DOI: http://dx.doi.org/10.26483/ijarcs.v8i8.4746

ISSN No. 0976-5697

Volume 8, No. 8, September-October 2017

International Journal of Advanced Research in Computer Science

SURVEY REPORT

Available Online at www.ijarcs.info

A SURVEY REPORT ON TECHNIQUES FOR DATA CONFIDENTIALITY IN CLOUD COMPUTING USING HOMOMORPHIC ENCRYPTION

Dhruva Gaidhani
Dept. of Computer Science
Fr. Conceicao Rodrigues College of Engineering
Mumbai, India

Joshua Koyeerath
Dept. of Computer Science
Fr. Conceicao Rodrigues College of Engineering
Mumbai, India

Neel Kudu
Dept. of Computer Science
Fr. Conceicao Rodrigues College of Engineering
Mumbai, India

Prof. Mahendra Mehra
Dept. of Computer Science
Fr. Conceicao Rodrigues College of Engineering
Mumbai, India

Abstract: **In order to store information, access content ubiquitously and run processes remotely, without the concern of existing infrastructure, a majority of users have resorted to using Cloud based technology. Cloud computing uses shared pools of configurable resources to provide varying computational capabilities to store and process data. This service is broadly classified into private clouds and third-party operated clouds. Because of the shared nature of this technology, cloud computing poses confidentiality concerns. Important data might be leaked either by the service provider itself, accidentally or intentionally, or by an external attacker who manages to gain unauthorized access. The solution is data encryption. However, encrypting data will make it unusable for computations or processing. Homomorphic Encryption techniques allow encrypted data to be processed like plain text data to give the same output. Thus the service provider will have access to encrypted data and can perform operations on it as well; however, the original data will be unknown. In this paper, we aim to present a survey on homomorphic encryption techniques from different categories and a comparative analysis of their application to maintain data confidentiality in cloud computing.**

Keywords: **Cloud Computing, Cloud Security, Confidentiality, Cryptography, Homomorphic Encryption Algorithms, Integrity**

I. INTRODUCTION

CLOUD computing has gained increasing popularity because of the wide range of services it provides. Apart from Infrastructure-as-a-service (IaaS), which is providing on-demand physical computer resources, Platform-as-a-Service (PaaS), which is providing appropriate environment to application developers and Software-as-a-Service (SaaS), which is providing the user with application software and databases, Cloud computing also features big data analytics, disaster recovery and cloud based backup. However, incidents like the 2014 Dropbox security breach and the iCloud leak [1] critically call into question the confidentiality of the user data stored in the Cloud; accessible to the service provider.

A solution to this issue is encryption of data prior to the storage. However, while encryption satisfies security constraints it largely reduces the usability of the data. Thus computations cannot be performed and the cloud is reduced to a remote storage. To circumvent this issue, we use Homomorphic Encryption Schemes. These techniques allow us to use encrypted data as an input to various processes. The results so obtained are also encrypted and can only be deciphered by the designated receiver who has the key [2]. Thus, the confidentiality of the original data is maintained.

However, while there are multiple variations of Homomorphic Encryption algorithms; performing arbitrary operations over encrypted data incurs a large penalty in terms of overhead. Time and computational power needs to be redirected for this. Hence, there is a need to identify the appropriate application of the various available Homomorphic Encryption based algorithms and evaluate them based on their feasibility of use.

In section 2 we briefly discuss about the background of Homomorphic encryption algorithms. In section 3 we discuss the functions and properties of homomorphic encryption algorithms. We move on to explain the available algorithms according to the respective categories. In section 4 we provide a comparative analysis of the aforementioned techniques. In section 5 we discuss possible paths research related to this topic might take. Finally, we conclude the paper in section 6.

II. HOMOMORPHIC ENCRYPTION

Homomorphic Encryption is the conversion of plain text data into cipher text that can be analyzed and processed as if it were in its original unencrypted form. Mathematically, we say that an encryption system is homomorphic if:

From Enc(x) and Enc(y), it is possible to calculate Enc(f(x, y)), without using the private key of the sender, where f can be:

+ (addition), × (multiplication), ⊕ (XOR) [3]

This concept was first put forth by Ronald Rivest, Leonard Adleman and Michael Dertuzos in 1978. In this section we will first discuss the extant properties of homomorphic encryption, followed by the general process of homomorphic encryption and some important classifications. The next section will be based on the categorization provided in this section.

