Hybrid architecture of LPV dynamical systems in the context of cybersecurity

Hamid Boukerrou, Gilles Millérioux, Marine Minier

Hybrid architecture of LPV dynamical systems in the context of cybersecurity ⋆

Hamid Boukerrou * **Gilles Millerioux** * **Marine Minier** **

* *Université de Lorraine, CNRS, CRAN, UMR 7039, France, email:* {*hamid.boukerrou,gilles.millerioux*}*@univ-lorraine.fr*
** *Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France, email:* *marine.minier@loria.fr*

Abstract: This paper deals with an hybrid architecture involving LPV dynamical systems for encryption purpose, having in mind the context of cybersecurity. Such an hybrid architecture is motivated by the fact that it is a natural model, recast in a control-theoretic framework, of a so-called statistical self-synchronizing stream cipher. It is shown that flatness is central to guarantee the necessary synchronization between the cipher and the decipher. In this context, beyond synchronization, security must be a constraint to be taken into account as well. We especially focus on diffusion as a security criterion. The hybrid architecture is motivated to take advantage of both properties simultaneously. An illustrative example presents a numerical application and must be considered as a proof-of-concept before further investigation.

Keywords: Structural analysis, LPV systems, cryptography, self-synchronization stream ciphers

1. INTRODUCTION

The considerable development of new technologies of communication leads nowadays to an increasing need for security of the exchanges of information. Data exchanges often take place over public networks. In this context, Cyber-physical systems (CPS) deserve a special attention since they are ubiquitous and play a central role. Indeed, they connect the physical and the digital worlds in more and more situations of everyday life. In particular, all the operators of vital importance (OIVs) rely on Cyberphysical systems: telecommunication, transportation, banking, manufacturing, power generation and distribution, Supervisory Control And Data Acquisition (SCADA) systems, medical systems, to mention a few. Besides, the concept of networked control systems (NCSs) Ge et al. (2017) consisting in controlling process over communication networks has replaced traditional point-to-point control systems. Strengthen the protection of the control systems against cyber (electronic) threat and assessing their vulnerabilities are of paramount importance and is clearly a timely challenge. Well-known examples like the cyber-attack against the Bushehr nuclear power plant in Iran with Stuxnet illustrate those considerations. To this end, many tools and frameworks borrowed from distinct areas as computer science, cryptography, automatic control are regularly proposed.

As far as automatic control is concerned, the literature grows up rapidly. The paper Abdelwahab et al. (2021) (and references therein) explores the use of a control-theoretic approach enabling a compromised networked controller to leak information to an eavesdropper who has access to the measurement channel. The underlying notion is *covert channel* originated in 1973 by Butler Lampson. A covert channel is a type of attack that creates a capability to transfer information objects between processes that are not supposed to be allowed to communicate by the computer security policy. To secure CPS, a possible strategy is to exploit an accurate mathematical model of the dynamics of the physical system under control, and analyze any discrepancy between the actual sensor measurements and the ones predicted by the model, to decide about the existence of an adversarial attack. Such an approach calls for the notion of state reconstruction, flatness and observability as detailed in Shoukry et al. (2015). More generally, the paper Sánchez et al. (2019) presents a bibliographical review of definitions, classifications and applications concerning cyber-attacks in networked control systems and Cyber-physical systems. This review tackles the topic from a control-oriented perspective, which is complementary to information or communication ones. It is shown that many concepts derived from diagnosis are really appropriate.

Another fully-fledged approach to secure CPS is cryptography Menezes et al. (1996). The aim is to directly protect data conveyed through public channels. Since 2015, several attempts to incorporate cryptography into networked control systems have appeared to enhance cybersecurity. There are two areas in cryptography: public key cryptography and secret key cryptography. In our work, we are interested in secret key cryptography which uses a same key to encrypt and decrypt. In secret key cryptography, the stream ciphers are used in secure data transmission at a very high rate and for electronic devices with limited hardware resources. There are two main classes of stream ciphers: Synchronous Stream Cipher (SSC) and Self-Synchronous Stream Cipher (SSSC). The SSSCs are the subject of our study.

The reader may refer for example to Darup et al. (2020) that addresses this topic. In this paper, still having in

⋆ This work was partly supported by the french PIA project "Lorraine Université d'Excellence", reference ANR-15-IDEX-04-LUE.

mind data protection in the context of CPS, we address in a theoretic-control point of view the design of a special class of ciphers called statistical self-synchronizing stream ciphers. This class of cipher deserve further investigation because it has not been addressed in a large extent, see Geraldy et al. (2001); Jung and Ruland (1999); Heys (2001) for exceptions. Essentially, it involves an architecture that combines the properties of both modes of operation. A special switching rule orchestrates the active modes. It must take into account on one hand, the inherent necessity of synchronization between the cipher and the decipher for proper decryption and on the other hand, the performances in terms of security and data rate. This being the case, control theory appears as a natural framework to tackle such a class of ciphers. We propose in this paper a new statistical self-synchronizing stream cipher involving automata in the form of LPV systems. The choice of the class of LPV systems will be motivated. The proposed framework and proof-of-concept results involve control-theoretic concepts such as flatness and structural analysis.

This paper is organized as follows. Section 2 introduces generalities on stream ciphers and especially the self-synchronized ones. Section 3 explains how a self-synchronizing stream cipher can be constructed from an LPV system and the choice of such a class is motivated. The construction is based on the interpretation of flatness in terms of the structure of a graph associated to the LPV system. Thus, the connection between flatness and SSSC is made. Section 4 is devoted to the design of an hybrid architecture used as a statistical Self-Synchronization Stream Cipher. The hybrid feature is motivated by a trade-off to be fulfilled between the consideration of synchronization and security. A numerical example is given. Finally, Section 5 concludes this paper.

