Modes of Operation for Compressed Sensing based Encryption

DISSERTATION
zur Erlangung des Grades eines Doktors
der Naturwissenschaften
Dr. rer. nat.

vorgelegt von
Robin Fay, M. Sc.

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2017

1. Gutachter: Prof. Dr. rer. nat. Christoph Ruland
2. Gutachter: Prof. Dr.-Ing. Robert Fischer
Tag der mündlichen Prüfung: 14.06.2017

To Verena ...

s7+OZThMeDz6/wjq29ACJxERLMATbFdP2jZ7I6tpyLJDYa/yjCz6OYmBOK548fer
76zoelzF8dNf/0k8H1KgTuMdPQg4ukQNmadG8vSnHGOVpXNEPWX7sBOTpn3CJzei
d3hbFD/cOgYP4N5wFs8auDaUaycgRicPAWGowa18aYbTkbjNfswk4zPvRIF++EGH
UbdBMdOWWQp4Gf44ZbMiMTlzzm6xLa5gRQ65eSUgnOoZLyt3qEY+DIZW5+NsBCAj
GBttjsJtaS6XheB7mIOphMZUTj5lJM0CDMNVJiL39bq/TQLocvV/4inFUNhfa8ZM
7kazoz5tqjxCZocBi153PSsFae0BksynaA9ZIvPZM9N4++oAkBiFeZxRRdGLUQ6H
e5A6HFyxsMELs8WN65SCDpQNd2FwdkzuiTZ4RkDCiJ1Dl9vXICuZVx05StDmYrgx
S6mWzcg1aAsEm2k+Skhayux4a+qt19sDJ5JcDLECo8acz+RL7/ovnzuExZ3trm+O
6GN9c7mJBgCfED keror5Af4VHUtZbD4vALyqWCr42u4yxVjSj5fWIC9k4aJy6XzQ
cRKGnsNrV0ZcGokFRO+IAcuWBIp4o3m3Amst8MyayKU+b94VgnrJAo02Fp0873wa
hyJlqVF9fYyRX+couaIvi5dW/e15YX/xPd9hdTYd7S5mCmpoLo7cqYHCVuKWyOGw
ZLu1ziPXKIYNEegeAP8iyeaJLnPInI1+z4447IsovnbgZxM3ktWO6k07IOH7zTy9
w+0UzbXdD/qdJI1rENyriAO986J4bUib+9sY/2/kL1L7nPy5Kxg3Et0Fi3I9/+c/
IYOwNYaCotW+hPtHlw46dcDO1Jz0rMQMf1XCdn0kDQ61nHe5MGTz2uNtR3bty+7U
CLgNPkv17hFPu/lX3YtlKvw04p6AZJTyktsSPjubqrE9PG00L5np1V3B/x+CCe2p
niojR2m01TK17/oT1p0enFvDV8C351BRnjC86Z2OlbadnB9DnQSP3XH4JdQfbtN8
BXhOglfobjt5T9SHVZpBbzhDzeXAF1dmoZQ8JhdZ03EEDHjzYsXD1KUA6Xey03wU
uwnrpTPzD99cdQM7vwCBdJnIPYaD2fT9NwAHICXdlp0pVy5NH20biAADH6GQr4Vc
DhGjzjaFnVbluHvsxKO0Q6IGDeW/F+xGb9iOPwDgTAxRho/4OIBaRDt2nhN4k9c0
1whVbff2T1VblRJUSjCAZFKRr/3P8siZ5gqVD7zNMCNmLwQDGKNQanOXpscVWuJ3
A4ETETcvRouPdL0GE0DioyiCvA5ljNfLtqTjNkuje3NKdJv/PZ6Xo13rCLZQM14o
KGJA8EybOUjsXXYJYqsUdm9Azk0yCblWu5QRIsssmGG6ftIo7jOlN2OAjTcegMfl
PK8GiRCHj8lb2YVurnuLVQpS6TAghOaYQrHeR/j+cSSlFrHGmUfDMO34v8VSzMPA
ynFNvSA6oZs79hSqOZxYHiSCcvjE5gISRHTmIGrcHhbvgpoO+7VEgL6sL95tJ3+N
RCuQbf3iHBk5Oy4hp/r1zd8IKa18qOrlV+viKJ188uwrSzB7PmORr3UPSYotJroH
WUVgZr+CaljaiLhWCwpZxBe8SLzJwWkCV6t9CnCBTg0skD1Aud4v8aOKbzgzKjc7
Ji1FVHo+VbjiIHsilSvgvKQS7AAIIWHbZJWqEEmgyALB9BQj5rloukBHJ08cFUXt
6jYSJhR4gWIY+MneoTNUM5M7vq2Oj5zcW/KxKhqdGBfGVnX0qBXZ5zrIqTqVIK06
wdMldXPpQwbmIltEshf+XmGGMJF5qKYu2I9OCUjvnYia1mgez0PRwcQpCQXxqAFH
C0qnjggf0Vo2PQV8UqngI3fh0fBtuR7yb0ULBAShDTnduTwtWD87BYgixLobIUkW
eN9s8itDfsu1gEDuUMNl2AB2h5/rphW28ULofSP5lQh0H8NfuX7A8ibIPr3GQe2+
LrGHEUgJox6QBNcFmKhwIjA1VjufYzbQR5v07x/4Kb7fwF5XubS195pPlci+fbGB
/yeZYjs9NjKo1R3ch8DAYlY0wu4JImpf0nb9q/EzVsSW5lWi3dgqCWW5g2STAyRf
AkzBvaPGSGDdin/Pseq59dQ4GKSa1XQZrI5XounNjBYsRVckiRAv7AZDb9kL9X6Q
XoLdacRYYBw4UcQhYN9PmNzwVNSv0hTS17v9qTQz3H/M5v010zFjD4doIHa5jr2c
MaPtAInhbzPsIhPd/Iadg3CSENHfTL2gLpyDS1vzA4aSc1S8oqMQSE9Uburo7hnH
RBZWwUtHiD0U0qs1ftEEVRZYqhaT5NA8IuzOEBoRr7orkArQQ1OLAohtbLF5Tqh9
ONi1HPHkobukE3J9aqXL+qJG5vEs9vWKr4kkIuPAxjxqVqvq3zkocYv2YdZil4Pq
VWV6YPmC2qqAxlpsAfcZpNBW7ie47rzeM0rnI2EtwVAwX1Rf7gE9EHgAnlHDB28E
vpwTYnvgVVrHH9i8h1U1zOkC6ZgE4O2Uz+vWuUKHlan4CBTIASRuHpRcZ9IPnK0L
GEcuJp8YPgXtyfKR49nwPBKjwf6EFRfd71XRXWE9vMGfF+mNfLcgIvVGZOKQGQus
Ayjsy5bD8bz8w2cklbcwQ7Up948P2dQl48FTFmeoSot7lplVYRvbfXh6yEA2cKtD
IWk7aAvESzDob4UwAypgujAe/rjWd1/X9VpHB6Zjx358AmvOUfyICTUL495NC3xq
RhndrJ1vGiCLs68vBvepSaoMJDG2cG7VBstOGYERuuG2DCDJZsZMtPSHM7vKZRjC
ootOT5H6GPjY8KEFvVR0REFib2wG/X9zSQlTdPKGqNweZ9ggSpU+VR8GamaisQFb
iT6sm2uHM6hcc7aqwl22OJEp75e4vrvuiwdwj94AiAVarvxKLFVgcvHCAYv2HbaJ
BTM1uU0BLKWDIZ5I/yRPLc0s+fma3pY2noYl6YwOL6sboi6FyYSXtmtRUm8SdqaP
DsBSoCVEU7IgzDgVAieDheNK0kAdASwUxGRFgI9UNzTVKBxRydwtlXIJ6pNSJtrb
b8gz/QLK5wbAdgo5Wsj3X0hox97fu47iUgAT2H543zgZbApP3cVwMpwL73GbmVRG
zcco/LbaLDtXxm+v+M8x5rF6HddpKNoECi5PF0j9WWR0PfGgDN10qCfj08tw3rlA
A6idjrnYQYfdeaWAOMtC5J5krPtFCW03Bk1nw1Zl73dZ8qhiKqJqHfplPzWYUsJ5
Tmwk3Oks7DUTOX6NGni6VHD0W+kw4SAbhZ1MgCxpTK8Wly6BrIt07c2DBr17UK2v
CRqNk2KmCwgMgcRJRiupnve5NBZsZtRiJBSx3jzphLyRezKiy8axnXbs8fLkxZrv
N2cQkNYcmJhacpDJeW6d2lzseUIZ9ZXHhIznpOBkL+lHLE8Z4xwMrMk1KMebQ4Ka
AN7mgdukJocfvJowBRlMvXGOt7KIrDY8/sJDF8vn9XahSQ+kJOLJI6BkafZ/eczD
yJwYepCJ/YvzIIYnBLgqp3Wvc6bra6BPKT5SPvqpGjys8ZbDfjE2O4/j0gRnnhiU
mXWNxHp5gkc5U+us55XA8XaGrTziPPKMy0hAanVSoe8nWGqDySncQsaA+JLzuVd
YFOfl+30WW8pVZrEes9sPg/ExXZIUxMKxVxXerX1OxTpex/58jEOXiZYpXW5AGAs
z/I9Od+UCKDiHE7YYmQ4WKmN+RqSDWAi7x2IMI7F53qgloXLw5HTzaiPnUNRHprg
WEZ+YbwUnQOYN+hSiOxYAU8C3z+G8KkT2HhE+9A5pHxTgNQEDjO/RfQOh1rbrxtF
SEVtDZX6LF2rKe8YAEFITrbYZpMYx8EmYG+ZvKAaD1Zz/ub5xG1xvFQ+tQo6WpsT
05lwKBLu7E5l17XfTRfou686PfWHtv1t05JVDLZjt3coZVgxek+hxqaGj0PHdHnK
e0etBo28nONtDnx3c+Jfp7nJAVw824F4boQ3Y+h/2s8225RuzipHC7ObfCCD05ph
1uy/hPV8GH4eWaQnjgJnsc/0AcIiNmrfCK9CPy0CTwRuvWuRXLuk7K7OxyhgAksz
blCtdzXN4Ar87JwdFobs7U6INXmDhO+twW8kcES1OGsibAvOzwcgOrndNlvNM0DJ
I+8HRK0ICS5fzVDJWQ6Wc9H1xq7vyBiwCIN8HjBMshon2qDtegAdftFn+RvQ8CBc
VIXv8hbdofcTxEQLPyDn9Vi9s6AcNLTmdWCsheZlw38fQR+8TO2Wi74s5v3aTLGl
geXAlaJCZzku+u9hI2DqNkfKeFGJyVWK8wCz+RXNbEx3VymGDLY9ZJeP7FYtIDOV
7kXUIw1ToW4dSZT8ZoB8r1t7OB9xQvT4mdQEpY4N4FR0xkliGJ4XRgPkoA45JjKh
lbwXc+PK9Sx+gDiLTDfKCMzT

Acknowledgement

This research was funded by the German Research Foundation (DFG) under project number RU 600-11/1.

Abstract

Compressed Sensing offers the possibility to jointly compress and encrypt sparse or compressible signals during the sampling process, i.e. directly at the sensor level. For the purpose of encryption, only the sampling matrix, which is also needed for the signal recovery, must be kept secret. However, this type of encryption scheme is no longer secure if multiple signals are encrypted, since the same plaintext will always yield the same ciphertext. This thesis proposes modes of operations for Compressed Sensing that allow the secure encryption of multiple signals.

First, different attacks and threat models are analyzed in order to develop security definitions for Compressed Sensing based encryption and Compressed Sensing modes of operations. The results from this analysis are used to design a general model for Compressed Sensing modes, which ensure confidentiality and additionally reduce the information leakage of Compressed Sensing encryption. The security and performance of cryptographic constructions that are suitable for implementing the general model are evaluated and three dedicated modes with different quality of service properties are derived from the general design. These modes of operation and their corresponding security analysis are the first main result of this thesis.

In addition to confidentiality, which is achieved through encryption, another important security service is data integrity. This work also examines so-called authenticated-encryption modes for Compressed Sensing, which ensure confidentiality and data integrity simultaneously. A general model for Compressed Sensing with authenticated-encryption is developed and dedicated schemes are derived from this model. The security of these schemes is reduced to the security of the underlying cryptographic constructions and primitives. Compressed Sensing with authenticated encryption provides security even in the case where an adversary has the ability to tamper with the ciphertext.

The final contribution of this thesis is the integration of the proposed Compressed Sensing modes into a distributed application. A software architecture for Industry 4.0 applications is developed and implemented. The designed system includes all necessary components for the usage of Compressed Sensing modes, like key-establishment and parameter exchange. Next to security, a main design goal of this system is usability. The user just determines the security service for his application and all security relevant operations are handled by the software, transparent for the user. This approach prevents security problems that are caused by inappropriate usage of cryptographic schemes.

The Compressed Sensing modes developed in this thesis are suitable for a wide range of real world applications due their joint sampling, compression and end-to-end security that starts at the sensor level. The proposed software-system completes the contribution of this work, since it realizes all necessary components for the usage of the designed modes.

Zusammenfassung

Compressed Sensing bietet die Möglichkeit spärlich besetzte oder komprimierbare Signale bereits bei der Abtastung - also im Sensor - zu komprimieren und gleichzeitig zu verschlüsseln. Dabei muss zur Verschlüsselung lediglich die Abtastmatrix, welche auch für die Signalrekonstruktion notwendig ist, geheim gehalten werden. Allerdings ist diese Verschlüsselungstechnik nicht mehr sicher, wenn mehrere Signale verschlüsselt werden, da derselbe Klartext immer denselben Schlüsseltext ergibt. Deshalb befasst sich diese Dissertation mit Betriebsarten für Compressed Sensing, die eine sichere Verschlüsselung von mehreren Signalen ermöglichen.

Zunächst werden verschiedene Angriffe und Angreifermodelle untersucht, um Sicherheitsdefinitionen für Compressed Sensing basierte Verschlüsselung und Compressed Sensing-Betriebsarten zu entwickeln. Basierend auf den Ergebnissen dieser Analyse wird ein generelles Modell für Compressed Sensing-Betriebsarten entworfen, welche Vertraulichkeit gewährleisten und darüber hinaus den Informationsverlust der Compressed Sensing-Verschlüsselung reduzieren. Nach einer Sicherheits- und Performancebewertung der für die Implementierung geeigneten kryptographischen Verfahren werden drei dedizierte Betriebsarten aus dem generellen Design abgeleitet, die verschiedene Dienstgüteeigenschaften wie Parallelisierbarkeit oder Selbstsynchronisation aufweisen. Diese Betriebsarten und die damit verbundenen Sicherheitsanalysen sind das erste Hauptergebnis dieser Dissertation.

Zusätzlich zur Vertraulichkeit, die durch Verschlüsselung erreicht wird, ist Datenunversehrtheit ein weiterer wichtiger Sicherheitsdienst. Deshalb werden in dieser Arbeit auch sogenannte Authenticated-Encryption-Betriebsarten für Compressed Sensing entworfen, die Vertraulichkeit und zeitgleich den Schutz der Datenunversehrheit gewährleisten. Auch hier wird zunächst ein generelles Modell für diese Betriebsarten entwickelt, aus dem dann konkrete Implementie-

rungen abgeleitet werden, deren Sicherheit auf die zugrundeliegenden kryptographischen Verfahren zurückgeführt wird. Compressed Sensing mit Authenticated-Encryption bietet auch dann Sicherheit, wenn ein Angreifer gezielt Schlüsseltexte manipulieren kann.

Ein letzter Aspekt, der in dieser Arbeit betrachtet wird, ist die Integration der entwickelten Betriebsarten in eine verteilte Anwendung. Dazu wird eine Softwarearchitektur für Industrie 4.0-Anwendungen entworfen und implementiert, welche die nötigen Komponenten für den Einsatz der Compressed Sensing-Betriebsarten, wie Schlüsselvereinbarung und Parameteraustausch, beinhaltet. Neben der Sicherheit ist ein Hauptziel des entworfenen Systems die Benutzerfreundlichkeit. Alle sicherheitsrelevanten Operationen werden für den Anwender transparent von der Software durchgeführt, sodass der Anwender lediglich festlegen muss, wie die Daten seiner Anwendung geschützt werden sollen. Auf diese Weise werden Sicherheitsprobleme vermieden, die durch unsachgemäße Verwendung von kryptographischen Verfahren entstehen.

Die in dieser Arbeit entwickelten Betriebsarten für Compressed Sensing sind durch die gleichzeitige Abtastung und Komprimierung für viele reale Anwendungsfälle interessant und ermöglichen echte Ende-zu-Ende-Sicherheit, die bereits im Sensor beginnt. Das vorgestellte Softwaresystem vervollständigt den Beitrag dieser Arbeit, da es die nötigen Komponenten für die Verwendung der entworfenen Betriebsarten realisiert.

Contents

List of Figures

Acronyms

AD	Associated-Data
AE	Authenticated-Encryption
AEAD	Authenticated-Encryption with Associated-Data
AES	Advanced Encryption Standard
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
CBC	Cipher Block Chaining (Mode)
CCA	Chosen Ciphertext Attack
CCM	Counter with CBC-MAC
CFB	Cipher Feedback Mode
CMA	Chosen Message Attack
COA	Ciphertext Only Attack
CPA	Chosen Plaintext Attack
CS	Compressive Sensing, Compressed Sensing or Compressive Sampling
CS-AE	Compressive Sensing with Authenticated-Encryption
CS-CBC	Compressive Sensing with Cipher Block Chaining
CS-CFB	Compressive Sensing with Cipher Feedback
CS-CTR	Compressive Sensing Counter Mode
CSE	Compressive Sensing Encryption Scheme
CTR	Counter Mode
CTS	Ciphertext Stealing

DIN	Deutsches Institut für Normung e.V.
ECB	Electronic Codebook (Mode)
ECDSA	Elliptic Curve Digital Signature Algorithm
GCM	Galois Counter Mode
HMAC	Keyed Hash Based Message Authentication Code
IND-CCA	Ciphertext Indistinguishability under Adaptive Chosen Ciphertext Attacks
IND-COA	Indistinguishability under Ciphertext Only Attacks
IND-CPA	Ciphertext Indistinguishability under Chosen Plaintext Attacks
ISO	International Organization for Standardization
JCA	JAVA Cryptography Architecture
KDF	Key Derivation Function
KPA	Known Plaintext Attack
LDAP	Lightweight Directory Access Protocol
MA	Message Authentication (scheme)
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
NMAC	Nested Message Authentication Code
Nonce	Number Only Used Once
OCB	Offset Codebook
OFB	Output Feedback Mode
PRF	Pseudorandom Function
PRNG	Pseudorandom Number Generator
PRP	Pseudorandom Permutation

PS	Publish-Subscribe
PSNR	Peak Signal-to-Noise Ratio
RIP	Restricted Isometry Property
RPC	Remote Procedure Call
SHA	Secure Hash Algorithm
SIV	Synthetic Initialization Vector
SLP	Service Location Protocol
TLS	Transport Layer Security
XOF	Extendable-Output Function

List of Symbols

A	a sampling matrix with dimension $m \times N$
$\mathcal{A}$	an efficient (i.e. polynomial time) algorithm called adversary
$\approxeq$	approximately equal or equal
$\langle u \rangle_p$	binary representation of some scalar $u \in \mathbb{R}$ with precision p (in accordance to IEEE 754 [IEE08])
$\rangle b \langle$	floating point representation of some binary string b with precision p (in accordance to IEEE 754 [IEE08])
$\perp$	symbol indicating that the *tag* verification has failed
$\lceil \cdot \rceil$	the ceiling function
c	ciphertext (encrypted energy) represented as binary string
$\checkmark$	symbol indicating that the *tag* verification was successful
CTR_i	a counter starting at an initial value CTR_0 incremented by i ($i > 0$)
d	the security parameter of a PRF or PRF*
$\varepsilon(\cdot)$	a negligible function
f_k	a PRF that is chosen from a pseudorandom function family F according to the key k
Gen_k	keyed matrix generation function
$\vec{\gamma}_i$	ciphertext vector, $\vec{\gamma}_i = (\hat{y}_i, \rangle c_i \langle)^T$

Γ	ciphertext set
$\mathcal{IV}$	initialization vector
k	a secret key
κ	a security parameter (treated as implicit parameter to polynomial time algorithms)
$\mathcal{K}$	key space
l	number of encrypted signals under a single key
L	output length of a fixed output length PRF or hash function
(M1), (M2)	the two sampling models from Def. 3.3
n	a natural number, mostly the block size of a block cipher
$\mathcal{N}$	a number only used once (nonce)
$\|\| \cdot \|\|_2^2$	squared Euclidean norm, energy
$[\cdot]$	optional input to a function or algorithm
π_k	a keyed IND-CPA secure encryption scheme
Ψ	a sparsifying basis with dimension $N \times N$
rec	a Compressive Sensing reconstruction algorithm
s	sparsity of a signal
tag	authentication tag outputted by a message authentication scheme, AE or AEAD scheme
$\hat{v}$	a normalized vector
$\vec{x}$	a compressible or sparse signal vector with dimension N
$\tilde{x}$	reconstructed signal vector
$\mathcal{X}$	set of all plaintext signals
$\vec{y}$	compressed measurement vector, where $\vec{y} \in \mathbb{R}^m$
$\mathcal{Y}$	set of all measurement vectors
$\|\|$	concatenation of binary strings or vectors

Chapter 1

Introduction

1.1. Motivation

The amount of data generated in today's world grows rapidly. Compression gains an increasingly important significance, since it makes the massive amount of data controllable. Data is compressed right after sampling in order to reduce the data quantity by keeping only the important information for storage and further processing. However, it would be more efficient and natural to measure the important parts of the data directly, instead of collecting a lot of data just to throw it away moments later.

Compressive Sensing (CS) is an inspiring technique that might offer a solution to the aforementioned problem. Candès, Donoho, Romberg and Tao are regarded as the founders of this sampling paradigm [CT05, CT06, CRT06, Don04, Don06]. CS allows to recover sparse signals from far less samples than the Nyquist rate pretends in accordance with the Whittaker-Shannon-Kotelnikow sampling theorem [Whi35, Sha49a, Kot33]. By this means, compression is integrated directly into the sampling process. Random sampling shows the most promising dimensionality reduction, i.e. compression, guarantees even in presence of noise. Interestingly, Candès and Tao mentioned in [CT06] that this random encoding can be used to implement a robust symmetric encryption scheme

just by treating the random sampling operator – a matrix – as a secret shared between legitimized parties. The additional overhead for this encryption is minimal and limited by the necessary key-establishment and -management. Hence, Compressive Sensing might offer real end-to-end encryption that is directly integrated into the sampling process, which means that data can be protected right at the sensor level.

When multiple signals are encrypted under the same sampling operator, the Compressed Sensing based encryption scheme is no longer randomized like a block cipher with a fixed key. As a consequence, identical plaintext signals will be encrypted to the same ciphertext. From a privacy point of view this means that an eavesdropper will recognize if the same plaintext is encrypted repeatedly. Even more critical, the system can easily be broken by an active and simple key recovery attack [Fay16, FR16].

The main problem with Compressive Sensing based encryption is the predictable behavior of the cipher after fixing the secret sensing matrix. In modern cryptography, a block cipher is used in an appropriate mode of operation that randomizes the encryption process and achieves security against active adversaries. This work introduces modes of operation to Compressive Sensing based encryption, which allow to encrypt multiple signals securely under a single key.

1.2. Outline

This work is organized as follows. The next chapter presents an overview of proposed Compressed Sensing encryption schemes and other work that is related to this thesis. The related work is discussed and categorized in order to give the reader an insight of the state-of-the-art techniques for Compressed Sensing encryption. After that, open problems partially extracted from the related work are stated in section 2.3. These outstanding issues and some more are addressed in this thesis.

Chapter 3 contains an introduction to Compressed Sensing and symmetric cryptography. Formal definitions are provided in order to establish the notation that is used throughout the thesis. At first, the concept of sparsity, different sampling models and the requirements on the sampling system are presented. The basic method for Compressive Sensing reconstruction is explained, which also shows why CS achieves some robustness against noise. Section 3.2 of-

fers an introduction to symmetric cryptography. Basic primitives and security models are defined. The security models are used to provide formal definitions for the security of various cryptographic schemes. These definitions express precisely what the word "secure" means for a scheme in a specific context. Standardized block cipher modes are presented and their properties are summarized. Finally, the basic concept of authenticated-encryption and authenticated-encryption with associated-data is introduced.

Chapter 4 explains the concept of Compressed Sensing based encryption. A security definition for this kind of encryption schemes is established for two important scenarios. At first, security for the case where just a single signal is encrypted with a fixed matrix is studied in section 4.2. The impact of different sampling matrix designs on the security of CS based encryption is discussed in section 4.2.2 and experimental results regarding the behavior of the compressed ciphertext are presented. Section 4.3 deals with the many-time encryption scenario, i.e. the case in which multiple plaintexts should be encrypted. Possible attacks on CS based encryption with a fixed sampling matrix are shown and implemented. The results from this analysis are used to establish a security definition for the many-time encryption scenario. This definition builds the basis for the security analysis of all CS modes of operation proposed in this thesis.

Compressive Sensing encryption modes are presented in chapter 5. These modes will ensure the confidentiality of plaintext signals in the many-time encryption scenario, and they are resistant to chosen-plaintext key recovery attacks. Some results of this chapter are published in [Fay16] and [FR16]. A general design for Compressed Sensing encryption modes is presented in section 5.2.1. The design can be split into two parts, namely matrix generation and normalized encryption, which are explained in section 5.2.2 and 5.2.3. Necessary implementation details for the matrix generation algorithm are discussed in section 5.2.2.2 together with suitable primitives, constructions and recommended parameters. A formal security analysis of the proposed constructions against generic attacks is presented in section 5.2.2.3 and their performance is analyzed in section 5.2.2.4. Dedicated modes derived from the general design are presented in section 5.3, 5.4 and 5.5. A security analysis is also provided for each of the designs. Section 5.6 compares the proposed confidentiality modes and their properties, while experimental results are presented in section 5.7.

Chapter 6 bases on [FR17] and deals with Compressed Sensing modes that

achieve confidentiality and integrity. Generic constructions, i.e. combinations of confidentiality schemes and message authentication schemes are analyzed. The findings of this analysis are used to establish a general design for Compressed Sensing authenticated-encryption schemes in section 6.3.1. Suitable constructions for the implementation of these schemes are presented, and it is discussed, under which conditions these schemes achieve the claimed security goal.

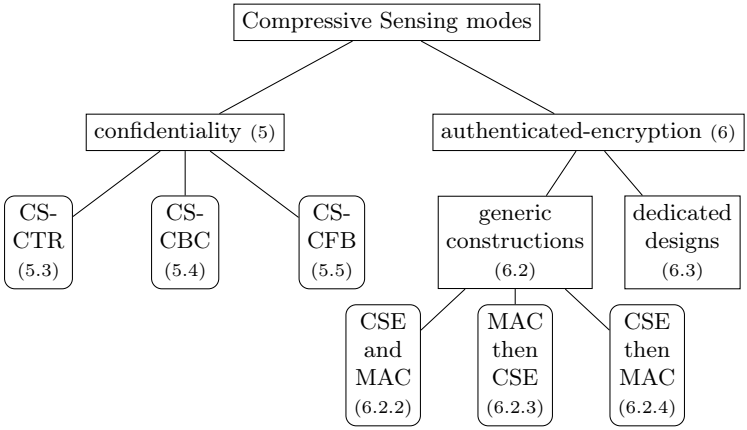


Figure 1.1.: Overview of the designed modes and the roadmap for the subsequent sections.

Fig. 1.1 presents an overview of the designed modes of operation that are studied in this thesis. The rectangular nodes describe the modes classification, divided into confidentiality modes and authenticated-encryption. The nodes with rounded corners show the proposed modes resp. the considered generic constructions. The corresponding section numbers are given in round brackets.

Chapter 7 introduces a framework for secure sensor networks, especially for Industry 4.0 applications. This framework is the use-case for the developed encryption schemes presented in the previous chapters. The proposed software framework allows implementers to create services that communicate securely with each other. However, one of the main design goals of this framework is usability and misuse resistance. Therefore, all cryptography and security related operations are handled by the system, transparent for the implementer who just chooses the security service he/she wants for the data.

The final chapter 8 concludes the thesis, summarizes their main contribution and proposes future work.

1.3. Remarks

An accurate mathematical notation is important for the understanding of scientific work. The provided *List of Symbols* contains the most important symbols used in this thesis. Unless otherwise noted, the basic mathematical operators used in this work are in accordance with ISO 80000-2. Terminology from the field of cryptography and information security can be found in [ISO16].

Chapter 2

Related Work

2.1. Security of Compressed Sensing Based Encryption

The goal of this thesis is to develop and analyze Compressive Sensing encryption modes that are usable to encrypt multiple signals securely with respect to precisely formalized threat models. Hence, the first question that comes to mind is: how secure is Compressive Sensing based encryption? Candès and Tao mentioned in [CT06] that Compressive Sensing offers security and robustness as long as the sampling operator is unknown to an eavesdropping adversary. It took two years until Rachlin and Baron published the first security analysis of Compressed Sensing measurements in [RB08]. They were able to show that, in general, Compressive Sensing cannot achieve perfect secrecy in the Shannon sense [Sha49b]. This means that an adversary learns information about the plaintext just by observing the ciphertext. However, Rachlin and Baron also showed that Compressive Sensing based encryption is computationally secure, which means that it is computational infeasible for an adversary to recover the original plaintext when he/she is just given the ciphertext but not the key. Of course, it must be assumed that some parameters like the key-size are sufficiently large. Further contributions by [OASB08] and [TZ12] support the results of Rachlin and Baron. Cambareri et al. [CMP+15a, CMP+15b] and Bianchi et al. [BBM14, BBM16] studied the information leakage in Compressive Sensing

with respect to different matrix designs and assumptions. Both found that the signals energy is in fact the only information that an adversary might extract from eavesdropped measurements. It must be noted that all this studies target one-time encryption.

2.2. Related Encryption Schemes

Various encryption schemes that make use of Compressive Sensing were proposed in the last years, see [CHP+13, HRU14, ZWLZ14, ZWX+14]. A literature review on Compressed Sensing encryption schemes can be found in [ZZZ+16]. In general, the proposed schemes can be divided into two categories. One type of systems use Compressive Sensing just for sampling and compression while encryption is performed by additional enciphering layers. Those schemes follow the *compress-then-encrypt* paradigm. An example for such a system is presented in [HRU14]. The authors use Compressive Sensing as sampling method and apply additional cipher layers in order to protect the compressed measurements during communication. Huang et al.'s system [HRU14] is able to encrypt multiple signals under a single key and the authors also claim security against chosen-plaintext attacks. However, systems from the compress-then-encrypt family do not offer joint encryption, compression and sampling. Therefore, it is not clear whether they can be integrated into a CS-sensor. Moreover, their security is mainly due to the additional cipher layers. The robustness of the compressed measurements against noise is also lost with high probability. Hence, one could just encrypt the measurements after sampling using standardized and mature ciphers.

The other type of systems contains encryption schemes that use the inherent security of Compressive Sensing to ensure the confidentiality of a plaintext signal. Examples include one-time encryption schemes like [CHP+13], but also systems like [ZWLZ14] that provide security in the case where multiple signals are encrypted. Zhang et al. [ZWLZ14] proposed a bi-level protection of the compressed measurements with a key dependent sampling basis besides the secret sampling matrix. However, in practice it might prove hard to find a suitable sampling basis for each application scenario that does not harm the recovery guarantees. On top, the only random components of this system are the random keys used to generate the matrices. If the same keys are used to encrypt iden-

tical plaintexts multiple times the encryption scheme is no longer randomized. Hence, an adversary will learn that the same plaintext is encrypted repeatedly, since same plaintext will produce same ciphertext. This shows an additional disadvantage of most of the proposed CS based encryption schemes: they are lacking a concrete and formal security treatment.

2.3. Open Problems

The related literature shows some open issues, but other interesting questions have not been asked yet. The following open problems form the objectives of this thesis.

Encrypting multiple signals

Compressive Sensing based encryption becomes deterministic when multiple signals are encrypted under one key. This means that the same plaintext will always produce the same ciphertext under a fixed key. Previously proposed encryption schemes are either lacking a concrete security treatment or they are just one-time key encryption schemes. The Compressive Sensing encryption modes presented in this thesis will use the inherent security of Compressive Sensing to protect multiple signals that are encrypted under a single key. The security of the proposed modes of operation is analyzed with respect to formal and well defined threat models.

Hiding the signals energy

[CMP+15a] and [BBM16] agreed that the signals energy is in fact the only information that an adversary might learn from the eavesdropped measurements. [BBM14] proposed to normalize the signals in order to hide their energy. The normalization value should be sent securely to the recipient for a correct signal reconstruction. However, no actual system with integrated protection of the signals energy has been proposed.

Integrity protection

Data integrity is an important security service, but the proposed CS based encryption schemes are only dealing with confidentiality. It is not even clear how

to combine Compressive Sensing based encryption with message authentication schemes.

System integration

It is another open problem to investigate how the proposed encryption schemes can be integrated into a complete system. The majority of the proposed encryption schemes assume that, for example, secret keys are already shared between communicating parties. Other parameters that are necessary for the system to work correctly are not discussed in detail. Hence, it is not clear which parameters might be publicly available and which must be secret.

Chapter 3

Fundamentals

3.1. Compressive Sensing

3.1.1. Signal representation

This section provides the necessary notation for an introduction to Compressed Sensing and the concept of sparsity. Let $\mathcal{V}$ be a vector space of dimension N over the field $\mathbb{F}$ and define a norm on $\mathcal{V}$ as follows (cf. [FR13, Def. A.1]):

Definition 3.1. (Norm)
A norm on $\mathcal{V}$, denoted by $\|\cdot\|$, is defined as a function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_0^+$ where the following axioms hold for all $\vec{x}, \vec{y} \in \mathcal{V}$ and all scalars $a \in \mathbb{F}$:

(I) $\|\vec{x}\| = 0$ iff $\vec{x} = \vec{0}$; $\|\vec{x}\| > 0$ otherwise
(II) $\|a \cdot \vec{x}\| = |a| \cdot \|\vec{x}\|$
(III) $\|\vec{x} + \vec{y}\| \leq \|\vec{x}\| + \|\vec{y}\|$

The p-norm of $\vec{x} \in \mathcal{V}$ is given by (cf. [BDDH11, Eq. (2.1)])

$$\|\vec{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p} = (|x_1|^p + |x_2|^p + \ldots + |x_n|^p)^{1/p}, \text{ for } p \geq 1. \tag{3.1}$$

In the case where $0 < p < 1$, Eq. (3.1) defines a so called quasi-norm, since the function $\|\cdot\|_p$ does not fulfill axiom (III) from Def. 3.1. Fig. 3.1 presents the unit circles for different (quasi-)norms in $\mathbb{R}^2$.

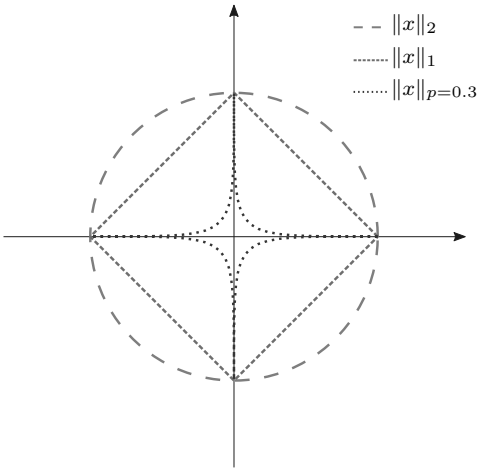


Figure 3.1.: Unit circles for different (quasi-)norms.

In case that p equals zero, let $\|\vec{x}\|_0$ be the number of nonzero elements in $\vec{x}$, e.g. the vectors Hamming weight. The sparsity s of a signal $\vec{s}$ is now defined using $\|\cdot\|_0$.

Definition 3.2. (Sparse signals)
A signal $\vec{s} \in \mathbb{R}^N$ is s-sparse if at most s entries of $\vec{s}$ are nonzero, meaning that $\|\vec{s}\|_0 \leq s$ holds, for some $s \in \mathbb{N}$ with $s \leq N$.

Sparsity plays an important role in CS, but natural signals, like images, are most likely not directly sparse in the domain where they were sampled originally. However, it turns out that these signals might be sparse in a different domain Ψ. If some domain Ψ exists where $\vec{x} = \Psi\vec{s}$ holds, with $\|\vec{s}\|_0 \leq s$, the vector $\vec{x}$ is called s-sparse in the sparsifying basis Ψ. Images, for example, are sparse or compressible in the wavelet domain, which is used in JPEG 2000 compression standardized in ISO 15444.