A. *Properties of Homomorphic Encryption*

We broadly categorize these into two:

Additive Homomorphic Encryption: Mathematically this can be described as [4]:

$$Ek\ (PT1 \oplus PT2) = Ek\ (PT1) \oplus Ek\ (PT2)$$

Additive homomorphic encryption allows the following identities:

i) The product of two cipher texts will decrypt to the sum of their corresponding plaintexts,

$$D\ (E\ (m1, r1) \cdot E\ (m2, r2) \bmod n^2) = m1 + m2 \bmod n.$$

ii) The product of a cipher text with a plaintext raising o will decrypt to the sum of the corresponding plaintexts,

$$D\ (E\ (m1, r1) \cdot o^{m}{}_{2} \bmod n^2) = m1 + m2 \bmod n.\ [5]$$

Multiplicative Homomorphic Encryption: Homomorphic encryption is multiplicative, if [5]:

$$Ek\ (PT1 \otimes PT2) = Ek\ (PT1) \otimes Ek\ (PT2)$$

B. *Categorization of Homomorphic Encryption*

We classify the algorithms based on the above mentioned properties into

- **Partially Homomorphic Encryption** (PHE): Permits operations on encrypted data like either multiplication or addition, but not both [6].

- **Somewhat Homomorphic Encryption** (SWHE): Permits more than one operation –multiplication and addition, but the number of operations is limited [6].

- **Fully Homomorphic Encryption** (FHE): Permits multiple – multiplication and addition operations without a restriction on the number of operations [6].

C. *General process*

The general process of the Homomorphic Encryption systems is: [7], [8], [9].

- Key Generation: The client generates a public key (pk) and a private key (sk).
- Encryption: The client encrypts data with encryption key (pk or pk+sk).
  C_t = E (Sk (Pk))
- Storage: This encrypted data C_t and pk, are stored in the cloud database.
- Request: The client requests the server to retrieve the data or to perform operations on encrypted data.
- Processing: The processing server processes the request and performs the operations requested by the client. This as per the required function using Pk.
  EvlPk (f, ESk (a), ESk (b))
  EvlPk (f, C_{t1}, C_{t2}) [9]
- Response: Cloud provider returns to the client the processed result.
- Decryption: The client decrypts the returned result, using sk.
  Result= DSk (EvlPk (f, ESk (a), ESk (b)))
  Result=DSk (EvlPk (f, C_{t1}, C_{t2}))

PT=DecSk (Pk, Ct) [9]

III. HOMOMORPHIC ENCRYPTION TECHNIQUES

Of the various techniques available we shall present at least one of each category. We shall also specify the operation that is permitted in each algorithm. In the next section we compare our findings.

A. *RSA – Multiplicative |PHE*

Rivest, Shamir and Adleman published their public key cryptosystem in 1978 [10]. Although it is a very basic algorithm it is one of the most crucial building blocks of homomorphic encryption which is why it has been included as an example of multiplicative partial homomorphic encryption technique. Other techniques include El Gamal.

1. Key Generation
Step 1: User creates a public/private key pair by selecting, two large primes at random - p, q Step 2: System modulus N is calculated = p.q Euler's totient ø(N) = (p-1)(q-1) Step 3: selecting at random the encryption key e Where, 1 < e < ø(N) | gcd(e,ø(N)) = 1 Step 4: Public key: PK = {e,N} Private key: SK = {d,p,q}

2. Encryption
Step 1: To be used – public key PK = {e,N} Step 2: computes: $C = M^e \bmod N$, where $0 \leq M$ Encrypted text C is obtained.

3. Decryption
Step 1: To be used - private key SK = {d,p,q} Step 2: computes: $M=C^d \bmod N$ to get back the original plain text message

Figure 1: RSA Algorithm

Following figure shows the homomorphic property of the RSA.

Suppose there are two cipher texts, CT1 and CT2.

$$CT1 = m_1^e \bmod n$$
$$CT2 = m_2^e \bmod n$$
$$CT1 \cdot CT2 = (m_1^e \cdot m_2^e\ ) \bmod n$$

Thus, multiplicative property: $(m1m2)^{e}$ mod n [9] is displayed.