2. GENERALITIES ON STREAM CIPHERS

2.1 Stream Ciphers Overview

For a stream cipher, it must be given an alphabet A, that is, a finite set of basic elements named symbols. The set A stands in this paragraph as a general notation without any specific alphabet. Typically, A could be composed of 1 or several bits elements. Hereafter, the index $t \in \mathbb{N}$ will stand for the discrete-time. On the transmitter part, the plaintext (also called information or message) $m \in \mathcal{M}$ ($\mathcal{M}$ is the message space) is a string of plaintext symbols $m_t \in A$. Each plaintext symbol is encrypted, by means of an encryption (or ciphering) function e, according to:

$$c_{t+r} = e(z_{t+r}, m_t), \tag{1}$$

where $z_t \in A$ is a so-called keystream (or running key) symbol delivered by a keystream generator. The function e is invertible for any prescribed z_t. The resulting symbol $c_t \in A$ is the ciphertext symbol. The integer $r \geq 0$ stands for a potential delay between the plaintext m_t and the corresponding ciphertext c_{t+r}. This is explained by computational or implementation reasons. Consequently, for stream ciphers, the way how to encrypt each plaintext symbol changes on each iteration and stands as an asset for this class of ciphers. The resulting ciphertext $c \in \mathcal{C}$ ($\mathcal{C}$ is called the ciphertext space), that is the string of symbols c_t, is conveyed to the receiver through a public channel.
At the receiver side, the ciphertext c_t is deciphered according to a decryption function d which depends on a running key $\widehat{z}_t \in A$ delivered, similarly to the cipher part, by a keystream generator. The decryption function d obeys the following rule. For any two keystream symbols $\widehat{z}_{t+r}, z_{t+r} \in A$, it holds that

$$\widehat{m}_{t+r} := d(c_{t+r}, \widehat{z}_{t+r}) = m_t \text{ whenever } \widehat{z}_{t+r} = z_{t+r}. \tag{2}$$

Equation (2) means that the running keys z_t and $\widehat{z}_t$ must be synchronized for a proper decryption. The generators delivering the keystreams are parameterized by a secret key denoted by $K \in \mathcal{K}$ ($\mathcal{K}$ is the secret key space). The distinct classes of stream ciphers (synchronous or self-synchronizing) differ each other by the way on how the keystreams are generated and synchronized. Next, we detail the special class of stream ciphers called Self-Synchronizing Stream Ciphers and denoted for brevity SSSC.

2.2 Keystream Generators for Self-Synchronizing Stream Ciphers

A well-admitted approach to generate the keystreams has been first suggested in Maurer (1991). It is based on the use of so-called finite state automata with finite input memory as described below. This is typically the case in the cipher Moustique Kasper et al. (2004). At the ciphering side, the automaton delivering the keystream takes the form:

$$\begin{cases} x_{t+1} = f_K(x_t, m_t), \\ z_{t+r} = h_K(x_t) \end{cases} \tag{3}$$

where $x_t \in A$ is the internal state, f is the next-state transition function parameterized by $K \in \mathcal{K}$. As previously stressed, the delay r is introduced to cope with special situations, in particular when the computation of the output (also called filtering) delivered by the function h involves r successive operations processed at time instants $t, \ldots, t+r$. Those operations will be here matrix multiplications as detailed later in Equation (13). Substituting m_t by its expression (2) and taking into account, from (3), that z_{t+r} is a function of x_t, give

$$\begin{cases} x_{t+1} = g_K(x_t, c_{t+r}), \\ z_{t+r} = h_K(x_t) \end{cases} \tag{4}$$

If such an automaton has finite input memory, it means that, by iterating (4) a finite number of times, there exists a function ℓ_K and a finite integer M such that

$$x_t = \ell_K(c_{t+r-1}, \ldots, c_{t+r-M}), \tag{5}$$

and thus,

$$z_{t+r} = h_K(\ell_K(c_{t+r-1}, \ldots, c_{t+r-M})). \tag{6}$$

Actually, the fact that the keystream symbol can be written in the general form

$$z_{t+r} = \alpha_K(c_{t-\ell}, \ldots, c_{t-\ell'}), \tag{7}$$

with α_K a function involving a finite number of shifted ciphertexts from time $t-\ell$ to $t-\ell'$ ($\ell, \ell' \in \mathbb{Z}$), is a common feature of the SSSC. Equation (7) is called the canonical equation.
At the deciphering side, the automaton takes the form

$$\begin{cases} \widehat{x}_{t+1} = g_K(\widehat{x}_t, c_{t+r}), \\ \widehat{z}_{t+r} = h_K(\widehat{x}_t) \end{cases} \tag{8}$$

where $\widehat{x}_t$ is the internal state. Similarly to the cipher part, the automaton having a finite input memory, it means

that, by iterating Equation (8) the same finite number of times as (4), one also obtains

$$\hat{x}_t = \ell_K(c_{t+r-1}, \ldots, c_{t+r-M}),$$

and thus,

$$\hat{z}_{t+r} = h_K(\ell_K(c_{t+r-1}, \ldots, c_{t+r-M})).$$

Hence, it is clear that after a transient time of maximal length equal to M, it holds that, for $t \geq M$,

$$\widehat{x}_t = x_t \text{ and } \widehat{z}_{t+r} = z_{t+r}. \tag{9}$$

In other words, the generators synchronize automatically after at most M iterations. Hence, the decryption is automatically and properly achieved after at most M iterations too. No specific synchronizing protocol between the cipher and the decipher is needed. This explains the terminology Self-Synchronizing Stream Ciphers. The integer M is called the delay of synchronization. The following remark is central for our purpose.

Remark 1. From (5), it is worth pointing out that the state vector x_t of the automaton (3) is expressed as a function of a finite number of its shifted outputs. Furthermore, from the equalities (2) and (9), the same property holds for the input m_t. And yet, from a control-theoretic point of view, this is nothing but the property of difference flatness of (3). Let us recall that difference flatness, (see Sira-Ramirez and Agrawal (2004), Chapter 5, for an introduction in the case of LTI systems), is the discrete-time counterpart of the so-called differential flatness dedicated to continuous-time systems first introduced in Fliess et al. (1995). The link between SSSC and flatness is central for our purpose as seen in next section.