A signal is called compressible in a domain Ψ, if only a small number of its transformed entries have large values, while most of the entries are close to zero, see Fig. 3.2 for an example. In this case, the compressible signal can be approximated using an s-sparse signal by setting the entries that are below a

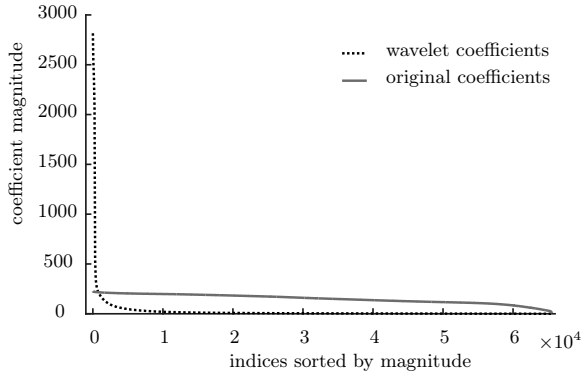


Figure 3.2.: The power-law decay of the wavelet coefficients of an image compared to the original coefficients from the spatial domain.

certain threshold to zero. Of course, this kind of approximation will result in a loss of information.

The CS systems considered in this thesis work by taking a small set of m linear measurements $\vec{y}$ from a finite N-dimensional, s-sparse or compressible, signal $\vec{x}$ by:

$$\vec{y} = A \cdot \vec{x} \approxeq A \cdot \Psi \cdot \vec{s}, \|\vec{s}\|_0 \leq s, \tag{3.2}$$

where the sampling matrix $A \in \mathbb{R}^{m \times N}$ is a linear map from $\mathbb{R}^N \to \mathbb{R}^m$ and $m \geq s$. It is assumed that N is much larger than m, hence the sampling process given by Eq. (3.2) contains an implicit dimensionality reduction.

One might either implement Eq. (3.2) by treating $\vec{x}$ as a vector of discrete samples obtained by classical Nyquist sampling or by acquiring $\vec{y}$ directly. Therefore, the following two sampling models are defined:

Definition 3.3. (Sampling models)

(M1) The vector $\vec{y}$ of compressed measurements is acquired directly by the compressed sampling system.

(M2) $\vec{x}$ is assumed to be a vector of discrete samples obtained by classical sampling in accordance to the Whittaker-Shannon-Kotelnikow sampling theorem [Whi35, Sha49a, Kot33], and $\vec{x}$ is compressed to $\vec{y}$ using Eq. (3.2).

Clearly, the preferred and most interesting sampling model is (M1), since the sampling process collects the important information directly. When $\vec{x}$ is s-sparse and (M2) is used, the system acquires a large number of samples just to discard most of them later. However, (M2) is often assumed for experimental or theoretical purpose.

3.1.2. Requirements on the sampling system

Consider a set $\mathcal{X} = \{\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_{l-1}\}$ of s-sparse or compressible signals that is sampled as stated in Eq. (3.2), and later the original signal should be recovered from the obtained measurements $\mathcal{Y} = \{\vec{y}_0, \vec{y}_1, \ldots, \vec{y}_{l-1}\}$. It must be ensured that the dimensionality reduction preserves the information contained in the signals. Otherwise, it would be impossible to recover the original signal from the compressed measurements. In the case where $m \ll N$, the system $A\vec{x} = \vec{y}$ is heavily underdetermined and has an infinite number of solutions in general. The fact that $\vec{x}$ is s-sparse relaxes the problem, since from the set of all possible solutions only the s-sparse solutions are from interest. The sampling matrix A must be designed in a way that ensures that the considered system of linear equations has a unique sparse solution. This can be formalized using the *restricted isometry property* (RIP) that was introduced by [CT05].

Definition 3.4. (Restricted isometry property)
A matrix $A \in \mathbb{R}^{m \times N}$ is said to satisfy the restricted isometry property (RIP) of order s if for all s-sparse $\vec{x}$ the following inequality holds

$$(1 - \beta_s)\,||\vec{x}||_2^2 \leq ||A \cdot \vec{x}||_2^2 \leq (1 + \beta_s)\,||\vec{x}||_2^2. \tag{3.3}$$

β_s is called the restricted isometry constant of the matrix A and $0 < \beta_s < 1$.

An RIP matrix of order $2s$ preserves the distances between any pair of s-sparse vectors [BDDH11, Sec. 3.3]. No pair of s-sparse vectors will be mapped to the same measurement vector, which ensures that the system has a unique solution. Moreover, the distance preservation introduces some robustness against noise.

Verifying that a certain matrix meets the RIP of order s was assumed to be computationally intractable for a long time and later it was proven to be an NP-hard problem by [TP14]. Surprisingly, [CT05][CT06] found that sub-Gaussian matrix ensembles satisfy the RIP with overwhelmingly high probability, i.e.

$1 - \exp(-C_1 \cdot m)$ [FPRU10]. For this results to hold, the minimal number of measurements, given by m, must be bounded by

$$m \geq C_2 \cdot s \cdot \log\left(\frac{N}{s}\right), \tag{3.4}$$

with some positive constant C_1, C_2 depending only on β_s [FPRU10]. As shown in [FPRU10], the bound from Eq. (3.4) is indeed optimal and according to [FR13, Sec. 5.4] most deterministic matrix constructions only yield $m \geq C \cdot s^2$, for some positive constant C.

Prominent examples for sub-Gaussian matrix ensembles are Gaussian matrices whose entries are drawn independently and identical from the standard normal distribution, and Bernoulli matrices whose entries are drawn uniformly at random from $\{+1, -1\}$. Another benefit of sub-Gaussian matrix ensembles is that the product $A\Psi$ satisfies Eq. (3.3) with very high probability for some orthonormal basis Ψ [BDDW08, Thm. 5.2], i.e. the RIP holds even in the case where $\vec{x}$ is s-sparse in Ψ. This property is sometimes called universality. In contrast, deterministic matrices are not universal in the sense that Ψ must be considered explicitly in the construction of A.

3.1.3. Reconstruction

In order to reconstruct a sparse signal $\vec{x} \in \mathbb{R}^N$ from compressed measurements $\vec{y} \in \mathbb{R}^m$ one must find the sparsest solution to the underlying system of linear equations. The conditions introduced in the last section will ensure that a unique solution exists, even for noisy measurements. The sparsest solution given $\vec{y} = A\vec{x}$ can be found by solving the following optimization problem:

$$\underset{\tilde{x} \in \mathbb{R}^N}{\operatorname{argmin}} \|\tilde{x}\|_0 \qquad \text{subject to } \vec{y} = A \cdot \tilde{x}. \tag{3.5}$$

When the measurements are contaminated with noise, the optimization problem becomes:

$$\underset{\tilde{x} \in \mathbb{R}^N}{\operatorname{argmin}} \|\tilde{x}\|_0 \qquad \text{subject to } \vec{y} = A \cdot \tilde{x} + \epsilon, \tag{3.6}$$

where ϵ is some bounded noise term, i.e. $\|\epsilon\|_2 \leq \eta$.

Unfortunately, the optimization problems from Eq. (3.5)/(3.6) are non-convex and therefore computationally hard to solve. A major breakthrough in the area of sparse signal recovery was the work of [Don04] and [CT05] who found that for RIP matrices replacing $\|\cdot\|_0$ by $\|\cdot\|_1$ in Eq. (3.5)/(3.6) yields the correct solution. Finding the solution to Eq. (3.5)/(3.6) with the minimal $\|\cdot\|_1$ norm relaxes the sparse signal recovery problem to a convex optimization problem, which can be solved by linear programming (see [FR13, chap. 15] for some example algorithms). In order to get a rough intuition why $\|\cdot\|_1$ is a adequate approximation for $\|\cdot\|_0$ in Eq. (3.5)/(3.6) the reader is referred to [BDDH11, Sec. 2.1/4.1]. In the case of compressible or noisy signals the reconstruction will introduce some error, which has to be considered. Let $\vec{y} = A \cdot \vec{x} + \epsilon$ with $\|\epsilon\|_2 < \eta$ and define the error of the best s-sparse approximation to $\vec{x} \in \mathbb{R}^N$ as (cf. [FR13, Def. 2.2]):

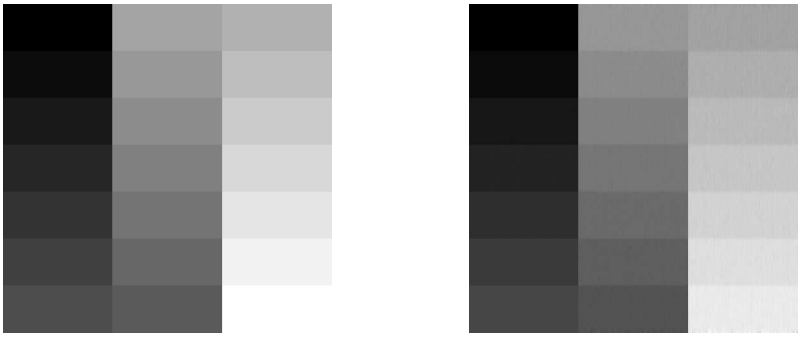
$$\Delta_s(\vec{x})_1 \coloneqq \min_{\|\vec{z}\|_0 \leq s} \|\vec{x} - \vec{z}\|_1. \tag{3.7}$$

Following [FR13, Thm. 4.22], the $\|\cdot\|_2$-error of the reconstruction algorithm can be estimated by

$$\|\vec{x} - \tilde{x}\|_2 \leq \frac{C}{\sqrt{s}} \Delta_s(\vec{x})_1 + D\eta, \tag{3.8}$$

where C, $D > 0$ are constants.

Fig. 3.3 presents exemplary the impact of noise in the reconstruction of the "gray-scale" test image. The original test image was sampled column-wise with a Gaussian random matrix and reconstructed with TVAL3 recovery [Li09], an algorithm specially suitable for the reconstruction of images. The rightmost columns of the original image have more zero-entries than the others due to the black area, hence the noise in the reconstructed columns is hardly visible. In contrast, the bright columns on the left-hand side are visibly corrupted with noise.



(a) (b)

Figure 3.3.: The impact of reconstruction noise in Compressed Sensing. (a) shows the original test image, while (b) presents the reconstruction of the original image with TVAL3 recovery [Li09].

3.2. Symmetric Cryptography

3.2.1. General

A symmetric encryption scheme that should provide data confidentiality is defined as a three-tuple $\Pi = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$, where $\mathcal{K}$ (the key space) is a finite and nonempty set of binary strings k called keys. Let $|\cdot|$ denote the length of a binary string in bits, e.g. $|k|$ is the length of the key k. The encryption algorithm $\mathcal{E}_k$ is a keyed deterministic or randomized and efficient (i.e. polynomial-time) algorithm. An algorithm is randomized if it uses internal sources of randomness during its execution, otherwise the algorithm is called deterministic. The encryption algorithm takes a key k and a plaintext message $\mathfrak{m} \in \{0,1\}^*$ as input and outputs a ciphertext $\mathfrak{c} \in \{0,1\}^*$. This process is denoted by $\mathfrak{c} \leftarrow \mathcal{E}_k(\mathfrak{m})$. Throughout this thesis, the key inputs to keyed functions and algorithms are written as subscript in order to differentiate between keys and other function inputs. The polynomial-time decryption algorithm $\mathcal{D}_k$ is a keyed deterministic algorithm that takes as input a ciphertext $\mathfrak{c}$ and outputs a message $\mathfrak{m}$, i.e. $\mathfrak{m} \leftarrow \mathcal{D}_k(\mathfrak{c})$. In general, decryption is defined to be whatever necessary, to be the inverse of encryption, which means that $\mathcal{D}_k(\mathcal{E}_k(\mathfrak{m})) = \mathfrak{m}$ holds for all $k \in \mathcal{K}$ and $\mathfrak{m} \in \{0,1\}^*$. A high level view of a symmetric encryption system using the

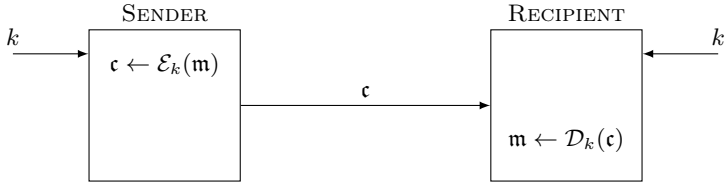


Figure 3.4.: High level view of a symmetric encryption system.

aforementioned notation is presented in Fig. 3.4, the legitimized communicating parties are sharing the same key k and therefore the system is called symmetric.

Encryption schemes are constructed from primitives, such as *pseudorandom functions* (PRF) and *pseudorandom permutations* (PRP) that are used to model the behavior of an idealized cipher. In practice, these primitives are replaced by concrete ciphers and constructions that have withstood an intensive cryptanalysis. This means that a lot of experienced cryptanalysts have tried to distinguish these ciphers from ideal primitives, but they were not able to find serious weaknesses. To date, an example for a secure PRP is the *Advanced Encryption Standard* (AES) while *HMAC* instantiated with *secure hash algorithm* (SHA)-2 is often used as a PRF.

Definition 3.5. (Pseudorandom functions)

A pseudorandom function family F is a map $F : \mathcal{K} \times \{0,1\}^{|\mathfrak{m}|} \rightarrow \{0,1\}^{|\mathfrak{c}|}$, where $|\mathfrak{m}|, |\mathfrak{c}| \leq n$ $(n \in \mathbb{N})$. For each key $k \in \mathcal{K}$ let $f_k(\mathfrak{m}) = F(k, \mathfrak{m})$ be a deterministic polynomial-time function $f_k : \{0,1\}^{|\mathfrak{m}|} \rightarrow \{0,1\}^{|\mathfrak{c}|}$. The function f_k is called pseudorandom if no efficient algorithm (called adversary) is able to distinguish f_k from a random function $\mathcal{F}$ from the same domain and range, when k is chosen uniformly at random (cf. [KL15, p. 77 f.]).

Definition 3.6. (Pseudorandom permutations)

A family of pseudorandom permutations P is defined as $P : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$, where $|\mathfrak{m}| = |\mathfrak{c}| = n$ $(n \in \mathbb{N})$. Let $\rho_k = P(k, \mathfrak{m})$ be an efficient deterministic bijection on $\{0,1\}^n$ and further let $\rho_k^{-1}(\mathfrak{c})$ be the efficiently computable inverse function. A permutation ρ_k is called pseudorandom if no efficient algorithm (called adversary) is able to distinguish ρ_k from a random permutation $\mathcal{P}$ on the same set, when k is chosen uniformly at random (cf. [KL15, p. 79

f.]). Each pseudorandom permutation is also a pseudorandom function, which is implied by the PRP/PRF switching Lemma [BR04, Lemma 1].

An efficient adversary $\mathcal{A}$ is modeled as an algorithm, and therefore, like all other efficient algorithms, $\mathcal{A}$ is supposed to run in polynomial-time with respect to some security parameter κ. κ is often determined by the key size such that $\kappa \leq |k|$. However, κ is treated as implicit parameter and unless otherwise noted it is omitted as function input. Now, consider that the adversary's goal is to distinguish a random function/permutation from a pseudorandom function/permutation. Hence, $\mathcal{A}$ is given access to an oracle O^{fun} resp. O^{perm} chosen with probability 1/2 to be either $O^{\text{fun}}(\mathfrak{m}) = f_k(\mathfrak{m})$, $O^{\text{perm}}(\mathfrak{m}) = \rho_k(\mathfrak{m})$ where k is chosen uniformly at random from $\mathcal{K}$, or $O^{\text{fun}}(\mathfrak{m}) = \mathcal{F}(\mathfrak{m})$, $O^{\text{perm}}(\mathfrak{m}) = \mathcal{P}(\mathfrak{m})$. The adversary has to guess how the oracle O was chosen and if the probability that he/she guesses correctly is not greater than $\frac{1}{2} + \varepsilon(\kappa)$ for some *negligible* function $\varepsilon(\kappa)$, the pseudorandom function/permutation is said to be *indistinguishable* from a random function/permutation (cf. [KL15, Def. 3.25/3.28]). The experiment is called the *real* or *random* experiment, abbreviated $\text{Exp}^{\text{ror}}_{\mathcal{A}}$. A function $\varepsilon(\kappa) : \mathbb{N} \to \mathbb{R}$ is called negligible, if for every positive polynomial $poly(\cdot)$ there exists a positive integer $\mathfrak{N}$ such that $\varepsilon(\kappa) < 1/(poly(\kappa))$ holds $\forall \kappa > \mathfrak{N}$ [KL15, Def. 3.4].

3.2.2. Security for one-time key encryption

The concept of indistinguishability plays an important role in the security analysis of encryption schemes. The security of an encryption scheme Π is often established using a reductionist approach, meaning that the security of Π is reduced to the security of the underlying primitive, e.g. a PRP. A well studied and widely applied technique used to define the security of an encryption scheme is based on interactive experiments (often called games) that involve an adversary $\mathcal{A}$ and an honest party called the challenger who uses Π (see [GM82, BDJR97, BR04]). The adversary's power and his/her goal must be specified in order to define what it means for a system Π to be secure against specific types of adversaries, regardless of their concrete implementation.

Consider the experiment from Fig. 3.5 that is denoted by $\text{Exp}^{\text{coa}}_{\mathcal{A},\Pi}(b)$. On input of a bit $b \in \{0, 1\}$ the challenger draws a key k randomly from the key space $\mathcal{K}$. The adversary chooses two plaintext messages $\mathfrak{m}_0, \mathfrak{m}_1$, both of equal length, and

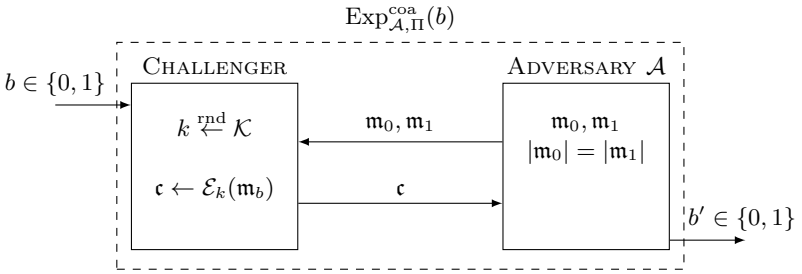


Figure 3.5.: The ciphertext only indistinguishability experiment $\mathrm{Exp}^{\mathrm{coa}}_{\mathcal{A},\Pi}(b)$.

sends them to the challenger. According to the bit b the challenger encrypts $\mathfrak{m}_b$ by $\mathfrak{c} \leftarrow \mathcal{E}_k(\mathfrak{m}_b)$ and sends the challenge ciphertext $\mathfrak{c}$ to the adversary. $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$ that represents his/her guess which message was encrypted and b' is also the output of the experiment. The adversary $\mathcal{A}$ observes only a single ciphertext encrypted under the key k and therefore this attack is called a *ciphertext only attack* (COA) (see [KL15, p. 54 f.]). In order to complete the threat model, the adversary's goal must be specified. Since $\mathcal{A}$ tries to guess the bit b, his/her goal is to distinguish between the encryption of $\mathfrak{m}_0$ and $\mathfrak{m}_1$. The advantage $\mathrm{Adv}^{\mathrm{coa}}_{\mathcal{A},\Pi} \in [0,1]$ of an adversary $\mathcal{A}$ against Π in the experiment $\mathrm{Exp}^{\mathrm{coa}}_{\mathcal{A},\Pi}(b)$ is defined as

$$\mathrm{Adv}^{\mathrm{coa}}_{\mathcal{A},\Pi} \coloneqq |\Pr[\mathrm{Exp}^{\mathrm{coa}}_{\mathcal{A},\Pi}(0)=1] - \Pr[\mathrm{Exp}^{\mathrm{coa}}_{\mathcal{A},\Pi}(1)=1]|. \tag{3.9}$$

Definition 3.7. (IND-COA SECURITY)

A symmetric encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$ has ciphertext indistinguishability in presence of a ciphertext only adversary, or it is said to be IND-COA secure for short, if

$$\mathrm{Adv}^{\mathrm{coa}}_{\mathcal{A},\Pi} \leq \varepsilon(\kappa)$$

holds for a negligible function $\varepsilon(\kappa)$ and all efficient adversaries $\mathcal{A}$.

The fact that $\mathcal{A}$ is allowed to choose the two messages $\mathfrak{m}_0$ and $\mathfrak{m}_1$ simulates that the adversary might have partial or full knowledge about the plaintext. Even in this case he/she should not be able to distinguish the plain-

texts given only a single ciphertext. The probabilities $\Pr[\text{Exp}^{\text{coa}}_{\mathcal{A},\Pi}(0)=1]$ and $\Pr[\text{Exp}^{\text{coa}}_{\mathcal{A},\Pi}(1)=1]$ from Eq. (3.9) are equal if the adversary is just guessing the bit b' at random and thus his/her advantage is zero. Hence, Def. 3.7 states that a system achieves IND-COA if no efficient adversary, regardless of his/her strategy, can do much better than guessing.

3.2.3. Modes of operation

3.2.3.1. Security in the many-time key scenario

The adversaries considered so far were only allowed to eavesdrop on one ciphertext encrypted under a single secret key k. This section discusses encryption schemes, so called modes of operation, which are secure against more powerful adversaries in the case where a single key is used to encrypt multiple plaintexts.

The experiment $\text{Exp}^{\text{cpa}}_{\mathcal{A},\Pi}(b)$ described in Fig. 3.6 is used to define a stronger security notation than IND-COA, since the adversary is now able to send multiple equal length messages $\mathfrak{m}_{i,0}$, $\mathfrak{m}_{i,1}$ to the challenger, who replies with $\mathfrak{c}_i \leftarrow \mathcal{E}_k(\mathfrak{m}_{i,b})$ for $0 \leq i \leq l-1$. The rest of the experiment is quite similar to the experiment $\text{Exp}^{\text{coa}}_{\mathcal{A},\Pi}(b)$ from the previous section. Note that the adversary is now able to perform a *chosen plaintext attack* (CPA) and use this knowledge for further queries, because he/she might choose a plaintext $\mathfrak{m}$ and send $\mathfrak{m} = \mathfrak{m}_{i,0} = \mathfrak{m}_{i,1}$, which leads to $\mathfrak{c}_i = \mathcal{E}_k(\mathfrak{m})$. The following security notation was introduced in the seminal work of [BDJR97] and is the most common security notation that confidentiality only encryption schemes are required to achieve in modern cryptography.

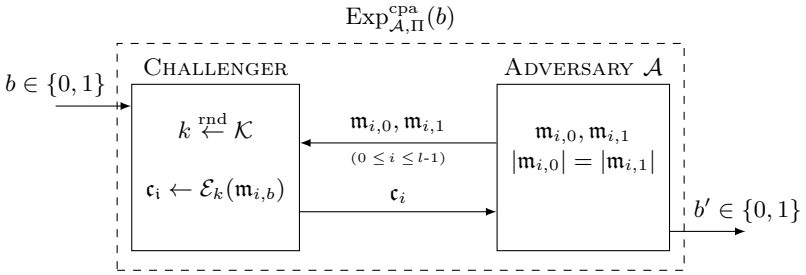


Figure 3.6.: The chosen plaintext indistinguishability experiment $\text{Exp}^{\text{cpa}}_{\mathcal{A},\Pi}(b)$.

Definition 3.8. (IND-CPA SECURITY)

A symmetric encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$ has ciphertext indistinguishability in presence of a chosen plaintext adversary, or it is said to achieve IND-CPA for short, if

$$\mathrm{Adv}^{\mathrm{cpa}}_{\mathcal{A},\Pi} \coloneqq |\Pr[\mathrm{Exp}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(0){=}1] - \Pr[\mathrm{Exp}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(1){=}1]| \leq \varepsilon(\kappa)$$

holds for a negligible function $\varepsilon(\kappa)$ and all efficient adversaries $\mathcal{A}$.

Remark 3.9. Note that IND-CPA includes IND-COA.

3.2.3.2. Standardized block cipher modes

The encryption scheme $\mathrm{ECB}[P] = (\mathcal{K}, \mathcal{E}^{\mathrm{ecb}}_k, \mathcal{D}^{\mathrm{ecb}}_k)$ is called *Electronic Codebook* (ECB) mode and is defined for a PRP family $P : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ by $\mathcal{E}^{\mathrm{ecb}}_k(\mathfrak{m}) = \rho_k(\mathfrak{m})$ and $\mathcal{D}^{\mathrm{ecb}}_k(\mathfrak{m}) = \rho^{-1}_k(\mathfrak{m})$. The choice of P also determines the key space and block size n, for example, with the block cipher AES-128 the key space is $\{0,1\}^{128}$ and $n = 128$. The key k is assumed to be shared between the legitimized communicating parties. $\mathrm{ECB}[P]$ achieves IND-COA (Def. 3.7) for a random k and $|\mathfrak{m}| = n$, because in order to distinguish the two experiments, the adversary must be able to distinguish the PRP ρ_k from a random permutation. By the assumption that ρ_k is a PRP, he/she is not able to do so, hence the security of $\mathrm{ECB}[P]$ reduces to the block ciphers indistinguishability from a PRP.

However, $\mathrm{ECB}[P]$ does not achieve the stronger IND-CPA notation. As an example, let the adversary $\mathcal{A}$ in $\mathrm{Exp}^{\mathrm{cpa}}_{\mathcal{A},\mathrm{ECB}[P]}(b)$ ask for the encryption of two n-bit messages $\mathfrak{m}_{0,0} = \mathfrak{m}_{0,1} = \mathfrak{m}$ and on his second query $\mathfrak{m}_{1,0} = \mathfrak{m}$, $\mathfrak{m}_{1,1} \neq \mathfrak{m}$. In the case where the adversary sees $\mathfrak{c}_0 = \mathfrak{c}_1$ he/she outputs $b' = 0$, otherwise the output is $b' = 1$. The adversary from this simple example has advantage $\mathrm{Adv}^{\mathrm{cpa}}_{\mathcal{A},\mathrm{ECB}[P]} = 1$, which stems from the fact that the same plaintext encrypted under the same key always yields the same ciphertext. As noted in [ISO06], the ECB mode is only secure if exactly one n-bit block is encrypted under a single key (which is given by IND-COA), or if it can be ensured that the same plaintext is never encrypted repeatedly. ECB mode should not be used for messages longer than one block, because it does not achieve IND-CPA.

Except for ECB, the other standardized modes from *ISO* 10116 [ISO06] or *NIST* SP 800-38A [Dwo01], namely *Cipher Block Chaining* (CBC), *Cipher Feedback mode* (CFB), *Output Feedback mode* (OFB) and *Counter mode* (CTR),

achieve IND-CPA under some specific requirements. All this modes of operation are based on an n-bit block cipher and require an initialization vector $\mathcal{IV} \in \{0,1\}^n$, which is often assumed to be drawn at random from the set of all n-bit strings at the beginning of the encryption algorithm $\mathcal{E}_k(\cdot)$. The $\mathcal{IV}$ is used to randomize the encryption algorithm in such a way that same plaintext messages will not be mapped to the same ciphertext. In order to decrypt properly, the $\mathcal{IV}$ is treated as the first ciphertext block and send in the clear to the recipient. Assuming a random $\mathcal{IV}$, the number of signals that might be encrypted securely under one key can be upper bounded by considering the number of adversarial queries q and the block length n. Bounds for CBC and CTR are given in [BDJR97], and for CFB in [AGPS02]. In addition, [BDJR97] also established IND-CPA security for CTR mode under the assumption that the $\mathcal{IV}$ is a *number only used once* (nonce). In practice, providing a random $\mathcal{IV}$ is sometimes not possible, however, constructing the $\mathcal{IV}$ by encrypting a nonce with a block cipher (i.e. PRP) using a different key is also secure for all modes.

Next to the IND-CPA security, all approved modes have different properties, like parallelizability or self-synchronization in case of bit slips. Table 3.1 summarizes some of this properties and the reader is referred to Annex-B of ISO 10116 for further information. A summary of the same modes, but in accordance with NIST standards, can be found in [Rog11, Figure 4.1].

Properties	*CBC*	*CFB*	*OFB*	*CTR*
IND-CPA established	for random $\mathcal{IV}$ [BDJR97]	for random $\mathcal{IV}$ [AGPS02]	for random $\mathcal{IV}^*$	for random or nonce $\mathcal{IV}$ [BDJR97]
Needs an invertible primitive	`Yes`	`No`	`No`	`No`
Parallel processing ($\mathcal{E}_k/\mathcal{D}_k$)	(`No`/`Yes`)	(`Yes`/`Yes`)†	(`No`/`No`)	(`Yes`/`Yes`)
Self-synchronizing (w.r.t. bit slips)	`No`, unless slip of full blocks	`Yes`, for bitwise encryption	`No`	`No`
Error propagation	`Yes`, an error in $\mathtt{c}_i$ corrupts $\mathtt{m}_i$ and $\mathtt{m}_{i+1}$	`Yes`, until corrupted bits are no longer used as feedback	`No`, bit errors affect only the corr. plaintext bit	`No`, bit errors affect only the corr. plaintext bit
Padding required if the plaintext is not a multiple of the processed block length	`Yes`, except for CBC-CTS‡	`No`, for bitwise encryption	`No`, for bitwise encryption	`No`

*analogous to CTR [ISO06].
†Not for every combination of the allowed parameters (cf. [ISO06, B.4.1]).
‡*Ciphertext stealing* (CTS).

Table 3.1.: Summary of some important properties of the IND-CPA secure ISO 10116:2006 encryption modes.

3.2.4. Data integrity and authentication-encryption

3.2.4.1. Message unforgeability

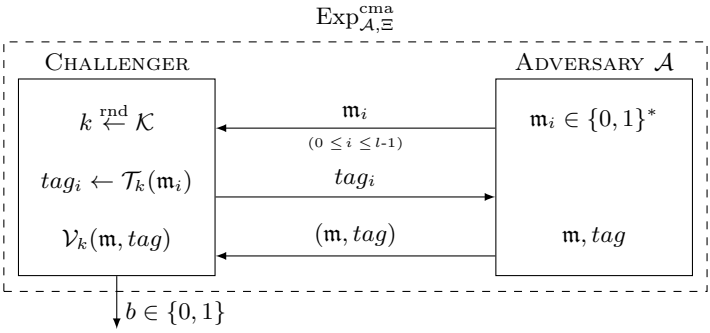


Figure 3.7.: The chosen message unforgeability experiment $\mathrm{Exp}^{\mathrm{cma}}_{\mathcal{A},\Xi}$.

Data integrity is another important security service, which ensures "[...] that data has not been altered or destroyed in an unauthorized manner" [ISO16]. Typical symmetric security mechanisms that establish data integrity are message authentication schemes based on *message authentication codes* (MACs) like HMAC or CBC-MAC [BCK96, BKR00]. MACs might also be called *message error detection codes*, since they cannot distinguish between an active forgery or a transmission error.

A *message authentication* (MA) scheme $\Xi = (\mathcal{K}, \mathcal{T}_k, \mathcal{V}_k)$ is defined for a key space $\mathcal{K}$ and two polynomial time algorithms $\mathcal{T}_k, \mathcal{V}_k$ (cf. [KL15, Def. 4.1]). The keyed tag generation algorithm $\mathcal{T}_k$ (e.g. the MAC) is a deterministic algorithm that takes a message $\mathfrak{m} \in \{0,1\}^*$ and outputs a fixed length authenticator $tag \in \{0,1\}^n$ ($n \in \mathbb{N}$). On input of $(\mathfrak{m}, tag)$ the verification algorithm $\mathcal{V}_k$ will either output a symbol ✓ indicating that the tag verification succeeded or $\perp$ meaning that the tag verification has failed. For all keys k and all messages $\mathfrak{m}$ it is essential that $\mathcal{V}_k(\mathfrak{m}, \mathcal{T}_k(\mathfrak{m}))$ will result in ✓. In general, the verification algorithm $\mathcal{V}_k(\mathfrak{m}, tag)$ works by recomputing the tag, i.e. $tag' = \mathcal{T}_k(\mathfrak{m})$, and comparing both tags. The verification is only successful if $tag' = tag$ otherwise the message will be regarded as not authentic.

To define security for a message authentication scheme Ξ, the adversary – not given the secret key – is allowed to perform an adaptive *chosen message attack* (CMA), meaning that he/she might ask for the *tag* of arbitrary messages up to a specific limit l. The adversaries goal is to come up with a new message/-tag pair that gets accepted by the verification algorithm. This pair is called an existential forgery, hence a secure message authentication scheme is (existentially) unforgeable under adaptive chosen message attacks. The experiment $\mathrm{Exp}^{\mathrm{cma}}_{\mathcal{A},\Xi}$ presented in Fig. 3.7 is used to establish this security definition more formally. The adversary might perform up to l chosen message queries $\mathfrak{m}_i$ and the challenger, who starts by generating a random key k, answers the queries with $tag_i \leftarrow \mathcal{T}_k(\mathfrak{m}_i)$. After that, the adversary outputs his forgery attempt $(\mathfrak{m}, tag)$ and sends it to the challenger. The challenger computes $\mathcal{V}_k(\mathfrak{m}, tag)$ and outputs $b = 1$ if the verification returns ✓ and the message was not submitted before, i.e. $\mathfrak{m} \notin \{\mathfrak{m}_0, \mathfrak{m}_1, ..., \mathfrak{m}_{l-1}\}$, otherwise the output is $b = 0$. The output of the experiment is given by the bit b. Now, the security of Ξ against adaptive chosen message attacks can be defined (see [KL15, p. 112]).

Definition 3.10. (Message unforgeability)

A message authentication scheme $\Xi = (\mathcal{K}, \mathcal{T}_k, \mathcal{V}_k)$ with security parameter $\kappa \leq |k|$ has message unforgeability under adaptive chosen message attacks, or is just said to be secure or unforgeable, if

$$\mathrm{Adv}^{\mathrm{cma}}_{\mathcal{A},\Xi} \coloneqq \Pr[\mathrm{Exp}^{\mathrm{cma}}_{\mathcal{A},\Xi}{=}1] \leq \varepsilon(\kappa)$$

holds for a negligible function $\varepsilon(\kappa)$ and all efficient adversaries $\mathcal{A}$.

A well established result due to [GGM85][GGM86][BKR00, Prop. 2.7] is that a secure fixed output PRF is also a secure MAC. However, even with a secure MAC, a message authentication scheme is not secure against *replay attacks*, i.e. when the adversary sends an eavesdropped message/tag pair to the original recipient. In order to protect a message authentication scheme against replay attacks, legitimized parties add time variable parameters like timestamps or sequence numbers to the messages before the authentication tags are computed.

3.2.4.2. Authenticated-encryption

An *authenticated-encryption* (AE) scheme is used to protect the data integrity (unforgeability) as well as data confidentiality (privacy). In general, an AE scheme can be implemented by either combining an IND-CPA-secure encryption scheme with a secure message authentication scheme, so called *generic constructions* [BN00], or by dedicated authenticated-encryption modes. In addition, Rogaway [Rog02] introduced the term *authenticated-encryption with associated-data* (AEAD) for schemes that ensure the confidentiality of a plaintext while simultaneously protecting its integrity as well as the integrity of additional *associated-data* (AD) that will not be encrypted, like package headers. An AEAD scheme is defined as a triplet $\Pi = (\mathcal{K},\mathcal{E}_k,\mathcal{D}_k)$, where k is a key drawn uniformly at random from $\mathcal{K}$. $\mathcal{E}_k$ takes an n-bit nonce $\mathcal{N}$, a binary string AD containing the associated-data and a message $\mathfrak{m}$ as input and outputs a ciphertext $\mathfrak{c}$. Note that AD might be empty, which defines an AE scheme. $\mathcal{D}_k$ takes $\mathcal{N}$, AD and $\mathfrak{c}$ as input and outputs a message $\mathfrak{m}$ or $\perp$ if it regards the input as invalid. As usual, it is required that decryption is the inverse of encryption.

Authenticated-encryption should ensure privacy in the IND-CPA manner and *unforgeable encryption*, which is a variation of message unforgeability but adopted to the AEAD definition of Π. Let $\mathcal{A}$ be an efficient adversary with access to an encryption oracle $\mathcal{E}_k(\mathcal{N},\mathrm{AD},\mathfrak{m})$ and denote the set of $\mathcal{A}'s$ encryption queries with Q. Π is said to achieve unforgeable encryption if the probability $\mathrm{Adv}^{\mathrm{auth}}_{\mathcal{A},\Pi}$ that $\mathcal{A}$ outputs $(\mathcal{N},\mathrm{AD},\mathfrak{c})$ with $\mathfrak{c} \neq \mathcal{E}_k(\mathcal{N},\mathrm{AD},\mathfrak{m})$, $\forall(\mathcal{N},\mathrm{AD},\mathfrak{m}) \in Q$ and $\mathcal{D}_k(\mathcal{N},\mathrm{AD},\mathfrak{c}) \neq \perp$ is negligible. Stated differently, the adversary should not be able to come up with a new decryption query $(\mathcal{N},\mathrm{AD},\mathfrak{c})$ that decrypts to a valid message (cf. [Rog02, Section 3]). It is important to note that in this model, the adversary is assumed to respect nonces, meaning that all $\mathcal{N}$'s that are submitted by $\mathcal{A}$ are unique values.

Another strong privacy notation for AE and AEAD is *ciphertext indistinguishability under adaptive chosen ciphertext attacks* (IND-CCA) [KL15, p. 96 f.], which is established with the experiment from Fig. 3.8.