It is apparent that RSA is a basic algorithm that provides us with limited computation options. This greatly reduces actual practical applications of this algorithm. However, it is a very important algorithm because it acts as a building block and many enhanced algorithms are based on RSA or use it for some part of the implementation. It is also important to note that comparatively RSA is fast and can be feasibly implemented. An alternative to RSA is the El Gamal algorithm [10].

B. Goldwasser-Micali System (GM) – Additive|PHE

The Goldwasser-Micali(GM) system is a probabilistic (asymmetric) public-key encryption scheme, developed by Shafi Goldwasser and Silvio Micali in 1982. It is an additive Homomorphic Encryption, but it can encrypt just a single bit [3][11]. It provides data confidentiality however it is not efficient in terms of space complexity because in several cases the cipher text generated is many times larger than the input plain text. GM algorithm as shown in the figure

1. Key Generation
Step 1: Each user generates a public/private key pair by selecting, two large primes at random - p, q Step 2: computing their system modulus N = p.q Choose b € B_n | b is a quadratic non-residue modulo n and (b/n) = 1 Step 3: the key pair is obtained as (pk, sk) Where pk = (n, b) {public key} And sk = (p, q) {private key}

2. Encryption
Input message M is made up of multiple bits m1,m2…m3 Step 1: For every bit $m_{i,}$ a random value y_i is generated | $gcd(y_i,N) = 1$ Step 2: compute: $C_i = y_i^2 x^{m}_{i}$ (mod N)

3. Decryption
Input message C is made up of multiple bits c1,c2…c3 Step 1: calculate: $e_i = (c_i/p)$, ∀ i € [1, t] Step 2: if e_i = 1then m_i = 0 else m_i = 1 Output message is M = m1,m2,…m3

Figure 2: Goldwasser-Micali Algorithm

Assume we have to encrypt two bit: m1 and m2 using the Goldwasser Micali cryptosystem.
Additive:
EncGM (m1) × EncGM (m2) ≡ $(b^{m1}\times r1^2).(b^{m2}\times r2^2)$ mod n
≡ b^{m1+m2} $(r1r2)^2$ mod n
≡EncGM(m1⊕m2 , pk) [3]

C. The Sander-Young-Yung Cryptosystem (SYY) – SWHE

Circuits involving OR and NOT gates in a bidirectional communication have been put forth by Sander, Young and Yung. In these studies one party is assumed to have knowledge of a circuit computing some secret function f, and the other has a secret input x for which it would like to learn this function and its behavior. Their protocol could be used to calculate AND by DeMorgan's theorem. They also present a separate AND-homomorphic cryptosystem. This system takes a major inspiration from the Goldwasser-Micali cryptosystem. [12]

1. Key Generation
Step 1: security parameter : *e* a positive non-zero integer: *l* Step 2: Create an instance of GM Cryptosystems using security parameter *e* Gen(*e, l)* = *(n, p, q, m)* Public key = (n, m, l) Private key = p

2. Encryption
Input message M is made up of multiple bits m1,m2…m3 Public key is pk PT=CT=Z_n Step 1: Enc(pk, m) generates a vector v ∈ (Z_2) l to return cipher text c Step 2: compute: c = E(pk, v) = (E(pk, v1), E(pk, v2),..., E(k, vl))

3. Decryption
Input message C is made up of multiple bits c1,c2…c3 Private key is sk Step 1: calculate: v = D(sk, c) = (D(sk, c1), D(sk, c2),..., D(sk, cl)) Where D is the decryption function from GM cyptosystem Step 2: M = Decode(v). Output message is M = m1,m2,…m3

Figure 3: SYY Algorithm

The homomorphic operation of the Goldwasser-Micali cryptosystem is performed by taking the product of two ciphertexts. This is a binary XOR.

Let us assume that we have two ciphertexts c1 and c2 which are respectively the encoded counterparts of plain text m1 and m2; the encryption of m1.m2 can be calculated by choosing two random non-singular matrices:

$A, B \in (Z_2)^{lxl}$,
Combined cryptotext, $c = Ac_1 + Bc_2$. [12] [13]

The homomorphic AND operation can be performed by a series of homomorphic XOR operations on the encrypted vectors which are given as input to the algorithm.