3. FLAT LPV AUTOMATA AND SSSC

3.1 State space equations of flat LPV automata

As a clue to simply obtaining nonlinear systems, we propose to investigate the class of Linear Parameter-Varying (LPV) systems as a special class of automata. Those automata will be called LPV automata. As usual in the context of cryptography, they must be defined over a finite field $\mathbb{F}$. They are described by the following state space representation:

$$x_{t+1} = A_{\rho(t)}x_t + Bm_t \tag{10}$$

$x_t \in \mathbb{F}^n$ is the state vector, $m_t \in \mathbb{F}$ is the input. The matrices $A \in \mathbb{F}^{n\times n}$ and $B \in \mathbb{F}^{n\times 1}$ are respectively the dynamical matrix and the input matrix. The matrix B is the input matrix and defines the component x_t^i on which the plaintext symbol m_t is added. The set of all varying parameters of A are collected on a vector denoted by

$$\rho(t) = \left[\, \rho^1(t), \rho^2(t), ..., \rho^L(t) \,\right] \in \mathbb{F}^L$$

where L is the total number of non-zero (possibly varying) entries. Such automata can exhibit nonlinear dynamics. Indeed, the nonlinearity is obtained by defining the varying parameters $\rho^i(t)$ as nonlinear functions $\varphi^i : \mathbb{F}^{s+1} \to \mathbb{F}$ of the output c_t (or a finite number of shifts) $\rho^i(t) = \varphi^i(c_t, c_{t-1}, \cdots, c_{t-s})$ with s a natural number. In the context of cryptography, those functions are implemented in the form of S-boxes

$$\begin{array}{rccc} \varphi^i : & \mathbb{F}^{s+1} & \to & \mathbb{F} \\ & (c_t, c_{t-1}, \cdots, c_{t-s}) & \mapsto & S(c_t, c_{t-1}, \cdots, c_{t-s}, SK_i) \end{array} \tag{11}$$

where SK_i is the subkey number i derived from the secret key K. S-boxes are usual in symmetric cryptography but they have been used in the context of LPV automata very recently and for the first time in Francq et al. (2020). The outcome of introducing LPV automata is that we can design a large class of nonlinear automata with a high flexibility that rests on the choice of S-boxes and on the way how they are introduced in the matrix A. Besides, LPV automata benefit from their linear structure and thus, are appropriate for a structural analysis-based design as we shall see in the sequel.

3.2 Constructing an SSSC from an LPV automaton

For brevity, we introduce the following notation. For $t_2 \geq t_1$, denote by $\prod_{l=t_2}^{t_1} A_{\rho(l)}$ the product of matrices $A_{\rho(l)}$ from t_2 to t_1. For $t_2 < t_1$, define $\prod_{l=t_2}^{t_1} A_{\rho(l)} = \mathbf{1}_n$ (the identity matrix of dimension n). Finally, let $\mathcal{T}$ be the scalar defined by $\mathcal{T} = C\prod_{l=t+r-1}^{t+1} A_{\rho(l)}B$.
Consider the LPV finite state automaton (10) with an output c_t defined as

$$c_t = Cx_t \tag{12}$$

with $C \in \mathbb{F}^{1\times n}$ the output matrix. Then, define the keystream with delay r as

$$z_{t+r} = C \prod_{l=t+r-1}^{t} A_{\rho(l)}x_t \tag{13}$$

and the ciphering function as

$$c_{t+r} = z_{t+r} + \mathcal{T}m_t, \tag{14}$$

On the other hand, consider the finite state automaton with internal state $\widehat{x}_t$ with dynamics given by

$$\widehat{x}_{t+1} = P_{\rho(t:t+r)}\widehat{x}_t + B\mathcal{T}^{-1}c_{t+r} \tag{15}$$

with

$$P_{\rho(t:t+r)} = A_{\rho(t)} - B\mathcal{T}^{-1}C \prod_{l=t+r-1}^{t} A_{\rho(l)} \tag{16}$$

along with the keystream $\widehat{z}_t$ defined as

$$\widehat{z}_{t+r} = C \prod_{l=t+r-1}^{t} A_{\rho(l)}\widehat{x}_t \tag{17}$$

and the deciphering function obeying

$$\widehat{m}_{t+r} = \mathcal{T}^{-1}(c_{t+r} - \widehat{z}_{t+r}). \tag{18}$$

Actually, Equations (15)-(18) define the left inverse system of (10) and the least natural integer r such that $\mathcal{T}$ is non zero is the *relative degree* of the system (10). By simple manipulations, it can be shown that the synchronization error $\varepsilon_t = x_t - \widehat{x}_t$ verifies

$$\varepsilon_{t+1} = P_{\rho(t:t+r)}\varepsilon_t \tag{19}$$

Two classes of SSSC can be defined according to the delay of synchronization:

- *Deterministic*: the delay of synchronization is bounded by the constant M a priori fixed.
- *Statistical*: the bound of the delay of synchronization is not constant but is a random variable with respect to the sequence of ciphertexts or the initial state vector.

If ε_t reaches zero after a finite transient time of length bounded by an a priori fixed natural integer M, thus,

the respective state vectors of (10) and (15) are self-synchronized. Then, by definition, the set of Equations (10)-(18) defines a deterministic SSSC. The integer M is the synchronization delay. From Remark 1, it can be claimed that any flat LPV automaton gives rise to a deterministic SSSC. The point is that the LPV automaton defined by (10) must be flat for any secret key K and any realization of $\rho(t)$. In other words, flatness must be a generic property of (10). For design perspectives, to meet such a requirement, it is interesting to realize that an LPV system can be considered as an admissible realization of a corresponding structured LTI system. Hence, if flatness is ensured for a structured LTI system, flatness will be preserved for the derived LPV system admitting the same structure. This is the clue that will be used as seen in next section.