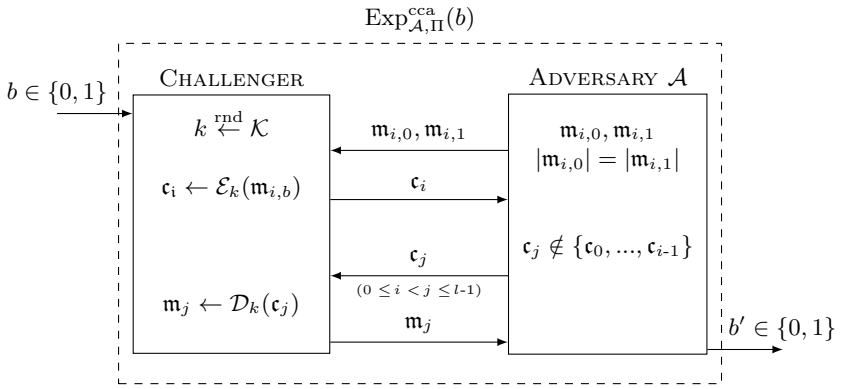


Figure 3.8.: The adaptive chosen ciphertext indistinguishability experiment $\mathrm{Exp}^{\mathrm{cca}}_{\mathcal{A},\Pi}(b)$.

Definition 3.11. (IND-CCA security)

A symmetric (authenticated-)encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$ has ciphertext indistinguishability in presence of an adaptive chosen ciphertext adversary, or it is said to achieve IND-CCA for short, if

$$\mathrm{Adv}^{\mathrm{cca}}_{\mathcal{A},\Pi} \coloneqq |\Pr[\mathrm{Exp}^{\mathrm{cca}}_{\mathcal{A},\Pi}(0)=1] - \Pr[\mathrm{Exp}^{\mathrm{cca}}_{\mathcal{A},\Pi}(1)=1]| \le \varepsilon(\kappa)$$

holds for a negligible function $\varepsilon(\kappa)$ and all efficient adversaries $\mathcal{A}$.

The first part of $\mathrm{Exp}^{\mathrm{cca}}_{\mathcal{A},\Pi}(b)$ is similar to the chosen plaintext indistinguishability experiment (see sec. 3.2.3.1). After the chosen plaintext part, the adversary is now enabled to ask for the decryption of chosen ciphertexts, as long as they were not obtained by a chosen plaintext query. At the end of the experiment, the adversary outputs a guess $b' \in \{0,1\}$. Hence, the adaptive chosen ciphertext adversary is more powerful than the chosen plaintext adversary.

Authenticated-encryption schemes are standardized in ISO 19772 [ISO09] or NIST SP 800-38 C/D. Examples for authenticated-encryption schemes are *Offset Codebook* (OCB) [Rog02], *Counter with CBC-MAC* (CCM) [WFH03], *Galois Counter Mode* (GCM) [MV04a] or encrypt-than-MAC [BN00].

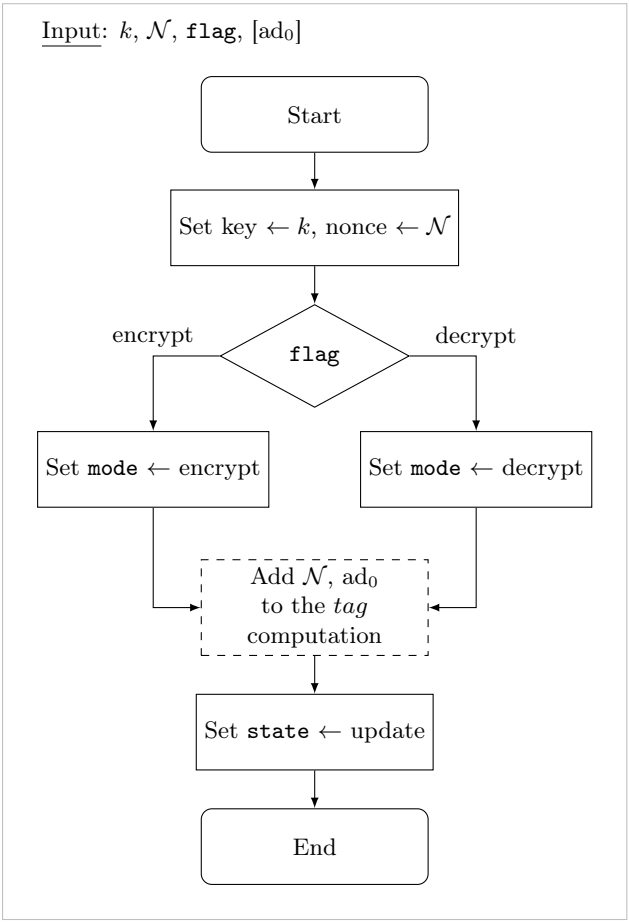
The AEAD *init*, *update* and *final* functions

Cryptographic *application programming interfaces* (APIs) like *OpenSSL libcrypto* or the *JAVA Cryptography Architecture* (JCA) define AEAD schemes as function-triplets consisting of three stateful functions: *init*, *update* and *final*. This definition abstracts from the concrete implementation of the scheme and it enables application developers to treat the AEAD scheme as a black box. Fig. 3.9 presents flow charts for the AEAD *init, update* and *final* functions in accordance to the DIN 66001 flowcharting notation. The dashed lines in Fig. 3.9 indicate abstract processes whose realizations differ depending on the AEAD scheme that is used for the implementation.

Let $\mathfrak{m}_i$ denote the plaintext, $\mathfrak{c}_i$ the ciphertext and ad_i the associated data, for $i = 0, ..., l-1$. The AEAD *init* function takes a key k, a nonce $\mathcal{N}$, optional associated-data ad_0 and a `flag` as input. The `flag` is a condition variable that is used to determine whether the scheme is going to perform encryption or decryption. $\mathcal{N}$ and ad_0 are added to the internal computation of the authentication *tag*. If `flag` = encrypt, the *update* function consumes plaintext $\mathfrak{m}_i$ and associated-data strings ad_i of arbitrary length. Internally, buffers are used to hold the data that will be processed, since most AEAD schemes work on n-bit blocks (see [ISO09]). In case that $|\mathfrak{m}_i| = \lambda \cdot n$, for some $\lambda \in \mathbb{N}^+$, the plaintext is encrypted and the *update* function outputs a ciphertext, which is a multiple of n-bit in length. If the plaintext is not a multiple of the block length, the *update* function encrypts n-bit plaintext blocks. The remaining bits are kept in the internal buffer and processed in further calls or during *final*. The *update* function adds each processed block to the internal *tag* computation. When the whole plaintext is consumed by the *update* function, the *final* function must be called. The function outputs the authentication *tag* and possibly a padded ciphertext block $\bar{\mathfrak{c}}$.

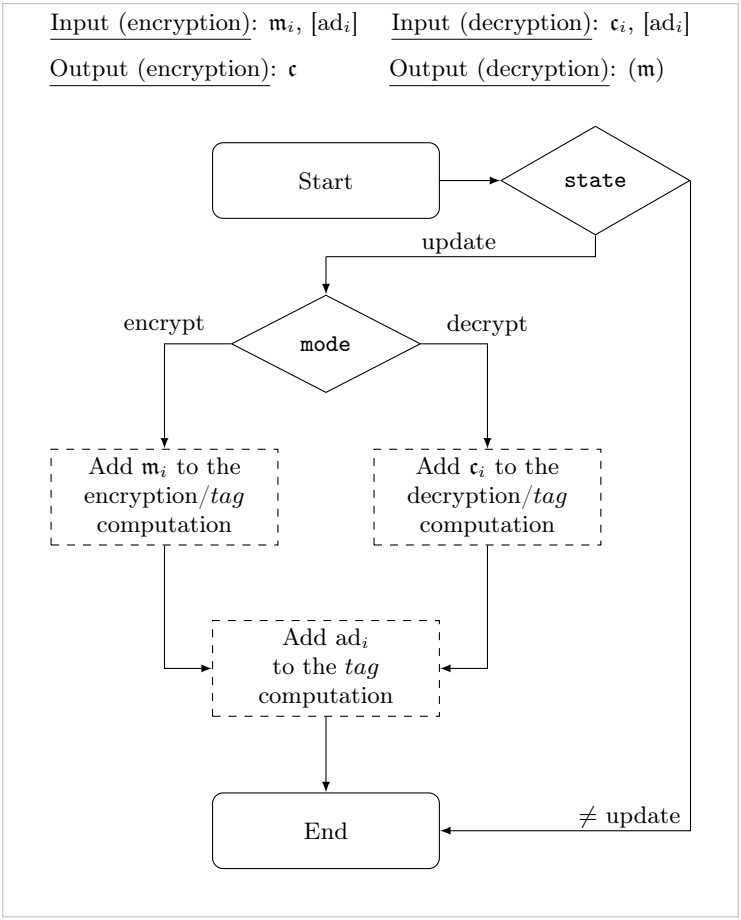
If *init* is called with the decryption `flag`, the *update* function processes the ciphertext and associated-data. $\mathcal{N}$ and ad_0 are added to the internal *tag* computation during *init*. The *update* function decrypts the ciphertext and it also adds $\mathfrak{c}_i$ together with the associated-data strings to the *tag* computation. The plaintext should only be released if the ciphertext is authentic. Hence, the *final* function is called with the received tag' as input and it releases the plaintext $\mathcal{M} = \{\mathfrak{m}_0, ..., \mathfrak{m}_{l-1}\}$ in case that $tag = tag'$ and otherwise it outputs $\perp$.

AEAD *init*


Input: k, $\mathcal{N}$, flag, [ad_0]
Start
Set key ← k, nonce ← $\mathcal{N}$
encrypt
flag
decrypt
Set mode ← encrypt
Set mode ← decrypt
Add $\mathcal{N}$, ad_0 to the *tag* computation
Set state ← update
End

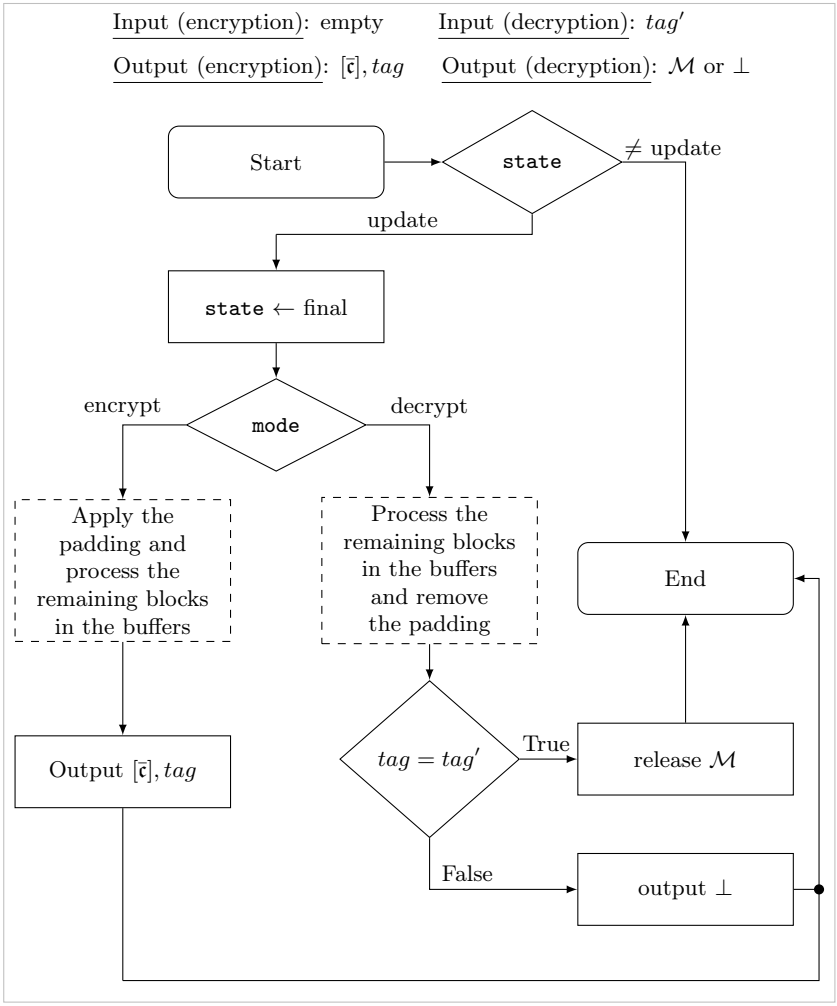
(a)

AEAD *update*



(b)

AEAD *final*



(c)

Figure 3.9.: The AEAD *init*, *update* and *final* functions.
(a) AEAD *init*.
(b) AEAD *update*.
(c) AEAD *final*.

Chapter 4

Compressive Sensing Based Encryption

4.1. General

A Compressive sensing based encryption system is a symmetric encryption system (see sec. 3.2.1) with the goal to protect the confidentiality of sampled signals $\vec{x}$. A high level description of a CS encryption system is depicted in Fig. 4.1. A sub-Gaussian *random* matrix $A \in \mathbb{R}^{m \times N}$ is treated as a secret key and is assumed to be shared between the legitimized parties. Encryption is performed by taking m random samples $\vec{y}$ from the plaintext signal $\vec{x}$ according to A, i.e. $\vec{y} = A\vec{x}$. Under the assumptions that $\vec{x}$ is sparse or compressible and since A achieves the RIP with very high probability [CT05, CT06], the signal can be reconstructed resp. decrypted from $\vec{y}$ using an appropriate algorithm that solves the sparse recovery problem (see sec. 3.1). Let *rec* denote this reconstruction algorithm, which takes as input the sampling matrix A together with the measurements $\vec{y}$ and an optional sparsifying basis Ψ and outputs $\tilde{x} = \vec{x} + \epsilon$, where ϵ is some bounded noise term introduced by the sampling system as discussed in section 3.1.3. The communicating parties publicly agree on an algorithm *rec* prior to encryption, hence it is expected that *rec* is also known to the adversary. Example algorithms for *rec* include *orthogonal matching pursuit* [TG07] or *basis pursuit* [CDS01], but for sake of simplicity and without loss of generality *rec* is viewed as a black box.

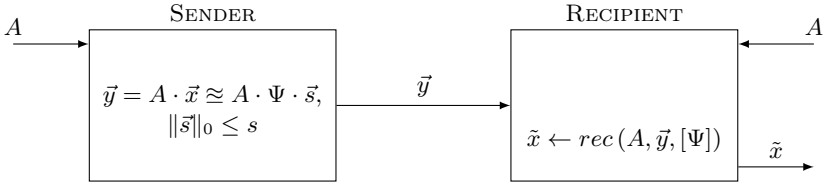


Figure 4.1.: High level description of CS based encryption.

Achieving confidentiality during the sampling process by keeping A as a shared secret was already mentioned in the seminal work of Candès and Tao [CT06]. They argued that a Compressive Sensing system achieves confidentiality against eavesdropping adversaries due to the randomness of the sampling matrix A. In order to reconstruct the signal from eavesdropped measurements, an adversary has to perform an exhaustive search to find the secret matrix, which is computational infeasible. Rachlin and Baron [RB08] were the first who studied the secrecy of the compressed measurements and they found that, while not perfectly secure [Sha49b] in general, CS is computational secure even if the adversary has some knowledge about the signals sparsity. The following section presents more detailed and recent results on the security of CS encryption in the one-time and many-time encryption scenario. Consequently, security of CS encryption schemes is defined more formally for different threat models.

4.2. Secrecy for One-Time Encryption

4.2.1. Establishing security definitions

A Compressive Sensing based encryption system Π is a symmetric encryption scheme of the form $\Pi = (Gen, \mathcal{E}_A, \mathcal{D}_A)$. Note the slight difference to the definition of a symmetric encryption scheme from section 3.2.1. Instead of fixing a finite key space, a probabilistic function Gen is defined, which is used to generate the key (i.e. the random matrix). The reason for this is that the key space might be an uncountable set, for example when the entries of the matrix A are drawn from a standard normal distribution. The encryption algorithm $\mathcal{E}_A$ takes a signal $\vec{x} \in \mathbb{R}^N$ as input and outputs $\vec{y} \in \mathbb{R}^m$, while the decryption algorithms $\mathcal{D}_A$ takes $\vec{y}$ and outputs $\tilde{x} = \vec{x} + \epsilon$. All algorithms are supposed run in polynomial time with respect to some implicit parameter κ.

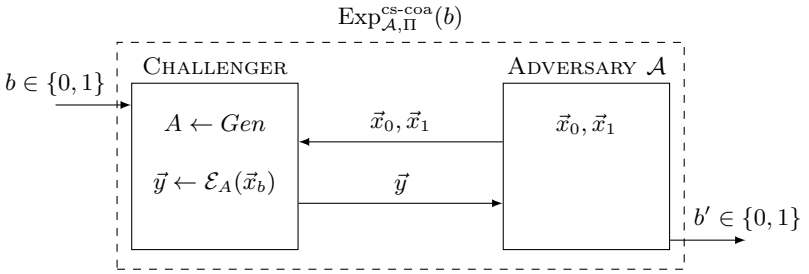


Figure 4.2.: The Compressive Sensing ciphertext only indistinguishability experiment $\text{Exp}^{\text{cs-coa}}_{\mathcal{A},\Pi}(b)$.

Now, consider the experiment $\text{Exp}^{\text{cs-coa}}_{\mathcal{A},\Pi}(b)$ presented in Fig. 4.2. Similar to the experiments from section 3.2.2, the adversary is allowed to choose two plaintext signals $\vec{x}_0, \vec{x}_1$ of equal length and sends them to the challenger. The challenger generates an RIP-matrix A by calling Gen and sends the encrypted signal $\vec{x}_b$ back to the adversary, i.e. $\vec{y} = A\vec{x}_b$. The adversary will output a guess b'. As before, the encryption scheme is called secure if the adversary cannot distinguish between the two experiments determined by the choice of b.

Unfortunately, the adversary wins $\text{Exp}^{\text{cs-coa}}_{\mathcal{A},\Pi}(b)$ with advantage 1 due to the nature and linearity of the encryption process. While the RIP from Eq. (3.3) is a necessary condition ensuring the reconstructability of the signal from compressed measurements, it also introduces some restrictions to the encryption process. For example, the adversary might ask for the encryption of $\vec{x}_0 = \vec{0}$ and $\vec{x}_1 = \vec{s}$, for some s-sparse vector $\vec{s}$, and he/she outputs $b' = 0$ if $\vec{y} = \vec{0}$ and $b' = 1$ otherwise. For all s-sparse signals, the RIP matrix A preserves the signals energy $\|\vec{s}\|_2^2$, consequently the adversary is able to distinguish encrypted signals with different energy.

Rachlin and Baron [RB08] were the first who noted this undesirable behavior of CS encryption and they used it to proof that CS does not achieve perfect secrecy. Later, Cambareri et al. [CMP+15a, CMP+15b] and Bianchi et al. [BBM14, BBM16] extended the security analysis of Compressed Sensing based encryption and they showed that for large enough sensing matrices the signals energy is in fact the only information an adversary learns from eavesdropped measurements. Another experiment must be defined in order to derive a suitable security definition for CS.

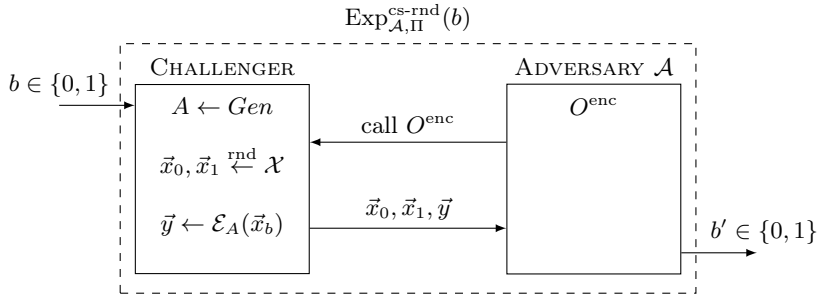


Figure 4.3.: The Compressive Sensing random plaintext indistinguishability experiment $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$

The experiment shown in Fig. 4.3 is used to establish a slightly weaker security definition for CS encryption, however, this notation is appropriate for practical Compressive Sensing systems with respect to the preferred sampling model (M1). The adversary gets access to a randomized encryption oracle O^{enc}, which will be simulated by the challenger. When the encryption oracle is called, it draws two plaintext signals $\vec{x}_0, \vec{x}_1$ uniformly at random from the set of all possible respectively applicable plaintexts, denoted by $\mathcal{X}$. According to the bit b, the oracle encrypts one signal $\vec{x}_b$ with a random sub Gaussian matrix A and sends the three-tuple $(\vec{x}_0, \vec{x}_1, \vec{y} \leftarrow A\vec{x}_b)$ back to the adversary. The experiments output is the adversary's guess b'.

The fact that an adversary might distinguish ciphertext obtained from plaintext signals with different energy remains, but now the challenger controls which plaintext is encrypted. If $\mathcal{X}$ is restricted to the unit $(N-1)$-sphere denoted by S^{N-1} all signals will have the same energy and are therefore asymptotically indistinguishable for large enough N as noted by [CMP+15a, BBM16]. Nevertheless, restricting the set of all applicable signals is not the most promising approach with respect to practical applications. Certainly, each signal with positive energy $\|\vec{x}\|_2^2 > 0$ can be projected to S^{N-1} by a simple normalization, i.e. $\hat{x} = \vec{x}/\|\vec{x}\|_2$, which was proposed originally by [BBM14] in order to increase security. This normalization can be integrated in the encryption algorithm $\mathcal{E}_A$, but the signals norm must be transmitted securely to the recipient for a proper decryption.

While the threat model considered in $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$ is weaker than the one of $\mathrm{Exp}^{\text{cs-coa}}_{\mathcal{A},\Pi}(b)$, the random plaintext indistinguishability experiment captures the fact that the adversary might not be able to choose plaintext messages in sampling model (M1) directly. CS based encryption is integrated into the sampling process, which means that whoever wants to encrypt chosen plaintext needs physical control over the sensing system. When the adversary has physical control over the sampling system, he/she might also obtain the sensing matrix, for example if the matrix is implemented by an array of micromirrors like the single pixel camera [WLD+06]. Now, a formal definition of the security of an CS encryption scheme can be given using $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$.

Definition 4.1. (ν-Indistinguishability)
A Compressive Sensing encryption scheme $\Pi = (Gen, \mathcal{E}_A, \mathcal{D}_A)$ is said to achieve ν-indistinguishability, if

$$\mathrm{Adv}^{\text{cs-rnd}}_{\mathcal{A},\Pi} \coloneqq |\Pr[\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(0)=1] - \Pr[\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(1)=1]| \leq \nu$$

holds for some negligible ν depending on the sampling matrix and all efficient adversaries $\mathcal{A}$.

Weak matrices

The matrix A achieves the RIP with very high probability if the entries of A are drawn at random from a sub-Gaussian distribution, which ensures that the equation $A\vec{x} = \vec{y}$ has a unique solution $\vec{x} \in \mathbb{R}^N$ (see sec. 3.1.2). However, it is still possible, but highly unlikely, that some weak A is chosen that has an undesired structure, for example if all entries of A are identical or the rows/-columns are too strongly related. In these cases, A does not fulfill the necessary conditions for a successful signal recovery. The system of interest has infinitely many solutions, which makes it infeasible for an adversary and also for the legitimized recipient to recover the original $\vec{x}$. More formally, let $\mathrm{rank}(A)$ denote the rank of the matrix A and let $\mathrm{null}(A)$ be the dimension of $\mathrm{nullspace}(A)$. For $A \in \mathbb{R}^{m \times N}$ with $N \gg m$ it holds that $\mathrm{rank}(A) \leq m$ and due to the rank-nullity theorem [KM03, Proposition 3.2.13] it follows that $\mathrm{null}(A) + \mathrm{rank}(A) = N$. Hence, $\mathrm{null}(A) = N - \mathrm{rank}(A) \geq N - m > 0$, which means that the dimension of A's nullspace is greater than zero. For every $\vec{v} \in \mathrm{nullspace}(A)$ the equation $A \cdot (\vec{x} + \vec{v}) = \vec{y}$ gives another possible solution.

4.2.2. Impact of different sensing matrix designs

Security implications

In contrast to the definitions from section 3.2, Def. 4.1 contains a measure ν that represents the level of achievable security, which depends on the sensing matrix design. Bianchi et al. established bounds on ν for Bernoulli random matrices and Gaussian random matrices in [BBM16]. They were able to prove that, at least in theory, measurements obtained with a Gaussian random matrix achieve $\nu = 0$ for all $\vec{x} \in S^{N-1}$, which is equivalent to perfect secrecy. This result stems from the fact that the indistinguishability is limited by $\delta(\Pr[\vec{y}|\vec{x}_0], \Pr[\vec{y}|\vec{x}_1]) \leq \nu$ where $\delta(\Pr[\vec{y}|\vec{x}_0], \Pr[\vec{y}|\vec{x}_1])$ denotes the statistical distance (i.e. total variation distance) between the two conditional probability measures (see [BBM16, Lemma 4]). In the case of normalized signals or $\vec{x} \in S^{N-1}$ the measurements cannot be distinguished from Gaussian random variables, which is compliant with the results from [CMP+15a], and $\Pr[\vec{y}|\vec{x}_0] = \Pr[\vec{y}|\vec{x}_1] = \Pr[\vec{y}]$ follows [BBM16, Prop. 1]. Unfortunately, those results are mainly of theoretical interest, because it is not practical to take random samples according to a continuous Gaussian distribution.

In the discrete case, it is expect that the entries of A vary from Gaussian, which is mainly due to the limited floating point precision, hence ν will depend on the difference of A from a Gaussian matrix. The generation of discrete Gaussian distributed numbers is an important problem in the field of lattice-based cryptography, especially for cryptosystems based on the *learning with error* problem [DG14, Fol14]. An overview of general techniques to obtain discrete Gaussian samples is given in [Dev86]. Usually, a uniform random bit-string from a random number generator builds the basis for the generation of discrete Gaussian samples by using large lookup tables that map from one distribution to the other. Example algorithms are variants of the Knuth-Yao random walk [SRVV14] or the discrete Ziggurat [BCG+14]. In order to derive bounds for the ν-indistinguishability of Compressive Sensing encryption schemes based on discrete Gaussian matrices, the quality and precision of the discrete Gaussian sampling and the underlying random number generator must be considered. It is also not clear how the discrete Gaussian sampling methods perform in CS cryptosystems with respect to properties like unpredictability or leakage of secret information, which are required for cryptographically secure pseudorandom

bit generators (cf. [ISO11]). It is still an open problem to verify if the bounds on ν for the continuous Gaussian matrices can be transferred to the discrete case.

Prominent Compressive Sensing applications, like the single pixel camera [WLD+06], are based on binary random matrices with values from $\{+1, -1\}$. Bernoulli sensing matrices are discrete random matrices where each entry is drawn uniformly at random from the set $\{+1, -1\}$. An advantage of these matrices is that they are implementable in hardware and they need far less memory than floating point based Gaussian matrices. The secure generation of Bernoulli random matrices can rely on trusted and well studied designs for deterministic random bit generators as standardized in ISO 18031 [ISO11]. The bounds from Bianchi et al. [BBM16, Lemma 5] on the parameter ν show that Bernoulli sensing matrices cannot achieve the same level of security as Gaussian matrices. In the asymptotic case, i.e. when $N \to \infty$ and m is large enough, the Lindeberg-Feller central limit theorem (cf. [Vaa00, Prop. 2.27]) applies and the measurements distribution converges to the standard normal distribution (cf. [CMP+15a, Prop. 2] and the corresponding proof). Hence, for Bernoulli matrices $\nu \to 0$ for $N \to \infty$ and m large enough. [BBM16] and [CMP+15a] verified that the convergence rate is about $\mathcal{O}(N^{-1})$.

Asymptotic indistinguishability with Bernoulli matrices

Due to the practical relevance of the Bernoulli matrix ensemble, the asymptotic behavior of Compressed Sensing with Bernoulli random matrices is investigated in an application scenario using the two sample Cramér-von-Mieses test [And62]. This statistical test determines if two random samples are drawn from the same underlying distribution. The test returns a p-value and if this value is greater than the significance $\alpha = 0.05$ the null hypothesis is accepted, meaning that the two samples are drawn from the same distribution. While p-values under the null hypothesis are uniformly distributed, the p-values under the alternative hypothesis will be close to zero. Compared to other applicable statistical tests, like the Kolmogorov-Smirnov test [Hod58], the Cramér-von-Mieses test shows a better accuracy [And62].

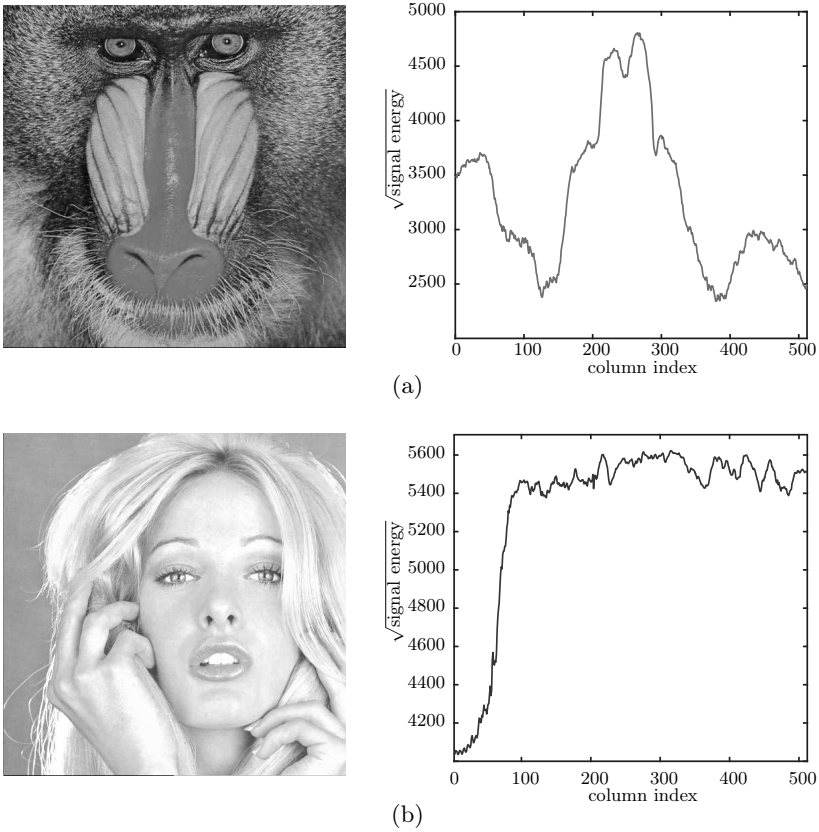


Figure 4.4.: An excerpt of the test set for the Cramér-von-Mieses test.
(a) The "baboon" test image with the corresponding energy values.
(b) The "tiffany" test image with the corresponding energy values.

The experiments are performed on a test set consisting of the 25 images of size 512×512 from volume three of the USC-SIPI image database*, except for the "ruler" image. Images are chosen as a test set for several reasons. On the one hand, CS-imaging is a possible application scenario for Compressed Sensing encryption. On the other hand, the signals are not drawn at random, since randomness might influence the results from the statistical test in an

*http://sipi.usc.edu/database/database.php?volume=misc

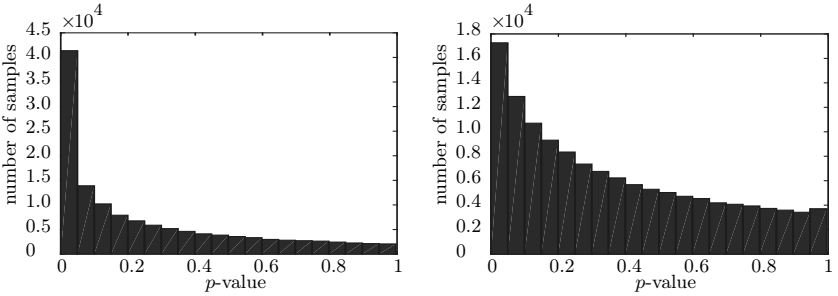


Figure 4.5.: Cramér-von-Mieses p-value histograms of
(a) the "baboon" test image and
(b) the "tiffany" test image, both with 512×512 pixel.

unintended way. Moreover, the usage of a publicly available and static image database makes the results reproducible. Fig. 4.4 presents two images from the test set with the corresponding energy values for each column vector. Due to the implications on the indistinguishability of random measurements, the signals energy is an important characteristic in the experiment. It can be seen that the energy values of the "tiffany" test image – roughly from column 90 on – differ only slightly from each other, meaning that the variation of the energy is not so strong. In contrast, the energy values of the "baboon" test image vary more strongly especially in the centered image area.

For the experiments, all images are processed column-wise with independent Bernoulli random matrices, meaning that each signal $\vec{x}_i$ is a column of the image consisting of $N = 512$ pixel values and each signal is sampled with a fresh Bernoulli random matrix ($i = 0, ..., 511$). The compression ratio $1 - \frac{m}{N}$ was set to 60%. All the resulting measurements are compared with the Cramér-von-Mieses test, which yields $\binom{512}{2} = 130816$ p-values for each image.

Fig. 4.5 shows the resulting p-value histograms for the two examples from Fig. 4.4. It can be seen that the statistical test is able to distinguish the compressed measurements by their distribution. The peaks around zero shows that there are a lot of p-values supporting the alternative hypothesis, however, there are also differences in the p-value histograms obtained from the different images. For signal sets with a stronger variation in the signals energy, like the "baboon",

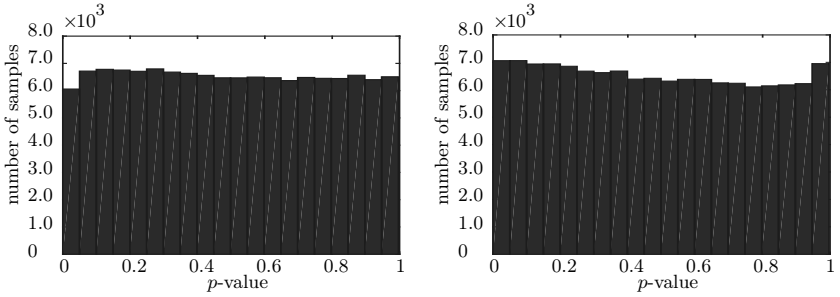


Figure 4.6.: Cramér-von-Mieses p-value histograms for normalized signals. (a) results from the "baboon" test image and (b) results from the "tiffany" test image.

it can be seen that more p-values are close to zero compared to images with a smaller variation in energy, like "tiffany" (see Fig. 4.4). This shows the connection between the signals energy and the indistinguishability of compressed measurements.

When the signals are normalized by $\hat{x}_i = \vec{x}_i / \|\vec{x}_i\|_2$, the statistical test is not able to distinguish the measurements. In order to see this, consider Fig. 4.6. The p-values tend to be uniformly distributed, which shows that the statistical test confirms the null hypothesis. This result also supports the results from [BBM16] and [CMP+15a] on the asymptotic indistinguishability of random measurements sampled with Bernoulli matrices.

To study the asymptotic behavior of the ν-indistinguishability, the test set is scaled down to 128×128 and 256×256 pixel by cubic interpolation. The p-value histograms for the normalized "baboon" and "tiffany" signal sets are given in Fig. 4.7. It can be seen that even for relatively small N the statistical test supports the null hypothesis when the signals are normalized to equal energy. By comparing the results from Fig. 4.6 and Fig. 4.7 one observes the asymptotic secrecy of the compressed measurements, since the distribution of the p-values tends stronger to uniform when N increases.

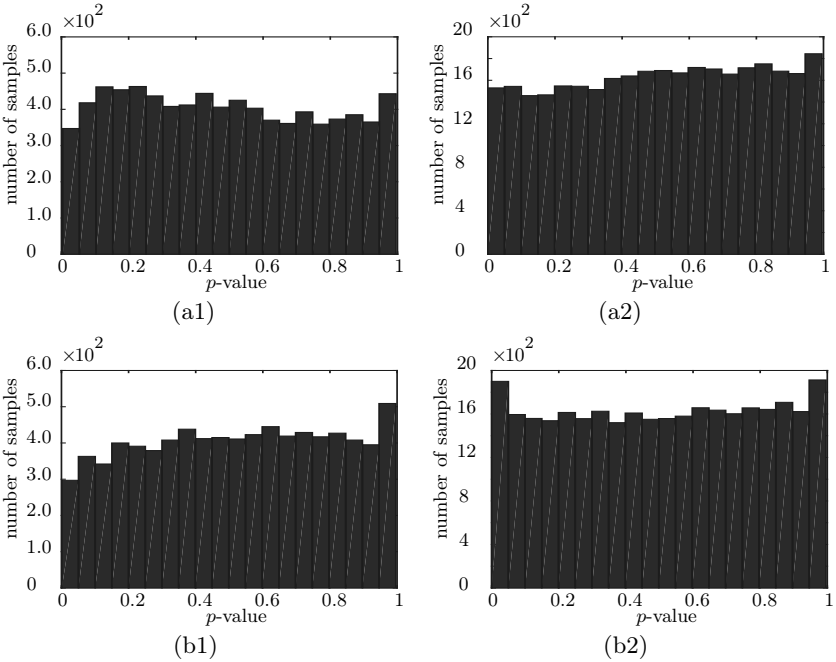


Figure 4.7.: Cramér-von-Mieses p-value histograms for different image sizes. (a1) shows the test outcome for "baboon" 128×128 and (a2) is "baboon" 256×256. (b1) "tiffany" 128×128, (b2) "tiffany" 256×256.