D. The Boneh-Goh-Nissim Cryptosystem (BGN) – SWHE

The cryptosystem devised by Boneh, Goh, and Nissim [14] was the first to allow both additions and multiplications with a constant-size ciphertext. There is a catch, however: while the additive property is the same as for the ElGamal variant, only one multiplication is permitted. The system is thus called "somewhat homomorphic."

1. Key Generation
Step 1: select, two large primes at random - p, q Step 2: Find a supersingular elliptic curve E/Fp with a point P of order n W = <P>. Step 3: Choose Q' ← W \ {∞} and set Q = [r]Q' ; then Q has order q. Let *e* : W × W → μn ⊂ F_p^2. Step 4: Output the public key pk = (E, *e*, n, P, Q) and the private key sk = q.

2. Encryption
Step 1: Choose t ← [1, n] and output C = [m]P + [t]Q.

3. Decryption
Step 1: Compute P` = [q]P and C` = [q]C, and output m' = $\log_{P`}$ C`.

Figure 4: BGN Algorithm

We know that this algorithm allows us multiple additions but only one multiplication. This is how we proceed with the encryption: [15]

Assume that we have two cipher texts C1 and C2. Pk is the public key.

Addition (pk, C1, C2):
Choose t' ← [1, n] and output C' = C1 + C2 + [t']Q ∈ G
-performed repeatedly

Multiplication (pk, C1, C2):
Let u ← [1, n] and output D = *e*(C1, C2) · e(Q, Q) ^{u} ∈ μ_n.

Performed only once because if repeated more than once we end up losing the key pairing on $\mu_{n.}$ Thus we will lose any relation with the encrypted data and retrieving the original data from the encrypted data will no longer be possible.

E. Enhanced Homomorphic Encryption Scheme Cryptosystem (EHES) – FHE

Enhanced homomorphic Encryption Scheme (EHES) for homomorphic encryption as well as decryption was posited by Gorti Vaikuntanathan Subba Rao, supplemented by research from Garimella Uma in 2013. This was carried out with the IND-CCA secure system. This Cryptosystem presents operations like addition, multiplication individually and mixed as well [16].

1. Key Generation
Step 1:select two large primes at random - p, q | q < p Step 2: computing their system modulus N= p.q Output is (pk, sk) Where pk = (n) {public key} And sk = (p, q) {private key}

2. Encryption
Input message M ∈ Z_p Step 1: generate a random number *r* Step 2: compute: C = m + r x p^q(mod n)

3. Decryption
Input message is C Step 1: calculate: m = c mod p Output: m ∈ Z_p

Figure 5: EHES Algorithm

Let a, b ∈ Zp, pk = (n) and sk =(p, q)

Multiplicative: EncEH (a × b) ≡ (EncEH (a) × EncEH (b)) (mod n), or
a × b = DecEH (EncEH (a) × EncEH (b))
≡ (EncEH (a) × EncEH (b)) (mod p)

Additive: EncEH (a + b) ≡ EncEH (a) + EncEH (b) (mod n), or
a + b = DecEH (EncEH (a) + EncEH (b))
≡ (EncEH (a) + EncEH (b)) (mod p)

F. ***Algebra Homomorphic Encryption Scheme based on Updated El Gamal (AHEE) – FHE***

The Algebra Homomorphic Encryption Scheme is a modified version of the DSS i.e., Digital Signature Standard proposed by NIST [17]. This algorithm is highly secure and can reliably maintain data confidentiality against plain text attacks and similar threats, thus resulting in varied applications. The AHEE can be referred to as subset of the fully homomorphism as it allows addition as well multiplication to be performed on encrypted data without any cap limits on the number of operations performed.

1. Key Generation
Step 1: select any two prime numbers - p and q, preferably large
Step 2: computing their system modulus N= p.q
Step 3: select random number *h* Select a root *r* of GF(p) | a, r < p.
Step 4: calculate $y = r^h \bmod p$.

2. Encryption
Input message M is made up of multiple bits m1,m2…m3
Step 1Select random integer number I E_1 (M) = (M+i*p) mod N.
Step 2: Choose a random integer k such that:
$E_r (M) = (a,b) = (r^k \bmod p, y^k E_1 (M) \bmod p)$

3. Decryption
Input message C is made up of multiple bits c1,c2…c3
Step 1: $M = b\times (a^h) -1 \pmod p$
Output message is M = m1,m2,…m3

Figure 6: AHEE Algorithm

Consider two messages M1 and M2 to be stored in an insecure environment:

Multiplicative:
EnAH (M1M2) = EnAH (M1)·EnAH (M2), or
M1.M2=DeAH (EnAH (M1)·EnAH (M2)).