3.3 Structured systems and digraphs

A structured linear system is a linear system only defined by the sparsity pattern of the state space realization matrices. In other words, for a structured linear system, we distinguish between the entries that are fixed to zero and the other ones that can take any value in $\mathbb{F}$, including the ones which are time-varying. Hence, a structured linear discrete-time system, denoted by Σ_Λ, is a system that admits the form:

$$\Sigma_\Lambda : \ x_{t+1} = I_A x_t + I_B m_t \tag{20}$$

The entries of the matrices of (20) are '0' or '1'. In particular, the entries $A(i,j)$ of I_A (*resp.* $B(i)$ of I_B) that are '0' mean that there are no relation (dynamical interaction) between the state x^i_{t+1} at time $t+1$ and the state x^j_t at time t (*resp.* the state x^i_{t+1} at time $t+1$ and the input m_t at time t). The entries that are '1' mean that there is a relation. As a simple example, let us consider an LPV system with the setting

$$A_{\rho(t)} = \begin{pmatrix} a & 0 \\ \rho^1(t) & \rho^2(t) \end{pmatrix} \text{ and } B = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

where a is a constant element in $\mathbb{F}$, $\rho^1(t)$ and $\rho^2(t)$ are varying parameters in $\mathbb{F}$. The dynamical matrix and the input matrix I_A and I_B of the corresponding structured linear system read:

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

As a consequence, if the structural linear system (20) derived from (10) is flat, the flatness will hold for any $\rho(t)$ or equivalently any nonlinearity φ^i (any S-box will be admissible). Hence, the challenge is to define a methodology to construct flat linear structural systems. To this end, the graph-based approach provided in Millérioux and Boukhobza (2015) can be used.

In our context, the triplet (n, r, n_a), the number of nonlinear functions φ^i and their location in the matrix I_A determine a family of flat LPV-based SSSC. Next subsection will aim at summarizing the steps needed for the design of such a family.

A digraph $\mathcal{G}(\Sigma_\Lambda)$ describing a structured linear system associated state equations, is the combination of a vertex set $\mathcal{V}$ and an edge set $\mathcal{E}$. The vertices represent the states and the input components of Σ_Λ while the edges describe the dynamic relations between these variables. One has $\mathcal{V} = \mathbf{X} \cup \{\mathbf{m}\}$ where $\mathbf{X}$ is the set of state vertices defined as $\mathbf{X} = \{\mathbf{x^1}, \ldots, \mathbf{x^n}\}$ and $\mathbf{m}$ is the input vertex. The edge set is $\mathcal{E} = \mathcal{E}_A \cup \mathcal{E}_B$, with $\mathcal{E}_A = \{(\mathbf{x^i}, \mathbf{x^j}) \,|A(i,j) \neq 0\}$ and $\mathcal{E}_B = \{(\mathbf{m}, \mathbf{x^i}) \,|B(i) \neq 0\}$. For convenience, we will denote by $\mathbf{v^j}$, $(j = 0, \ldots, n)$ a vertex of the digraph $\mathcal{G}(\Sigma_\Lambda)$ regardless whether it is the input or a state vertex.

- A directed path $\mathbf{P}$ is a sequence of successive edges directed in the same direction which connect a sequence of vertices. It is said that the path $\mathbf{P}$ covers a vertex if this vertex is the begin or the end vertex of one of the edges of $\mathbf{P}$;
- In a directed path from a vertex $\mathbf{v^i}$ to a vertex $\mathbf{v^j}$, it is said that $\mathbf{v^j}$ is a successor of $\mathbf{v^i}$ and conversely, $\mathbf{v^i}$ is a predecessor of $\mathbf{v^j}$;
- A simple path is a path which contains no repeated vertices;
- The length of a directed path $\mathbf{P}$ is equal to the number of edges involved in $\mathbf{P}$. We let $\ell(\mathbf{v^i}, \mathbf{v^j})$ denote the minimal length of a path connecting $\mathbf{v^i}$ to $\mathbf{v^j}$;
- $V_{ess}(\mathbf{v^i}, \mathbf{v^j})$ is the set of vertices, called essential vertices from $\mathbf{v^i}$ to $\mathbf{v^j}$, which are common to all the paths connecting $\mathbf{v^i}$ to $\mathbf{v^j}$.

We recall from Millérioux and Boukhobza (2015) the necessary and sufficient conditions which must be satisfied for any vertex $\mathbf{v^i}$ $(i \in \{1, \ldots, n\})$ of the digraph $\mathcal{G}(\Sigma_\Lambda)$ to be associated to a flat output $c^F_t = x^i_t$ $(i \in \{1, \ldots, n\})$. In such a case, the output state space matrix $C_{\rho(t)} = C$ is constant and has zero entries except the entry located at the column number i which is equal to one. The direct transfer matrix $D_{\rho(t)} = D$ is the zero matrix.

Theorem 1. The output $c^F_t = x^i_t$ $(i \in \{1, \ldots, n\})$ of the structured linear system (20), associated to the vertex $\mathbf{v^F} \in \mathbf{X}$ in the associated digraph $\mathcal{G}(\Sigma_\Lambda)$, is generically a flat output iff, the three following conditions hold:
C0. $\mathbf{v^F}$ is a successor of $\mathbf{m}$;
C1. Any simple paths from $\mathbf{m}$ to $\mathbf{v^F}$ have the same length equal to $\ell(\mathbf{m}, \mathbf{v^F})$;
C2. Any cycles cover at least an element of $V_{ess}(\mathbf{m}, \mathbf{v^F})$.

Hence, Conditions **C0-C2** are instrumental for the construction of structural flat dynamical systems. An example of systematic construction of digraphs fulfilling **C0-C2** can be found in the paper Francq et al. (2020).

3.4 Summary for the construction of SSSC from a flat LPV-based automaton

The following Steps Si summarize the way how to construct an SSSC from a flat LPV automaton. Choose a triplet (n, n_a, r) with n the dimension of the state, r the relatve degree and n_a the number of non-zero entries of matrix A.

Step S1: Choose a component x^i_t on which the plaintext symbol m_t is added. It follows that $B = (0 \ldots 1\ 0 \ \ldots 0)^T$ (the entries 1 is located at column i, ^{T} stands for transposition).

Step S2: Choose a component x_t^i ($i \in \{1, \ldots, n\}$) as the desired flat output $y_t = x_t^i$. It follows that $C = (0 \cdots 0\ 1\ 0\ \ldots 0)$ (the only entry 1 is located at the i-th column of C). It can be shown that for the special case when $B = (1\ 0\ \ldots)$, the relative degree r coincides with i.