4.3. Secrecy for Many-Time Encryption

The considered Compressive Sensing encryption schemes achieve a decent level of security in the one-time encryption scenario that mainly depends on the sensing matrix design. When multiple signals are encrypted under the same matrix, the encryption algorithm is no longer randomized with the consequence that an adversary will recognize if the same plaintext signal is encrypted multiple times. Moreover, the RIP (cf. Def. 3.4) ensures that the distance between two s-sparse vectors will be preserved, which implies continuity, since neighboring signals will be encrypted to similar measurements. Normalizing the signals to equal energy defends against this information leakage, but more powerful attacks can easily break the encryption scheme. Assume that the adversary

is able to ask for the encryption of N plaintext signals and let $\vec{e}_i$ denote the i-th unit vector of the standard basis $\mathbb{R}^N$. On input $\vec{e}_i$, the challenger outputs $\vec{a}_i = A\vec{e}_i$, i.e. the columns of the secret matrix, which enables the adversary to decrypt all subsequent ciphertexts after N queries. Even if the signals are normalized to equal energy, the adversary would gain enough useful information to break the system. For example, assume that Bernoulli matrices are used. In this case, the adversary is just interested in the measurements sign, which does not change due to normalization. With Gaussian matrices, the obtained sensing matrix is equal to the original matrix up to a scaling factor and still useful for reconstruction (cf. [BDDH11, Chapter 3]).

Fig. 4.8 presents an analysis of a chosen plaintext attack against a system using a fixed Bernoulli random matrix A. First, the adversary obtains encrypted measurements $\mathcal{Y} = \{\vec{y}_0, ..., \vec{y}_{l-1}\}$ of the "baboon" 256×256 test image with a color depth of $D = 8$ bit per pixel, sampled row-wise, i.e. $N = 256$. The adversary's goal is to obtain enough content information in order to decide which image is encrypted. The adversary starts by generating a Bernoulli random matrix A' and with each unit vector query, the corresponding column of A' is replaced by $\vec{a}_i$. Fig. 4.8 presents the *peak signal-to-noise ratio* (PSNR) (Eq. (4.1)) of the image that the adversary recovers by using A' in the publicly known reconstruction algorithm compared to the original image.

The PSNR is computed by [BL11]:

$$\mathrm{PSNR}(img_1, img_2) = 10 \cdot log_{10} \left(\frac{\left(2^D - 1\right)^2}{\mathrm{MSE}} \right) \ [\mathrm{dB}] \tag{4.1}$$

where D denotes the color depth of the images. Further, MSE is the mean squared error of the two $l \times N$ images [BL11]:

$$\mathrm{MSE}(img_1, img_2) = \frac{1}{l \cdot N} \sum_{i=0}^{l-1} \sum_{j=0}^{N-1} \left(img_1(i,j) - img_2(i,j) \right)^2 \tag{4.2}$$

As it can be seen in Fig. 4.8, the PSNR increases with the number of chosen plaintext queries. After about 150 queries, the silhouette of the "baboon" becomes more and more visible and after 220 queries the PSNR is around 10 dB.

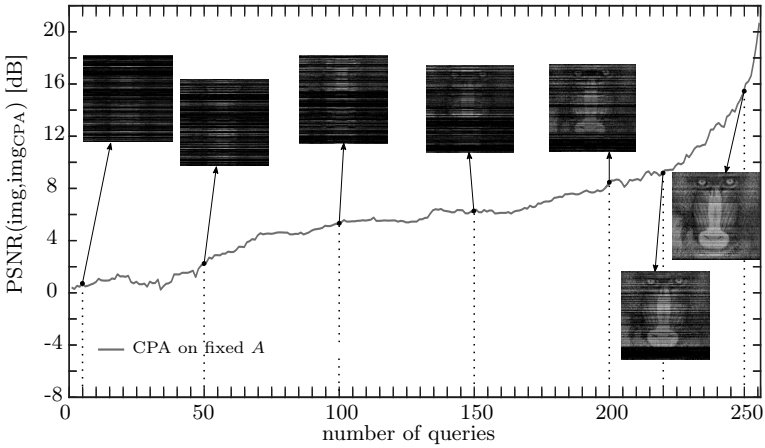


Figure 4.8.: Analysis of a chosen plaintext attack against CS with a fixed matrix $A \in \{+1, -1\}^{m \times N}$.

In contrast, consider Fig. 4.9, where each signal was encrypted using an independent random Bernoulli matrix A_i. The PSNR is close to zero, independently of the number of chosen plaintext queries. The reason for this is that the adversary obtains columns of independent matrices that are not suitable for the reconstruction of the eavesdropped measurements. The only visible structure comes from the reconstruction algorithm, which outputs some sparse vector (cf. section 3.1.3). However, since sampling was performed with another matrix, the reconstruction algorithm is not able to recover the original signal.

If a CS encryption scheme is used to encrypt multiple signals it can only be secure if each signal is encrypted with a different random sampling matrix [Fay16]. If the matrices are independent of each other, the adversary has to solve multiple instances of the $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$ experiment (cf. Fig. 4.3). Hence, if the challenger normalizes the signals to equal energy, the resulting measurements achieve ν-indistinguishability [BBM16]. An CPA-adversary only obtains columns of independent random matrices by the aforementioned chosen-plaintext key recovery attack, which does not help him/her to predict further matrices or to trace previous matrices.

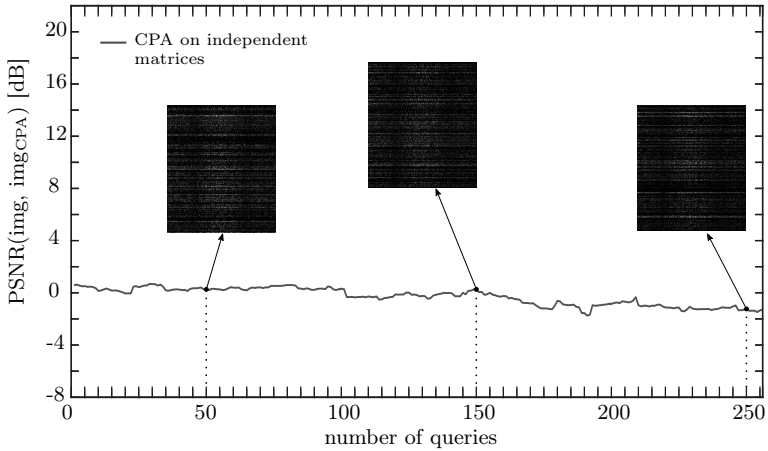


Figure 4.9.: Analysis of a chosen plaintext attack against CS with fresh random matrices $A_i \xleftarrow{\text{rnd}} \{+1, -1\}^{m \times N}$.

A formal security definition for many-time encryption

Formally, a Compressive Sensing encryption scheme $\Pi = (Gen, \mathcal{E}, \mathcal{D})$, which achieves ν-indistinguishability for all $\vec{x} \in S^{N-1}$ in a many-time encryption scenario, works by generating fresh random matrices A_i for each signal $\vec{x}_i$, where $0 \leq i \leq l-1$. This can be formulated using another experiment, depicted in Fig. 4.10.

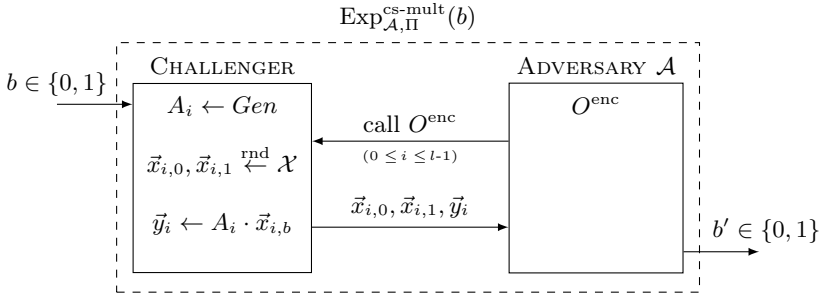


Figure 4.10.: The Compressive Sensing multiple random plaintext indistinguishability experiment $\text{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(b)$.

Definition 4.2. (ν-Indistinguishability for many-time encryption)
A Compressive Sensing encryption scheme $\Pi = (Gen, \mathcal{E}, \mathcal{D})$ achieves ν-indistinguishability for many-time encryption, if

$$\mathrm{Adv}^{\text{cs-mult}}_{\mathcal{A},\Pi} := |\Pr[\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(0)=1] - \Pr[\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(1)=1]| \leq \nu$$

holds for some negligible ν depending on the sensing matrix design and all efficient adversaries $\mathcal{A}$.

For statistically independent matrices, $\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(b)$ can be viewed as multiple instances of the $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$ experiment (cf. Fig. 4.3). The adversary is allowed to query the encryption oracle O^{enc} at most l times and because $\mathcal{A}$ is efficient, the number of queries is at most polynomial in the running time of $\mathcal{A}$.

Under the assumption that ν is negligible in the experiment $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$ and since the sum of negligible functions is still negligible [KL15, Prop. 3.6] the adversary does not gain a notable advantage in the $\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(b)$ experiment. It must be noted that for Bernoulli random matrices ν is only negligible in the asymptotic case, i.e. for large enough N, m (cf. section 4.2.2).

Limited signal spaces and known plaintext attacks

A possible strategy to break Π in experiment $\mathrm{Exp}^{\text{cs-rnd}}_{\mathcal{A},\Pi}(b)$ would be to use $\vec{x}_{i,0}, \vec{x}_{i,1}$ and $\vec{y}_i$ in order to find the secret sensing matrix A_i. In practice, the set of possible $\vec{x}$ might be limited by the application. For example $\vec{x}$ could represent an image with a color depth of 8 bit per pixel. Assume that an adversary knows a pair of $\vec{x}$ and $\vec{y}$ and tries to find the matrix A that was used for encryption. When the entries of A are drawn from a continuous distribution there are infinitely many possible solutions for A. Even in the case where A is discrete, for example when A is a Bernoulli random matrix, there are still many candidate solutions to $A\vec{x} = \vec{y}$ for a given $\vec{x}$, $\vec{y}$ pair. [CMP+15b] investigated the complexity of such a known-plaintext attack against Bernoulli random matrices. They proved that the adversary's problem is equivalent to multiple instances of a subset sum problem, a special case of knapsack problem. While the subset sum problem is in general NP-complete, [CMP+15b] showed that the solution to a *known plaintext attack* (KPA) on CS with Bernoulli random matrices can be found in polynomial time. In order to establish this results, [CMP+15b] assumed

that $\vec{x}$ has only nonzero entries. This is a very strong restriction in general, but it favors the attacker. However, the obtained solutions are not unique and for each row there exists an enormous number of candidate solutions that grows with $\mathcal{O}(\sqrt{N}^{-1} \cdot 2^N)$ (see [CMP+15b, Theorem 1/2]). Hence, the success probability of a known plaintext attack is negligible for large enough N. Furthermore, it is very likely that $\vec{x}$ is the only signal that is encrypted with the matrix A. Even if the adversary would be successful with his/her known plaintext attack, the obtained information does not help to predicting another sampling matrix that is used in the encryption scheme.

Chapter 5

Compressive Sensing Encryption Modes

5.1. Introduction

Similar to block cipher modes of operation, a Compressive Sensing encryption mode should ensure the confidentiality of plaintext that consists of multiple signals. The findings from the previous sections can now be used to develop CS encryption schemes that meet the security requirements and besides that have other interesting properties. While ν-indistinguishability is the strongest security notation that Compressive Sensing based encryption can achieve, the modes of operation should not harm security significantly. Therefore, the modes of operations presented in this thesis will be analyzed with respect to their security. In addition, the proposed modes are compared and their properties are examined.

Generating sensing matrices from a key

The legitimized communicating parties must share many random sensing matrices in order to encrypt multiple signals. Given the large size of these matrices it seems unreasonable to exchange them all previous to encryption. It is more convenient to rely on trusted key establishment mechanisms, like the authenticated ephemeral (elliptic curve) Diffie-Hellman key exchange, to establish a shared key k, which is then used to generate sensing matrices. Moreover, generating

sensing matrices from a shared key allows to create sensing matrices when they are needed for sampling, such that signals can be sampled "on the fly".

Let $\Pi = (Gen, \mathcal{E}, \mathcal{D})$ be a Compressed Sensing encryption scheme that achieves ν-indistinguishable for many-time encryption by encrypting each signal $\vec{x}_i \in S^{N-1}$ with a statistically independent sampling matrix A_i (cf. Def. 4.2). Further let $\bar{\Pi} = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$ be a modified version of Π. The difference between the two schemes is that in $\bar{\Pi}$ a key k is chosen uniformly at random from $\mathcal{K}$ and at each call of $\mathcal{E}_k, \mathcal{D}_k$ the keyed *pseudorandom number generator* (PRNG) Gen_k is used to generate A_i. With the modified encryption scheme, the communicating parties just need to exchange a single key that is used to generate sensing matrices deterministically. The following lemma establishes the security of $\bar{\Pi}$ based on the assumption that Gen_k is a cryptographically secure PRNG.

Lemma 5.1. *Let $\mathcal{A}$ be an efficient adversary that makes at most l queries and let Π be a Compressive Sensing encryption scheme achieving ν-indistinguishability in $\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(b)$. Further let $\bar{\Pi} = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$ be a modified encryption scheme where each A_i is generated using a PRNG Gen_k with a random k. When the probability to distinguish the output Gen_k from random is at most $\varepsilon_{\mathrm{prng}}(\kappa)$, for some negligible function $\varepsilon_{\mathrm{prng}}(\kappa)$, the advantage of $\mathcal{A}$ against $\bar{\Pi}$ is limited by*

$$\left|\Pr[\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\bar{\Pi}}(b) = b] - \Pr[\mathrm{Exp}^{\text{cs-mult}}_{\mathcal{A},\Pi}(b) = b]\right| \leq \varepsilon_{\mathrm{prng}}(\kappa) \tag{5.1}$$

Lemma 5.1 is confirmed using a transition between the two experiments based on the indistinguishability of the output of the PRNG from random following the seminal framework of [BR04] and [Sho04]. The reader is referred to [KL15, Section 3.3.2] for an introduction to proofs by reduction.

Proof. Let $\mathcal{B}$ be an efficient adversary that has oracle access to an algorithm O and tries to tell if he/she is interacting with an oracle that outputs random matrices, denoted by O^{rnd}, or an oracle O^{real} that outputs pseudorandom matrices using Gen_k with a random k. Now, $\mathcal{B}$ is using $\mathcal{A}$ as a black box in order to distinguish O^{rnd} from O^{real}. Therefore, $\mathcal{B}$ simulates $\mathrm{Exp}^{\text{cs-mult}}$ for $\mathcal{A}$ and on the event that $\mathcal{A}$ succeeds in the experiment, $\mathcal{B}$ guesses that O is pseudorandom, otherwise O is regarded as random. Formally, $\mathcal{B}$ is constructed as follows:

1. fix $b \in \{0, 1\}$.
2. choose $\vec{x}_{i,0}, \vec{x}_{i,1} \xleftarrow{\mathrm{rnd}} S^{N-1}$ and call O to receive an A_i.

3. give $\vec{x}_{i,0}, \vec{x}_{i,1}$ and $\vec{y} \leftarrow A_i \cdot \vec{x}_{i,b}$ to $\mathcal{A}$.

4. continue to simulate $\text{Exp}^{\text{cs-mult}}$ (i.e. repeat step 2.-3.) for $\mathcal{A}$ while $i < l$.

5. When $\mathcal{A}$ outputs a bit b', output 1 if $b' = b$ and 0 otherwise.

Two cases need to be considered according to the oracle O:

I) *$\mathcal{B}$ is interacting with O^{rnd}*:
In this case, $\mathcal{B}$ mimics $\text{Exp}_{\Pi}^{\text{cs-mult}}(b)$ exactly from the perspective of $\mathcal{A}$. Each measurement is encrypted using an independend random matrix A_i as it would be in Π. Let $\mathcal{B} \rightarrow 1$ denote the event that $\mathcal{B}$ outputs 1, then

$$\Pr[\mathcal{B}(O^{\text{rnd}}) \rightarrow 1] = \Pr[\text{Exp}_{\mathcal{A},\Pi}^{\text{cs-mult}}(b) = b]. \tag{5.2}$$

II) *$\mathcal{B}$ is interacting with O^{real}*:
Here, $\mathcal{B}$ simulates $\text{Exp}_{\bar{\Pi}}^{\text{cs-mult}}(b)$ from the perspective of $\mathcal{A}$. Each measurement is encrypted using a pseudorandom matrix A_i obtained by calling Gen_k with a random k, as it would be in $\bar{\Pi}$, such that

$$\Pr[\mathcal{B}(O^{\text{real}}) \rightarrow 1] = \Pr[\text{Exp}_{\mathcal{A},\bar{\Pi}}^{\text{cs-mult}}(b) = b]. \tag{5.3}$$

According to the assumption that the output of Gen_k cannot be distinguished from random with probability greater than $\varepsilon_{\text{prng}}(\kappa)$ it follows that

$$\left|\Pr[\mathcal{B}(O^{\text{real}}) \rightarrow 1] - \Pr[\mathcal{B}(O^{\text{rnd}}) \rightarrow 1]\right| \leq \varepsilon_{\text{prng}}(\kappa) \tag{5.4}$$

Using Eq. (5.2) and Eq. (5.3), Eq. (5.4) can be written as

$$\left|\Pr[\text{Exp}_{\mathcal{A},\bar{\Pi}}^{\text{cs-mult}}(b) = b] - \Pr[\text{Exp}_{\mathcal{A},\Pi}^{\text{cs-mult}}(b) = b]\right| \leq \varepsilon_{\text{prng}}(\kappa) \tag{5.5}$$

and the claim follows. □

Lemma 5.1 shows that the security loss against computationally bounded adversaries by using a cryptographically secure PRNG to generate matrices can be reduced to the security of the PRNG.

5.2. The General Design

5.2.1. Design goals

The following sections present the general design for Compressive Sensing based confidentiality modes. This general design is developed according to the following design goals derived from the analysis of Compressed Sensing based encryption presented previously.

- ▷ **ν-indistinguishability in the many-time encryption scenario**: Compressive Sensing encryption modes should achieve asymptotic ν-indistinguishability in the many-time encryption scenario (cf. Def. 4.2). In order to handle signals with different energy, the signals should be normalized and the normalization value should be transmitted securely next to the compressed measurements. Legitimized recipients who share the secret encryption key should be able to decrypt the norm and obtain the correctly scaled plaintext signal.

- ▷ **Resistance against chosen plaintext key recovery**: Each signal should be encrypted using a fresh pseudorandom matrix generated from a shared key k. Hence, a CS encryption mode turns the deterministic behavior of the encryption algorithm $\mathcal{E}_A$ after fixing the matrix into a randomized encryption scheme $\mathcal{E}_k$.

- ▷ **Reconstructability and adaptability**: The matrices generated by a Compressive Sensing encryption mode should fulfill the necessary conditions for sparse recovery like the RIP, as introduced in section 3.1.2. By this means, the modes of operation are adaptable in the sense that multiple Compressive Sensing reconstruction algorithms can be used.

- ▷ **Security by design**: Dedicated CS confidentiality modes should be designed in such a way that their security can be examined and justified. Therefore, they should be based on trusted cryptographic primitives and construction. In addition, the restrictions under which the security of a dedicated design holds must be stated explicitly. This will help that implementers, who are no experts in cryptography, are able to implement the modes of operation correctly.

Apart from the general design goals, certain modes may have different desirable properties like *parallelizability* or *self-synchronization* in case of ciphertext losses or errors.

5.2.2. Matrix generation

5.2.2.1. Matrix generation algorithm

The CS encryption modes developed in [Fay16, FR16, FR17] are using a matrix generation function Gen_k to obtain Bernoulli random matrices. The decision to use Bernoulli random matrices instead of other sub Gaussian ensembles has several reasons: on the one hand, binary matrices need less memory than floating point matrices and on the other hand approved and trusted constructions can be used to generate cryptographically secure pseudorandom binary matrices. Moreover, Gaussian matrix ensembles are mainly from theoretical interest and prominent Compressive Sensing applications rely on binary matrices. The asymptotic secrecy of Bernoulli matrices is convenient for most Compressive Sensing applications, since the signal dimensions are in general very large.

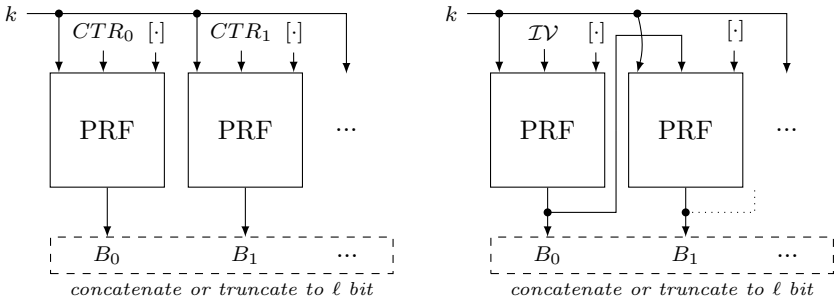
Gen_k is defined as a function that takes as input a key and an n-bit nonce $\mathcal{N}_i$ and outputs a sampling matrix $A_i \in \{+1, -1\}^{m \times N}$ that is indistinguishable from a truly random matrix from the same set ($i = 0, \ldots, l-1$). Therefore, the matrices satisfy the conditions for sparse recovery, like the RIP, with overwhelmingly high probability (see section 3.1.2). Various Compressive Sensing reconstruction algorithms that are based on the RIP can be used to recover the signals even from noisy measurements given the correct key. From now on, binary strings are treated as either from $\{0,1\}^*$ or $\{+1,-1\}^*$, since they can be mapped easily from one set to the other by a simple conversion function.

The matrix generation algorithm is modeled using an unlimited output pseudorandom function (PRF*) defined by $f_k^\ell : \mathcal{K} \times \{0,1\}^* \to \{+1,-1\}^\ell$. Note that the output length ℓ might be arbitrary large and ℓ is used as an implicit function parameter. $Gen_k(\mathcal{N}_i)$ works by calling $f_k^{m \cdot N}(\mathcal{N}_i)$ in order to produce $m \cdot N$ bits. The pseudorandom bits are then reshaped into the $m \times N$ matrix A_i, which is the output of Gen_k. More formally stated, if the pseudorandom bits outputted by $f_k^{m \cdot N}(\mathcal{N}_i)$ are denoted by $b_0, b_1, \ldots, b_{mN-1}$, fill the rows of $A_i = (\alpha_0, \alpha_1, \ldots, \alpha_{m-1})^T$ as described in Eq. (5.6).

$$\begin{aligned}\alpha_0 &= (b_0, b_1, \ldots, b_{N-1}) \\ \alpha_1 &= (b_N, b_{N+1}, \ldots, b_{2N-1}) \\ &\vdots \\ \alpha_{m-1} &= (b_{(m-1)N}, b_{(m-1)N+1}, \ldots, b_{mN-1}).\end{aligned} \tag{5.6}$$

5.2.2.2. Implementation details

In general, unlimited output pseudorandom functions are built from PRFs $f_k : \mathcal{K} \times \{0,1\}^* \to \{+1,-1\}^L$ with a limited and implicit output length L. In order to build a PRF* from f_k, the PRF output is extended by running f_k in *counter mode* or *feedback mode* as shown in Fig. 5.1. The counter mode construction works by incrementing an n-bit counter CTR_i with a random initial value CTR_0 (cf. [Che09, Section 5]). The counter is given as an input to the PRF such that each PRF call yields an distinct output $B_i \in \{+1,-1\}^L$. The feedback mode requires an n-bit random initialization vector $\mathcal{IV}$ as input to the first PRF call. The PRF output is then used as feedback for the next call. Both constructions allow optional input, denoted by $[\cdot]$. After $\lceil \ell / L \rceil$ calls to the PRF, the sequence of outputs $B_0, B_1, ...$ is concatenated and, if necessary, truncated to ℓ bit.



(a) Counter mode (b) Feedback mode

Figure 5.1.: Example constructions to build a PRF* from a PRF [Che09].

Suitable primitives to implement the PRF

Various primitives can be used to implement the PRF in the aforementioned PRF $\rightarrow$ PRF* constructions. Natural candidates are keyed cryptographic hash functions or block ciphers. When ℓ is large, like it is the case in most Compressive Sensing applications, keyed hash functions are more suitable due to their larger output length of up to 512 or even 1024 bit. In comparison, standardized block ciphers are limited by their 64 or 128 bit block size.

The simplest construction to implement a PRF using a hash function H is the *key prefix method*, which works by prepending a key to the message and hashing the concatenated bit string, i.e. $H(k \,||\, \mathfrak{m})$. Unfortunately, hash functions that are built from the iterative construction from ISO 10118-1 [ISO00] are no PRFs in the key prefix construction. Dedicated and standardized hash functions like SHA1 or SHA256 are built from this design. In order to see that these hash functions are no PRFs, when the key is prepended to the message, consider Fig. 5.2. A compression function f that processes L bit blocks is used to build the hash function H. At first, the message $\mathfrak{m}$ is padded to a multiple of L bits and the padded message $\bar{\mathfrak{m}}$ is split into L bit blocks. During the iteration phase, each block $\mathfrak{m}_1, ..., \mathfrak{m}_{\lceil |\bar{\mathfrak{m}}|/L \rceil}$ is processed by the compression function f and the final hash h is obtained after an optional truncation step.

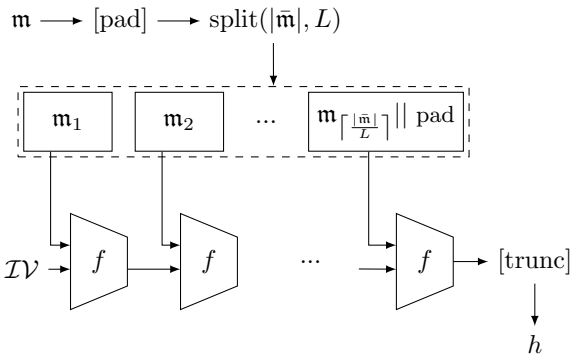


Figure 5.2.: The general design of a hash function H built from a compression function f, originally proposed by Merkle and Damgård [Mer90][Dam90].

The compression function f and the $\mathcal{IV}$ must be specified in order to define a dedicated hash function. Now, if an adversary obtains a message $\mathfrak{m}$ together with $H(k \,||\, \mathfrak{m})$ he/she is able to extend $\mathfrak{m}$ for example by a block x and compute $f(H(k \,||\, \mathfrak{m}), x) = H(k \,||\, \mathfrak{m} \,||\, x)$. This is a well known length extension attack [Tsu92]. Hence, the adversary is able to distinguish the keyed hash function from a PRF.

Secure constructions for keyed hash functions are *NMAC* [BCK96]:

$$\mathrm{NMAC}_{k=(k_1||k_2)}(\mathfrak{m}) \coloneqq H\left(k_1 \,||\, H\left(k_2 \,||\, \mathfrak{m}\right)\right) \tag{5.7}$$

and a special instance of NMAC called HMAC [BCK96]:

$$\mathrm{HMAC}_k(\mathfrak{m}) \coloneqq H\left(k \oplus opad \,||\, H\left(k \oplus ipad \,||\, \mathfrak{m}\right)\right). \tag{5.8}$$

The constants *opad* and *ipad* in HMAC are standardized and used to derive different keys for the inner and outer hash function calls (cf. [Tur08]). The security of NMAC and HMAC was proven in [Bel06] under the assumption that the compression function f is a PRF for a random k. Therefore, NMAC and HMAC can be used with all standardized hash functions to build a secure PRF. NIST [Che09] recommends HMAC as PRF for *key derivation functions* (KDFs) in counter or feedback mode (see Fig. 5.1). KDFs can be regarded as a special use case of the aforementioned PRF $\rightarrow$ PRF* constructions.

Extendable-output functions

Another group of interesting constructions that can be used to build PRF* straightforwardly are *extendable-output functions* (XOFs). An XOF is a hash function with an unlimited output length. Examples for XOFs include the recently standardized SHAKE [NIS14] – a variation of SHA-3 resp. Keccak [BDPA11a] – and another finalist of the SHA-3 competition called Skein [FLS$^+$10]. Both function families are built on a different design than the one presented in Fig. 5.2. The sponge construction, that is used in Keccak and SHA-3, is presented in Fig. 5.3. Two parameters, the rate r and the capacity c can be chosen. At first, the message is padded to a multiple of r bit and then the padded message is processed in r bit blocks $\mathfrak{m}_1, ..., \mathfrak{m}_{\lceil |\bar{\mathfrak{m}}|/L \rceil}$. The round function f is a permutation working on $w = r + c$ bits, where c can be regarded

as security parameter of the hash function. After processing the message (the absorbing phase in Fig. 5.3) the round function is used to output r bits on each further invocation, which is called the squeezing phase. In comparison to the construction from Fig. 5.2 with a fixed output length, the squeezing phase can be extended until the desired number of ℓ bits is produced. The sponge construction shows how the design of hash functions that support variable output length differ from the classical ISO 10118-1:2000 design [ISO00]. The main advantage of XOFs is that they do not need an additional output extension through counter mode or feedback mode unlike limited output length hash functions. It must be noted that an XOF with output length ℓ and security parameter c cannot provide more than $\ell/2$ resp. $c/2$ bits security against collisions, depending on whichever is less (cf. [NIS14, A.1]). Hence, the security of an XOF is given by $d = \min(\ell/2, c/2)$, under the assumption that f is a PRF.

In addition to the variable output length, Keccak, SHAKE and Skein are secure PRFs when a random key is used as a prefix to the message and both function families can be used securely in the NMAC or HMAC construction [BDPA11b][BKL+09]. Therefore, the keyed versions of Keccak, SHAKE and Skein can be defined by $H(k \,||\, \mathfrak{m})$.

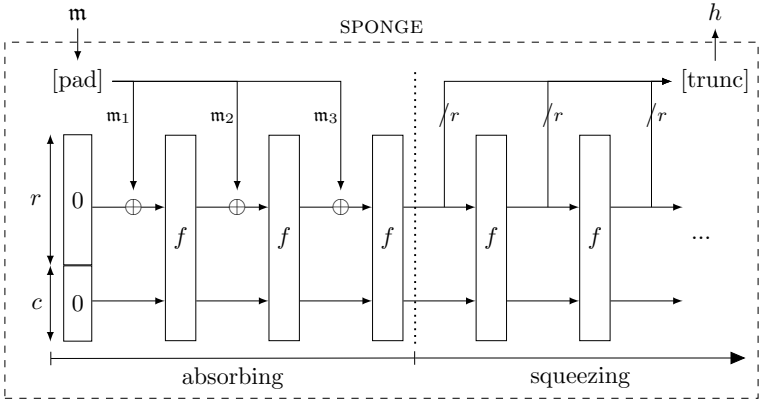


Figure 5.3.: The sponge construction used in Keccak, SHA3 and SHAKE (adapted from [BDPVA12]).

Recommended constructions and parameters

PRF*	*Output length of the primitive*	*Security level of the primitive (in bit)*†	*Number of primitive calls for* $\ell = m \cdot N$ *bits*	*Standardized in*
keyed SHAKE256‡	arbitrary	$\min(256, \ell/2)$	1	[NIS14]
keyed XOF-Skein512	arbitrary	$\min(256, \ell/2)$	1	-
counter mode HMAC-SHA3-512	$L = 512$	256	$\lceil \frac{m \cdot N}{L} \rceil$	[Che09] [NIS14]
counter mode HMAC-SHA-512	$L = 512$	256	$\lceil \frac{m \cdot N}{L} \rceil$	[Che09] [ISO04]

†A security level of $\mathfrak{b}$ bit means that the computational complexity of a generic attack against the primitive is $\geq 2^{\mathfrak{b}}$ (in case of hash functions the best generic attack is a birthday attack).

‡Remark: The 256 in SHAKE256 describes the security level of the function (i.e. $c/2$) and unlike the common notation it does not describe the output length of the function or the size of its inner state. This notation is due to [NIS14] and used here for conformity.

Table 5.1.: List of recommended PRF* construction and primitives for the usage in Gen_k.

Table 5.1 presents a list of recommended PRF* constructions that might be used in Gen_k. The list is ordered, meaning that the entries are sorted descendingly by suitability and security as well as performance, all with respect to the usage in Gen_k. Keyed XOFs are preferred over HKDF constructions due to their simplicity and better performance. SHAKE is preferred over Skein, because the construction is standardized and hardware support of SHAKE/SHA3 is very likely in the near future. The recommended constructions from Table 5.1 are built from cryptographic hash functions with a security level of 256 bit against generic attacks, which is high enough in the foreseeable future. Note that a generic attack is an attack against a cryptographic primitive that can be performed independently of the concrete implementation of the primitive, for example an exhaustive key search or a birthday attack.

Even with respect to attacks with quantum computers, especially Grover's algorithm [Gro97], the hash functions security decreases by the order of a square root. The security of the candidates from Table 5.1 against Grover's algorithm and variants thereof is around 128 bit [Ber09], which is still sufficient.

5.2.2.3. Security analysis

Table 5.1 contains a summary of the security level of the primitives that are recommended for the usage in the matrix generation function. This section establishes the security of $Gen_k(\mathcal{N}_i)$ by analyzing the attack complexity of generic attacks against the PRFs and the PRF $\rightarrow$ PRF* constructions. This way, concrete bounds for the security of the matrix generation function are given that establish the security of the encryption scheme for equal energy signals according to Lemma 5.1.

PRF security

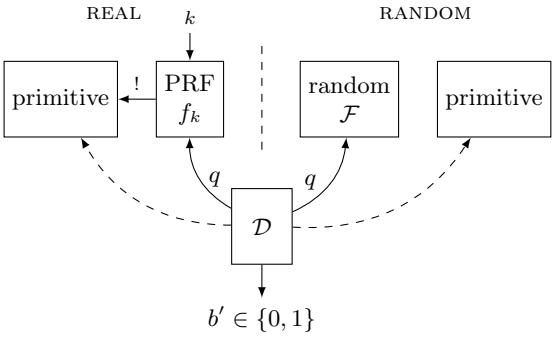


Figure 5.4.: Real or random distinguishing experiment - $\text{Exp}^{\text{ror}}_{\mathcal{D}}$ (with reference to [BDPA11b]).

Let $\mathcal{D}$ be an algorithm that tries to distinguish between the PRF f_k drawn from the pseudorandom function family F according the random key k and a random function $\mathcal{F}$. A high level view of the experiment, denoted by $\text{Exp}^{\text{ror}}_{\mathcal{D}}$,

is given in Fig. 5.4 (cf. sec. 3.2.1). Two kinds of computational costs should be considered for the PRF security analysis [BDPA11b].

- ▷ Online complexity: An attack is said to be *online* if the distinguisher is allowed to interact with the oracle O^{fun}, which is either REAL, i.e. f_k or RANDOM, i.e. $\mathcal{F}$ (cf. Fig. 5.4). Online complexity describes the costs of an online attack by considering the number of queries q to the oracle O^{fun}. It should be noted that the access to the pseudorandom function can be limited in practice.

- ▷ Offline complexity: In contrast to online attacks, *offline* attacks do not need access to O^{fun} and can therefore not be prevented or mitigated in practice. The offline complexity models queries to the underlying primitive of the oracle.

Clearly, the security of HMAC and the keyed versions of Skein and SHAKE is limited by the key size $|k|$, since an exhaustive key search can be performed with an offline complexity of $2^{|k|}$. For other kinds of generic attacks except exhaustive key search, it is clear that $\mathcal{D}$ succeeds if it recognizes the connection between the primitive and f_k when interacting with O^{fun}. This connection is marked with an exclamation mark in Fig. 5.4. The importance of this connection stems from the fact that the random function $\mathcal{F}$ does not depend on the primitive, but f_k does.

The PRF-security of HMAC was first established by [BCK96] and later confirmed under weaker assumptions on the underlying primitive by [Bel06]. Recent research results by [NW16] show that HMAC instantiated with SHA3 achieves a PRF-security of $\mathcal{O}(q^2 \cdot 2^{-L})$, where L is the fixed output length of the hash function (see Fig. 5.3). Stated differently, the PRF advantage of a distinguisher $\mathcal{D}$ in $\text{Exp}^{\text{ror}}_{\mathcal{D}}$ against HMAC-SHA3 is

$$\text{Adv}^{\text{ror}}_{\mathcal{D},F} \coloneqq |\Pr[\mathcal{D}_{f_k} \to 1] - \Pr[\mathcal{D}_{\mathcal{F}} \to 1]| \leq \frac{q}{2^{L/2}}. \tag{5.9}$$

The PRF-security of the keyed versions of KECCAK/SHAKE and Skein is similar [BDPA11b][BKL+09], but $L/2$ must be replaced by $d = \min(\ell/2, c/2)$. In comparison, HMAC-SHA2 achieves a PRF-security of $\mathcal{O}(\mathfrak{t} \cdot q^2 \cdot 2^{-L})$, where $\mathfrak{t}$ is the maximal length of the HMAC input [NW16].

Note that the stated bounds hold under the assumption that the primitive underlying the hash function has no structural weakness.