Additive:
EnAH (M1+M2) = EnAH (M1) ⊕ EnAH (M2), or
M1+M2=DeAH (EnAH (M1) ⊕ EnAH (M2)).

These equations demonstrate additive and multiplicative properties of Homomorphic Encryption as seen in Algebra Homomorphic Encryption Scheme Based on Updated El Gamal (AHEE).

IV. Comparative study of the existing techniques

As we have seen in the previous section, security is an important concern of the cloud computing environment. There are multiple techniques belonging to various categories that have been introduced to provide confidentiality. However, a fine balance between feasibility, speed and confidentiality is desired. Since users are looking for techniques that will provide the optimum results for them, we compare the techniques based on multiple relevant parameters – the category, the type of operations that can be performed and the possible applications based on this data to assist with the decision making process.

HE Technique	Category	Operation possible
RSA	Partial HE	Multiplicative
GM Cryptosystem	Partial HE	Additive
SYY	Somewhat HE	AND Operation
BGN	Somewhat HE	Unlimited additions, one multiplication
EHES	Fully HE	Mixed
AHEE	Fully HE	Mixed

Figure 7: Categorical classification of Algorithms

HE Technique	Suggested real life application
RSA	Credit Card transactions and Net Banking applications
GM Cryptosystem	For confidential E-voting and auctions, biometric-based authentication. [21]
SYY	Mobile Ad hoc networks for basic computations
BGN	Multiparty restricted computation
EHES	Consumer privacy in advertising, Financial Privacy [18]
AHEE	Medical applications, Data mining, Forensic image recognition [18]

- **Figure 8: Possible applications**

V. Future Works

When it comes to making a decision about the application of a particular Homomorphic encryption technique, we need to take into several factors such as the operations possible, the overhead involved in terms of time and space complexity and the type data to be encrypted. To combine the advantages observed in two standalone techniques, we have come across research that aims to create a hybrid Homomorphic Encryption algorithm [19]. As the research goes on to explain how RSA and El Gamal techniques are used in tandem to provide enhanced data confidentiality thereby augmenting the applications of such techniques. To provide a brief idea of this research, the algorithm used is such that plain text data is encrypted with RSA at the outset. This encrypted data is treated as the input data to a system which further encrypts it again with El Gamal. Computations are performed on this data and the decryption process follows the exact same process in the reverse manner.

We believe that, combining existing techniques in such a manner as to reduce the shortcomings while enhancing the advantages provided individually is promising area to carry out research. In our future works, we propose to compare the results of performing data mining on information stored on a system built using OwnCloud [20] which is encrypted using a hybrid algorithm with that of existing fully homomorphic encryption techniques like AHEE and EHES.

VI. Conclusion

Cloud computing offers individuals and businesses- small or large a wide range of services as mentioned above.

However, potential users are still hesitant when it comes to performing sensitive operations using cloud based technology. Most of this hesitation stems from lack of data confidentiality due to possible data leak. With this paper, we provide the reader with a basic understanding of Homomorphic encryption which is proves to be an outstanding solution to the issues regarding data confidentiality. We also provide a categorization of the various Homomorphic encryptions based schemes and provide examples of each category. We further provide the reader with an overview of the operations possible with these algorithms and the real life suggested applications of the studied algorithms. The comparisons provided and the possibility of future research mentioned is done so with the aim of providing assistance to extend the existing and on-going research in this field to improve on data confidentiality in cloud based computing.

VII. REFERENCES

[1]. “Google Drive, Dropbox, Box and iCloud Reach the Top 5 Cloud Storage Security Breaches List”.psg.hitachi-solutions.com.Retrieved 2015-11-22.

[2]. Coron, Jean-Sébastien, Tancrede Lepoint, and Mehdi Tibouchi. "Practical multilinear maps over the integers." Advances in Cryptology–CRYPTO 2013. Springer Berlin Heidelberg, 2013. 476-493

[3]. X. Yi et al., “Homomorphic Encryption and Applications,” SpringerBriefs in Computer Science, chapter 2, 2014, pp. 27 – 46

[4]. Tebaa, Maha, Saïd El Hajji, and Abdellatif El Ghazi. "Homomorphic encryption applied to the cloud computing security." In Proceedings of the World Congress on Engineering, vol. 1, pp. 4-6. 2012.