Step S3: Construct a digraph $\mathcal{G}(\Sigma_\Lambda)$ fulfilling Conditions **C0-C2** with n vertices and n_a edges. Derive the matrices I_A and I_B of the structured linear system Σ_Λ. It can be carried out from the adjacency matrix, denoted by $\mathcal{I}$, associated to the digraph $\mathcal{G}(\Sigma_\Lambda)$. It is the $(n+1) \times (n+1)$ matrix

$$\mathcal{I} = \left(\begin{array}{c|c} 0 & I_B^T \\ \hline 0 & \\ \vdots & I_A^T \\ 0 & \end{array}\right) \tag{21}$$

where I_A^T and I_B^T stands respectively for the transpose of the structured matrices I_A and I_B. The entries $\mathcal{I}_{ij}$ are defined as follows for $1 \leq i, j \leq n$

$$\mathcal{I}_{ij} = \begin{cases} 1 \text{ if there exists an edge from } \mathbf{v^j} \text{ to } \mathbf{v^i} \\ 0 \text{ otherwise.} \end{cases} \tag{22}$$

Step S4: Replace some of the non-zero entries of I_A by a nonlinear function $\rho^i(t) = \varphi^i(c_t, c_{t-1}, \cdots, c_{t-s})$ to construct the matrix $A_{\rho(t)}$ of (10) and set $B = I_B$. Not all '1' entries of I_A must be assigned to a non-linear function. Some of them can be merely constant. The choice must obey a trade-off between complexity of the architecture and security (a matter discussed in next section). The construction ensures flatness provided that the non-zero entries assigned to the directed path connecting the input vertex to the flat output one are constant.

Step S5: Complete the design by deriving the set of Equations (10)-(18).

As an example, the following graph corresponds to a flat structured LTI system with $(n = 7, n_a = 15, r = 3)$.

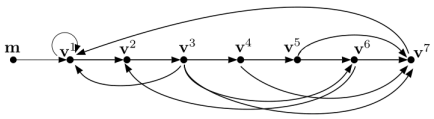


Fig. 1. Digraph $\mathcal{G}(\Sigma_\Lambda)$ related to a flat LTI system with relative degree $r = 3$. The vertex $\mathbf{v}^3$ is the flat output.

The following matrices correspond respectively to the adjacency matrix $\mathcal{I}$, the extracted structured matrix I_A, the matrix $A_{\rho(t)}$ involving nonlinearities $\varphi(t)$ and finally the matrix $P_{\rho(t:t+3)}$ computed from the formula (16).

$$\underset{\mathcal{I}}{\begin{pmatrix} 0&1&0&0&0&0&0&0\\ 0&1&0&1&0&0&0&1\\ 0&1&0&0&0&0&1&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&1&0&1&0&0\\ 0&0&0&1&1&1&1&0 \end{pmatrix}} \quad \underset{I_A}{\begin{pmatrix} 1&0&1&0&0&0&1\\ 1&0&0&0&0&1&0\\ 0&1&0&0&0&0&0\\ 0&0&1&0&0&0&0\\ 0&0&0&1&0&0&0\\ 0&0&1&0&1&0&0\\ 0&0&1&1&1&1&0 \end{pmatrix}}$$

$$\underset{A}{\begin{pmatrix} 1&0&\varphi(t)&0&0&0&1\\ 1&0&0&0&0&1&0\\ 0&1&0&0&0&0&0\\ 0&0&1&0&0&0&0\\ 0&0&0&1&0&0&0\\ 0&0&\varphi(t)&0&1&0&0\\ 0&0&\varphi(t)&1&\varphi(t)&1&0 \end{pmatrix}} \quad \underset{P}{\begin{pmatrix} 0&0&-\varphi(t)&0&-1&0&0\\ 1&0&0&0&0&1&0\\ 0&1&0&0&0&0&0\\ 0&0&1&0&0&0&0\\ 0&0&0&1&0&0&0\\ 0&0&\varphi(t)&0&1&0&0\\ 0&0&\varphi(t)&1&\varphi(t)&1&0 \end{pmatrix}}$$

4. A STATISTICAL SELF-SYNCHRONIZING HYBRID ARCHITECTURE

This section stands as the main contribution of the paper. The complexity of the SSSC depends on the triplet (n, n_a, r) and on the type of S-boxes. The larger the integers n, n_a and r, the more the complexity. Clearly, a relevant design must obey a trade-off between complexity and security. As for security, an SSSC must be safe with respect to distinct attacks or criteria like Time-memory-data trade-off attacks, Guess-and-Determine attacks, Differential / Linear Cryptanalysis, Algebraic Attacks, Cube Attacks. The reader may refer to Francq et al. (2020) and references therein for details of those attacks. Roughly speaking, the more the integers n and n_a, the more the security. It is rather intuitive. The role of the integer r is less clear and deserves a special attention. It is the purpose of the issue addressed below.

4.1 The role of the relative degree r

Due to the matrices multiplication involved in (16) at the deciphering side, the size of the corresponding hardware circuit grows up rapidly when r increases. Therefore, it is natural to attempt to reduce the relative degree to its minimum value, $r = 1$ ($r = 0$ would mean that the ciphertext c_t is equal to the plaintext m_t and so makes no sense).

We focus here on a security criteria called diffusion and explained below. Let p denotes the power of a matrix $Z \in M_n(\mathbb{F})$. The diffusion delay, introduced in Arnault et al. (2011), is the smallest value, denoted by d_0, of p such that Z^p does not have any zero coefficient. In other words, it is the smallest value of p such as each element of the initial internal state x_0 has influenced every element of x_t for $t \geq d_0$. Hence, the investigation of such properties relies on symbolic matrices A_S and P_S obtained by replacing any non zero entries of A and P by the symbol 'S'. It is then a structural property. From design perspectives, the smaller d_0, the better the diffusion. Hence, it is natural to investigate this property when $r = 1$.

Let us consider a digraph $\mathcal{G}(\Sigma_\Lambda)$ related to $r = 1$ as depicted on Figure 2.