PRF $\rightarrow$ PRF* security

In addition to the PRF security of the recommended hash function, the PRF $\rightarrow$ PRF*construction has an impact on the security of Gen_k. Recap that q denotes the number of HMAC resp. the keyed hash function queries and let l be the number of signals encrypted under a single key. In the counter mode PRF $\rightarrow$ PRF* construction, q increases by $\lceil \frac{m \cdot N}{L} \rceil$ at each call of Gen_k. In contrast, for XOFs q increases just by one for each generated matrix and by using the PRF-security bounds it can be seen that $l \leq q \leq 2^d$ signals can be encrypted securely under a single key, which is indeed optimal with respect to the PRF security ($d = \min(\ell/2, c/2)$). For PRFs with the counter mode output extension the number of signals that can be encrypted under a single key decreases by $\lceil \frac{m \cdot N}{L} \rceil$ at each call of Gen_k. The security margin of the recommended functions is large enough such that this degeneration is only from theoretical interest and will hardly have an impact in practice. However, because of this undesired behavior it is recommended to use keyed XOFs rather than the counter mode output extension.

5.2.2.4. Performance analysis

This section presents a performance analysis of the standardized HMAC key derivation functions in counter mode (HKDF) [Che09] and keyed XOFs. The Skein hash function is used in both constructions, because of its good software performance [CPB+12]. Fig. 5.5 shows the average results from ten runs of a performance evaluation (output length vs. time), based on the code from the final round NIST submissions (v1.3). The code was compiled with the native Visual Studio 2015 compiler optimized for speed, i.e. with the flag /O2. The evaluation was performed on a single core of an Intel® Core™ i7-4770 with 3.40 GHz and 16 GB RAM. The results for XOF-Skein are almost identical for all state sizes, because of that, only XOF-Skein512 is plotted for better visibility. The number of PRF calls that are needed to generate an $m \times N$ matrix depend strongly on the output size of the hash function. Hence, hash functions with bigger output size perform best in the counter mode HKDFs, when the overall

number of output bits increases. However, Fig. 5.5 shows that XOFs outperform HKDF significantly. The reason for this is that XOFs work by iterating the inner round function until enough bits are produced. The repeated padding, splitting and finalization needed in HKDF comes at high computational costs compared to the simpler XOFs. This is another reason why XOFs are preferred over HKDF like schemes.

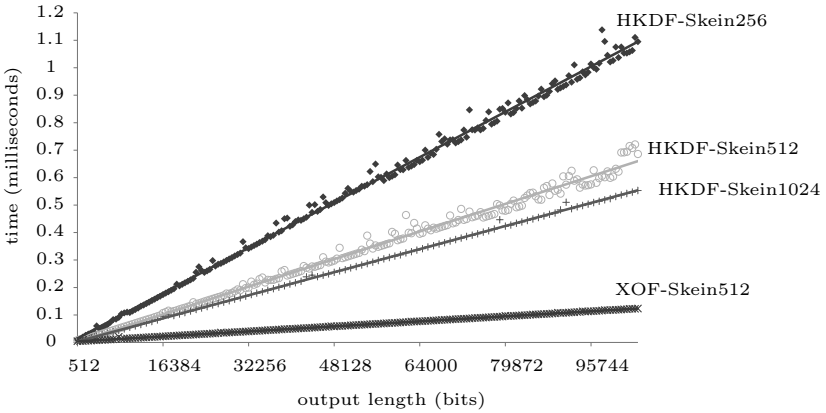


Figure 5.5.: Performance analysis of some PRF* constructions.

5.2.3. Normalized encryption

A Compressive Sensing based encryption scheme can only achieve ν-indistinguishability if the signals that should be encrypted have equal energy (cf. section 4.2/4.3). There are three different approaches for the normalization. A sparse or compressible signal $\vec{x}_i$ $(i = 0, ..., l-1)$ can be normalized by $\hat{x}_i = \vec{x}_i / \|\vec{x}_i\|_2$, if the signal is either known prior to sensing or if the normalization value is obtained using a separate sensor. Now the Compressive Sensing encryption function $\mathcal{E}_k$ can be defined as

$$\hat{y}_i = A_i \cdot \hat{x}_i, \tag{5.10}$$

where $A_i \leftarrow Gen_k(\mathcal{N}_i)$ for an n-bit nonce $\mathcal{N}_i$ (cf. section 5.2.2).

Another possible normalization method is to scale the entries of the sampling matrix by $1/\|\vec{x}_i\|_2$, thus the encryption function becomes

$$\hat{y}_i = \left(\frac{1}{\|\vec{x}_i\|_2} \cdot A_i \right) \cdot \vec{x}_i \,, \text{where } A_i \leftarrow Gen_k(\mathcal{N}_i) \,. \tag{5.11}$$

Unfortunately, scaling the matrix arbitrarily is not possible in practical applications [BDDH11, 3.3.1]. As in the first method, the signals norm must be known prior to sensing. This is a strong assumption for systems relying on the sampling model (M1), since the signals norm must be obtained separately.

The RIP ensures that the sampling process approximately preserves the signals energy (cf. Def. 3.4), leading to the information leakage described and analyzed in section 4.2. As a consequence, the third approach is to normalize the measurements $\vec{y}$. (M1)-systems have access to the measurements $\vec{y}$ and it is simple to compute their norm. Therefore, it is proposed that CS encryption modes work by normalizing the measurements after the sampling and compression process. This means that the encryption function obtains the compressed measurements with $\vec{y}_i = A_i \cdot \vec{x}_i$ (see sec. 4.1) and after that, the measurements are normalized using the measurements norm, that is $\hat{y}_i = \vec{y}_i / \|\vec{y}_i\|_2$.

The correct scaling factor needs to be known to the recipient, otherwise he/she will not be able to recover the original signal with sufficient quality. The secure encryption of $\|\vec{y}_i\|_2$ should be included in Compressed Sensing encryption modes. In addition, the encrypted norm can be used to provide interesting properties like self-synchronization (see section 5.4). Let π_k be an encryption scheme that achieves IND-CPA (cf. Def. 3.8) and further let π_k^{-1} denote the appropriate decryption algorithm. For now, one may think of π_k as a block cipher in an appropriate mode of operation like CTR or CBC. π_k is treated as a placeholder in the general design of CS encryption modes. Later it will be integrated in the dedicated modes of operation.

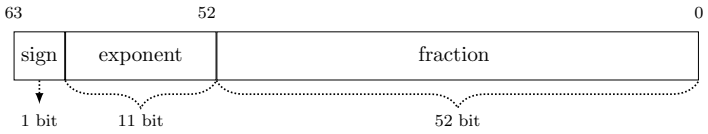


Figure 5.6.: Example of the IEEE 754 floating point representation with precision $p = 64$. (with reference to [IEE08]).

π_k processes binary strings, but $\|\vec{y}_i\|_2$ is a real number. The conversion from the reals to binary and vice versa is performed with the IEEE 754 floating point arithmetic [IEE08]. Let $\langle u \rangle_p$ denote the binary representation of a scalar $u \in \mathbb{R}$ as a p-bit floating point number with precision $p \in [64, 128]$. Fig. 5.6 presents an example of the IEEE 754 floating point representation with $p = 64$. Further let $\rangle b \langle$ be the floating point representation of a binary string b, also with precision p. The conversion functions can be found in [IEE08]. It must be noted that the limited floating point precision also affects the security, since independently from π_k an adversary might perform a dictionary attack in order to find $\langle \|\vec{y}_i\|_2 \rangle_p$. Therefore, it is recommended that p is at least 64 bit.

The encryption side in accordance with the general model is presented in Fig. 5.7. The ciphertext vector $\vec{\gamma}_i$ contains the compressively encrypted signal consisting of the compressed measurements $\hat{y}_i$ and the encrypted measurements norm denoted by $\rangle c_i \langle$, precisely:

$$\vec{\gamma}_i = (\hat{y}_i, \rangle c_i \langle)^T, \text{ where } c_i = \pi_k(\langle \|\vec{y}_i\|_2 \rangle_p) \text{ and } \hat{y}_i = \vec{y}_i / \|\vec{y}_i\|_2. \tag{5.12}$$

The set of all ciphertext vectors is called $\Gamma = (\vec{\gamma}_0, \vec{\gamma}_1, ..., \vec{\gamma}_{l-1})$.

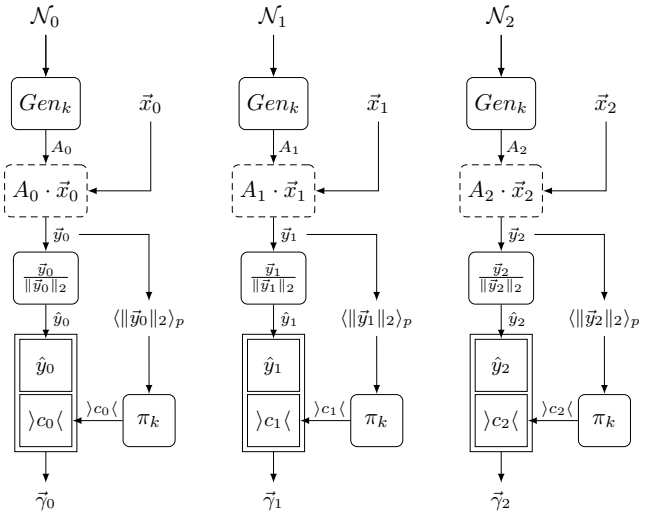


Figure 5.7.: Encryption side of the general model for CS confidentiality modes.

5.2.4. Decryption

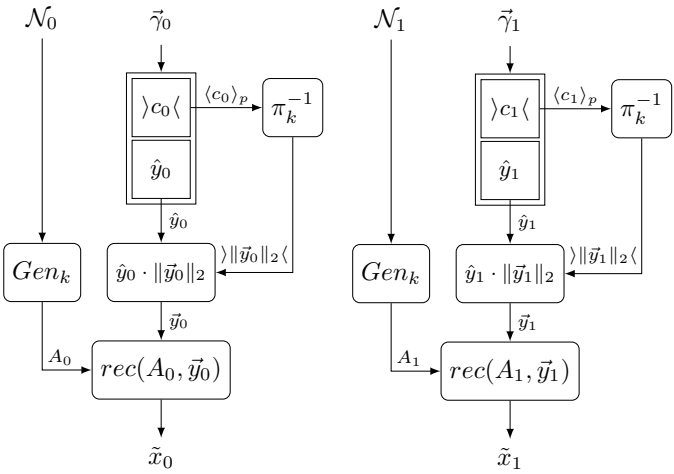


Figure 5.8.: Decryption side of the general model for CS confidentiality modes.

As usual, decryption is defined as the inverse of encryption. The decryption algorithm must use a suitable recovery algorithm *rec* to recover the original signal $\tilde{x}$. Unlike classical block cipher modes of operations, like CTR, it is not possible to rely only on the encryption functionality of the block cipher. A CS encryption mode is never inverse-free, which means that the mode has to use *rec* for decryption.

Legitimized recipients are meant to share the secret key k. The decryption starts by generating a pseudorandom sampling matrix via $A_i \leftarrow Gen_k(\mathcal{N}_i)$. It is crucial that encryption and decryption are synchronized, which means that the same nonce is used during matrix generation for corresponding plain- and ciphertexts. For now, it is assumed that the nonces are known to the recipient, but it is not predetermined how they are exchanged or derived. Each dedicated mode will define this separately. The scaling factor is decrypted by computing $\|\vec{y}_i\|_2 = \rangle \pi_k^{-1}(c_i) \langle$. The signal is reconstructed using the matrix A_i by calling $rec(A_i, \|\vec{y}_i\|_2 \cdot \hat{y}_i)$, which outputs $\tilde{x}_i = \hat{x}_i + \epsilon$ (cf. section 3.1.3). Fig. 5.8 presents the decryption side of the general model.

5.3. Compressive Sensing Counter Mode

5.3.1. Design and concept

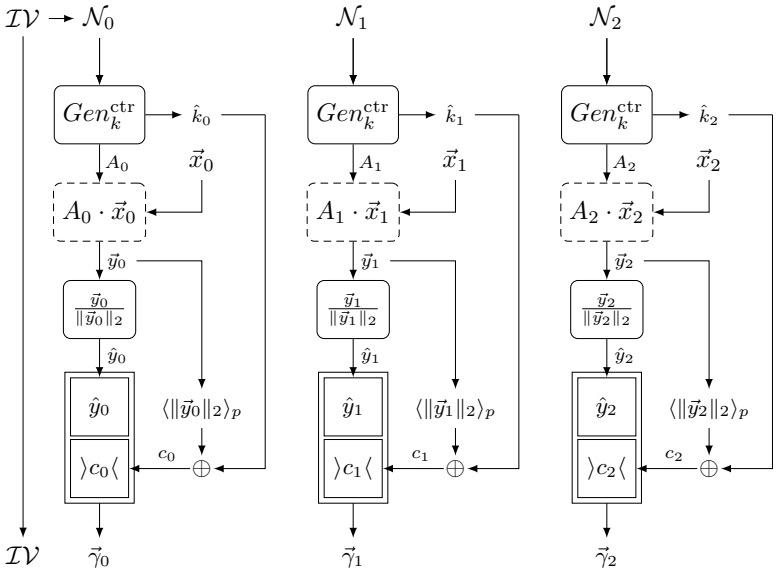


Figure 5.9.: Encryption with the Compressive Sensing Counter Mode.

The *Compressive Sensing Counter Mode* (CS-CTR) was proposed in [Fay16] using a fixed matrix generation function and without normalized encryption. The design was extended in [FR16]. Now, it can be used with various matrix generation functions and normalized encryption is integrated in the mode of operation without additional block ciphers. The mode of operation follows the design rationals from the general design presented in section 5.2. In addition, the main design goal behind CS-CTR is high performance, which means minimal overhead due to the normalized encryption and parallelizability of encryption and decryption. Moreover, the design allows precomputation of sensing matrices, which offers a time-memory trade-off. The sender side of CS-CTR is presented in Fig. 5.9. The notation used in this section is in conformance with section 5.2.

According to the general design, the mode of operation has to define how the nonces are derived and exchanged. The Compressive Sensing Counter Mode uses a counter as nonce, which gives the mode its name. The encryption algorithm starts by generating an n-bit initialization vector $\mathcal{IV}$, for example, one might draw the $\mathcal{IV}$ uniformly at random from the set of all n-bit strings. Let $\mathcal{N}_0 = \mathcal{IV}$ and derive the other nonces by counting onward, for example:

$$\mathcal{N}_i = (\mathcal{N}_0 + i) \bmod 2^n,\ i = 1, \ldots, l-1,\ l < 2^n. \tag{5.13}$$

In practice, the mode might also be used with another counting method then the one given in Eq. (5.13) as long as encryption and decryption are using the same method. However, it must be ensured that the $\mathcal{IV}$ and all intermediate counter values are nonces, which means that they must never be used more than once under one key.

Now, fix a pseudorandom function family F, a PRF $\rightarrow$ PRF* construction and draw a key k uniformly at random using an unpredictable source of randomness (see [ISO11]). The matrix generation algorithm in CS-CTR, denoted by Gen_k^{ctr}, produces $m \cdot N + p$ pseudorandom bits on each call using the underlying PRF*. Given a nonce $\mathcal{N}_i$ as input, $f_k^{m \cdot N + p}(\mathcal{N}_i)$ outputs the pseudorandom bits $b_0, b_1, \ldots, b_{mNp-1}$ and the leading $m \cdot N$ bits are reshaped to form the matrix $A_i \in \{+1, -1\}^{m \times N}$ as in Eq. (5.6). The remaining p bits form a pseudorandom keystream denoted by $\hat{k}_i \in \{0,1\}^p$, which is used to encrypt the measurements norm:

$$c_i = \langle \|\vec{y}_i\|_2 \rangle_p \oplus \hat{k}_i\ . \tag{5.14}$$

CS-CTR implements the encryption scheme π_k by means of a stream cipher that is integrated directly into the matrix generation algorithm. Observe that this stream cipher construction is nothing else than a pseudorandom function in CTR mode (cf. [ISO06]). The additional p bits output of $f_k^{m \cdot N + p}(\mathcal{N}_i)$ add only an insignificant computational overhead compared to the output of $m \cdot N$ bits that are anyway needed for the sampling matrix. In addition, $\hat{k}_i$ and A_i do not depend on the encryption of other signals, which means that they can be computed in parallel and also prior to encryption.

The plaintext signal $\vec{x}_i$ is encrypted by

$$\vec{y}_i = A_i \cdot \vec{x}_i \,, \text{where } A_i \leftarrow Gen_k^{\text{ctr}}(\mathcal{N}_i) \tag{5.15}$$

and the measurements are normalized

$$\hat{y}_i = \frac{\vec{y}_i}{\|\vec{y}_i\|_2} \,. \tag{5.16}$$

Finally, the ciphertext is formed by prepending the $\mathcal{IV}$ to Γ, such that $\Gamma = \{\mathcal{IV}, \vec{\gamma}_0, \vec{\gamma}_1, ..., \vec{\gamma}_{l-1}\}$ with $\vec{\gamma}_i = (\hat{y}_i, \rangle c_i \langle)^T$. The $\mathcal{IV}$ is used as the initial counter value and therefore it is needed for decryption when the ciphertext vectors are sent to another party, stored on a hard drive or in the cloud.

As usual, decryption is defined to be the inverse of encryption. Given the ciphertext Γ, which contains $\mathcal{IV} = \mathcal{N}_0$ as input, the decryption algorithm must derive the correct nonces for calling $Gen_k^{\text{ctr}}(\mathcal{N}_i)$ using the counter function (see Eq. (5.13)). The matrix generation algorithm outputs A_i and $\hat{k}_i$, which are used to decrypt $\vec{\gamma}_i$. The measurements norm is decrypted

$$\|\vec{y}_i\|_2 \ = \rangle c_i \oplus \hat{k}_i \langle \,, \tag{5.17}$$

and the signal is recovered by

$$\tilde{x}_i = rec(A_i, \ \|\vec{y}_i\|_2 \cdot \hat{y}_i). \tag{5.18}$$

5.3.2. Security analysis

An adversary might distinguish CS-CTR from an encryption scheme Π that achieves ν-indistinguishability by either breaking π_k or the matrix generation function. Therefore, the security of CS-CTR reduces to the indistinguishability of the output of $Gen_k^{\text{ctr}}(\mathcal{N}_i)$ from random. On the one hand, the random bits are used as a pseudorandom sensing matrix and on the other hand the random bit stream encrypts the measurements norm. Thus, π_k is integrated into $Gen_k^{\text{ctr}}(\mathcal{N}_i)$. The counter is used to call the inherent PRF* on distinct inputs, which ensures fresh output. This means that the output of $Gen_k^{\text{ctr}}(\mathcal{N}_i)$ is never used to encrypt more than one plaintexts. This mode of operation, i.e. the Counter mode, is standardized for the usage with n-bit block ciphers in [Dwo01][ISO06]. However, an n-bit block cipher is invertible and modeled as a

PRP, rather than a PRF. The PRP/PRF switching lemma [BR04, Lemma 1] implies that no efficient adversary can distinguish a PRP $\rho_k : \{0,1\}^n \to \{0,1\}^n$ from a PRF $f_k : \{0,1\}^n \to \{0,1\}^n$ with an advantage greater than $q \cdot 2^{-(n/2)}$, where q is the number of oracle queries. It is easy to see that the adversary can distinguish ρ_k from f_k by observing whether their occurs an output collision after hitting the birthday bound at around $2^{n/2}$ distinct queries or not. When the oracle is a pseudorandom function, the probability of a collision increases significantly after $2^{n/2}$ queries, while there are no output collisions with a PRP, since ρ_k is a bijection. This observation is then used to prove the switching lemma.

Most security bounds are first established for PRFs rather than PRPs, because this makes the analysis easier. The results are then lifted to the PRP setting by applying the switching lemma (see for example [BDJR97]). Interestingly, only the encryption functionality of the block cipher is used for encryption and decryption in CTR mode, which means that one does not need a PRP and might as well use a PRF directly. This has a significant impact on the security of CTR mode [BKR98]. According to [BDJR97, Theorem 13] and the corresponding proof, the IND-CPA advantage of an adversary $\mathcal{A}$ that asks at most polynomial many queries q in $\mathrm{Exp}^{\mathrm{cpa}}_{\mathcal{A},\mathrm{CTR}[F]}(b)$ is

$$\mathrm{Adv}^{\mathrm{cpa}}_{\mathcal{A},\mathrm{CTR}[F]} \leq 2 \cdot \mathrm{Adv}^{\mathrm{ror}}_{\mathcal{A},F}. \tag{5.19}$$

Remark. It should be noted that for CTR[P] the $q \cdot 2^{-n/2}$ security degeneration from the switching lemma must be added on the right side of Eq. (5.19).

The intuition behind the proof of the bound stated in Eq. (5.19) is that CTR mode would be ideal with a random function. This means that the advantage of any efficient adversary would be zero if the encryption is performed with a uniform random bit string, like in the case of the one-time-pad. The security loss from Eq. (5.19) arises when the random function is replaced with a pseudorandom function family F.

From the CTR[F] security bound (5.19), the PRF-security bounds from 5.2.2.3 and Lemma 5.1 it can be seen that the "loss" of security when CS-CTR is used instead of Π is negligible.

Restrictions on the $\mathcal{IV}$

Equation (5.13) contains a very important restriction on the number of encrypted signals, precisely $l < 2^n$, where $n = |\mathcal{IV}|$ in bit. An independent requirement next to the PRF security is that nonces never repeat under a single key and therefore l must be restricted in a way that guarantees that no counter value repeats. When the $\mathcal{IV}$ is drawn uniformly at random from the set of all n-bit strings, n must either be sufficiently large such that collisions in different $\mathcal{IV}$s and intermediate counter values have only a negligible probability, or this uniqueness must be checked separately. The same restrictions hold if the $\mathcal{IV}$ is generated by encrypting for example an incremented counter under a different key.

Unlike a secret key, the $\mathcal{IV}$ must not be stored or transmitted confidential and the fact that the $\mathcal{IV}$ is known to the adversary does not harm the IND-CPA-security of the CTR mode. In case of a more powerful *chosen ciphertext attack* (CCA) the $\mathcal{IV}$ as well as the rest of the ciphertext must be protected against tampering by an appropriate message authentication scheme (see section 3.2.4.1).

5.3.3. Properties

This section presents further properties of CS-CTR next to its security.

- ▷ **General**: Gen_k^{ctr} requires implementers to specify a PRF* and CS-CTR is secure under the assumption that f_k is a PRF. When the confidence in a certain function family fades due to stronger attacks, one can easily replace the PRF with another. The adaptable $\mathcal{IV}$ size n may increase the number of signals which can be safely encrypted under a single key, whereas the variable precision p affects the systems performance and reconstruction quality.

- ▷ **Error propagation**: An error during the transmission of $\vec{\gamma}_i$ has no effect on the output of $Gen_k^{ctr}(\mathcal{N}_{i+1})$. Hence, errors are not propagated into the next $\vec{\gamma}_{i+1}$. If the error occurs in $\hat{y}_i$, the Compressed Sensing offers some robustness, e.g. against noise, if an appropriate recovery algorithm is used. Similar to CTR, a bit error in c_i will result in an error of the corresponding plaintext bit.

- ▷ **Parallelizability**: Parallel sampling and reconstruction of multiple signals is a huge benefit in Compressive Sensing. CS-CTR processes each signal independently, such that encryption and decryption are fully parallelizable. In addition, this property also ensures *random access*, which means that a particular ciphertext vector can be decrypted without decrypting any other ciphertext. Of course, the corresponding nonce must be used for decryption.

- ▷ **Synchronization**: The critical variable with respect to synchronization is the implicit counter used to derive $\mathcal{N}_i$ $(i > 0)$ synchronously for encryption and decryption. Whenever the counters lose synchronization, e.g. through the loss of some $\vec{\gamma}_i$ during transmission or changes in the ciphertext order, the decryption algorithm will use a different nonce to generate A_i and the decryption will not succeed correctly. CS-CTR cannot restore synchronization by itself. To compensate for this, one may use sequence numbers of connection-oriented protocols to derive the nonces.

Connection-less protocols are sometimes preferred because they are faster than connection-oriented protocols. For example, large sensor networks and internet of things (IoT) applications often rely on UDP. In this case, the Compressive Sensing confidentiality modes presented in the next two sections are suitable, since they offer self-synchronization after a few ciphertexts are decrypted.

5.4. Compressive Sensing with Cipher Block Chaining

5.4.1. Design and concept

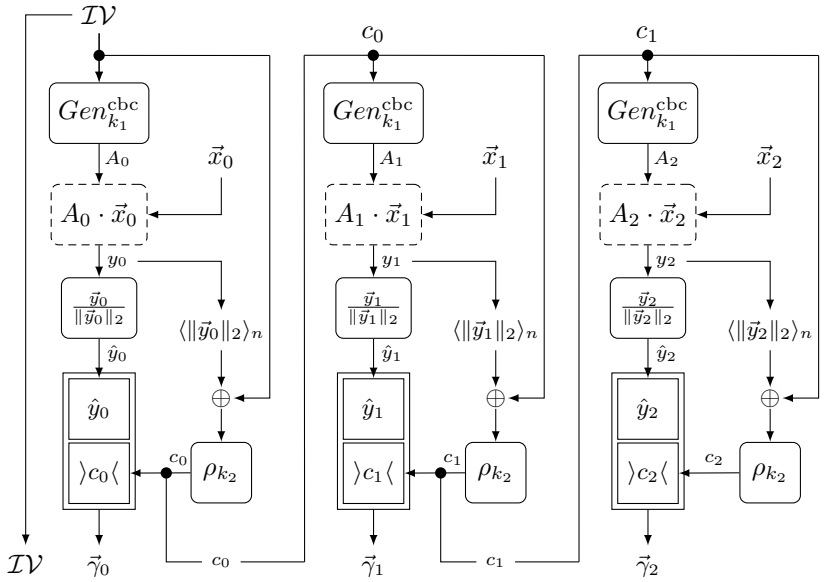


Figure 5.10.: Compressive Sensing with Cipher Block Chaining encryption.

The Compressive Sensing confidentiality mode presented in this section is designed to ensure self-synchronization in the case where ciphertext vectors got lost in transmission or when their order is changed. It is clear that this kind of self-synchronization can only be ensured when consecutive ciphertext vectors are somehow interdependent. Fig. 5.10 presents the encryption algorithm of the proposed mode of operation, which is called *Compressive Sensing with Cipher Block Chaining* (CS-CBC) (cf. [FR16]). The CS-CBC mode follows the design rationals from section 5.2, therefore it must be defined how the nonces used as input to the matrix generation function are derived and how the normalized encryption is performed.

The encryption algorithm starts by generating a random n-bit $\mathcal{IV}$ and setting $\mathcal{N}_0 = \mathcal{IV}$. Fix a pseudorandom function family F, a PRF $\rightarrow$ PRF* construction and draw a key $k = k_1 \,||\, k_2$ at random using an unpredictable source of randomness. In addition, choose a family of pseudorandom permutations P (i.e. a block cipher family) and define the encryption function as $\rho_{k_2} : \{0,1\}^n \rightarrow \{0,1\}^n$. Note that ρ_{k_2} works on n-bit blocks and its inverse function is $\rho_{k_2}^{-1}$. Further let $p \leq n$, such that $\langle \|\vec{y}_i\|_2 \rangle_n$ denotes the binary representation of $\|\vec{y}_i\|_2$ (see [IEE08]), which is optionally padded to n bits when $p < n$. The encryption scheme π_{k_2} is implemented by a PRP in CBC mode and the resulting ciphertext blocks are used as input to generate the next sensing matrix. By this means, the mode of operation makes use of the fact that the measurements norm should be transmitted confidential anyway. On top of this, the encrypted norm is accessible to the encryption and decryption algorithm and can therefore be used to achieve the intended self-synchronization.

Formally, the encryption of the measurements norm is defined as

$$c_i = \rho_{k_2}(\mathcal{N}_i \oplus \langle \|\vec{y}_i\|_2 \rangle_n), \text{where } \begin{cases} \mathcal{N}_0 = \mathcal{IV} \\ \mathcal{N}_i = c_{i-1} \quad (i > 0) \end{cases} \tag{5.20}$$

The matrix generation function $Gen_{k_1}^{\text{cbc}}$ works by calling $f_{k_1}^{m \cdot N}(\mathcal{N}_i)$ to obtain $m \cdot N$ pseudorandom bits $b_0, b_1, \ldots, b_{mN-1}$, which are reshaped into A_i as in Eq. (5.6). The signals are jointly sampled, encrypted, and compressed by

$$\vec{y}_i = A_i \cdot \vec{x}_i \,, \text{where } A_i \leftarrow Gen_{k_1}^{\text{cbc}}(\mathcal{N}_i) \tag{5.21}$$

and the measurements are normalized

$$\hat{y}_i = \frac{\vec{y}_i}{\|\vec{y}_i\|_2} \,. \tag{5.22}$$

After l signals are encrypted, the ciphertext is formed by prepending the $\mathcal{IV}$ to Γ. Here, l must be restricted to $l \ll 2^{n/2}$ for security reasons, which are explained in section 5.4.2.

The decryption algorithm, which is given Γ obtains $\|\vec{y}_i\|_2$ by

$$\|\vec{y}_i\|_2 = \rangle \mathcal{N}_i \oplus \rho_{k_2}^{-1}(c_i) \langle, \text{ where } \begin{cases} \mathcal{N}_0 = \mathcal{IV} \\ \mathcal{N}_i = c_{i-1} \quad (i > 0) \end{cases} \tag{5.23}$$

The compressed measurements are reconstructed and rescaled to form the plaintext signal by $A_i \leftarrow Gen_{k_1}^{cbc}(\mathcal{N}_i)$ and

$$\tilde{x}_i = rec(A_i, \|\vec{y}_i\|_2 \cdot \hat{y}_i). \tag{5.24}$$

5.4.2. Security analysis

Distinguishing CS-CBC from a CS encryption scheme Π that achieves ν-indistinguishability might be done by either breaking π_{k_2} or $Gen_{k_1}^{\text{cbc}}$. CS-CBC is designed with two different keys for the two functions. The reason for this decision is that both functions are sharing the same $\mathcal{IV}$ and when they are built from the same underlying primitive, using the same key and $\mathcal{IV}$ might result in undesirable and insecure behavior. Hence, it was decided to follow the "one key for one application" paradigm, like when combining a IND-CPA secure encryption scheme with a MAC.

The security analysis of the matrix generation algorithm was already presented in section 5.2.2.3 and is not repeated here. Lemma 5.1 bridges the gap between the pseudorandom matrix generation and random matrices. What plays an important role in the security analysis of CS-CBC is the impact of the CBC encryption, since the chained ciphertext blocks are treated as nonces. The ciphertext blocks must be distinct to ensure that $Gen_{k_1}^{\text{cbc}}(c_{i-1})$ produces fresh pseudorandom matrices for each signal. Interestingly, the IND-CPA security of the CBC block cipher mode holds up to the point where a collision in the ciphertexts blocks gets likely. To see this, let $\mathcal{A}$ be an efficient chosen plaintext adversary attacking CBC[P] with a random $\mathcal{IV}$ that asks at most q queries. Now consider the following bound on the security of CBC from [BDJR97, Theorem 17]

$$\text{Adv}_{\mathcal{A},\text{CBC}[P]}^{\text{cpa}} \leq 2 \cdot \text{Adv}_{\mathcal{A},P}^{\text{ror}} + \frac{q}{2^{n/2}}, \tag{5.25}$$

where P is a PRP-family with members $\rho_k : \{0,1\}^n \to \{0,1\}^n$. The proof of this bound is given in [BDJR97].

Since the output of the PRP cannot be distinguished from random, one would expect repetitions of c_i's roughly at the birthday bound, i.e. after $2^{n/2}$ blocks. This is exactly the security degeneration that is stated in Eq. (5.25). An adversary that achieves the stated advantage might for example ask for the encryption

of a long message where all plaintext blocks are the same n-bit string. When the adversary observes an output collision of the form $c_i = c_j$ where $i \neq j$ he/she is able to confirm that the encryption oracle is CBC by verifying if $c_{i+1} = c_{j+1}$ (cf. [Rog11, 4.5]). This attack shows that the bound stated in Eq. (5.25) is tight, which means that CBC mode cannot achieve a higher level of security.

Additionally, the security bound from Eq. (5.25) has the consequence that after the encryption of $2^{n/2}$ signals in CS-CBC mode the key must be changed even if the underlying functions have a greater security margin. If $l \geq 2^{n/2}$ one can either expect a collision in some c_i, which violates the assumption that c_i is a nonce, or an adversary is able to break CBC encryption. In contrast to CS-CTR, it can be seen that the CBC mode introduces very strict restrictions, since n is not variable and determined by the choice of the block cipher.

Restrictions on the $\mathcal{IV}$

The IND-CPA security guarantees of CBC stated in Eq. (5.25) hold only for a random $\mathcal{IV}$. Using a nonce like an incremented counter as $\mathcal{IV}$ is not sufficient to ensure IND-CPA security with CBC. The same applies to an encrypted nonce under the same key, but a nonce encrypted under a different key is sufficient to ensure security (cf. [Rog04]). Moreover, using the last ciphertext block as $\mathcal{IV}$ for the next run of the encryption algorithm is also not secure. Ignoring this requirements has led to various practical attacks on protocols that are using CBC mode like SSH and SSL [BKN02][Bar06]. Therefore, it is recommended to use an unpredictable $\mathcal{IV}$, like a nonce encrypted under a different key, if it is not possible to generate the $\mathcal{IV}$ at random. Similar to every other $\mathcal{IV}$ based encryption mode from ISO 10116 the $\mathcal{IV}$ must not be kept confidential to ensure IND-CPA.

5.4.3. Properties

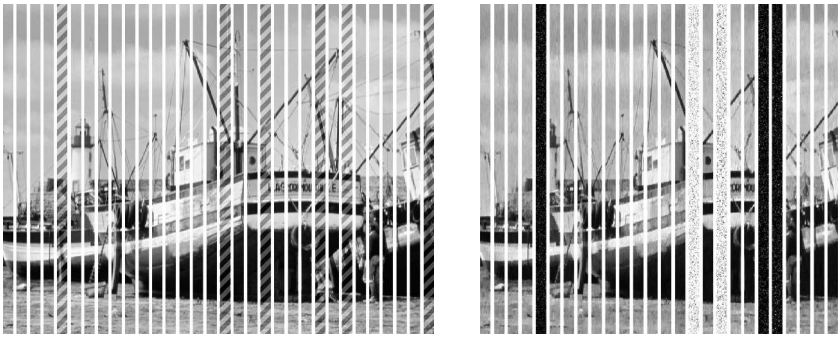
- ▷ **General**: The CS-CBC design does not restrict implementers to use a specific primitive or construction. However, CS-CBC is not as flexible as CS-CTR, mainly due to the PRP, which limits the number of encrypted signals under a single key by $l \ll 2^{n/2}$. In addition, if $p < n$ the plaintext blocks must be padded to a multiple of n bits, which will introduce further overhead.

- ▷ **Error propagation**: An error in just a single bit of c_i leads to a wrong scaling of the corresponding decrypted signal. The following ciphertext vector is not decrypted correctly because of this error, but the error does not propagate further. Compared to that, the error through noisy measurements $\hat{y}_i$ might be mitigated by the inherent robustness of CS against noise.

- ▷ **Parallelizability**: Encryption can not be performed in parallel due to the cipher block chaining, whereas decryption is parallelizable, if Γ is received prior to decryption. This is a benefit in Compressed Sensing, since the recovery algorithm is more expensive then the sensing process.

- ▷ **Synchronization**: The main goal of CS-CBC is to ensure self-synchronization after the loss or interchange of some $\vec{\gamma}_i$, while offering confidentiality in the many-time encryption scenario. This property is a great advantage for example in sensor networks used in internet of things applications, where connection less protocols are preferred because of their performance. Self-synchronization is introduced by the cipher block chaining, which also provides the nonces used for matrix generation. The ciphertext vectors are interdependent meaning that the decryption of $\vec{\gamma}_i$ depends on $\vec{\gamma}_{i-1}$. For example, assume that $i \geq 2$ and let $c_0 = \mathcal{IV}$. The loss of $\vec{\gamma}_{i-1}$ has the consequence that $\hat{y}_i$ is reconstructed with the wrong matrix $A_{i-1} \leftarrow Gen_{k_1}^{cbc}(c_{i-2})$ instead of $A_i \leftarrow Gen_{k_1}^{cbc}(c_{i-1})$. The resulting plaintext signal is just a random sparse signal, since the Compressive Sensing reconstruction algorithm is not able to reconstruct the original signal by using the wrong matrix. The corresponding ciphertext is also not decrypted properly by the PRP. However, synchronization can be recovered directly, such that the following vectors are decrypted correctly if there were no further transmission errors. A similar behavior is shown in the case where the order of ciphertext vectors is changed.

Fig. 5.11 demonstrates an example of the CS-CBC self-synchronization in the case where 20 % of the ciphertext vectors got lost during transmission. It can be seen that a ciphertext vector who follows directly after a ciphertext vector that was lost during transmission cannot be recovered correctly. The next vector can be reconstructed as long as its predecessor was received in order.



