Khulna University of Engineering & Technology
Department Of Computer Science And Engineering

CSE 4120: Technical Writing and Seminar

ANS-based compression and encryption with 128-bit security

Authored By:
Seyit Camtepe, Jarek Duda, Arash Mahboubi, Pawel Morawiecki, Surya Nepal, Marcin Pawlowski, Josef Pieprzyk

Source:
Springer IJIS
DOI:
https://doi.org/10.1007/s10207-022-00597-4

Presented By:
Partho Choudhury Shoumya
Roll: 1807021

Outline

- Problem Statement
- Objectives
- Introduction
- Related Works
- Methodology
- Result Analysis
- Discussion & Conclusion
- References

Problem Statement

- Bulk of internet interaction requires data to squeeze out redundant bits and then get encrypted to reduce communication bandwidth and provide security.
- Requires 2 independent algorithms.
- Single algorithm that does compression and encryption (called compcrypt) should provide benefits in terms of efficiency and security.

Objectives

To design a compcrypt algorithm using the ANS entropy coding so that:

- cost(compcrypt) << cost(compression) + cost(encryption)
- security(compcrypt) = security(encryption)
- comp_rate(compcrypt) ≈ comp_rate(compression)

Introduction

- In a noisy and unreliable communication channel, there remains a problem of data transmission.
- Error correcting codes were developed to deal with the errors precisely.
- Reverse problem still persists: redundancy of data.

Inroduction (Cont...)

- Entropy encoding mainly focuses on how to remove redundancy from transmitted data which is extremely important for ever growing internet applications.
- Asymmetric Numeral Systems (ANS) are a new and promising invention that provides both efficient and close to optimal entropy encoding.
- It can ensure Confidentiality, Integrity and Authenticity

Related Works

Author	Approach	Problem
Duda j. [1]	Applying Haffman coding	Suboptimal
Duda J. et all. [2]	Arithmetic Range Coding	A heavy computation overhead
Duda and Niemiec [3]	Plain ANS	Completely defenceless against integrity attack