The structured matrix I_A and the matrix A related to digraph $\mathcal{G}(\Sigma_\Lambda)$ are respectively given by:

$$\underset{I_A}{\begin{pmatrix} 1&0&1&0&1&0&1\\ 1&0&0&0&0&0&0\\ 1&1&0&0&0&0&0\\ 1&0&1&0&0&0&0\\ 1&0&0&1&0&0&0\\ 1&0&1&0&1&0&0\\ 1&0&1&1&1&1&0 \end{pmatrix}} \quad \underset{A}{\begin{pmatrix} 1&0&\varphi(t)&0&\varphi(t)&0&\varphi(t)\\ 1&0&0&0&0&0&0\\ 1&1&0&0&0&0&0\\ 1&0&1&0&0&0&0\\ 1&0&0&1&0&0&0\\ 1&0&\varphi(t)&0&1&0&0\\ 1&0&1&1&1&1&0 \end{pmatrix}}$$

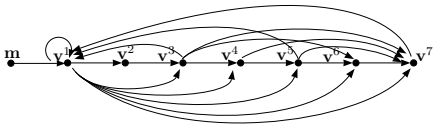


Fig. 2. Digraph $\mathcal{G}(\Sigma_\Lambda)$ related to a flat LTI system with relative degree $r = 1$.

The matrix P obtained from the matrix A obeys (16). If $P_{\rho(t:t+1)}[j]$ $(j = 1, \ldots, n-1)$ stands for the row number j of P, one has:

$$\begin{array}{l} P_{\rho(t:t+1)}[0] = A_{\rho(t)}[0] - A_{\rho(t)}[0] = 0 \\ P_{\rho(t:t+1)}[i] = A_{\rho(t)}[i], \quad 1 \leq i \leq n-1 \end{array} \tag{23}$$

and thus,

$$P = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & \varphi(t) & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

In our context, the study of the diffusion must focus on the successive powers A_S^p and P_S^p as p increases. Indeed, the power of matrices results from the successive iterations of the ciphering and the deciphering equations given by (10) and (15) respectively.

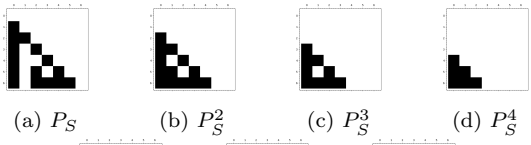
Figures 3 shows that a complete diffusion is achieved for A_S and the delay of diffusion is equal to 3. On the other hand, the successive powers of P^P undergo a bad diffusion as illustrated in Figure 4. Indeed, it can be noticed that P is lower triangular. Actually, it can be shown, by construction of the digraph fulfilling Conditions **C0-C2**, that it is a general property for $r = 1$. And yet, it can be easily proved that the product of two triangular matrices is a triangular matrix. Indeed, for $j > i$, for two triangular matrices A, $B \in \mathcal{M}_{n\times n}(\mathbb{F}_{16})$, then $A \cdot B = \sum_{k=1}^{n} a_{ik}b_{jk} = \sum_{k=1}^{i} a_{ik} \cdot 0 + \sum_{k=i+1}^{n} 0 \cdot b_{jk} = 0$. Besides, P_S^7 reaches the null matrix. It is not surprising since P is nilpotent by construction. Indeed, in view of (19), if the system is self-synchronizing, ε_t must reach zero after a finite transient time, even in the LTI case where P is constant.

(a) A_S (b) A_S^2 (c) A_S^3

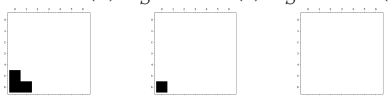
Fig. 3. Powers of A_S^p with $p \in [1, 3]$, $r = 1$. Black squares represent diffusion.

4.2 Introducing a hybrid architecture

In order to conceal the triangularization of the powers of P, one non-zero entry in the upper triangular part of the matrix I_A is added (2nd row, last column), having in mind



(a) P_S (b) P_S^2 (c) P_S^3 (d) P_S^4



(e) P_S^5 (f) P_S^6 (g) P_S^7

Fig. 4. Powers of P_S^p for $p \in [1, 7]$, $r = 1$. Black squares correspond to non zero entries.

that the complexity must be preserved as most as possible. In such a case, according to (23), the triangular pattern of P is broken (let us notice that the non zero entry must not be added on the first row of A since the first row of matrix P is null by construction, see (23)). Hence, the new structural matrix I'_A and the corresponding matrix P' are obtained and given below.

$$\underset{I'_A}{\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}} \quad \underset{P'}{\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & \varphi(t) & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}}$$

It is equivalent to state that a new edge is added in digraph $\mathcal{G}(\Sigma_\Lambda)$ yielding the new digraph $\mathcal{G}'(\Sigma_\Lambda)$ depicted in Figure 5

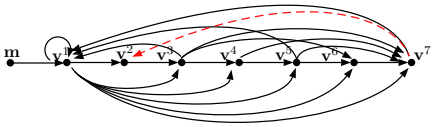
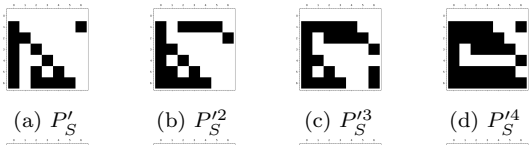
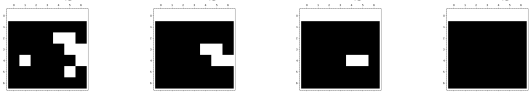


Fig. 5. Digraph $\mathcal{G}'(\Sigma_\Lambda)$ related to the new LTI system with relative degree $r = 1$. The dotted edge is added.

Figure 6 gives the diffusion property in this new mode of operation. Clearly, the diffusion is now much better achieved.



(a) P'_S (b) P'^2_S (c) P'^3_S (d) P'^4_S



(e) P'^5_S (f) P'^6_S (g) P'^7_S (h) P'^8_S

Fig. 6. Powers of P_S^p for $p \in [1, 8]$, $r = 1$. Black squares correspond to non zero entries.