(a) (b)

Figure 5.11.: CS-CBC self-synchronization in case of ciphertext losses.
(a) shows the original "boats" image consisting of 32 signals, the signals that are lost during transmission are marked with stripes.
(b) is the recovered image decrypted from the 26 received ciphertext vectors using TVAL3 [Li09].

5.5. Compressive Sensing with Cipher Feedback

5.5.1. Design and concept

The price that is payed to achieve the self-synchronization property of CS-CBC is the additional block cipher that is needed for the cipher block chaining resp. feedback to the matrix generation algorithm. An additional key should be used for this block cipher and if $p < n$ the plaintext must be padded. The block size n of the block cipher restricts the number of signals that can be encrypted securely under a single key in such a way that the security margin of CS-CBC is not as good as it is with CS-CTR. On top of this, n is no longer a variable that can be adapted freely in accordance to security and application requirements.

Compressive Sensing with Cipher Feedback (CS-CFB) achieves self-synchronization and limited error propagation in a more flexible fashion than CS-CBC, since the feedback comes from a stream cipher that is integrated in the matrix generation algorithm, like in CS-CTR. The main advantage over CS-CBC is that no additional block cipher must be used and implementers are not restricted to a fixed n. As before, the design follows the general model from section 5.2.1. The encryption side of CS-CFB is shown in Fig. 5.12.

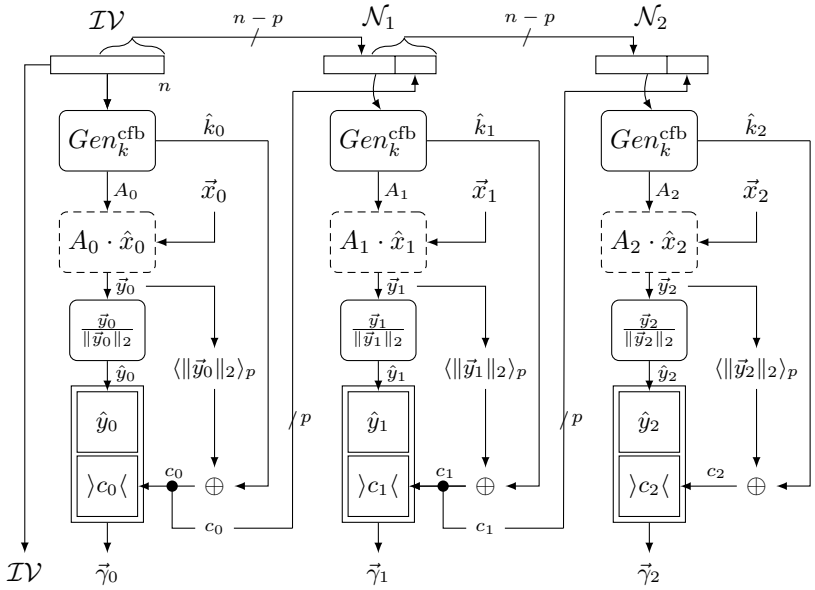


Figure 5.12.: Encryption side of the Compressive Sensing with Cipher Feedback mode.

Let $\mathcal{IV} = \mathcal{N}_0$ be a bit string of length n that is drawn uniformly at random from the set of all n-bit strings. Given a random key k and a nonce as input, $Gen_k^{\text{cfb}}(\mathcal{N}_i)$ outputs a pseudorandom matrix $A_i \in \{+1, -1\}^{m \times N}$ and a p-bit keystream $\hat{k}_i$. Recap that p is the number of bits used to represent $\langle\|\vec{y}_i\|_2\rangle_p$ and let $n = \lambda \cdot p$, for $\lambda \in \mathbb{N}^+$. Gen_k^{cfb} is essentially the same as Gen_k^{ctr} and the reader is referred to section 5.3.1 for a concrete description of this function. CS-CFB uses the ciphertexts obtained by encrypting the measurements norm as feedback in order to derive the nonces that are used for matrix generation. Let $\vec{x}_i$ be a sparse or compressible signal that is encrypted by

$$\vec{y}_i = A_i \cdot \vec{x}_i \,, \text{where } A_i \leftarrow Gen_k^{\text{cfb}}(\mathcal{N}_i). \tag{5.26}$$

The measurements are normalized as described in Eq. (5.27).

$$\hat{y}_i = \frac{\vec{y}_i}{\|\vec{y}_i\|_2} \,. \tag{5.27}$$

The additional keystream outputted by Gen_k^{cfb} is used to encrypt the measurements norm,

$$c_i = \langle \|\vec{y}_i\|_2 \rangle_p \oplus \hat{k}_i \,. \tag{5.28}$$

The ciphertext is formed by prepending the $\mathcal{IV}$ to Γ. π_k is implemented as a pseudorandom function in cipher-feedback mode where the plaintext length is limited to p.

Let $\text{lsbit}^t(u)$ denote the t least significant bits of the bit string u and further let

$$\mathcal{N}_{i+1} = \text{lsbit}^{n-p}(\mathcal{N}_i) \,||\, c_i \,. \tag{5.29}$$

The feedback from Eq. (5.29) might be implemented using an n-bit shift register as feedback buffer. The register is initialized with $\mathcal{IV}$ and shifted by p-bit after each signal (see Fig. 5.12).

The decryption algorithm extracts the $\mathcal{IV}$ from Γ, initializes the shift register to decrypt $\vec{\gamma}_0$ and updates it as in Eq. (5.29) for each following ciphertext vector. Decryption is performed by

$$\tilde{x}_i = rec\left(A_i, \langle c_i \oplus \hat{k}_i \langle \cdot \, \hat{y}_i\right) . \tag{5.30}$$

5.5.2. Security analysis

Similar to CS-CTR, the security of CS-CFB is reduced to the indistinguishability of the output of Gen_k^{cfb} from random in order to apply Lemma 5.1 and establish the CPA security of π_k. The CFB mode of operation is standardized for the usage with an n-bit block cipher in [Dwo01] and [ISO06]. Due to the stream cipher encryption, the mode of operation only needs the encryption routine of the block cipher. This means that no invertible primitive is necessary for CFB and the block cipher can be replaced with a non-invertible PRF.

However, the important factor with respect to the security of CS-CFB is the cipher feedback, which is used to derive nonces for Gen_k^{cfb}. Clearly, an adversary is able to distinguish the output of Gen_k^{cfb} from random if the input to the matrix generation function is no longer unique. In contrast to CS-CTR,

the nonces are not independent of the underlying PRF* and collisions in the feedback buffer become likely after roughly $2^{n/2}$ signals are encrypted. This can also be seen from the following bound that states the IND-CPA security of CFB for a random function family F against an adversary $\mathcal{A}$ that asks at most q CPA-queries:

$$\mathrm{Adv}^{\mathrm{cpa}}_{\mathcal{A},\mathrm{CFB}[F]} \leq 2 \cdot \mathrm{Adv}^{\mathrm{ror}}_{\mathcal{A},F} + \frac{q}{2^{n/2}} \tag{5.31}$$

This bound was first established and proven in [AGPS02] by using the typical transition based on the indistinguishability of the pseudorandom function from a random function. It should be noted that the bound from Eq. (5.31) is tight, since an attack comparable to the one discussed for CBC mode in section 5.4.2 can be applied to CFB. All randomized modes with feedback, i.e. CBC, CFB and OFB, show the $q^2 \cdot 2^{-n}$ security degeneration. Therefore, the key must be changed when l approaches $2^{n/2}$, even if the underlying PRF has a greater security margin. In contrast to CS-CBC, n is no longer determined by the block cipher.

Restrictions on the $\mathcal{IV}$

The security of CS-CFB can only be guaranteed if either a random $\mathcal{IV}$ or a nonce encrypted under a different key is used as $\mathcal{IV}$.

5.5.3. Properties

- ▷ **General**: The CS-CFB design does not need an invertible primitive to achieve synchronization when ciphertext vectors got lost. As a consequence, the mode of operation works with a single key, since π_k is integrated into Gen^{cfb}_k. The mode is also flexible, because n might be adapted, for example, to increase the number of signals that can be securely encrypted under one key. On top of this, no padding is required to encrypt $\|\vec{y}_i\|_2$. In contrast to the CFB block cipher mode of operation (cf. [ISO06]), CS-CFB uses all the random bits that are produced by the underlying PRF, which means that not a single pseudorandom bit is wasted.

- ▷ **Error propagation**: The relationship between n and p plays an important role with respect to error-propagation caused by corrupted $c_i's$. As

long as $n = p$, a bit error in c_i propagates into the decryption of the next ciphertext vector. When c_{i+1} is received correctly, the following ciphertext vector is decrypted correctly. In case that $p < n$, errors in c_i affect the decryption of subsequent ciphertext vectors as long as the corrupted bits are inside the feedback buffer. Implementers should consider this behavior and choose n, p in accordance with their security goals and application. Errors or noise in $\hat{y}_i$ does not affect other ciphertext vectors and might be mitigated by the inherent robustness of Compressive Sensing.

- ▷ **Parallelizability**: The CS-CFB mode is not easily parallelizable, but pipelining can be used to enhance performance. The state of the feedback buffer might be precomputed for a parallel decryption when the mode is implemented in software and all ciphertext vectors are received prior to decryption.

- ▷ **Synchronization**: After the loss of some $\vec{\gamma}_i$, synchronization is recovered with the next correctly received ciphertext vector for $n = p$. In the more general case where $p < n$, synchronization is not achieved directly. Sender and recipient are working synchronously, when their feedback buffers have the same state. Fig. 5.13 shows an example of the synchronization property of CS-CFB for $n = p$ and $n = 2 \cdot p$. In Fig. 5.13, 20 % of the 32 signals got lost during transmission. It can be seen that synchronization is recovered faster when $n = p$, since the state of the feedback buffer is updated completely with each received ciphertext vector. A similar behavior is shown when the order of the ciphertext vectors is changed.

(a)

(b)

(c)

Figure 5.13.: Synchronization with CS-CFB for different parameters.
(a) shows the original image that consists of 32 signals, the signals that are lost during transmission are marked with stripes.
(b) is the recovered image with $n = p$ using TVAL3 [Li09].
(c) is the recovered image with $n = 2 \cdot p$ using TVAL3 [Li09].

5.6. Comparison of the Proposed Modes

All proposed confidentiality modes follow the design rationals from section 5.2 and achieve the design goals stated in section 5.2.1. The presented modes rely on Bernoulli sensing matrices and therefore asymptotic ν-indistinguishability in the many-time encryption scenario is the best achievable security notation. Table 5.2 summarizes further properties and security restrictions of the proposed Compressive Sensing confidentiality modes. As before, let $d = \min(\ell/2, c/2)$ denote the security parameter of the PRF, let l be the number of signals that should be encrypted and further let n be the size of $\mathcal{IV}$ in bit.

For a fixed PRF with d bits of security, it is clear that the PRF-security fades with $\mathcal{O}(q \cdot 2^d)$ and that an exhaustive key search has an offline complexity of $2^{|k|}$ (see section 5.2.2.3 for further details). Table 5.2 states the number of signals that can be securely encrypted under a single key when $n < d$, since n might be chosen freely in CS-CTR and CS-CFB. In this case, the security of the mode of operation against online attacks is limited by the choice of n rather than the PRF security as an implementer might believe. Note that increasing n such that $n > d$ does not increase security beyond the PRF-security. In contrast, the online security of CS-CBC is limited by the underlying n-bit block cipher.

The ubiquitous confusions about the $\mathcal{IV}$ for an n-bit block cipher mode have led to various practical attacks, see for instance [BKN02, Bar06] or the BEAST attack on *Transport Layer Security* (TLS) 1.0. Table 5.2 contains the restrictions on the $\mathcal{IV}$ under which the proposed Compressive Sensing confidentiality modes work as designed and the reader is referred to section 5.3.2, 5.4.2 and 5.5.2 for more details. CS-CTR is the only proposed mode that is secure when the $\mathcal{IV}$ is a nonce rather than a random n-bit string. In some application scenarios it might be easier to provide a nonce instead of a random string. Unfortunately, it seems that misuse resistant modes like *Synthetic Initialization Vector* (SIV) [RS07] cannot be adapted for the usage in Compressive Sensing. A misuse resistant mode achieves a decent and formally defined security goal even if it is used incorrectly. To attain misuse resistance, SIV uses a CBC-MAC of the plaintext as $\mathcal{IV}$, but in the CS sampling model (M1) from Def. 3.3 it is impossible to perform computations on the signal.

	CS-CTR	*CS-CBC*	*CS-CFB*
Maximal number of signals until the key must change ($n < d$)	$\leq 2^n - 1$	$\ll 2^{n/2}$	$\ll 2^{n/2}$
Restrictions on the $\mathcal{IV}$	nonce (or random)	random	random
Needs a block cipher for encryption	`No`	`Yes`	`No`
Number of keys	`1`	`2`	`1`
Parallel processing ($\mathcal{E}_k/\mathcal{D}_k$)	(`Yes`/`Yes`)	(`No`/`Yes`)	(`No`/`Possible`)
Precomputation of A_i *possible*	`Yes`	`No`	`No`
Self-synchronizing (w.r.t. losses or reordering)	`No`	`Yes`	`Yes`, impact depends on n, p
Error propagation (w.r.t. c_i)	`No`, an error in c_i affects the scaling of $\hat{y}_i$	`Yes`, an error in c_i affects the scaling of $\hat{y}_i$ and $\tilde{x}_{i+1}$ cannot be recovered	`Yes`, impact depends on n, p
Robustness against noisy measurements	`Yes`	`Yes`	`Yes`
Padding required (if $p < n$)	`No`	`Yes`	`No`
Free parameters (independent of any primitive)	n, p	p	n, p

Table 5.2.: Comparison of the proposed confidentiality only modes.

The other properties of the proposed modes offer the possibility to choose a mode of operation in accordance to its use case. The main advantages of CS-CTR is its full parallelizability, simplicity and the limited impact of errors. Each signal is treated independently and therefore, the mode cannot recover from losses of ciphertext vectors or changes in their order. To compensate for this, external values like sequence numbers of connection-oriented protocols might be used to derive the correct counter value.

CS-CBC and CS-CFB will restore synchronization after a few ciphertext vectors in case of losses or reordering. CS-CFB is more flexible than CS-CBC mainly due to the stream cipher design used for the encryption of the measurements norm. Both modes do not support parallel encryption, because of the feedback resp. cipher block chaining. The Compressive Sensing reconstruction algorithm is more expensive then the sampling process, hence parallel decryption is a desirable feature. Performing parallel decryption is simple in CS-CBC as long as the required portions of the ciphertext are accessible. In case of CS-CFB, the parallel decryption property depends on the implementation of the feedback buffer.

5.7. Experimental Results

5.7.1. Proof of concept

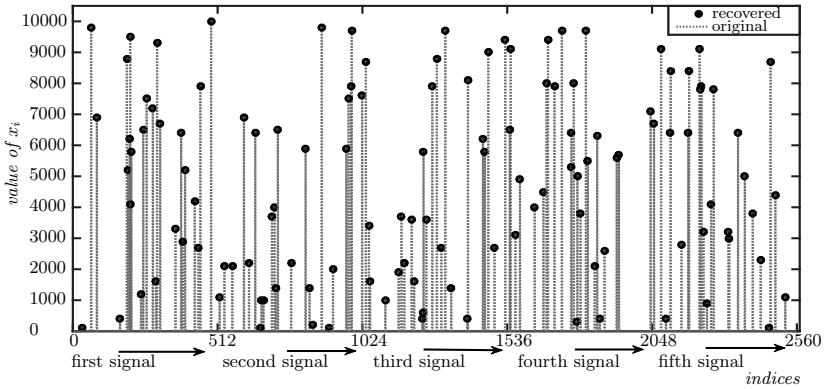


Figure 5.14.: Perfect recovery of five random sparse signals with CS-CTR using basis pursuit.

Random sparse signals

A design goal for the proposed confidentiality modes is reconstructability. This means that the modes of operation should not harm the necessary conditions for a successful signal recovery, this is the reason why the sampling matrices used in CS encryption modes should achieve the RIP. However, it is important to investigate how the encryption modes work with respect to different signals. At first, all modes are tested on s-sparse signals with randomly drawn entries, since the theoretical results from section 3.1 can be used to establish sufficient parameters. Fig. 5.14 presents the result from a simulation performed with CS-CTR using SHAKE128 as underlying PRF*. Five random signals, each with $N = 512$ entries and a sparsity of $s = 25$, are sampled with $m = 120$. As it can be seen from Fig. 5.14, the signals are lossless recovered with basis pursuit [CDS01]. The basis pursuit simply performs the ℓ_1-optimization as it is stated in Eq. (3.5)/(3.6). The results from simulations with more signals and the other proposed modes of operation are similar, which shows that Compressed Sensing

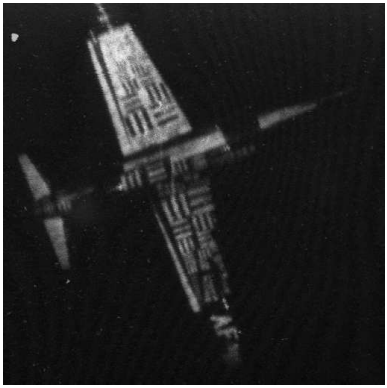
encryption modes are applicable for a wide range of use cases where the signals of interest are actually sparse.

Imaging application

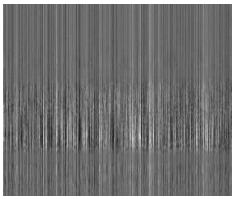
Imaging is another popular application for Compressed Sensing, since most natural images are sparse in a wavelet domain for instance. The examples that were presented in section 5.4.3 and 5.5.3 already showed that the proposed modes can be applied in Compressed Sensing imaging. The imaging application has another advantage when it comes to simulations: the encrypted image visualizes the view of an eavesdropper. For most of the presented simulations, the images are sampled column wise or with signals that consist of multiple columns. This was mainly done for performance reasons and in a practical application one would probably sample the hole image at once.

Fig. 5.15 presents the encryption of the 512×512 "plane" test image with CS-CTR based on SHAKE128 and CS with a fixed binary matrix $A \in \{-1, 1\}^{m \times N}$. All the adversary sees in case of CS-CTR is just a bunch of random measurements without any visual structure. In contrast, the image encrypted with a fixed matrix shows some structure, which can be regarded as information leakage due to the fact that similar signals are encrypted to similar measurements. With a measurement dimension of $m = 205$ and TVAL3 recovery [Li09] the PSNR between the original image and the recovered one is around 33 dB, which is still an acceptable PSNR value for lossy image compression. The recovery error is a result from the sparse representation of the image and it is also influenced by the recovery algorithm. For more suitable images, like the "phantom" test image presented in the top left corner of Fig. 5.16, the PSNR values are much better.

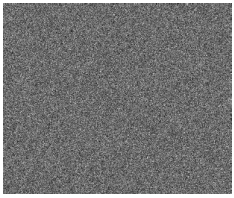
A natural question to ask is if the proposed confidentiality modes introduce further recovery errors compared to the classical Compressive Sensing with a fixed Bernoulli matrix. Fig. 5.16 presents the results from 100 runs performed on the "phantom" test image with CS-CTR, CS-CBC and CS with a fixed matrix per run. It can be seen that the proposed modes do not introduce more recovery errors than the classical CS. Further experiments with other signals and parameters support this statement. The results from the "phantom" image are presented here, because it shows a good PSNR in general and if the modes of operation would introduce further errors these would certainly be visible.



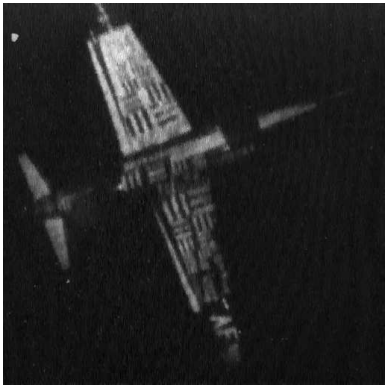
(original)



(encrypted w/ a fixed A)



(encrypted w/ CS-CTR)



(decrypted w/ CS-CTR
& TVAL3 [Li09])

Figure 5.15.: Encryption examples for an imaging application ($m = 307, N = 1024, l = 256$).

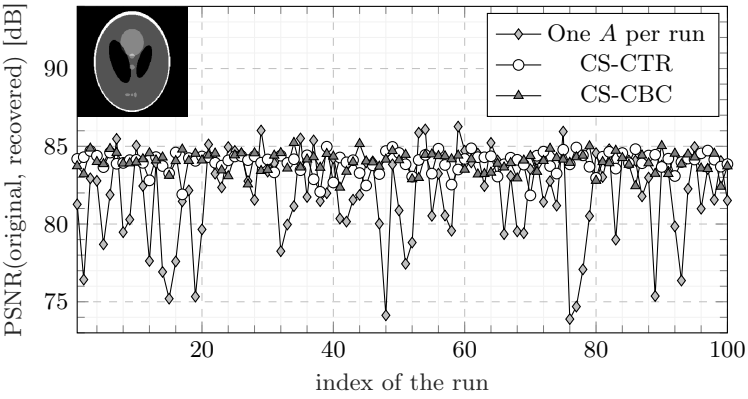


Figure 5.16.: Comparison of the recovery PSNR for the "phantom" test image.

The results presented in Table 5.3 show that the recovery quality of the "plane", "baboon" and "boats" test images are not sufficient if the recipient cannot apply the correct scaling. This means that in case of general images the measurements norm must be transmitted to the recipient and this transmission cannot be omitted for performance reasons.

Image (size)	PSNR(orig., rec.) $\hat{y}$ scaled with the correct norm	PSNR(orig., rec.) $\hat{y}$ normalized , i.e. $\|\hat{y}\|_2 = 1$
"phantom" (256×256)	83.0826 dB	43.6868 dB
"plane" (256×256)	33.0826 dB	15.8967 dB
"baboon" (256×256)	20.5456 dB	4.7898 dB
"boats" (512×512)	30.3159 dB	5.6377 dB

Table 5.3.: The TVAL3 [Li09] recovery PSNR(original, recovery) for correctly scaled measurements and measurements without proper scaling.

5.7.2. Measurement distribution

In order to achieve asymptotic ν-indistinguishability, the ciphertext vectors should be indistinguishable from random vectors drawn from a normal distribution (see section 4.2). The parameters N, m and s will have a significant impact on the measurements distribution, especially when Bernoulli random matrices are used. The measurements obtained by running one of the proposed modes with their normalized encryption tend to the standard normal distribution in accordance to the central limit theorem. Consider Fig. 5.17 for an example of the measurements obtained by the encryption of the "plane" test image with CS-CTR, again sampled column wise. The normal probability plot shows that the distribution of the measurements is very close to a normal distribution. Statistical tests like the Cramér-von-Mieses test or the Anderson-Darling test [And62] cannot distinguish the measurements by their distribution. The results from the statistical tests are as presumed very similar to the results presented in section 4.2.2 for Bernoulli random matrices and therefore not repeated here.

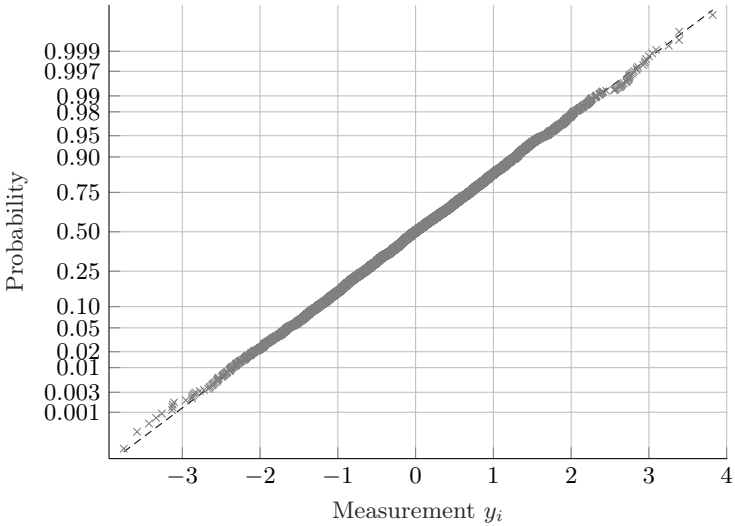


Figure 5.17.: Normal probability plot of the encrypted 512×512 "plane" test image with CS-CTR.

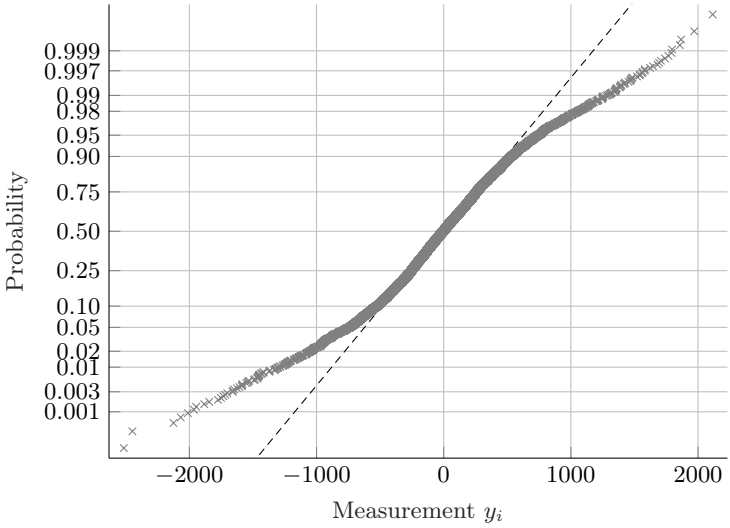


Figure 5.18.: Normal probability plot of the encrypted 512×512 "plane" test image with CS-CTR but without normalization.

In contrast, when the measurements are not normalized during the encryption process, the distribution of the measurements can be distinguished by a statistical test. Fig. 5.18 shows the probability plot of the "plane" image with CS-CTR but without normalization. It can be seen that the results are not aligned with the dashed line that represents a normal distribution.

Special care must be taken in cases where the signals are either too sparse or the sampling rate is too small. Following Eq. (3.4) the sampling rate depends on the signals sparsity and if the sparsity is known a-priory one might choose a very small number of measurements, which harms the measurements indistinguishability. In order to see the impact of s and m on the measurements distribution, consider Fig. 5.19, which shows the histograms of two random sparse signals encrypted with CS-CTR. When the number of measurements is too small, e.g. with a compression ratio (m/N) of nearly 5% as in Fig. 5.19, the measurements are not normally distributed. The signals in this example are pretty sparse and have a large dimension. This type of construction is not too likely in real applications, where N and m are both quite large.

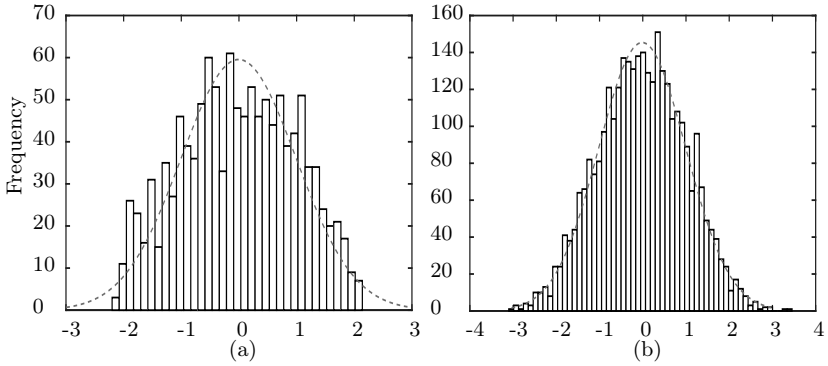


Figure 5.19.: Measurement histogram compared to a normal distribution.
(a) shows measurements obtained from the encryption of a random sparse signal with CS-CTR and $N = 512$, $m = 24$, $s = 5$.
(b) another set of CS-CTR measurements but from a different signal with $N = 512$, $m = 65$, $s = 20$.

Noisy measurements

Another interesting idea is to investigate whether the inherent robustness of Compressive Sensing can be used to enhance the security of general Compressive Sensing applications. The sampling process will introduce some noise to the measurements and this noise might be helpful to hide the signals energy from an eavesdropper. The legitimized recipient, however, would be able to recover a noisy signal and normalization could be avoided. Unfortunately, simulations with various signal-to-noise ratios (SNR) show that even under adaptive white Gaussian noise (AWGN) with an SNR of 5 dB as reported in Fig. 5.20 the measurement distribution can be distinguished from a normal distribution. In contrast to the example from Fig. 5.18 the distribution becomes better with respect to indistinguishability, but it will not be as good as with normalized measurements (cf. Fig. 5.17). For this reason it is important to perform the normalization in the proposed modes of operation as long as it cannot be ensured that the sampled signals all have similar energy.

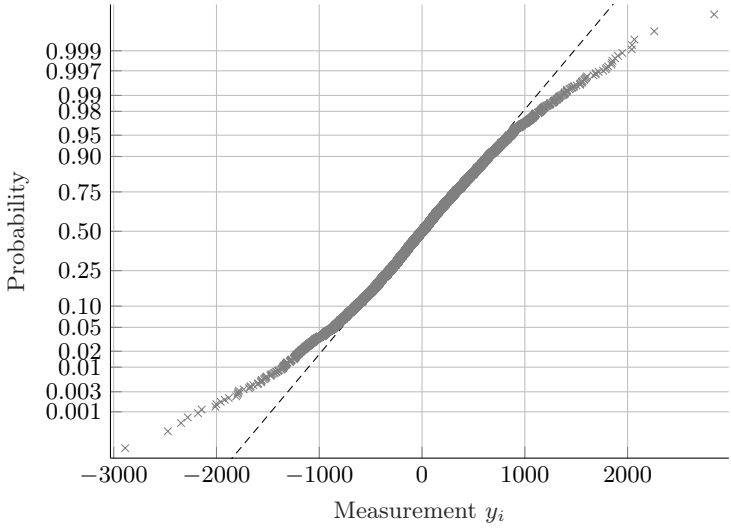


Figure 5.20.: Normal probability plot with noisy measurements of the "plane" test image encrypted with CS-CTR but without normalization (AWGN with 5 dB SNR).

5.7.3. Error sensibility

A mayor advantage of Compressed Sensing based encryption against compress-then-encrypt schemes like [HRU14] is the robustness of the encrypted measurements against noise. The CS encryption modes proposed in this thesis prevent the information leakage introduced by the RIP with their normalized encryption (cf. section 5.2.3). The encrypted measurements norm c is included in the ciphertext vector $\vec{\gamma}$, but c does not benefit from the robustness of CS. Hence, the impact of bit errors in c must be investigated.

Recap, that the measurements norm $\|\vec{y}\|_2$ is converted into a bit string with precision p using the IEEE 754 floating point arithmetic [IEE08]. The resulting bits are then encrypted with π_k, which is implemented using a block cipher mode in case of CS-CBC, or with an XOR cipher in case of CS-CFB and CS-CTR. Fig. 5.21 reports the impact of a single bit error in different bit positions of $c = \pi_k(\langle\|\vec{y}\|_2\rangle_p)$ for the three proposed confidentiality modes.

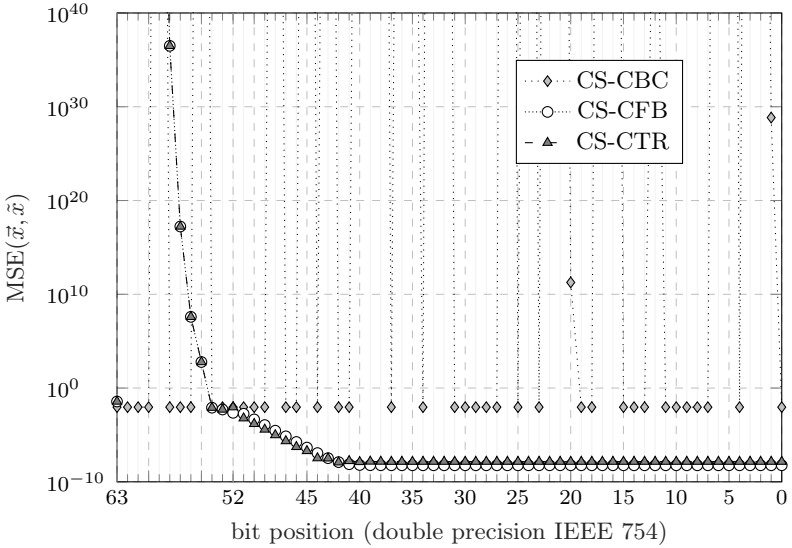


Figure 5.21.: The impact of a bit error in different positions of c.

A randomly generated signal $\vec{x} \in \mathbb{R}^{512}$ with $\|\vec{x}\|_2 = 10$ and $s = 30$ is encrypted with double precision, i.e. $p = 64$. All modes used the same key for the matrix generation, SHAKE128 as PRF* and a randomly chosen $\mathcal{IV}$. CS-CBC applied AES with a random 128 bit key and zero padding. Fig. 5.21 presents the mean squared error (see Eq. (4.2)) of the output of the decryption using the smoothed ℓ_0 recovery [MBZJ09] compared to the original signal $\vec{x}$ for each bit of c.

In case of CS-CFB and CS-CTR, it can be seen that the MSE decreases towards the low-order bits of c. This behavior stems from the stream cipher design used for the encryption scheme π_k. By looking at the IEEE 754 double precision floating point representation (cf. Fig. 5.6 on page 63), one can see that an error in the high-order bits will affect the exponent that is used for the floating point representation. Errors in the exponent have a greater impact than errors in the low-order fraction bits. Errors in the sign bit do not show a huge impact (MSE ≈ 0.0367). A bit error in the fraction bits can be mitigated by the robustness of the CS recovery as the results from Fig. 5.21 demonstrate.

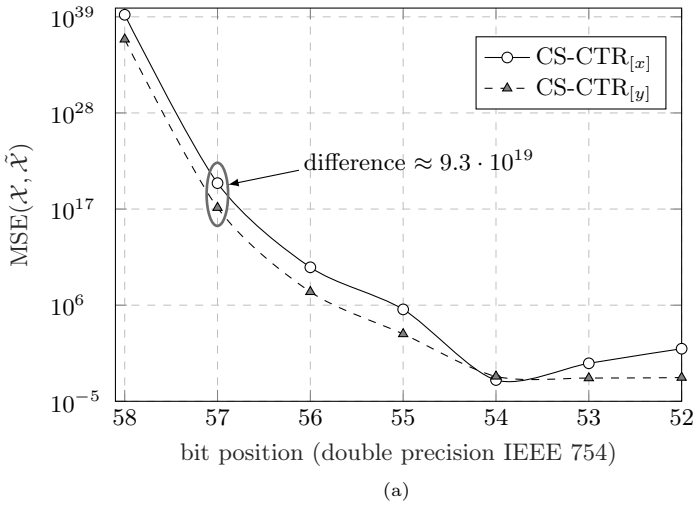
In contrast, CS-CBC shows a worse error behavior due to the block cipher. A bit error has the consequence that each decrypted bit changes with a probability of $\approx 50\%$, independently from the position of the corrupted bit. Hence, the MSE values for CS-CBC are pseudo-randomly distributed. Higher MSE values than 10^{40} are only indicated in Fig. 5.21 for better visibility.

Different normalization methods and error propagation

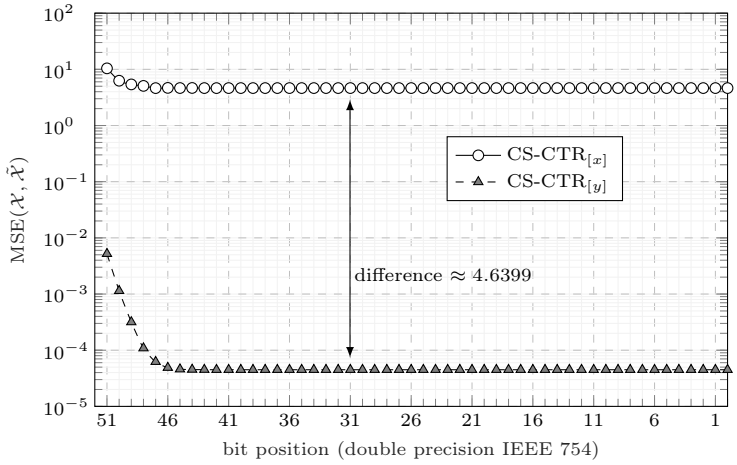
Section 5.2.3 described two different methods for the necessary normalization in CS encryption modes. The behavior of different normalization methods in case of bit errors in c must be considered. In addition to the practicality reasons stated in section 5.2.3, this analysis will justify the choice of the normalization method.