Table 1: Comparison of related works

Proposed Methodology

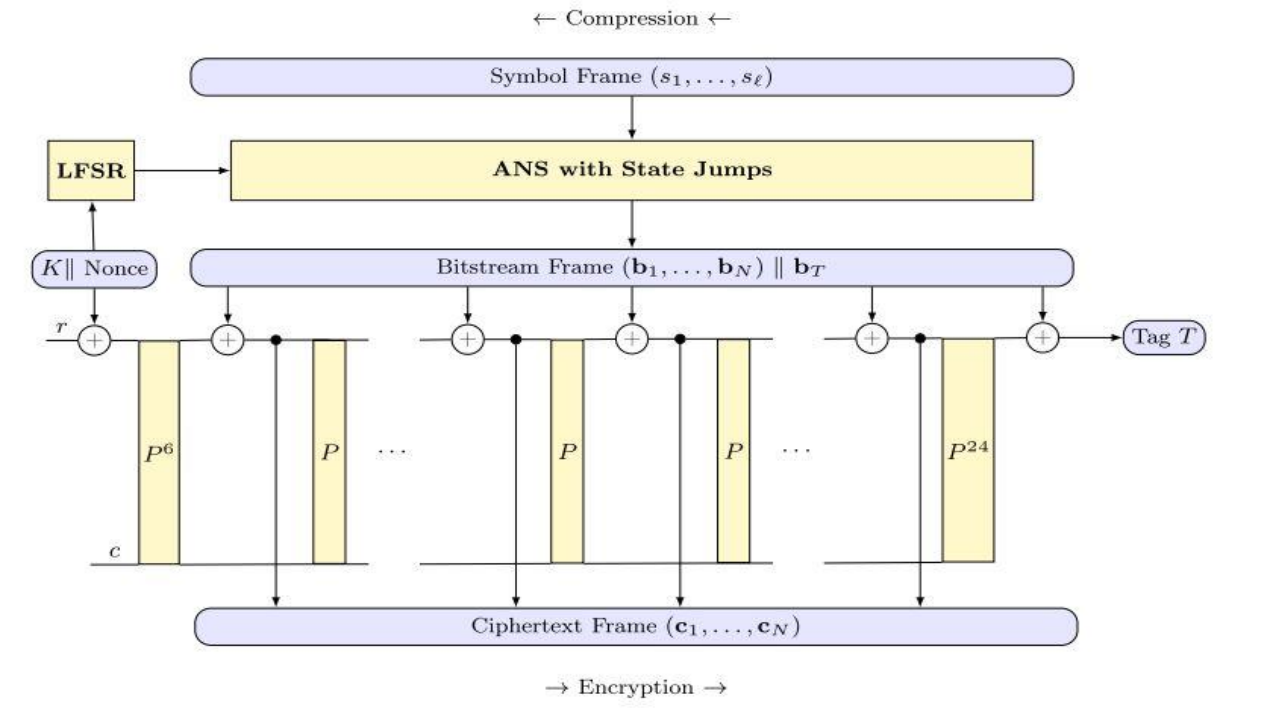

Fig1: Compcrypt based on ANS and sponge, where r = 512 bits, c = 1088 bits and size of sponge state is b = 1600bits

Methodology(cont.)

- Algorithm depicting compression and encryption in a single process.

(h) **Data**: A symbol frame $\mathcal{S} = (s_1, s_2, \ldots, s_\ell)$, a 128-bit secret key K and a 128-bit random nonce α.
Result: A ciphertext frame $\mathcal{C} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N)$ of compressed bit stream together with a 128-bit tag T, where $\mathbf{c}_i$ is a 512-bit long block; $i = 1, \ldots, N$.
begin
(1) Upload the symbol frame and compute symbol probabilities $\{p_s | s \in \mathbb{S}\}$;
(2) Initialize a sponge for $K \| \alpha$ by running $f_{start} = P^6$, where the number of iterations $n_{start} \geq \lceil \frac{R \cdot \ell}{512} \rceil$;
(3) Design ANS instance for the symbol statistics;
(4) Compress $\mathcal{S}$ in the reverse order. This creates a bitstream frame;
(5) Split the bitstream frame into 512-bit blocks $(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, where the last block is padded to the full length;
(6) Encrypt the decoding table and the bitstream frame $(\mathbf{b}_1, \ldots, \mathbf{b}_N)$ into $(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N)$ using the sponge;
(7) Generate the 128-bit tag T. First, extract 128-bit string $\mathbf{b}_T$ from ANS states when their jumps are controlled by LFSR only (no symbol compression) and then, create the tag $T = \mathbf{b}_T \oplus P(\mathbf{c}_N)|_{128}$, where $P(\mathbf{c}_N)|_{128}$ denotes the top 128 bits of the sponge state;
(8) Send the ciphertext frame and the tag T to the receiver.

Fig 2: Compcrypt algorithm

Result Analysis

Comparisons with other techniques indicating efficiency is shown in below table:

Algorithm	Efficiency MB/s			Efficiency Drop			128-bit	Comp.
	PC	RPI3	RPI4	PC	RPI3	RPI4	Security	Rate Loss
Plain ANS	220	17	60				✗	0
ANS*	**182**	**15**	**49**	**17%**	**12%**	**18%**	✗	≈0.5%
ANS* + *Keccak*	**178**	**13**	**48**	**19%**	**24%**	**20%**	✓	≈0.5%
ANS+Full_KECCAK	135	9	27	39%	47%	55%	✓	0
ANS+AES-NI	152	8	29	31%	53%	52%	✓	0
ANS+AES	100	8	29	55%	53%	52%	✓	0
ANS+ChaCha20	95	6	16	57%	65%	73%	✓	0

Table 2: Comparison of algorithms

Discussion & Conclusion

Prominent features:

1. Randomized ANS usage provides protection against statistical attack.
2. 128 bit security provides protection against brute-force attack.
3. Flexible enough to scale the security and speed levels.

Some Caveats:

1. Compression rate is slow which could lead to bigger problems for large systems.
2. Compression loss is shown to be 0.5% which can lead to significant penalty if the algorithm is applied to large system.

References

- [1] Duda, J.: Asymmetric Numeral Systems as Close to Capacity Low State Entropy Coders
- [2] Duda, J., Tahboub, K., Gadgil, N.J., Delp, E.J.: The use of asymmetric numeral systems as an accurate replacement for Huffman coding. In: Picture Coding Symposium (PCS), Cairns, QLD, Australia, 2015, pp. 65–69
- [3] Jarek Duda and Marcin Niemiec. Lightweight compression with encryption based on asymmetric numeral systems. arXiv preprint arXiv:1612.04662, 2016

Thank you