However, it is worth pointing out that adding a non zero entry in the matrix I_A and so on the matrix P induces a corresponding digraph $\mathcal{G}'(\Sigma_\Lambda)$ that does no longer meet Conditions **C0-C2** guaranteeing flatness. And yet, let

us recall that flatness is necessary to ensure the self-synchronization. This is the reason how a hybrid architecture is proposed as detailed below.

The hybrid architecture with switching rule σ obeys the following state space representation at the cipher part:

$$x_{t+1} = \begin{cases} A_{\rho(t)}x_t + Bm_t \text{ if } \sigma(t) = 1 \\ A'_{\rho(t)}x_t + Bm_t \text{ if } \sigma(t) = 2 \end{cases} \quad (24)$$

with $x_t \in \mathbb{F}^n$ is the state vector of the cipher, $m_t \in \mathbb{F}$ is the plaintext input. The matrices $A \in \mathbb{F}^{n\times n}$ and $A' \in \mathbb{F}^{n\times n}$ are the dynamical matrices of respective modes 1 and 2 , $B \in \mathbb{F}^{n\times 1}$ is the input matrix of the cipher.
The hybrid architecture obeys the following state space representation at the decipher part:

$$\widehat{x}_{t+1} = \begin{cases} P_{\rho(t:t+1)}\widehat{x}_t + B\mathcal{T}^{-1}c_{t+1} \text{ if } \sigma(t) = 1 \\ P'_{\rho(t:t+1)}\widehat{x}_t + B\mathcal{T}^{-1}c_{t+1} \text{ if } \sigma(t) = 2 \end{cases} \quad (25)$$

All the equations (11)-(14) and (16)-(18) still hold except that A is replaced by A' for the mode 2.
Now, let us characterize each mode.

- For the mode $\sigma(t) = 1$, the system with dynamical matrix A is flat. The corresponding digraph $\mathcal{G}(\Sigma_\Lambda)$ fulfills Conditions **C0-C2** and the system gets the self-synchronization property. In other words, for any initial conditions of the cipher and the decipher and any plaintext message, the synchronization is guaranteed with a delay bounded by the dimension n of the system. However, the property of diffusion is not optimal because of the inherent triangular structure of P.
- For the mode $\sigma(t) = 2$, the system with dynamical matrix A' is not flat. The corresponding digraph $\mathcal{G}'(\Sigma_\Lambda)$ does not fulfill Conditions **C0-C2** and since those conditions are Necessary and Sufficient ones, the system does not get the self-synchronization property. On the other hand, the diffusion property is much more significant.

This being the case, the following switching rule σ is proposed. The hybrid architecture is equipped with a supervisor at both sides that manages the synchronization as follows. At the initialization, both the cipher and the decipher are in the mode $\sigma(t) = 1$. After n iterations, since the synchronization is ensured, the cipher and the decipher switch on the mode $\sigma(t) = 2$. In this mode, since the synchronization had already occurred, although the system is not flat, the synchronization is preserved (see Equation (19)). However, it may happen that a transmission error produces a bad cryptogram c_t that the decipher will receive. The cipher is not affected. Recalling that the dynamical matrices A and P of both the cipher and the decipher depend on the cryptogram (through time-varying entries in the form of S-boxes yielding an LPV dynamics), they will not coincide and a desynchronization will occur. To overcome this issue, in the mode $\sigma(t) = 2$, the supervisor at both sides is scanning on-line the sequence c of cryptograms. Let us notice that both the cipher and the decipher have knowledge of this sequence. The sequence is compared on-line with a cryptogram of reference (called sync pattern) of length $l \in \mathbb{N}$. It is denoted by $(p_0, \ldots, p_{l-1})$ with $p_i \in \mathbb{F}$. Then, when they coincide, the supervisor of both the cipher and the decipher switches in mode $\sigma(t) = 1$. The same strategy applies continuously.

It must be pointed out that the sync pattern occurs in a statistically distributed way in the cryptogram c. Hence, the time before resynchronization is statistical and no longer deterministic. This is the price to pay to meet the trade-off between the quality of synchronization and the safety regarding the diffusion. Such a strategy and such hybrid architecture should not be considered as a pure engineering-based design. Actually, it follows a well-admitted framework as motivated for the first time in Jung and Ruland (1999). Essentially, the concept of statistical self-synchronization embeds the properties of two modes operations. In the context of the paper Jung and Ruland (1999), the first mode benefits from a deterministic synchronization property but suffers from low data rate while the second mode suffers from a so-called error propagation weakness but allows higher encryption rate. The LPV-based hybrid architecture presented in this section is in the same vein. It is motivated by achieving a trade-off between synchronization quality and diffusion property for security purpose.

4.3 Proof-of-Concept Example

This section gives an example that illustrates the way how the LPV-based statistical self-synchronization operates. To this end, let us considers an automaton operating over the finite field $\mathbb{F}_{16}$ defined by $\mathbb{F}_2[X]/(X^4 + X + 1)$. Then, let us consider a plaintext m with symbols $m_t \in \mathbb{F}_{16}^\ell$ to be encrypted. The nonlinearity defined by φ is given by

$$\varphi : \mathbb{F}_{16} \to \mathbb{F}_{16}$$
$$c_t \mapsto \frac{1}{c_t} + \alpha^2 \text{ if } c_t \neq 0, \alpha^2 \text{ otherwise}$$

where α is a primitive element of $\mathbb{F}_{16}$ given by $\alpha^4 + \alpha + 1 = 0$. Two sync patterns of respective length 1 and 2 are used. The first one is a singleton α^2+1 and the second one is the sequence $(0, \alpha^2 + 1)$. The statistical SSSC has been implemented with Sagemath. Figure 7 gives the result by depicting the error $m_t - \hat{m}_{t+1}$ (let us notice that due to the relative degree, there is a delay $r = 1$ between the plaintext symbol and the deciphered one as seen in the Equation (2)).

