Outline

THE CASCADE

THEorie du **C**ontrôle **A**ppliquée à la **S**ynchronisation des **C**ommunic**A**tions **D**iscr**E**tes

AGENCE NATIONALE DE LA RECHERCHE ANR

AIRBUS DEFENCE & SPACE

CRAN

LIASD LABORATOIRE INFORMATIQUE AVANCÉE DE SAINT-DENIS

Identity

ANR	Programme INS Edition 2013	Projet : THE CASCADE DOCUMENT SCIENTIFIQUE

Acronyme **Acronym**	THE CASCADE
Titre du projet	THEorie du Contrôle Appliquée à la Synchronisation des CommunicAtions DiscrEtes
Proposal title	Control theory for synchronization issues in private communications
Axe(s) thématiques/ theme(s)	1. Sécurité et sûreté des systèmes numériques
Type of research	■ Basic Research ☐ Industrial Research ☐ Experimental Development
Aide totale demandée: 334361 € **Grant request**	**Durée du projet** : 42 months **Project duration**
Partenaire coordinateur **Coordinator partner/**	MILLERIOUX GILLES Centre de Recherche en Automatique de Nancy (CRAN UMR CNRS 7039) Université de Lorraine

Classes of ciphers

- ciphers
  - public–key
  - private–key
    - block ciphers
    - stream ciphers
      - synchronous
      - self–synchronous

Issues and related WP of THE CASCADE

- WP1: Synchronization (resp: Gilles Millérioux (CRAN))
- WP2: Security (resp: Philippe Guillot (LAGA))
- WP3: Hardware-oriented issues (resp: Julien Francq (ADS))

People

People involved in the project

Surname	Name	Current Position	PM	Contribution to the project
Partner 1 (CRAN)				
Boukhobza	Taha	PU Université de Lorraine	7	control theory, graph-oriented approaches for characterizing the self-synchronization
Collin	Floriane	MdC Université de Lorraine	7	control theory, elimination theory, identifiability for algebraic attacks
Millerioux	Gilles	PU Université de Lorraine	20	Main coordinator Scientific manager of Task 1 control, synchronization, hybrid systems
Partner 2 (LAGA)				
Guillot	Philippe	MdC Université Paris 8	20	Scientific manager of Task 2 security, Booleans functions, cryptography
Phan	Hieu Duong	MdC Université Paris 8	11	provable security
Partner 3 (LIASD)				
El Mrabet	Nadia	MdC Université Paris 8	11	cryptography, side-channel attacks, embedded systems, arithmetic
Partner 4 (ADS)				
Francq	Julien	Head of Security, Mathematics, Applied Cryptography and Hardware Team in ADSC	11	Scientific manager of Task 3 expertise in cryptography and SCADA systems

Table 1: Summary of involved researchers

CRAN (UMR 7039): Centre de Recherche en Automatique de Nancy (Partner 1)

LAGA (UMR 7539): Laboratoire Analyse, Géométrie et Applications (Partner 2)

LIASD (EA 4383): Laboratoire d'Informatique Avancée de Saint-Denis (Partner 3)

ADS: Airbus Defence & Space - Cybersecurity (Partner 4)

Applications of control theory

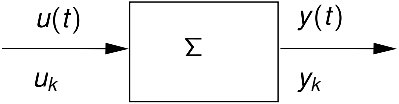
Models of linear dynamical systems

(Input/output models in continuous-time - Differential equations)

$$a_0 y(t) + a_1 \frac{dy(t)}{dt} + \ldots + \frac{d^n y(t)}{dt^n} = b_0 u(t) + b_1 \frac{du(t)}{dt} + \ldots + b_m \frac{d^m u(t)}{dt^m}$$

(Input/output models in discrete-time - Difference equations)

$$a_0 y_k + a_1 y_{k+1} + \ldots + y_{k+n} = b_0 u_k + b_1 u_{k+1} + \ldots + b_m u_{k+m}$$



Models of linear dynamical systems

(State space models)