[5]. Melchor, Carlos Aguilar, et al. "Improving Additive and Multiplicative Homomorphic Encryption Schemes Based on Worst-Case Hardness Assumptions}." IACR Cryptology ePrint Archive 2011 (2011): 607.

[6]. M. Ogburn, C. Turner, P. Dahal, “Homomorphic Encryption,” In Complex Adaptive Systems, Publication 3, Cihan H. Dagli, Editor in Chief Conference Organized by Missouri University of Science and Technology 2013 - Baltimore, MD, Elsevier, 2013, pp. 502 – 509.

[7]. K. EL MAKKAOUI , A. EZZATI, A. BENI HSSANE, “Challenges of Using Homomorphic Encryption to Secure Cloud Computing,” In Proceedings of the International Conference on Cloud Computing Technologies and Applications, Marrakesh, MOROCCO, 2– 4 June 2015, pp. 1 –7.

[8]. M. TEBAA and S. EL HAJJI, “Secure Cloud Computing through Homomorphic Encryption,” International Journal of Advancements in Computing Technology (IJACT), Vol.5, No.16, 2013, pp. 29 –38.

[9]. Payal V. Parmar, Shraddha B. Padhar, Shafika N. Patel, Niyatee I. Bhatt and Rutvij H. Jhaveri “Survey of Various Homomorphic Encryption algorithms and Schemes,” In International Journal of Computer Applications (0975 - 8887), Vol. 91 -No. 8, 2014 , pp. 26 – 32.

[10]. El Gamal, Taher. "A public key cryptosystem and a signature scheme based on discrete logarithms." In Advances in Cryptology, pp. 10-18. Springer Berlin Heidelberg, 1985.

[11]. S. Goldwasser, S. Micali, “Probabilistic encryption and how to play mental poker keeping secret all partial information,” In Proceedings of 14th Symposium on Theory of Computing , 1982, pp. 365 – 377.

[12]. K. Henry, “The theory and applications of homorphic cryptography”. Thesis, University of Waterloo, 2008, pp. 23 – 68.

[13]. Graig Gentry, “A fully homomorphic encryption scheme,” Ph.D. dissertration, Stanford University, 2009. Available: http://crypto.stanford.edu/craig/craigthesis.pdf.

[14]. D. Boneh, E.-J. Goh, and K. Nissim, “Evaluating 2-DNF formulas on ciphertexts,” TCC 2005, http://crypto.stanford.edu/~dabo/pubs/papers/2dnf.pdf

[15]. David Mandell Freeman “Homomorphic encryption and the BGN Cryptosystem”,2011 http://theory.stanford.edu/~dfreeman/cs259c-fll/lectures/bgn

[16]. G. VNKV Subba Rao et al., “Data Security in Bioinformatics,” In International Journal of Advanced Research in Computer Science and Software Engineering 3(11), November - 2013, pp. 590 – 598.

[17]. Smid, Miles E., and Dennis K. Branstad. "Response to comments on the NIST proposed Digital Signature Standard." Advances in Cryptology—Crypto‘92. Springer Berlin Heidelberg, 1993.

[18]. Fredrick Armknecht et al., “A guide to fully Homomorphic Encryption”,2015, pp 6-10

[19]. https://eprint.iacr.org/2015/1192.pdf

[20]. D. Chandravathi et al., “A New Hybrid Homomorphic Encryption Scheme for Cloud Data Security” Advances in Computational Sciences and Technology ISSN 0973-6107 Volume 10, Number 5 (2017) pp. 825-837

[21]. Shraddha Massih and Sanjay Tanwani “Distributed framework for data mining as a service on private cloud” Int. Journal of Engineering Research and Applications ISSN : 2248-9622, Vol. 4, Issue 11(Version 1), November 2014, pp.65-70

[22]. Julien Bringer et al., “An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication” The 12th Australasian Conference on Information Security and Privacy (ACISP ’07). (2–4 july 2007, Townsville, Queensland, Australia) J. Pieprzyk, H. Ghodosi and E. Dawson Ed. Springer-Verlag, LNCS 4586, pages 96–106.