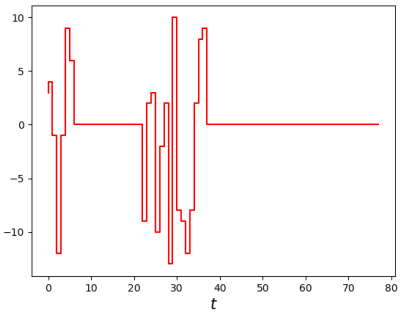


Fig. 7. Time evolution of the error $m_t - \hat{m}_{t+1}$.

A time $t = 0$, the cipher and the decipher are in mode $\sigma(t) = 1$. As expected, the synchronization is performed

after $t = 7$. Indeed, the flatness guarantees synchronization after a transient time no greater than $n = 7$. Then, for $t \geq 7$, the supervisor of both the cipher and the decipher switches to mode $\sigma(t) = 2$. At $t = 21$, mimicking a disturbance (bit slip for example), the cipher c_t is corrupted by changing the value delivered by the cipher. As expected, that causes a desynchronization and no self-synchronization can occur since the mode $\sigma(t) = 2$ is not flat. The bit pattern appears at $t = 30$. Then, the supervisor switches the cipher and the decipher in mode $\sigma(t) = 1$. After $t = 37$, the self-synchronization is again recovered. If no disturbance occurs, the synchronization error remains indefinitely at zero and the deciphering is achieved properly.

Finally, to illustrate the impact of the length of the sync pattern, for arbitrary messages and arbitrary initial conditions, 1000 runs have been performed. The ratio between the number of successful resynchronizations after a time t and the total number of runs has been reported in Figure 8 for a pattern's length l equal to 1 and 2 respectively. As expected, the plots show that the more the length of the sync pattern, the more the time before resynchronization. Indeed, the probability that the sync pattern occurs decreases with respect to the length l. Besides, the plots tend towards 1 as the time t before synchronization tends towards infinity as expected as well.

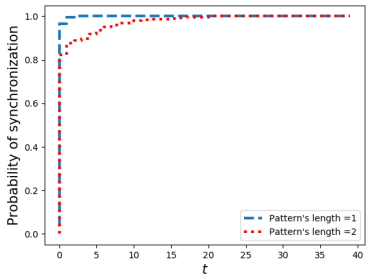


Fig. 8. ♯ successful resynchronizations after a time t/ ♯ total number of runs for a pattern's length equal to 1 and to 2 respectively

5. CONCLUSION

In this paper, we have proposed a new architecture of statistical self-synchronizing stream ciphers. It involves an hybrid architecture with two modes of operations, each one governed by an LPV model exhibiting a nonlinear dynamics. It has been explained how the design problem can be recast as a switched system design under security constraints. It has been shown that control-theoretic concepts like flatness and structural analysis are relevant to this end. We have focus on a specific security criteria, that is diffusion. The results should be considered as a proof-of-concept but they are encouraging and deserve further work. Checking the scalability of the proposed framework and addressing other security properties before defining a complete real-world cipher will be challenging issues in a near future.

REFERENCES

Abdelwahab, A., Lucia, W., and Youssef, A. (2021). Covert channels in cyber-physical systems. *IEEE Control Systems Letters*, 5(4), 1273–1278.

Arnault, F., Berger, T.P., Minier, M., and Pousse, B. (2011). Revisiting lfsrs for cryptographic applications. *IEEE Transactions on Information Theory*, 57(12), 8095–8113.

Darup, M.S., Alexandru, A., Quevedo, D., and Pappas, G.J. (2020). Encrypted control for networked systems - an illustrative introduction and current challenges. *ArXiv*, abs/2010.00268.

Fliess, M., Levine, J., Martin, P., and Rouchon, P. (1995). Flatness and defect of non-linear systems: introductory theory and examples. *Int. Jour. of Control*, 61(6), 1327–1361.

Francq, J., Besson, L., Huynh, P., Guillot, P., Millerioux, G., and Minier, M. (2020). Non-triangular self-synchronizing stream ciphers. *IEEE Transactions on Computers*, 1–1. doi:10.1109/TC.2020.3043714.

Ge, X., Yang, F., and Han, Q.L. (2017). Distributed networked control systems: A brief overview. *Information Sciences*, 380, 117–131.

Geraldy, A., Pfitzmann, B., and Sadeghi, A.R. (2001). Optimized self-synchronizing mode of operation. In *Proceedings of Fast Software Encryption International Workshop (FSE'2001)*.

Heys, H.M. (2001). An analysis of the statistical self-synchronization of stream ciphers. In *In Proceedings of IEEE INFOCOM 2001*, 22–26.

Jung, O. and Ruland, C. (1999). Encryption with statistical self-synchronization in synchronous broadband networks. In Ç.K. Koç and C. Paar (eds.), *Cryptographic Hardware and Embedded Systems*, 340–352. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kasper, E., Rijmen, V., Bjorstad, E., Rechberger, C., Robshaw, M., and Sekar, G. (2004). Correlated Keystreams in MOUSTIQUE. Technical report, ESTREAM Project.

Maurer, U.M. (1991). New approaches to the design of self-synchronizing stream ciphers. In *Advances in Cryptology - EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, 458–471. Springer.

Menezes, A.J., Oorschot, P.C., and Vanstone, S.A. (1996). *Handbook of Applied Cryptography*. CRC Press.

Millérioux, G. and Boukhobza, T. (2015). Characterization of flat outputs for lpv discrete-time systems: a graph-oriented approach. In *54th Conference on Decision and Control (CDC 2015)*. Osaka, Japan.

Sánchez, H.S., Rotondo, D., Escobet, T., Puig, V., and Quevedo, J. (2019). Bibliographical review on cyber attacks from a control oriented perspective. *Annual Reviews in Control*, 48, 103–128.

Shoukry, Y., Nuzzo, P., Bezzo, N., Sangiovanni-Vincentelli, A.L., Seshia, S.A., and Tabuada, P. (2015). Secure state reconstruction in differentially flat systems under sensor attacks using satisfiability modulo theory solving. In *2015 54th IEEE Conference on Decision and Control (CDC)*.

Sira-Ramirez, H. and Agrawal, S.K. (2004). *Differentially Flat Systems*. Marcel Dekker, New York.