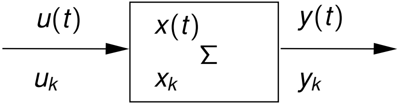
$x[1] = y,\, x[2] = \frac{dy}{dt}, \dots$ *or* $x_k[1] = y_k,\, x_k[2] = y_{k+1}, \dots$

Continuous time — *Discrete time*

$$\begin{cases} \frac{dx(t)}{dt} & = & Ax(t) + Bu(t) \\ y(t) & = & Cx(t) \end{cases} \qquad \begin{cases} x_{k+1} & = & Ax_k + Bu_k \\ y_k & = & Cx_k \end{cases}$$

$$\text{with } A = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & \dots \\ & & \dots & & \\ 0 & \dots & \dots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-1} \end{bmatrix},\ B = \begin{bmatrix} \vdots \\ 0 \\ b_m \\ \vdots \\ b_0 \end{bmatrix},\ C = [1\ 0\ \dots\ 0]$$

$$\Psi(\lambda) = det(A - \lambda\mathbf{1}) = \lambda^n + \dots + a_1\lambda + a_0$$



Models of nonlinear discrete-time dynamical systems

(Differential equations)

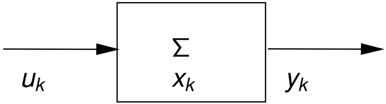
$$g(y_k, y_{k+1}, \ldots, y_{k+n}, [u_k, u_{k+1}, \ldots, u_{k+m}]) = 0$$

(State space equations)

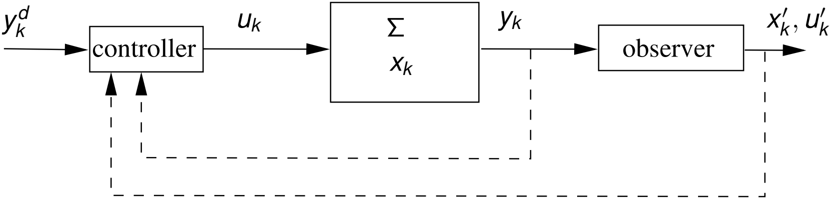
$$\left\{ \begin{array}{lcl} x_{k+1} & = & f(x_k, [u_k]) \\ y_k & = & h(x_k, [u_k]) \end{array} \right.$$

$$\begin{array}{l} f^1(u_k) \\ f^2(x_k[1]) \\ f^3(x_k[1], x_k[2]) \\ \vdots \\ f^n(x_k[1], \ldots, x_k[n]) \end{array}$$

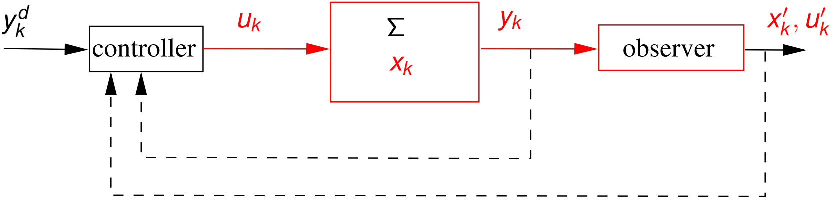
Triangular next-state transition function f



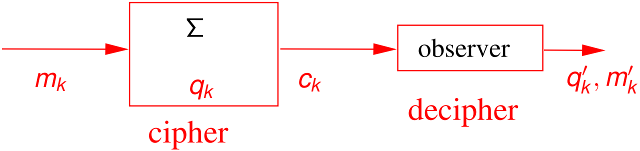
Connection between control theory and ciphering



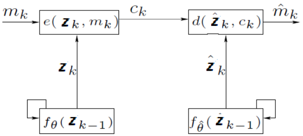
Connection between control theory and ciphering



Connection between control theory and ciphering

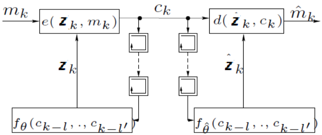


Stream ciphers and dynamical systems



Synchronizing Stream Cipher (SSC)