The proposed confidentiality modes work by normalizing the measurements after sampling and they encrypt the measurements norm. In the following experiments, these modes are marked with $[y]$ as subscript. The modes with the alternative normalization method are marked with $[x]$, which indicates that the signal is normalized prior to encryption and the signals norm is encrypted. The definitions of CS-CBC$_{[x]}$, CS-CFB$_{[x]}$ and CS-CTR$_{[x]}$ are given in appendix A on page 129. A set of sparse signals $\mathcal{X} = \{\vec{x}_0, ..., \vec{x}_{14}\}$ is encrypted with either CS-CTR$_{[x]}$ or CS-CTR$_{[y]}$ and for all of the signals the same bits in c_i are changed ($i = 0...14$). The remaining parameters of the experiment are the same as in the experiment previously presented in this section. CS-CTR is the only possible mode for this experiment, since it has no error propagation. The other confidentiality modes use c_i for the matrix generation, which is why decryption cannot succeed if all c_i's are corrupted. Fig. 5.22 (a) shows the impact of bit errors in the exponent bits of c_i, while Fig. 5.22 (b) reports the behavior in case of errors in the fraction bits, both for $\|\vec{x}_i\|_2 = 10$. For a better visibility, only the seven least significant exponent bits are plotted in Fig. 5.22 (a).

The results from Fig. 5.22 show that normalizing the measurements is less sensible to bit errors in c_i than normalizing the signals prior to sensing. This behavior can be explained by the different signal reconstructions. The proposed CS encryption modes perform the reconstruction by $\|\vec{y}_i\|_2 = \rangle \pi_k^{-1}(c_i) \langle$ and $\tilde{x}_i = rec(A_i, \|\vec{y}_i\|_2 \cdot \hat{y}_i) = \hat{x}_i + \epsilon$. Hence, an error in $\|\vec{y}_i\|_2$ might be mitigated by the CS recovery algorithm.



(a)



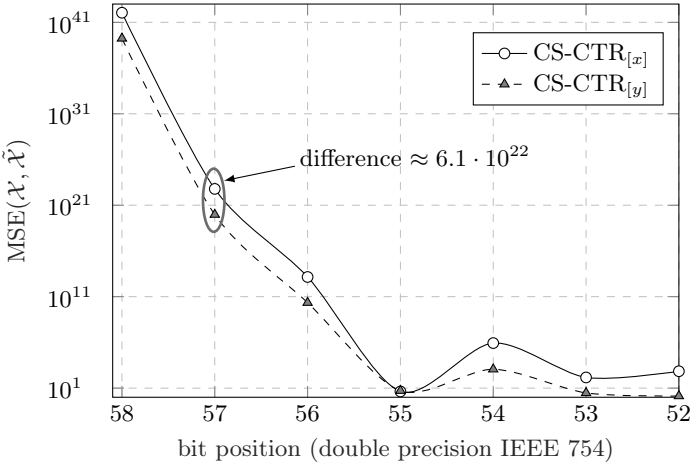
(b)

Figure 5.22.: The impact of a bit errors in c_i with $p = 64$ for $\|\vec{x}_i\|_2 = 10$.
(a) seven least significant exponent bits
(b) fraction bits

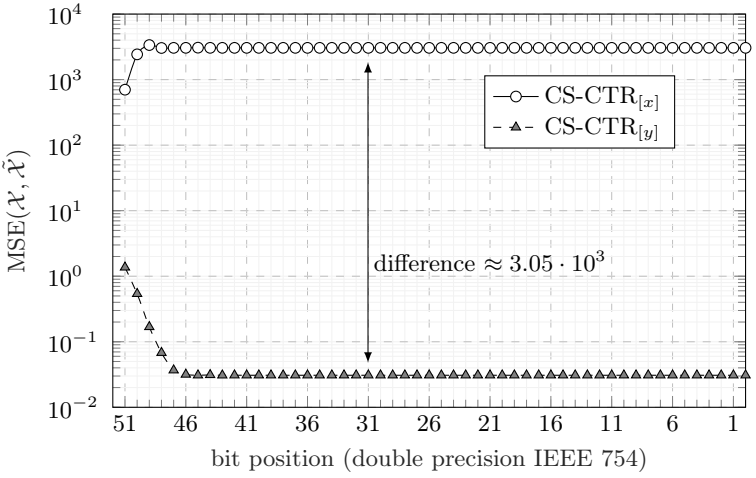
In case of the alternative normalization method, errors in $\|\vec{x}_i\|_2$ are multiplied with the reconstruction error ϵ, which follows from the signal reconstruction:

$$\begin{aligned} \|\vec{x}_i\|_2 &= \rangle \pi_k^{-1}(c_i) \langle \\ \tilde{x}_i &= rec(A_i, \hat{y}_i) \cdot \|\vec{x}_i\|_2 \\ &= (\hat{x}_i + \epsilon) \cdot \|\vec{x}_i\|_2. \end{aligned} \tag{5.32}$$

Fig. 5.23 presents the results from another simulation with a different set of signals $\mathcal{X} = \{\vec{x}_0, ..., \vec{x}_{14}\}$ where $\|\vec{x}_i\|_2 = 256$ $(i = 0...14)$. It can be seen that the general MSE is higher than in the experiments from Fig. 5.22, which shows that the error also depends on the original norm. Moreover the difference between CS-CTR$_{[x]}$ and CS-CTR$_{[y]}$ is even more visible.



(a)



(b)

Figure 5.23.: The impact of a bit errors in c_i with $p = 64$ for $\|\vec{x}_i\|_2 = 256$.
(a) seven least significant exponent bits
(b) fraction bits

Chapter 6

Compressive Sensing with Authenticated-Encryption

6.1. Introduction

The Compressive Sensing encryption modes presented in the previous chapter are designed to ensure confidentiality. Another important security service is data integrity. The proposed confidentiality only modes, for now just called *Compressive Sensing encryption schemes* (CSEs), are not designed to withstand active forgery attacks. Hence, an adversary might for example tamper with the ciphertext, change its order, or replay eavesdropped messages, which leads to significant attacks in practice. This chapter presents Compressed Sensing based authenticated-encryption schemes, which are modes of operations that ensure confidentiality as well as message unforgeability. The focus of this work is on the security of messages during transport or data at rest and message unforgeability under chosen message attacks (cf. section 3.2.4.1), i.e. hard authentication, is the demanded integrity goal.

6.2. Generic Constructions

6.2.1. General

First, three possible compositions of a secure CSE $\Pi = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$ and a secure message authentication (MA) scheme, so called generic constructions, are analyzed with respect to their security, implementability and behavior. Remember that an MA scheme $\Xi = (\mathcal{K}, \mathcal{T}_k, \mathcal{V}_k)$ consists of a key space $\mathcal{K}$, a keyed tag generation algorithm $\mathcal{T}_k$ (e.g. a MAC) and the keyed tag verification algorithm $\mathcal{V}_k$ (see section 3.2.4.1). $\mathcal{T}_k$ takes as input a message $\mathfrak{m}$ and outputs a authenticator $tag \leftarrow \mathcal{T}_k(\mathfrak{m})$. $\mathcal{V}$ takes a message $\mathfrak{m}'$ and a tag as input and outputs ✓ if the tag' computed over the message $\mathfrak{m}'$ equals the input tag and $\perp$ otherwise. The three generic constructions are presented in Fig. 6.1. Following general conventions, different keys should be used for the two primitives.

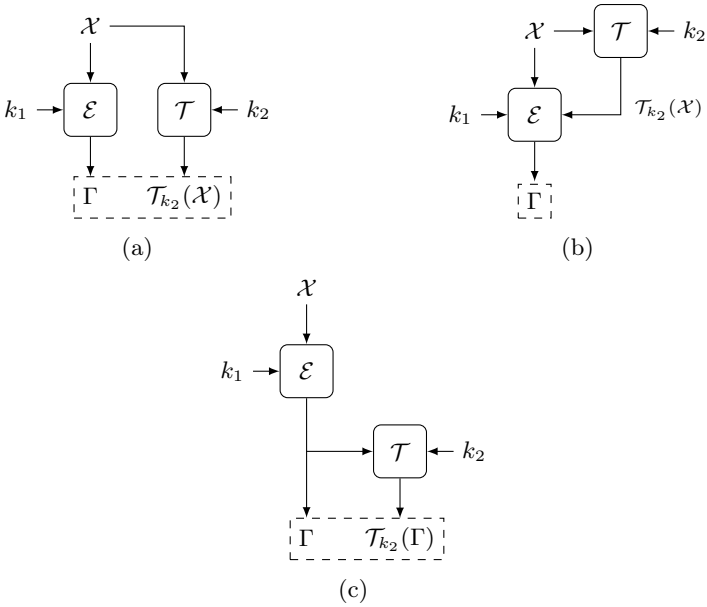


Figure 6.1.: Overview of the generic constructions with explicit key inputs.
(a) Compressed Sensing encryption and MAC (CSE&MAC).
(b) MAC then Compressed Sensing encryption (MAC-then-CSE).
(c) Compressed Sensing encryption then MAC (CSE-then-MAC).

6.2.2. CS-encrypt-and-MAC

The first construction to consider is called CS-encrypt-and-MAC, abbreviated CSE&MAC, and it is shown in Fig. 6.1 (a). The plaintext $\mathcal{X}$ that probably consists of multiple signals $\vec{x}_i$ is encrypted with an appropriate CS confidentiality mode (e.g. CS-CTR) and the MAC is computed over the plaintext, e.g. $tag \leftarrow \mathrm{HMAC}_k(\mathcal{X})$ (cf. Eq. 5.8 on page 56).

Encrypting and authenticating the plaintext is not recommended in general, since this type of constructions are only secure under very narrow conditions [BN08, NRS14]. The reason for this is that encryption and authentication should provide two different security services. While the encryption scheme ensures that the plaintext is confidential, the MA scheme should guarantee that every tampering attempt will be noticed. However, the *tag* computed by the MA scheme might leak some information about the plaintext, since it is not designed to achieve confidentiality. For example, imagine that the same plaintext is encrypted and authenticated multiple times in the CSE&MAC fashion. The ciphertext outputted by the randomized encryption algorithm is different for each run, so no adversary will recognize that the same plaintext is encrypted multiple times. Since the MAC is also computed on the plaintext, the *tag*s will be equal as long as the plaintext is equal and therefore, by observing the *tag*s, the adversary learns that the same plaintext is used multiple times. Moreover, when the $\mathcal{IV}$ is generated by the CS encryption algorithm $\mathcal{E}_k$ it will not be protected by the MAC, because the $\mathcal{IV}$ is treated as a part of the ciphertext. This means that in contrast to the definition of the proposed CS confidentiality modes from chapter 5 the $\mathcal{IV}$ must be provided externally and used as input to the CSE and MA scheme.

CSE&MAC can be implemented easily assuming a system is working with sampling model (M2). In the preferred sampling model (M1) it is not possible to compute the MAC on the plaintext unless the MAC computation is directly integrated into the Compressive Sensing system. It should be noted that using the measurements as *tag* cannot achieve strong message unforgeability under chosen message attacks, due to the additivity of the compressed measurements.

Another issue with CSE&MAC is that the reconstructed signal is often contaminated with noise. A secure MAC is designed to treat even a single bit flip as an malicious forgery attend, which means that $\mathcal{V}$ outputs $\perp$ if the recovered

plaintext differs from the original one. Thus, computing the MAC on the plaintext is inappropriate in most CS applications when perfect recovery cannot be guaranteed.

6.2.3. MAC-then-CS-encrypt

The next construction, presented in Fig. 6.1 (b), is called MAC-then-CSE. As the name suggests, the MAC is computed on the plaintext and then both, i.e. the *tag* and the plaintext, are provided as input to the CSE. On the one hand, this construction is impossible to implement in sampling model (M1), because the plaintext signal is not known prior to sampling. With respect to (M2) on the other hand it is pretty simple to build a MAC-then-CSE construction by slightly tweaking the proposed confidentiality modes to be capable to encrypt the MAC. The integrity of the $\mathcal{IV}$ must be ensured and it must be provided externally. Similar to CSE&MAC authenticating the plaintext seems inappropriate in case of error prone signal recovery.

An example: CS-CTR-SIV

Although the MAC-then-CSE constructions are not implementable in (M1), a dedicated Compressed Sensing authenticated-encryption mode can be build quite easily for the usage in systems that rely on (M2). The mode of operation is an adaptation of the SIV mode proposed in [RS07]. First a synthetic initialization vector is computed for a nonce $\mathcal{N}$ by $\mathcal{IV} \leftarrow \text{HMAC}(\mathcal{N} \,||\, \mathcal{X})$ and this $\mathcal{IV}$ is used as input to CS-CTR (cf. section 5.3). The ciphertext $\Gamma = (\mathcal{IV}, \vec{\gamma}_0, ..., \vec{\gamma}_{l-1})$ and $\mathcal{N}$ are needed as input for decryption, hence the nonce is just send in the clear. The receiver side decrypts the ciphertext with CS-CTR and obtains $\hat{\mathcal{X}}$. Finally, the decryption algorithm computes $tag \leftarrow \text{HMAC}(\mathcal{N} \,||\, \hat{\mathcal{X}})$ and outputs $\perp$ if $\mathcal{IV} \neq tag$ and otherwise it releases the authentic plaintext $\hat{\mathcal{X}}$. Note that the original SIV mode uses CBC-MAC rather than HMAC.

An interesting property of this mode, called CS-CTR with synthetic initialization vector (CS-CTR-SIV), is that it is misuse resistant. Here, misuse resistance means that the security of the mode does not fall apart when an implementer chooses an $\mathcal{N}$ that is not a nonce (see [RS07]).

As mentioned before, the mode can only be implemented in (M2) and it is only suitable when the signals can be lossless recovered. To visualize this, the

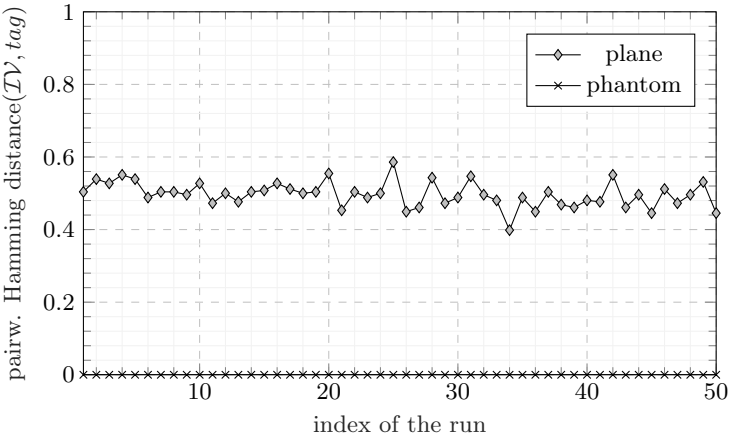


Figure 6.2.: Pairwise Hamming distance between the *tag* and $\mathcal{IV}$ in CS-CTR-SIV for lossless and lossy recovery.

pairwise Hamming distance between the $\mathcal{IV}$ and the *tag* are shown in Fig. 6.2 for different signals. The simulation is performed for the "plane" and "phantom" test image presented already in section 5.7. While the "phantom" image is sparse enough to ensure a good recovery quality, the "plane" image shows a higher recovery error. To mitigate the recovery error, the recovered signals are rounded to the next integer. It can be seen that the pairwise Hamming distance between the *tag* and the $\mathcal{IV}$ is zero in case of the "phantom" image. This shows that the recovery together with the simple post processing is enough to ensure that the reconstructed signal is the same as its original. However, for the "plane" image, the image content is clearly visible (PSNR(original,recovered) $\approx$ 35dB), but the recovered signal is not identical to the original. Hence, the pairwise Hamming distance between *tag* and $\mathcal{IV}$ is around 0.5, which is the expected value for a secure MAC.

6.2.4. CS-encrypt-then-MAC

The third generic composition of a CSE and MA scheme is called CS-encrypt-then-MAC. First, the signals are encrypted with a Compressive Sensing confidentiality mode and the integrity of the ciphertext vectors is protected by

employing a MAC on the ciphertext Γ. Computing the MAC on the ciphertext has the advantage that the MAC does not leak something useful about the plaintext to an adversary, since the ciphertext vectors are asymptotically indistinguishable from random. Treating the $\mathcal{IV}$ as part of the ciphertext ensures that each tampering attempt will result in $\mathcal{V}$ rejecting the ciphertext. Moreover, the integrity of the ciphertext is verified prior to decryption. This has the advantage that manipulated ciphertexts will not be decrypted and no unauthentic plaintext will be released.

CSE-then-MAC can be implemented in both sampling models and it is the only construction that can be implemented easily in (M1). On top of this, the noise introduced during the sampling resp. recovery has no impact on the *tag*. As a consequence, CSE-then-MAC can be applied even when lossless recovery cannot be guaranteed.

6.3. CS Authenticated-Encryption Modes

6.3.1. Design and concept

A Compressive Sensing authenticated-encryption (CS-AE) mode is defined as a three tuple $\Pi = (\mathcal{K}, \mathcal{E}_k, \mathcal{D}_k)$. The encryption algorithm $\mathcal{E}_k$ takes a set of signals $\mathcal{X} = \{\vec{x}_0, \vec{x}_1, ..., \vec{x}_{l-1}\}$ as input and – in contrast to a Compressive Sensing confidentiality mode – it outputs $\Gamma = \{\mathcal{IV}, \vec{\gamma}_0, \vec{\gamma}_1, ..., \vec{\gamma}_{l-1}, tag\}$. The decryption algorithm takes Γ as input and outputs the recovered plaintext $\tilde{\mathcal{X}}$ in case that the ciphertext is authentic and $\perp$ otherwise.

In order to derive a general design for CS-AE, the general model for the design of Compressive Sensing confidentiality modes can be extended. Following the results from section 6.2 the design should use the CSE-then-MAC approach. Now recap that a CSE consists of two main parts, matrix generation and normalized encryption. Protecting the integrity of either the measurements or the encrypted norm is not enough, since the adversary might tamper with the unprotected part. The proposed extension to the CSE general design is straightforward. The signals are sampled with fresh pseudorandom matrices derived from a shared key and the measurements are normalized to ensure their confidentiality as discussed in section 5.2.1. Instead of encrypting the measurements norm with an IND-CPA secure encryption scheme, it is proposed to use

an appropriate AEAD scheme, like OCB (see section 3.2.4.2 for more details). This way, the integrity and confidentiality of the normalization value is assured. The measurements that are already encrypted are given as associated-data (AD) input to the AEAD scheme, which ensures their integrity.

The general CS-AE model

Let k be a random key of length $|k|$ and derive to sub-keys k_1 and k_2 of equal size from k. The key k_1 is used for the AEAD scheme and k_2 controls the pseudorandom matrix generation. Further, define an AEAD scheme as a function-triplet consisting of the stateful functions *init*, *update* and *final* (cf. section 3.2.4.2 and Fig. 3.9 on page 32). The *init* function takes an n-bit $\mathcal{IV}$, a key k_1, an optional associated-data string AD and a `flag` as input. The `flag` is a condition variable that indicates whether the AEAD scheme performs encryption or decryption. It is required that the $\mathcal{IV}$ is a nonce and if it is drawn at random the uniqueness of the $\mathcal{IV}$ must be ensured separately. The $\mathcal{IV}$ and AD strings are added to the internal computation of the authentication *tag*.

When initialized for encryption, the *update* function consumes plaintext and associated-data strings of arbitrary length and it outputs ciphertext blocks $\mathtt{c}_i$ $(0 \leq i \leq l-1)$. The *update* function adds each processed block to the internal *tag* computation. When the whole plaintext is consumed by the *update* function, the *final* function is called and it outputs the authentication *tag*.

In the case that *init* is called with the decryption `flag`, the *update* function processes the ciphertext and associated-data blocks. The $\mathcal{IV}$ and AD strings are added to the internal *tag* computation during *init*. The *update* function decrypts the ciphertext and it also adds the ciphertext together with the associated-data strings to the *tag* computation. The plaintext is just released if the ciphertext is authentic. Therefore, the *final* function is called with the received tag' and it outputs $\perp$ if $tag \neq tag'$, otherwise it outputs the authentic plaintext.

The general model for CS-AE encryption is presented in Fig. 6.3. The encryption algorithm starts by initializing the AEAD scheme with the encryption `flag`, $\mathcal{IV}$, k_1 and optional AD. The AD might for example be used to include time variable parameters in the *tag* computation to prevent replay attacks.

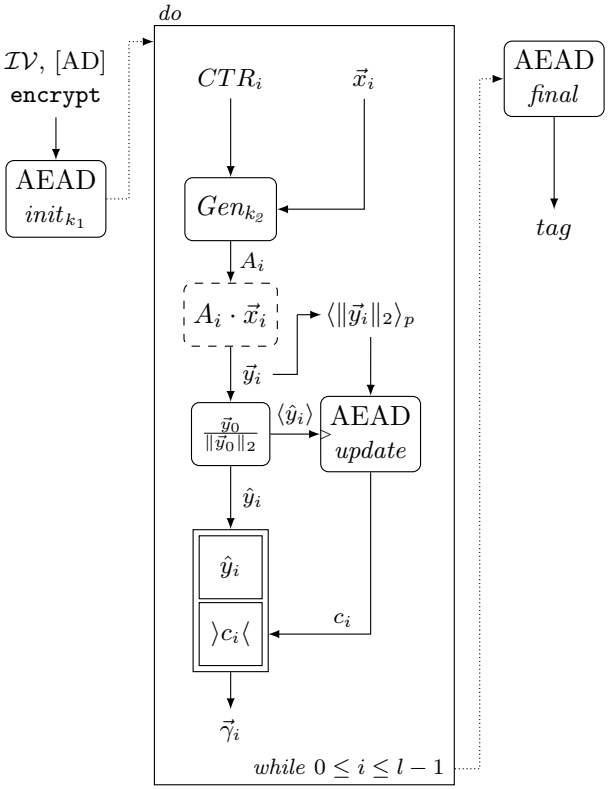


Figure 6.3.: The general design of an encryption algorithm for Compressive Sensing with authenticated-encryption. The associated-data input to the AEAD *update* function is marked with "$>$".

Let l be the number of signals that should be encrypted under one k. The matrix generation algorithm Gen_{k_2} takes a counter CTR_i, i.e. a value that is meant to be distinct and that never repeats, as input. Set $CTR_0 = \mathcal{IV}$ and use an appropriate counting method to increment the counter, see Eq. (6.1) for an example.

$$CTR_i = (\mathcal{IV} + i) \bmod 2^n,\ i = 1, 2, \ldots, l-1,\ l < 2^n. \tag{6.1}$$

Gen_{k_2} outputs a pseudorandom matrix $A_i \in \{+1, -1\}^{m \times N}$. Gen_{k_2} is similar to the counter mode matrix generation function and the reader is referred to section 5.2.2 and 5.3 for more details. There is no reason against using the counter mode construction here, since an CS-AE mode cannot benefit from the self-synchronization property of other type of constructions from chapter 5. Every error will be treated as a tampering attempt and even if the mode of operation might restore synchronization after the loss of some ciphertext, the resulting plaintext will not be authentic. On top of this, the counter mode construction is even more secure than feedback modes (cf. 5.6). The plaintext signals are encrypted as before,

$$\vec{y}_i = A_i \cdot \vec{x}_i, \tag{6.2}$$

and the measurements are normalized

$$\hat{y}_i = \vec{y}_i / \|\vec{y}_i\|_2. \tag{6.3}$$

The AEAD *update* function is used to encrypt the measurements norm represented as a binary string $\langle \|\vec{y}_i\|_2 \rangle_p$ and simultaneously it takes the binary representation of $\hat{y}_i$ as associated-data input. $\langle \hat{y}_i \rangle$ denotes the floating point representation of the vector $\hat{y}_i$, where each entry of $\hat{y}_i$ is represented with p-bit precision. The output of the *update* function is the encrypted norm c_i, which is concatenated with $\hat{y}_i$ in order to form the ciphertext vectors $\vec{\gamma}_i = (\hat{y}_i, \rangle c_i \langle)^T$. After processing all signals, the AEAD *final* function is called to produce the authentication *tag*. The ciphertext is $\Gamma = \{\mathcal{IV}, \vec{\gamma}_0, \vec{\gamma}_1, ..., \vec{\gamma}_{l-1}, tag\}$ and the optional AD might be send in the clear, for example as package header.

The general design for the decryption algorithm of a CS-AE scheme is shown in Fig. 6.4. The AEAD scheme is now initialized with the decrypt `flag`. The AEAD *update* function decrypts the measurements norm given c_i and it processes $\langle \hat{y}_i \rangle$ as associated-data for the computation of the authentication *tag*. The matrix generation is exactly the same as for encryption. A_i and $\vec{y}_i$ is given to the Compressed Sampling recovery algorithm *rec*, which recovers the signal $\tilde{x} = \hat{x}_i + \epsilon$. When all ciphertext vectors are processed, the received tag' is given to the *final* function, which outputs $\perp$ if tag' does not match the *tag* computed on the processed data. Otherwise, the scheme releases the authentic plaintext $\tilde{\mathcal{X}} = \{\tilde{x}_0, \tilde{x}_1, ..., \tilde{x}_{l-1}\}$. In general, the CS-AE scheme might output

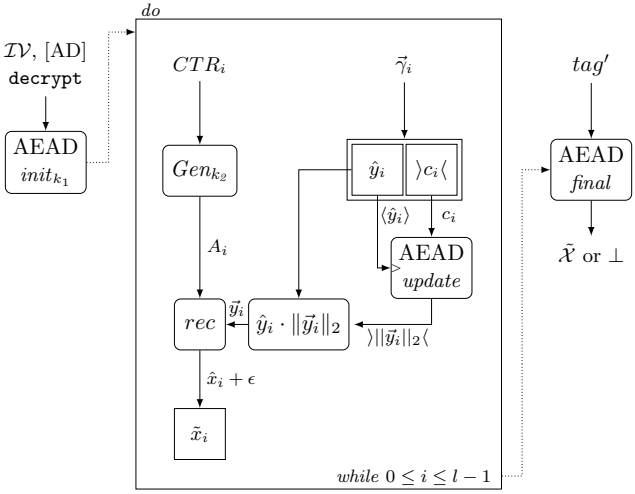


Figure 6.4.: The general design of a decryption algorithm for Compressive Sensing with authenticated-encryption.

each $\tilde{x}_i$ immediately when it is processed. For example, this is necessary when the decryption device does not offer enough memory to store all plaintext signals or when the system should achieve some real-time requirements. However, releasing the plaintext before it is verified is not recommended and raises serious security risks. The reader is referred to [ABL+14] for a formal analysis of this problem with respect to some prominent AEAD schemes.

6.3.2. Implementation and security

6.3.2.1. Implementation details

This section presents implementation details for CS-AE and its main focus is on the AEAD schemes. The reader is referred to section 5.2.2.2 for implementation details regarding the matrix generation function. Suitable AEAD schemes for CS-AE must support the simultaneous processing of associated-data and plaintext. This is due to way AEAD *update* is used in CS-AE. The measurements norm and the compressed measurements are processed together in a single *update* call. From the authenticated-encryption schemes of ISO 19772 [ISO09],

the most prominent schemes, namely GCM and OCB, support this simultaneous processing. CAESAR [CAE17] is a current competition that seeks for authenticated-encryption schemes. The goal of the CAESAR competition is to find strong AEAD schemes primarily as a successor of GCM, since the widely employed GCM has some drawbacks. For example, when the $\mathcal{IV}$ in GCM is not a nonce, confidentiality and integrity is lost completely [Fer05]. This means that a small implementation mistake, like drawing the $\mathcal{IV}$ at random without ensuring its uniqueness, reduces the number of plaintext and AD blocks to which GCM can be applied by the order of a square root. [BZD+16] found that a large number of servers use random $\mathcal{IV}$s for their TLS connections while running GCM and might therefore be vulnerable to nonce collision attacks.

The third round candidates of the CAESAR competition were announced just recently. Four out of the 16 third round candidates support simultaneous processing of plaintext and associated-data – one of them is OCB. Even more of the proposed AEAD schemes might offer this functionality when implemented accordingly. Table 6.1 on the following page compares AEAD schemes from ISO 19772 and the third round of CAESAR that are suitable for CS-AE implementations. The preferred modes for CS-AE implementations are Keyak and OCB. The reason for choosing OCB is that it is standardized, simple and fast (see [KR11]). Keyak on the other hand, is based on the Keccak-f permutation, which is also the primitive used in SHAKE (cf. section 5.2.2.2). Hence, matrix generation and authenticated-encryption might be performed by SHAKE resp. Keyak and only a single primitive must be implemented. This is especially interesting in environments with limited code space or when the primitive is implemented in hardware. On top of this, Keyak offers the possibility to output intermediate authentication *tags* that can be used to ensure the integrity of parts of the ciphertext. With intermediate *tags*, the decryption algorithm is able to verify parts of the ciphertext as they are decrypted and these parts of the plaintext can be released earlier.

	AES-OTR	*Deoxys*$^{\neq}$	*GCM*	*Keyak*	*OCB*
Underlying primitive	AES	AES	128 bit block cipher	KECCAK-f	128 bit block cipher
Simultaneous processing of AD and plaintext	`Yes`	`Yes`	`Yes`	`Yes`	`Yes`
Nonce misuse resistance	`No`	`No`	`No`	`No`	`No`
Intermediate tags	`No`	`No`	`No`	`Yes`	`No`
Parallel (enc./dec.)	`Yes/Yes`	`Yes/Yes`	interleaving possible	`Yes/No`	`Yes/Yes`
Formal security analysis	`Yes`	`Yes`	`Yes`	`Yes`	`Yes`
Intellectual property	patent pending	might fall under OCBs patents	free from patents	free from patents	patented, but free licenses available
Literature source	[Min15]	[JNP15]	[Dwo07, MV04b]	[GJM$^+$15]	[KR14]

Table 6.1.: Summary of suitable AEAD modes for CS-AE implementations.

6.3.2.2. Security analysis

The two important parts with respect to security of CS-AE modes are the counter mode matrix generation algorithm and the AEAD scheme. Both parts are independent of each other, since it is required that distinct keys are used. The security of Gen_{k_2} and the asymptotic ciphertext indistinguishability of the compressed measurements were already discussed in section 5.3.2 and 5.2.2.3. Hence, this results will not be repeated here.

All proposed AEAD schemes for CS-AE come with a formal security analysis against generic attacks and the reader is referred to the literature presented in Table 6.1 for the concrete security bounds of each scheme. The security analysis of AEAD schemes is split into a privacy (IND-CPA) and a integrity (unforgeable encryption) analysis (see section 3.2.4.2). It is crucial that the $\mathcal{IV}$ is a nonce and as noted in Table 6.1, none of the proposed schemes guarantees security if a nonce is repeated. Another important parameter that might be chosen by an implementer is the length (in bit) of the authenticator tag, denoted by $|tag|$. An adversary that just guesses a tag at random will succeed with probability $1/2^{|tag|}$, hence after q decryption queries the probability becomes $q/2^{|tag|}$. This applies to all the modes from Table 6.1 except for GCM. In case of GCM, $|tag|$ is more important, because an adversary might forge a tag with probability $(q \cdot (l+1))/2^{|tag|}$, where l is the total number of AD and plaintext blocks processed under a single key (see [MV04b, Theorem 1/2]). A concrete attack on GCM with short $|tag|$ is presented in [Fer05]. This attack is practical even for $|tag| = 64$. A throughout discussion of GCM can be found in [Rog11, Section 12]. The recommended tag size for CS-AE is 128 bit, but for other AEAD schemes then GCM smaller sizes are also reasonable for some applications as long as the probability of a successful forgery is small enough.

Replay protection

Recognizing replay attacks is quite simple with the proposed CS-AE general design. The optional AD string that is an input to the AEAD *init* function can be used to include time variable parameters (TVP) like time stamps or sequence numbers. The recipient must verify the plausibility of the received TVP separately. Random numbers are also possible, but the legitimized recipient must keep track of all received random numbers. If an adversary tampers with

the TVP during transport, the CS-AE mode will not output the unauthentic plaintext. When Keyak is used with intermediate tags, it is possible to add the TVP as associated-data into the *update* step that produces the intermediate *tag*.

6.3.2.3. Proof of concept

Figure 6.5 shows the encryption and decryption of the "cameraman" image with a dedicated CS-AE mode called *Compressive Sensing with Offset Codebook* (CS-OCB). This CS-AE mode uses the block cipher AES128 in OCB mode [KR14] as AEAD scheme. Fig. 6.5 (a) shows the plaintext, and the output of the CS-AE encryption is presented in Fig. 6.5 (b) together with the tag', i.e. the last 128-bit of the ciphertext (cf. 6.3.1). SHAKE128 is used as PRF* for the matrix generation and $N = l = 512$, $m = 205$. Using the TVAL3 [Li09] recovery, the decryption algorithm outputs the plaintext from Fig. 6.5 (c), since the tag that is computed during the decryption equals the tag' on the ciphertext.

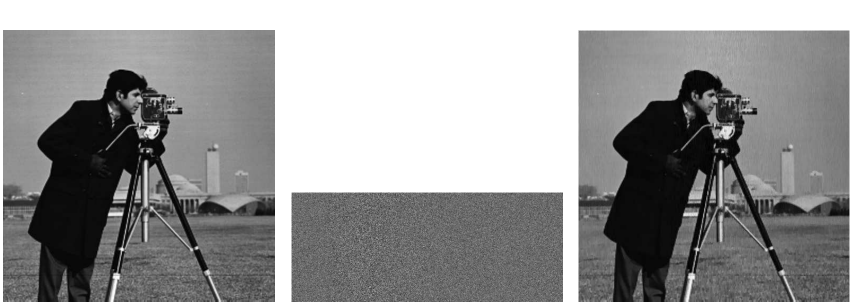


Figure 6.5.: Encryption and decryption of the "cameraman" image with CS-OCB.

Chapter 7

Fog Computing Security by Compressive Sensing Modes

7.1. System Design

Most companies feel uncomfortable when their production or manufacturing data is handled "somewhere" in the Cloud. Fog computing is an architecture in which data is aggregated and processed with close proximity to the data-source and -sink [BMZA12]. This offers various benefits like rapid data analytics and fast response times. On top of this, the data stays under control and only the necessary information might leave the local network.

This section proposes a system for secure Fog computing in Industry 4.0 and IoT applications, with a focus on (industrial) sensor networks. Some results from this section are published in [FBK+17]. Here, a software framework is presented that serves as a use case for the Compressed Sensing encryption modes presented in this thesis. Compressed Sensing modes with their joint encryption, compression and sampling offer a huge advantage in this setting. The mechanisms that are needed for the usage of the CS modes, like parameter arrangement, key-exchange resp. -management, authentication and many more are included in the presented system. In addition, also block cipher based encryption schemes and cryptographic hash functions are provided by the frame-

work. By this means, the system is usable even without Compressed Sensing devices. The developed system is designed in such a way that it can be integrated in an existing network without exposing the sensible (sensor-)data to other network users or applications. The underlying security mechanisms are handled by the framework to enhance usability and avoid usage errors. This means that the software hides the security mechanisms from the implementer and generates keys, $\mathcal{IV}$s etc. in the correct way.

Network topology

Fig. 7.1 presents the network topology and system design. It is supposed that the system is integrated in an existing network infrastructure and there are no specific requirements on this network.

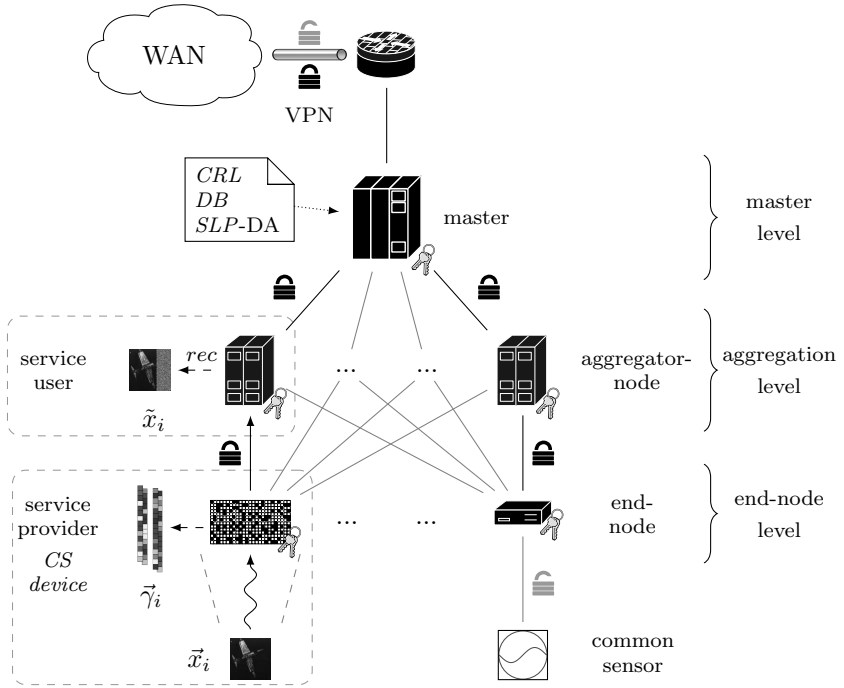


Figure 7.1.: The system design with Compressed Sensing encryption.

The components of the system are called *nodes* and communication inside the network is based on *services*. Each node is offering services that can be used by other nodes, like it is shown exemplary on the lower left side of Fig. 7.1. Services that a node provides are announced and registered centrally using the *Service Location Protocol* (SLP) [GPVD99]. Other nodes are able to locate a service by asking a central instance, the SLP *directory agent* (SLP-DA). The SLP-DA answers with a service reply that contains the *Uniform Resource Locator* (URL) under which the service is accessible.

