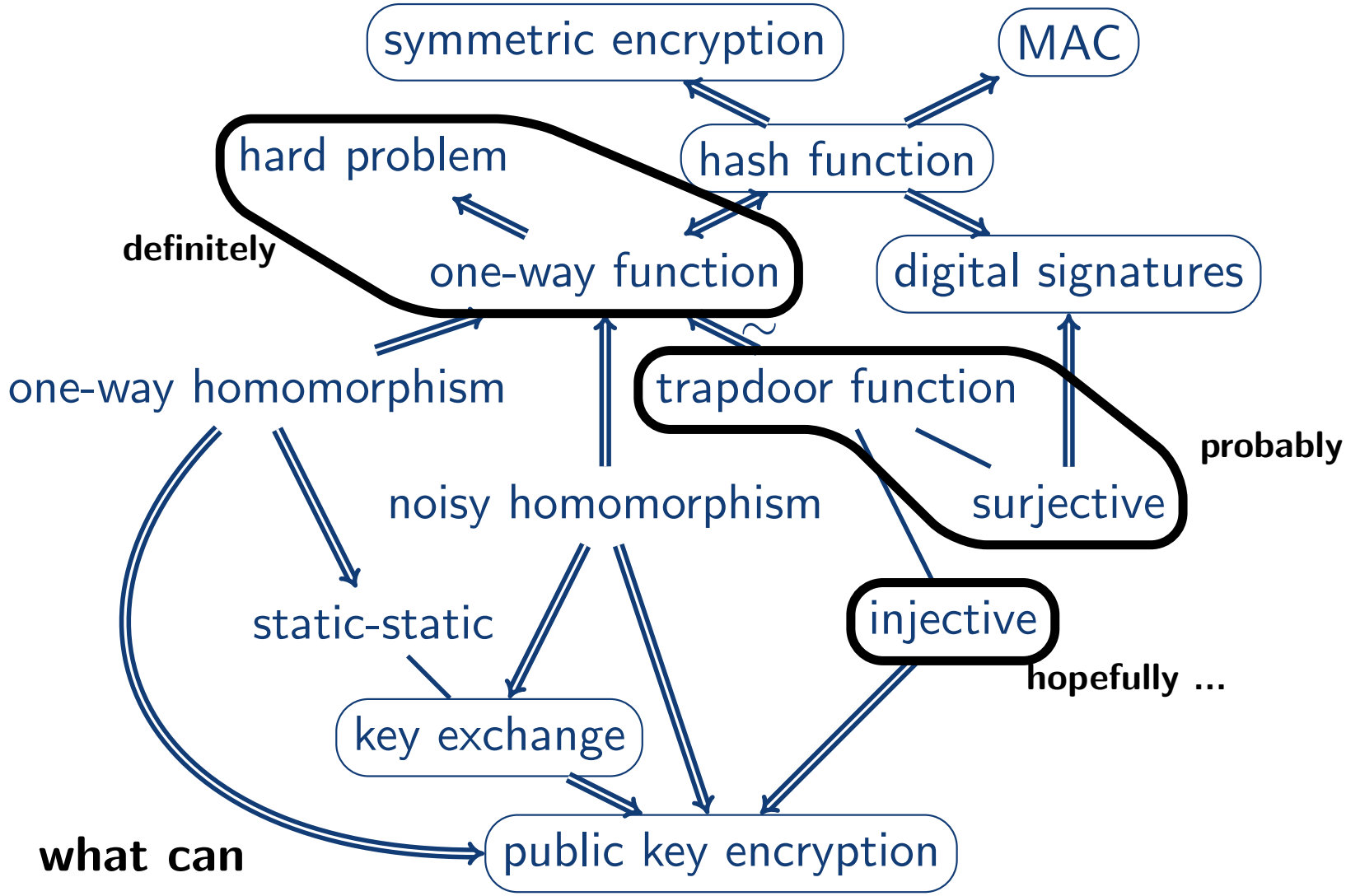
# Map of Public Key Crypto



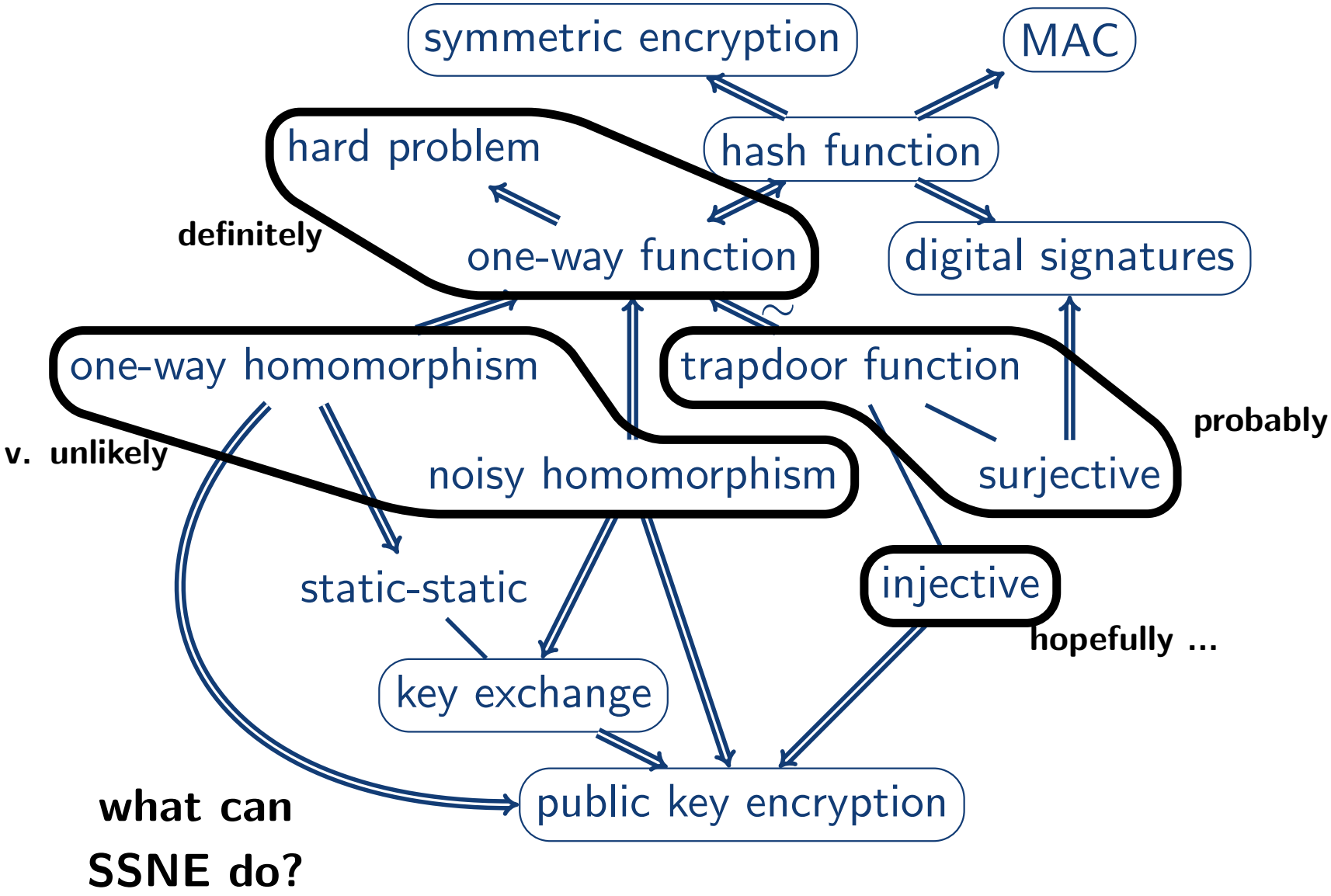
# Map of Public Key Crypto



# Map of Public Key Crypto



# Map of Public Key Crypto




**what can SSNE do?**

# Summary

- SSNE $\approx$ SIS $+$ MQ

# Summary

- SSNE $\approx$ SIS + MQ
- 6 design principles $\rightarrow$ brute force $\rightarrow$ scales well

# Summary

- SSNE $\approx$ SIS + MQ
- 6 design principles $\rightarrow$ brute force $\rightarrow$ scales well
- design principle 6: algebra+lattice hybrid attack

# Summary

- SSNE $\approx$ SIS + MQ
- 6 design principles $\rightarrow$ brute force $\rightarrow$ scales well
- design principle 6: algebra+lattice hybrid attack
- hash function ✓

# Summary

- SSNE $\approx$ SIS + MQ
- 6 design principles $\rightarrow$ brute force $\rightarrow$ scales well
- design principle 6: algebra+lattice hybrid attack
- hash function ✓ (signature x)

# Summary

- SSNE $\approx$ SIS $+$ MQ
- 6 design principles $\rightarrow$ brute force $\rightarrow$ scales well
- design principle 6: algebra+lattice hybrid attack
- hash function ✓ (signature x)
- co-design hard problem $\leftrightarrow$ public key functionality