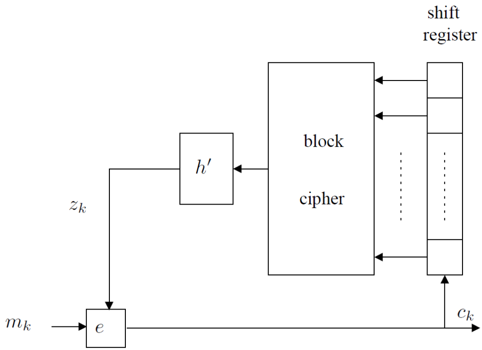
$$\begin{cases} \mathbf{z}_k = f_\theta(\mathbf{z}_{k-1}) \\ c_k = e(\mathbf{z}_k, m_k) \end{cases}.$$



Self Synchronizing Stream Cipher (SSSC)

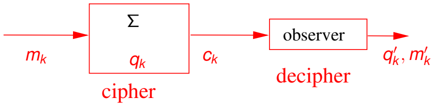
$$\begin{cases} \mathbf{z}_k = f_\theta(c_{k-l}, \ldots, c_{k-l'}) \\ c_k = e(\mathbf{z}_k, m_k) \end{cases}$$

Block cipher in CFB mode



$$\begin{cases} z_k &= \mathcal{F}_\theta(c_{k-1}, \ldots, c_{k-M}) \\ c_k &= e(z_k, m_k) \end{cases}$$

Maurer's approach (1991): use of finite input memory automata



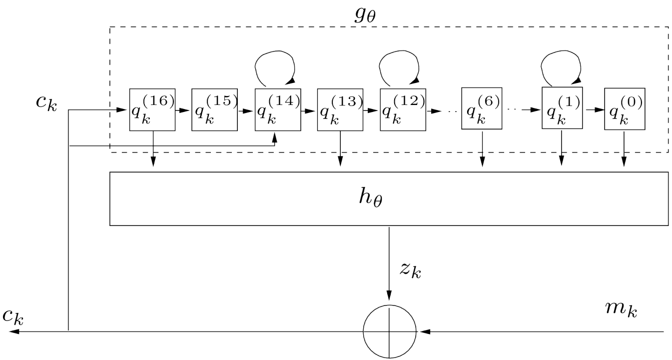
State equations
$$\left\{\begin{array}{lll} q_{k+1} & = & g_\theta(q_k, c_k) \\ z_k & = & h_\theta(q_k) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lll} q'_{k+1} & = & g_\theta(q'_k, c_k) \\ z'_k & = & h_\theta(q'_k) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If g_θ is triangular, for $k \geq M$,

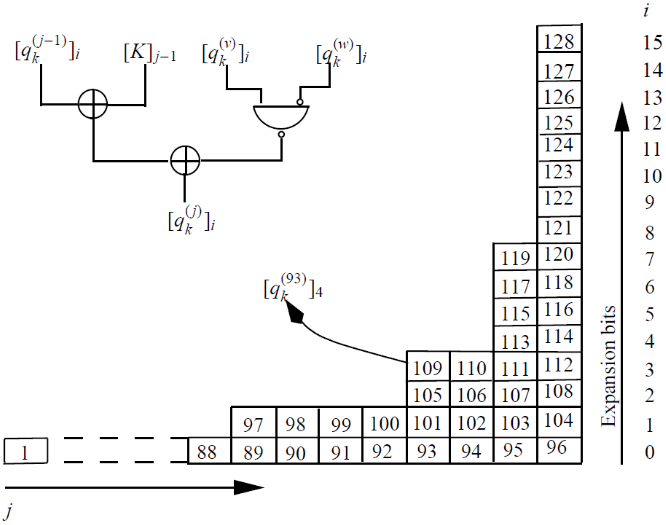
Canonical equations
$$\left\{\begin{array}{lll} q_k & = & l_\theta(c_{k-1}, \ldots, c_{k-M}) \\ z_k & = & \mathcal{F}_\theta(c_{k-1}, \ldots, c_{k-M}) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lll} q'_k & = & l_\theta(c_{k-1}, \ldots, c_{k-M}) \\ z'_k & = & \mathcal{F}_\theta(c_{k-1}, \ldots, c_{k-M}) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If g_θ is triangular, for $k \geq M$, $q'_k = q_k \Rightarrow m'_k = m_k$

Example: SSS with triangular function (2004)



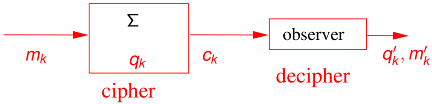
Example: Moustique (2005)



Objectives: design of SSSC with the following features

- Automata with finite input memory
- Non triangular state transition functions

Linear automata with finite input memory



$$\left\{\begin{array}{lll} q_{k+1} & = & Pq_k + Qc_k \\ z_k & = & h_\theta(q_k) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lll} q'_{k+1} & = & Pq'_k + Qc_k \\ z'_k & = & h_\theta(q'_k) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If P is nilpotent, for $k \geq M$,

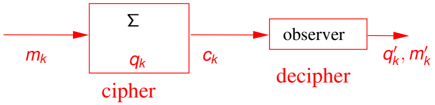
$$\left\{\begin{array}{lll} q_k & = & \cancelto{0}{P^M q_{k-M}} + I_\theta(c_{k-1}, \dots, c_{k-M}) \\ z_k & = & \mathcal{F}_\theta(c_{k-1}, \dots, c_{k-M}) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lll} q'_k & = & \cancelto{0}{P^M q'_{k-M}} + I_\theta(c_{k-1}, \dots, c_{k-M}) \\ z'_k & = & \mathcal{F}_\theta(c_{k-1}, \dots, c_{k-M}) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If P is nilpotent, for $k \geq M$, $q'_k = q_k \Rightarrow m'_k = m_k$

Linear automata with finite input memory

. . . but linearity is not suitable

LPV automata with finite input memory



$$q_{k+1} = P(\rho_k) q_k$$

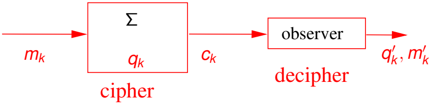
with

$$\rho_k = \left(\rho_1(c_k, \dots, c_{k-s}), \dots, \rho_L(c_k, \dots, c_{k-s})\right)^T$$

$$\Leftrightarrow$$

$$\begin{pmatrix} q_{k+1}[1] \\ \vdots \\ q_{k+1}[i] \\ \vdots \\ q_{k+1}[n] \end{pmatrix} = \begin{pmatrix} q_{11} & \cdots & \rho_1(c_k, \dots, c_{k-s}) & q_{1i} & \cdots & q_{1n} \\ \vdots & & & & & \\ q_{j1} & \rho_r(c_k, \dots, c_{k-s}) & \cdots & q_{ji} & \cdots & q_{jn} \\ \vdots & & & & & \\ q_{n1} & q_{n2} & \cdots & \rho_L(c_k, \dots, c_{k-s}) & \cdots & q_{nn} \end{pmatrix} \begin{pmatrix} q_k[1] \\ \vdots \\ q_k[i] \\ \vdots \\ q_k[n] \end{pmatrix}$$

LPV automata with finite input memory



$$\left\{\begin{array}{lcl} q_{k+1} & = & P(\rho_k)q_k + Q(\rho_k)c_k \\ z_k & = & h_\theta(q_k) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lcl} q'_{k+1} & = & P(\rho_k)q'_k + Q(\rho_k)c_k \\ z'_k & = & h_\theta(q'_k) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If $\Pi_{l=k}^{l=k+M} P(\rho_l) = 0, for\ k \geq M,$

$$\left\{\begin{array}{lcl} q_k & = & \cancelto{0}{\Pi_{l=k}^{l=k+M} P(\rho_l)}q_{k-M} + I_\theta(c_{k-1}, \dots, c_{k-M}) \\ z_k & = & \mathcal{F}_\theta(c_{k-1}, \dots, c_{k-M}) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lcl} q'_k & = & \cancelto{0}{\Pi_{l=k}^{l=k+M} P(\rho_l)}q'_{k-M} \\ & & +I_\theta(c_{k-1}, \dots, c_{k-M}) \\ z'_k & = & \mathcal{F}_\theta(c_{k-1}, \dots, c_{k-M}) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If $\Pi_{l=k}^{l=l=k+M} P(\rho_l) = 0, for\ k \geq M, q'_k = q_k \Rightarrow m'_k = m_k$

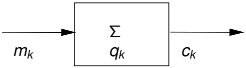
LPV automata with finite input memory

. . . but mortality is undecidable

Flatness

Let us consider the Mealy machine

$$\begin{cases} q_{k+1} = f_\theta(q_k, m_k) \\ c_k = s_\theta(q_k, m_k) \end{cases}$$



Definition

The system is *flat*, if there exists an output c_k, referred to as *flat output*, and a function l, such that

$$q_k \quad = \quad l(c_{k-1}, \ldots, c_{k-M})$$

Property (left inverse)

If the system is flat, there always exists a finite input memory automaton

$$q'_{k+1} \quad = \quad g(q'_k, c_k)$$

whose sequence $\{q'_k\}_{k \geq M}$ *coincides with* $\{q_k\}_{k \geq M}$

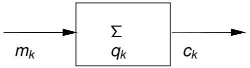
A flat system

allows to design an SSSC

Flatness and LPV systems

Let us consider the Mealy machine

$$\begin{cases} q_{k+1} = A_\theta(\rho_k) q_k + B_\theta(\rho_k) m_k \\ c_k = q_k[i] + m_k \end{cases}$$



Proposition

If the system is flat with flat output c_k, there always exists a function $I_\rho(A_\theta(\rho_k), B_\theta(\rho_k), i)$ such that

$$q_k = I_\rho(c_{k-1}, \ldots, c_{k-M})$$

Property (left inverse)

There always exists a finite input memory automaton ($\Pi_{l=k}^{l=k+M} P(\rho_l) = 0$)

$$q'_{k+1} = P_\theta(\rho_k) q'_k + Q_\theta(\rho_k) c_k$$

whose sequence $\{q'_k\}_{k \geq M}$ coincides with $\{q_k\}_{k \geq M}$

Flatness and LPV systems

How to construct a flat system ?

Structured linear systems

$$\Sigma_\rho : \begin{cases} q_{k+1} & = & A(\rho_k)q_k + B(\rho_k)m_k \\ c_k & = & q_k[i] \end{cases}$$

$$\Sigma_\Lambda : \begin{cases} q_{k+1} & = & I_A q_k + I_B m_k \\ c_k & = & q_k[i] \end{cases}$$

where

- Only the sparsity pattern of the matrices $I_A \in \mathbb{R}^{n\times n}$ and I_B is known ('0' or '1' entries)
- To the '1' entries are assigned the time-varying parameter ρ_k^i of the LPV system

Example:

$$A(\rho_k) = \begin{pmatrix} 1 & \rho_k^1 \\ 0 & \rho_k^2 \end{pmatrix},\; B(\rho_k) = \begin{pmatrix} 0 \\ \rho_k^3 \end{pmatrix},$$

$$I_A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix},\; I_B = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

Flatness for structured linear systems and LPV systems

generic flatness for the LPV system Σ_ρ

$$\Leftrightarrow$$

Structural flatness of the structured system Σ_Λ

Digraph

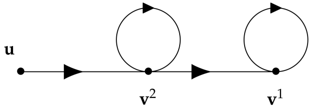
A digraph associated to Σ_ρ is a couple $(\mathcal{V}, \mathcal{E})$ where

- $\mathcal{V}$ is the vertex set associated to Σ_ρ
- $\mathcal{E}$ is the edge set associated to Σ_ρ

Example:

$$A(\rho_k) = \begin{pmatrix} 1 & \rho_k^1 \\ 0 & \rho_k^2 \end{pmatrix}, \; B(\rho_k) = \begin{pmatrix} 0 \\ \rho_k^3 \end{pmatrix},$$

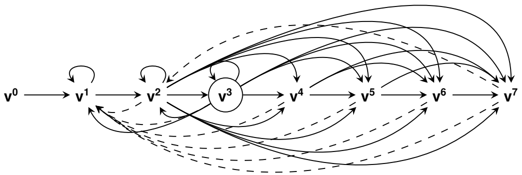
$$I_A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \; I_B = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$



$\Rightarrow$ We have derived conditions $C0$, $C1$, $C2$ to guarantee that the vertex v_i which corresponds to $c_k = q_k[i]$ is a flat output

Summary

- Choose a dimension n (number of components of the state q_k) and a number of edges n_a (number of non zero entries of I_A and I_B)
- Construct a digraph fulfilling the flatness conditions $C0$, $C1$, $C2$



Summary

- Derive from the adjacency matrix the matrices I_A and I_B

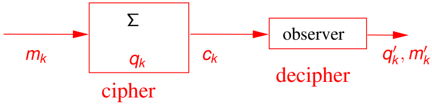
$$\left\{\begin{array}{l} q_{k+1} = I_A q_k + I_B m_k \\ c_k = q_k[i] \end{array}\right. , \; I_A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}, \; I_B = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

- Replace some of the '1' entries by nonlinear functions $\varphi(c_k, \ldots, c_{k-s})$ (S-boxes)

$$\left\{\begin{array}{l} q_{k+1} = A(\rho_k) q_k + B(\rho_k) m_k \\ c_k = q_k[i] \end{array}\right. , A(\rho_k) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \rho_k^1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & \rho_k^2 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}, B(\rho_k) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Maurer's approach (1991): use of finite input memory automata

- Derive the finite automaton with finite input memory

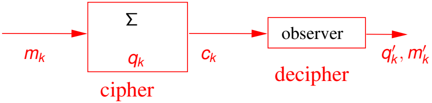


$$\begin{cases} q_{k+1} = P(\rho_k)q_k + Q(\rho_k)c_k \\ z_k = h_\theta(q_k) = q_k[i] \\ c_k = e(z_k, m_k) \end{cases} \qquad \begin{aligned} q'_{k+1} &= P(\rho_k)q'_k + Q(\rho_k)c_k \\ z'_k &= h_\theta(q'_k) = q_k[i] \\ m'_k &= d(z'_k, c_k) \end{aligned}$$

If $\Pi_{l=k}^{l=k+M} P(\rho_l) = 0$, *for* $k \geq M$, GUARANTEED

$$\begin{cases} q_k = \cancelto{0}{\Pi_{l=k}^{l=k+M} P(\rho_l)} q_{k-M} + I_\theta(c_{k-1}, \ldots, c_{k-M}) \\ z_k = \mathcal{F}_\theta(c_{k-1}, \ldots, c_{k-M}) \\ c_k = e(z_k, m_k) \end{cases} \qquad \begin{aligned} q'_k &= \cancelto{0}{\Pi_{l=k}^{l=k+M} P(\rho_l)} q'_{k-M} \\ &\quad + I_\theta(c_{k-1}, \ldots, c_{k-M}) \\ z'_k &= \mathcal{F}_\theta(c_{k-1}, \ldots, c_{k-M}) \\ m'_k &= d(z'_k, c_k) \end{aligned}$$

Maurer's approach (1991): use of finite input memory automata



State equations

$$\left\{\begin{array}{lcl} q_{k+1} & = & g_\theta(q_k, c_k) \\ z_k & = & h_\theta(q_k) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lcl} q'_{k+1} & = & g_\theta(q'_k, c_k) \\ z'_k & = & h_\theta(q'_k) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

If g_θ is triangular, for $k \geq M$, NO LONGER REQUIRED

Canonical equations

$$\left\{\begin{array}{lcl} q_k & = & l_\theta(c_{k-1}, \ldots, c_{k-M}) \\ z_k & = & f_\theta(c_{k-1}, \ldots, c_{k-M}) \\ c_k & = & e(z_k, m_k) \end{array}\right. \qquad \begin{array}{lcl} q'_k & = & l_\theta(c_{k-1}, \ldots, c_{k-M}) \\ z'_k & = & f_\theta(c_{k-1}, \ldots, c_{k-M}) \\ m'_k & = & d(z'_k, c_k) \end{array}$$

Conclusion

- A method to construct SSSC with non triangular next state transition functions
- Based on control theory (flatness), LPV systems, Graph approach
- Investigation of the most suitable dimension *n*, type of nonlinearities (S-boxes), number and position of S-boxes in the state transition matrix
- Security
- Effective implementation

Conclusion