The system components are organized hierarchically in three abstract levels. The highest level is the *master level* and it contains a single master-node. This master offers central services that are provided by third party software, like the SLP-DA or a local database (DB) connection. Moreover, the master-node controls the information that leaves the Fog network or the data that is stored in the database. Authorized external users might access the master-node over a secure VPN connection to perform system management tasks. The master is highly important for the system and it might become a single point of failure. Hence, suitable mechanisms are implemented that increase reliability and availability. In case that a master-node fails, an election algorithm is used to select a new master-node from the nodes of the underlying *aggregation level* (see Fig. 7.1). The winner of this election will take over the services performed by the original master-node, like the SLP-DA. Services that are already located by the service user are still accessible even if the SLP-DA on the master-node is not reachable, but searching and registering services will not work. SLP services must be re-registered in a timely manner and if a SLP service provider does not receive alive messages from the SLP-DA it will register the service again at another available SLP-DA.

Nodes on the aggregation level are called *aggregator-nodes* or *aggregators* for short. Their task is to collect, process or categorize the data that they receive from the *end-node level*. After aggregation, the data is transmitted to the master-node for further processing or storage if applicable. The hardware of an aggregator and master-node should have more performance, memory and storage in comparison to end-nodes. Suitable hardware for end-nodes are single-board computers like the Raspberry Pi 3, since they offer a good trade-off between costs and performance and many end-nodes might be needed in the system. The task of an end-node is quite simple, it should collect data from data

sources, e.g. an optical or capacitive sensors. The communication between the end-node and the data source is outside of the controlled Fog network and cannot be protected. This is the point where Compressed Sensing devices become interesting, equipped with Compressed Sensing encryption modes they offer real end-to-end encryption that is integrated into the sampling process.

Design goals

The system is designed with respect to the following main goals:

- **Security by design**: The data inside the Fog network is protected on the application layer by security mechanisms in accordance to a given *security service* (e.g. authenticity). A security service is a technology independent description of a security measure in order to react to the threats and risks of a distributed system. Each service in the Fog network must define the security service that indicates how the data of the service is protected. The communication with third party software that is used by the system, for example the local database, is protected by existing protocols like TLS.

- **Usability**: Although, security is the main design goal of the system, another important factor is usability. Handling cryptographic keys, generating the correct $\mathcal{IV}$ for a given scheme or determining an appropriate *tag* length is a problem for many implementers. Therefore, the realization of the security services and the key establishment are handled by the system and the implementer must only develop the service and determine its security service.

- **Scalability**: The system consists of various components and is easily expandable by adding further nodes. The design allows multiple Fog networks that are locally and technically separated from each other, but controlled and monitored by an external instance, e.g. a central provider.

Security services

A service must specify a security service. Three different security services are offered by the framework.

- ▷ **Authenticity**: The integrity of the transmitted data and its origin is ensured by a symmetric message authentication scheme. Sequence numbers and time stamps are used to protect against replay attacks. The transmitted data is send in the clear.

- ▷ **Confidentiality and Authenticity**: Authenticated-encryption with associated-data or CS with authenticated-encryption (cf. chapter 6) is used to ensure confidentiality and integrity. Sequence numbers and time stamps are treated as associated-data. Confidentiality without authenticity is not offered, since it cannot resist powerful active attacks.

- ▷ **Non-Repudiation**: Digital signatures, together with time stamps and sequence numbers are used to achieve non-repudiation. Non-repudiation means that a recipient is able to proof that the transmitted message was sent from the entity that has signed the message.

The abstraction through security services has the advantage that implementers who are no experts in cryptography are able to understand in which way a service is protected. All security relevant cryptographic operations, including key-establishment and -management, are handled by the developed software, transparent for the user.

Public key infrastructure

Each node of the system has a private key pair to perform the *Elliptic Curve Digital Signature Algorithm* (ECDSA) and an X.509v3 certificate that guarantees the authenticity of the corresponding public key. The certificate must be signed by a certificate authority (CA) that is trusted by all nodes inside the system. It is also possible to use other algorithms than ECDSA, but the proposed system uses only digital signatures and no asymmetric encryption. The master-node has access to all certificates and is used to provide the certificates to other nodes, for example with the *Lightweight Directory Access Protocol* (LDAP) protected by TLS. Additionally, the master-node publishes a certificate revocation list (CRL) that contains revoked certificates. Certificates that are in the CRL are no longer trusted and requests that are signed with a revoked key are dropped by the system, e.g. during authentication.

7.2. Implementation

7.2.1. Software architecture

The following sections describe the proof of concept implementation of the system described in section 7.1. The main software framework is implemented using JAVA SE 1.8, but user-provided services might rely on functionality implemented in another programming language like C++ or Python. This heterogeneity of programming is established by two components, *RabbitMQ* and *Protocol Buffers* [Piv16, Goo16a].

RabbitMQ – used in version 3.6.5 – is a cross-plattform message oriented middleware that supports several popular languages. It implements the *Advanced Message Queuing Protocol* (AMQP), which allows binary communication on the application layer. Messaging with RabbitMQ is asynchronous, fast and reliable. Acknowledgements sent by the middleware are used in the proposed system to recognize malicious message blocking.

Protocol Buffers uses an interface definition language to provide a mechanism for serializing structural data that is platform- and language-neutral. It is smaller, faster and simpler than XML. The desired data structure is defined once with name-value pairs and some attributes like data types (see [Goo16b] for further information). The Protocol Buffers code generator creates source code that includes write and read methods for serialization and deserialization.

7.2.2. The service framework

Providing a service

The software supports two different types of services, *publish-subscribe* (PS) services and *remote procedure calls* (RPCs). In general, PS services are used in the case where a service provider (called publisher) sends messages to an arbitrary and possibly empty set of service users. If a service user is interested in the publisher's information, the service user subscribes to the service. Service provider and service user authenticate each other in the presented system and the service provider verifies if the requesting subscriber is authorized to use the service. RPCs are implemented to allow distributed inter-process communication. Figure 7.2 presents an UML class diagram containing the most important classes for a service provider.

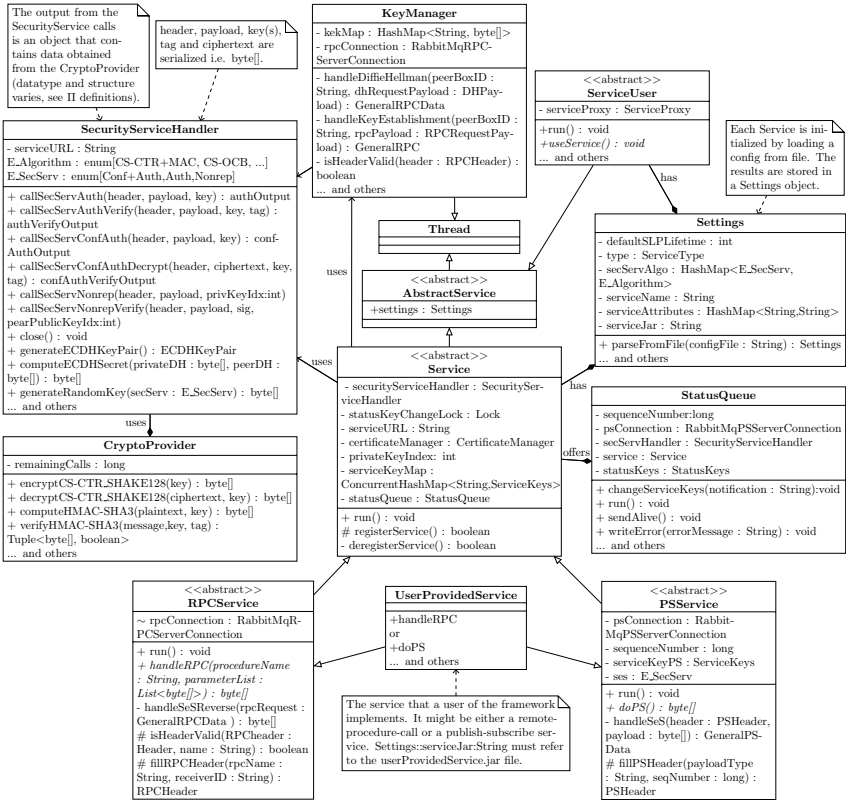


Figure 7.2.: UML class diagram for offering a service in the JAVA prototype.

Fig. 7.3 shows the Protocol Buffer message formats for the two different service types. The message is defined by different fields that consist of key-value pairs of the form `Name:`*`DataType`*. In addition, a field might contain other fields. Optional fields are surrounded by $[\cdot]$ and fields that might be repeated arbitrarily are marked with $\{\cdot\}$. Default values for a field are stated in round brackets. Whether the fields that are marked with the dashed brackets in Fig. 7.3 are authentic and confidential or just authentic depends on the security service that is specified for the service.

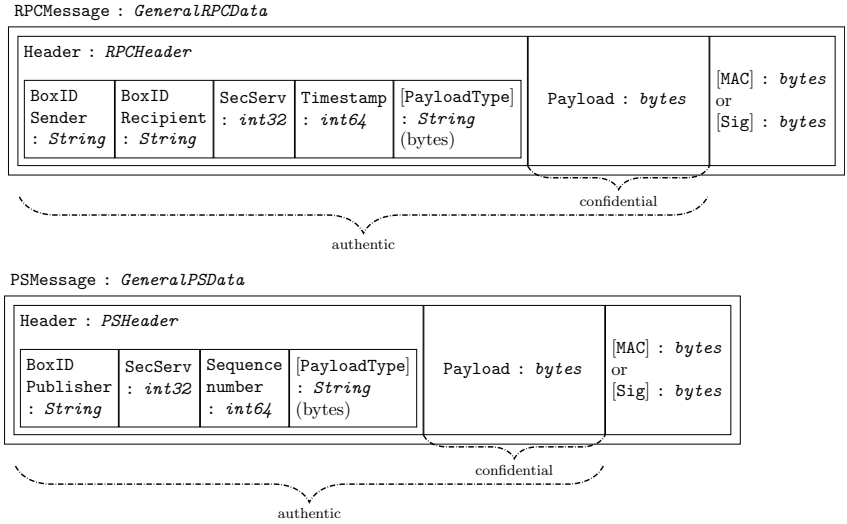


Figure 7.3.: Message format of a general RPC and PS message using the Protocol Buffers notation.

Publish-subscribe services use sequence numbers for replay protection whereas RPC messages include time stamps. This is simply due to the different concepts. It is assumed that a PS session contains multiple messages that are sent periodically, while a remote procedure call is in general not performed too frequently. The most natural use cases for Compressed Sensing encryption modes in (M1) are publish-subscribe services. For example, they could be used to provide end-to-end protection of periodically published sensor data. However, due to the lack of real Compressed Sensing devices it was just possible to simulate CS modes in the fashion of sampling model (M2).

The service provider fixes the security service used for the communication and accordingly it generates service keys. This is done during the service initialization, i.e. in the constructor of the "Service" class. Each service has a "Settings" object for the initial configuration that is loaded from a file (see Fig. 7.2). This service configuration file contains the SLP-service name, SLP-attributes, the service type and the location of the source code for the service. The classical, i.e. non-CS, security algorithms that are used to perform the cryptographic op-

erations are predetermined for each security service. However, the user might change the algorithm to another one provided by the framework via the service configuration file.

Using a service

A service is used with the help of a "ServiceProxy" object that handles all necessary tasks like authentication and key establishment. Fig. 7.4 presents the most important classes for a service user or client (see also Fig. 7.2 for missing classes). Similar to a service, the service proxy might either be an publish-subscribe or remote procedure call proxy.

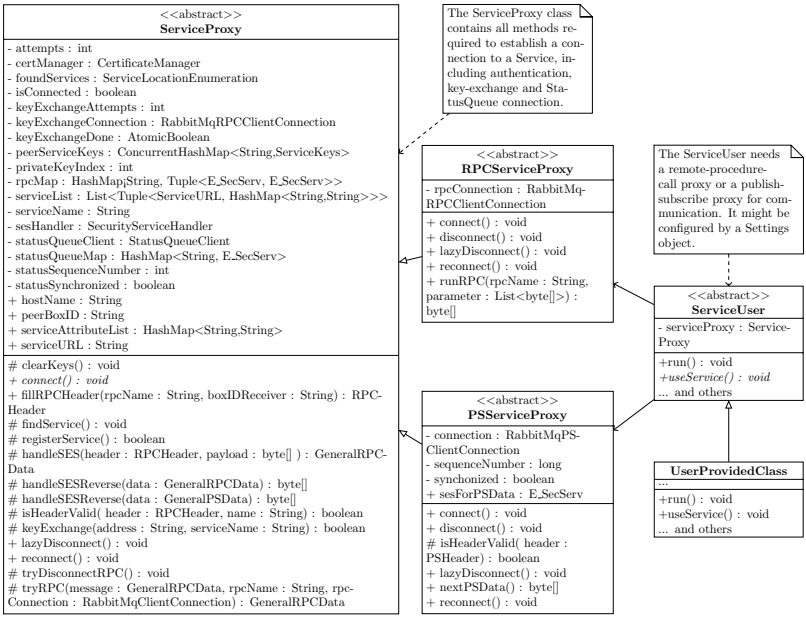


Figure 7.4.: UML class diagram for a service user in the JAVA prototype.

As soon as a client wants to use a service, the service proxy sends an SLP service requests in order to locate the service. After the service is located, the client performs a specific remote procedure call – the *Diffie-Hellman (DH)-RPC* – which all service providers in the system offer. The DH-RPC is used to carry

out a bilateral authenticated and ephemeral Elliptic Curve DH key exchange. This way, the key exchange achieves perfect forward secrecy. It is crucial that each connection uses fresh public/secret Diffie-Hellman parameters, but this is handled by the framework.

The structure of the messages that are exchanged during the key establishment are presented in Fig. 7.5. The DH-RPC message is a special type of the `RPCMessage` known from Fig. 7.3. While the `RPCMessage` is a generic message type for general RPCs, the DH-RPC message is only used during key establishment. The security service for a DH-RPC message is non-repudiation, but the digital signatures are just used to ensure the integrity of the message and its origin. The ***`DHPayload`*** contains the RPCs name, the public Diffie-Hellman parameter and an optional algorithm that is used to ensure confidentiality and authenticity of the service key exchange. A default algorithm will be used if no specific algorithm is given in this field.

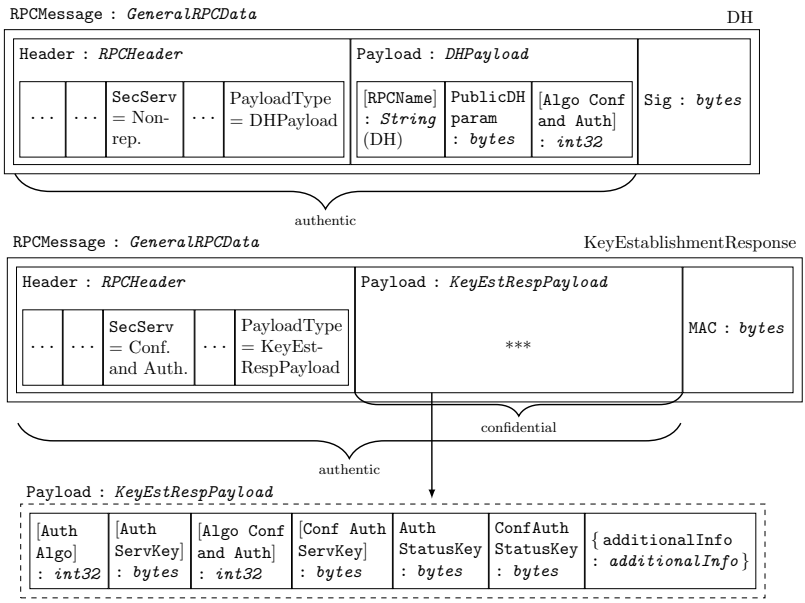


Figure 7.5.: Message format of a DH and KeyEstablishmentResponse message. The fields that contain "..." are the same as in Fig. 7.3.

The service provider handles the DH-RPC call by verifying the signature and the header. For example, the service provider checks if the sender and recipient ID fields are semantically correct and if the time stamp is plausible or not. After a successful verification, the service provider calls the DH-RPC of the service user, such that both sides are able to compute the DH secret.

The service user calls the KeyEstablishment-RPC and the service provider answers with a KeyEstablishmentResponse message, which contains the encrypted service key to the requested service and the security algorithm. The corresponding Protocol Buffers message format is shown in Fig. 7.5. The message is protected by an authenticated-encryption scheme. The key for this scheme, called key-encryption-key (KeK) in Fig. 7.2, is derived from the shared DH secret using a key derivation function.

The KeyEstablishmentResponse message also contains the keys for the *status queue*, which is a special PS sub-service that belongs to the service. All services offer this status queue in order to inform their clients, for example, when the service keys are renewed. The `additionalInfo` field is used for parameter exchange and might be repeated or empty. It contains a list of security services used for the offered RPCs or the security service for a publish-subscribe service. In case that the service uses Compressed Sensing encryption modes, this field contains the parameters m, N, Ψ and possibly $rec(\cdot)$. Fig. 7.6 summarizes the PS service usage in form of an UML sequence diagram. A prototype application of the proposed system is presented in [FBK+17].

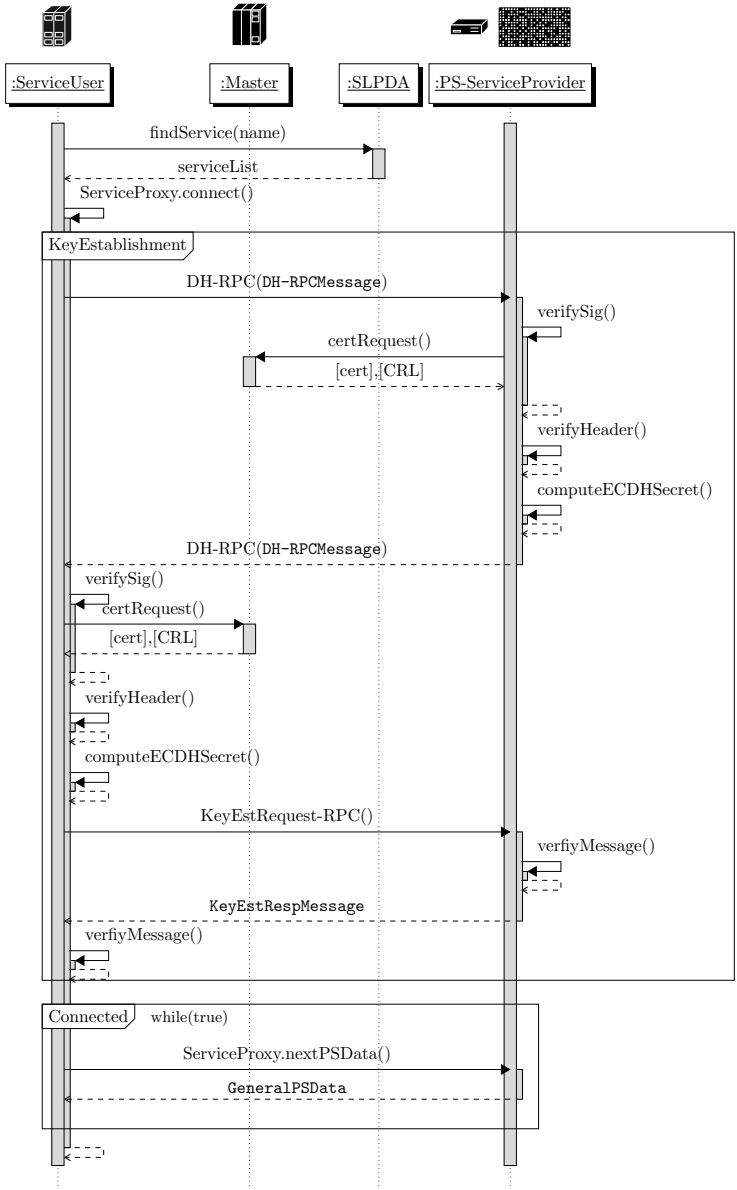


Figure 7.6.: UML sequence diagram for a PS service usage.

Chapter 8

Conclusion

8.1. Contributions

Compressive Sensing confidentiality modes and Compressive Sensing with authenticated-encryption were proposed in this thesis. A software framework for secure Fog networks in Industry 4.0 applications was presented and implemented as a use-case for the proposed modes. Some results from this thesis are published in [FBK+17, FR17, FR16, Fay16]. The main research contributions are summarized below.

Section 4.2.1 presented a formal security notation for Compressive Sensing based encryption in the one-time encryption scenario based on results from the related literature. The impact of weak random matrices was studied and it was shown that weak matrices cannot fulfill the necessary requirements for a successful signal recovery. The results from section 4.2.2 indicate that Bernoulli sampling matrices achieve asymptotic ν-indistinguishability even in a practical use case. The statistical test applied in the presented experiment cannot distinguish the measurements in distribution as long the they are obtained from signals with equal energy. The results from sec. 4.3 show how Compressive Sensing based encryption with a fixed matrix might be attacked. Countermeasures were discussed and a formal security definition for secure Compressed Sensing encryption was presented with respect to the many-time encryption scenario.

Lemma 5.1 shows that the loss of security is negligible when random matrices are replaced with the output of a secure pseudorandom number generator seeded with a random key. Hence, the security of the encryption scheme reduces to the security of the underlying PRNG. A general design for Compressive Sensing confidentiality modes was presented in section 5.2. Implementation details are addressed in section 5.2.2.2. Suitable constructions and primitives were analyzed and a summary of recommended constructions and parameters was given based on the security and performance considerations presented in section 5.2.2.3 resp. 5.2.2.4. The results show that the recently proposed XOFs outperform iterative PRF→PRF* designs in performance and security aspects. Three dedicated modes of operation with different interesting properties were derived from the general model and each of them is analyzed with respect to security. While CS-CTR is highly parallelizable, CS-CBC and CS-CFB offer self-synchronization in case of losses or changes in the ciphertext order. The proposed modes are summarized and compared in section 5.6. The experimental results from section 5.7 show that the proposed modes do not introduce further recovery errors. Additionally, the distribution of the compressed measurements was analyzed in 5.7.2 and the impact of different parameters on the indistinguishability of the compressed measurements was presented.

The results from chapter 6 show that Compressed Sensing encryption followed by a MAC is the only generic composition that can be implemented in sampling model (M1). Moreover, computing the MAC on the plaintext is inappropriate in most Compressive Sensing applications as long as perfect reconstruction cannot be guaranteed. Because of this results, the general model for Compressive Sensing with authenticated-encryption, proposed in section 6.3.1, follows the CSE-then-MAC approach. Suitable AEAD constructions for dedicated CS-AE modes were presented, analyzed and compared. Next to the security analysis from section 6.3.2.2, it was shown how replay protection can be achieved with the proposed CS-AE constructions.

Finally, an application scenario for CS modes including key-establishment and parameter exchange was presented in chapter 7. The developed software framework offers a secure service based approach for Industry 4.0 applications. All cryptography related actions are performed by the framework such that implementers without a deep knowledge in cryptography can easily build a secure application. The software is also usable in absence of CS devices just

with classical encryption schemes. With the proposed encryption modes, one achieves real end-to-end security that is directly integrated into the sensor.

8.2. Future Work

The general models for Compressive Sensing confidentiality modes and CS-AE should inspire researchers to develop further modes. Other interesting open issues are discussed in this section.

Hardware implementations

The experimental results presented in this thesis were achieved through simulations. All proposed designs were developed such that they can be used in sampling model (M1), but due to the lack of CS hardware it was not possible to investigate the behavior of the proposed encryption modes in (M1) systems. Clearly, implementing the proposed modes in hardware, e.g. as imaging sensor, is a challenging and interesting future work.

Security of CS with discrete Gaussian matrices

As discussed in section 4.2.2, it is not sure whether discrete Gaussian sampling matrices achieve the same level of security as the continuous Gaussian matrices. It must be investigated how the discrete Gaussian sampling methods from the field of lattice-based cryptography perform in the Compressive Sensing setup. It is clear that those methods will introduce further costs and therefore the performance security trade-off needs to be studied.

Performance comparison with compress-then-encrypt

Encryption schemes from the compress-then-encrypt family cannot offer a joint robust encryption and compression. However, they are targeting similar security goals than CS modes. The performance analysis published in [FR17] shows that Compressed Sensing followed by classical encryption in the binary domain with a suitable encryption mode is more than twice as fast as dedicated CS encryption modes. The presented results are only based on simulations and different results are expected if the encryption modes are implemented in hardware. In addition, no special encoding and quantization were used, but the compression ratio and

thus the throughput of CS can be enhanced by further suitable mechanisms (see [LLY+14] for an example). Hence it is left as future work to show whether an optimized CS system with the proposed encryption modes can outperform compress-then-encrypt schemes in speed and compression ratio.

Misuse-resistant modes

Misuse resistance has become an important factor in recently proposed AEAD modes (see the candidates from [CAE17]). CS-CTR-SIV, presented as example in section 6.2.3, shows some kind of nonce-misuse resistance. However, this mode can only be applied in (M2). In general, misuse resistance is achieved by performing computations on the plaintext. In case of SIV one computes a MAC on the plaintext and uses the MAC as $\mathcal{IV}$. This might only be possible in Compressive Sensing when the MAC generation is integrated into the sampling process in a CSE&MAC construction, but till now no concrete scheme is proposed for this purpose. This leads to the following open issue.

Integration of soft verification

Computing a MAC on the plaintext signal is a problem when the signals are not strictly sparse but compressible or when the measurements are contaminated with noise. The recovery error will lead to *tag*s and data that will be rejected by the verification algorithm. Soft verification resp. soft authentication describes message authentication schemes that are robust against some sort of noise (see for example [TZ14]). With soft verification, a Compressive Sensing encryption scheme could offer confidentiality and robust authentication that is directly integrated into the sampling and reconstruction process.

Appendix A

Alternative Confidentiality Mode Designs

This annex contains alternative definitions of the proposed CS confidentiality modes. Instead of encrypting the measurements norm, the modes work by encrypting the signals norm. The reader is referred to section 5.2.3 for more details. An experimental comparison of these modes with the modes proposed in chapter 5 can be found in section 5.7.3.

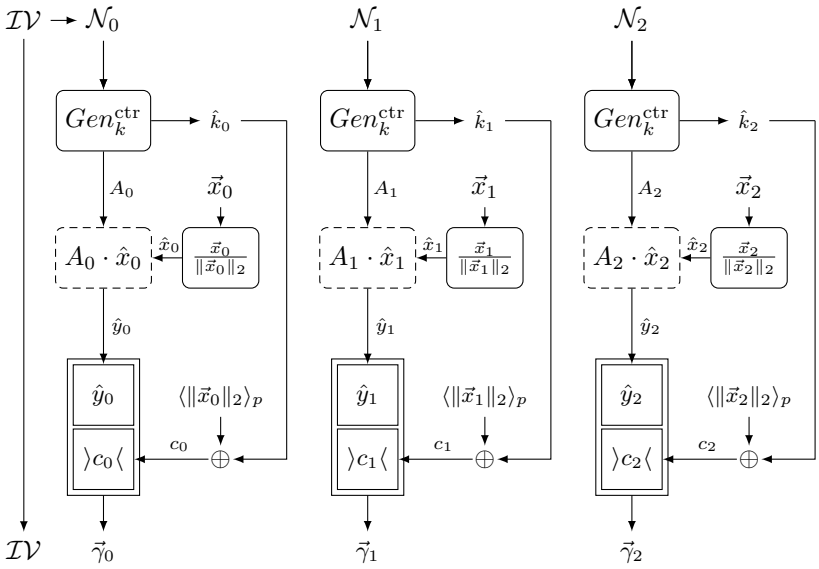


Figure A.1.: Encryption side of the Compressive Sensing Counter Mode with normalized signals abbreviated CS-CTR$_{[x]}$.

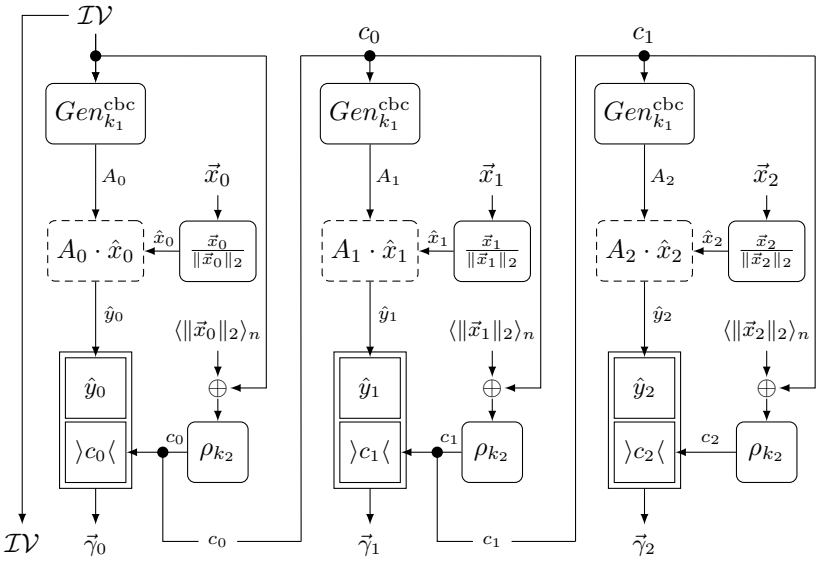


Figure A.2.: Encryption side of Compressive Sensing with Cipher Block Chaining and normalized signals (CS-CBC$_{[x]}$).

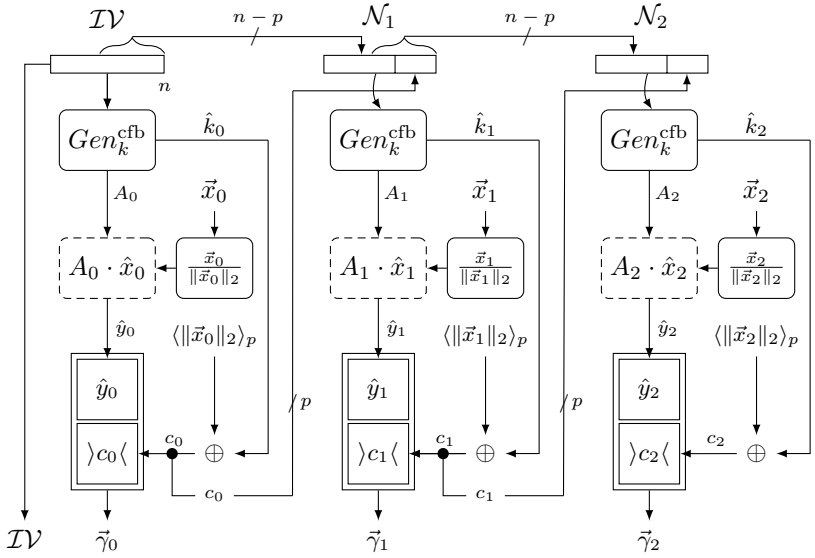


Figure A.3.: Encryption side of Compressive Sensing with Cipher Feedback and normalized signals (CS-CFB$_{[x]}$).

Bibliography

[ABL+14] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, N. Mouha, and K. Yasuda, "How to Securely Release Unverified Plaintext in Authenticated Encryption," in *Advances in Cryptology – ASIACRYPT 2014*, P. Sarkar and T. Iwata, Eds. Springer Berlin Heidelberg, 2014, vol. 8873, pp. 105–125.

[AGPS02] A. Alkassar, A. Geraldy, B. Pfitzmann, and A.-R. Sadeghi, "Optimized Self-Synchronizing Mode of Operation," in *Fast Software Encryption*, G. Goos, J. Hartmanis, J. van Leeuwen, and M. Matsui, Eds. Springer Berlin Heidelberg, 2002, vol. 2355, pp. 78–91.

[And62] T. W. Anderson, "On the distribution of the two-sample Cramer-von Mises criterion," *The Annals of Mathematical Statistics*, pp. 1148–1159, 1962.

[Bar06] G. V. Bard, "A Challenging but Feasible Blockwise-Adaptive Chosen-Plaintext Attack on SSL." in *SECRYPT*, 2006, pp. 99–109.

[BBM14] T. Bianchi, V. Bioglio, and E. Magli, "On the security of random linear measurements," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 3992–3996.

[BBM16] ——, "Analysis of One-Time Random Projections for Privacy Preserving Compressed Sensing," *Information Forensics and Security, IEEE Transactions on*, vol. 11, no. 2, pp. 313–327, 2016.

[BCG+14] J. Buchmann, D. Cabarcas, F. Göpfert, A. Hülsing, and P. Weiden, "Discrete Ziggurat: A Time-Memory Trade-Off for Sampling from a Gaussian Distribution over the Integers," in *Selected Areas in Cryptography – SAC 2013: 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, T. Lange, K. Lauter, and P. Lisoněk, Eds. Springer Berlin Heidelberg, 2014, pp. 402–417.

[BCK96] M. Bellare, R. Canetti, and H. Krawczyk, "Keying Hash Functions for Message Authentication," in *Advances in Cryptology — CRYPTO '96*, G. Goos, J. Hartmanis, J. van Leeuwen, and N. Koblitz, Eds. Springer Berlin Heidelberg, 1996, vol. 1109, pp. 1–15.

[BDDH11] R. Baraniuk, M. A. Davenport, M. F. Duarte, and C. Hegde, *An Introduction to Compressive Sensing*. Connexions e-textbook, 2011.

[BDDW08] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "A Simple Proof of the Restricted Isometry Property for Random Matrices," *Constructive Approximation*, vol. 28, no. 3, pp. 253–263, Dec. 2008.

[BDJR97] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, "A concrete security treatment of symmetric encryption," in *38th Annual Symposium on Foundations of Computer Science, Proceedings*. IEEE, 1997, pp. 394–403.

[BDPA11a] G. Bertoni, J. Daemen, M. Peeters, and G. Assche, "The keccak reference," *Submission to NIST (Round 3), available online http://keccak.noekeon.org/Keccak-reference-3.0.pdf*, 2011, last accessed 20.02.2017.

[BDPA11b] ——, "On the security of the keyed sponge construction," *Symmetric Key Encryption Workshop (SKEW)*, 2011.

[BDPVA12] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, "Permutation-based encryption, authentication and authenticated encryption," *Directions in Authenticated Ciphers*, 2012.

[Bel06] M. Bellare, "New proofs for NMAC and HMAC: Security without collision-resistance," in *Advances in Cryptology-CRYPTO 2006*. Springer Berlin Heidelberg, 2006, pp. 602–619.

[Ber09] D. J. Bernstein, "Cost analysis of hash collisions : Will quantum computers make SHARCS obsolete?" *SHARCS'09 Workshop Record, Proceedings 4th Workshop on Special-purpose Hardware for Attacking Cryptograhic Systems, Lausanne, Switzerland, September 9-10*, pp. 105–116, 2009.

[BKL+09] M. Bellare, T. Kohno, S. Lucks, N. Ferguson, B. Schneier, D. Whiting, J. Callas, and J. Walker, "Provable security support for the Skein hash family," *available online http://www.skein-hash.info/sites/default/files/skein-proofs.pdf*, 2009, last accessed 02.02.2017.

[BKN02] M. Bellare, T. Kohno, and C. Namprempre, "Authenticated encryption in SSH: Provably fixing the SSH binary packet protocol," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM, 2002, pp. 1–11.

[BKR98] M. Bellare, T. Krovetz, and P. Rogaway, "Luby-Rackoff backwards: Increasing security by making block ciphers non-invertible," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer Berlin Heidelberg, 1998, pp. 266–280.

[BKR00] M. Bellare, J. Kilian, and P. Rogaway, "The Security of the Cipher Block Chaining Message Authentication Code," *Journal of Computer and System Sciences*, vol. 61, no. 3, pp. 362–399, Dec. 2000.

[BL11] K. Bredies and D. Lorenz, *Mathematische Bildverarbeitung*. Wiesbaden: Vieweg+Teubner, 2011.

[BMZA12] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things." Proceedings of the first

edition of the MCC workshop on Mobile cloud computing, ACM Press, 2012, pp. 13–16.

[BN00] M. Bellare and C. Namprempre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," in *Advances in Cryptology — ASIACRYPT 2000*, G. Goos, J. Hartmanis, J. van Leeuwen, and T. Okamoto, Eds. Springer Berlin Heidelberg, 2000, vol. 1976, pp. 531–545.

[BN08] ——, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," *Journal of Cryptology*, vol. 21, no. 4, pp. 469–491, Oct. 2008.

[BR04] M. Bellare and P. Rogaway, "Code-Based Game-Playing Proofs and the Security of Triple Encryption." *Cryptology ePrint Archive, Report 2004/331*, 2004.

[BZD+16] H. Böck, A. Zauner, S. Devlin, J. Somorovsky, and P. Jovanovic, "Nonce-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS," *Cryptology ePrint Archive, Report 2016/475*, 2016.

[CAE17] CAESAR, "CAESAR submissions," https://competitions.cr.yp.to/caesar-submissions.html, 2017, last accessed 02.02.2017.

[CDS01] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[Che09] L. Chen, "Recommendation for key derivation using pseudorandom functions," *NIST Special Publication 800-108*, 2009.

[CHP+13] V. Cambareri, J. Haboba, F. Pareschi, H. R. Rovatti, G. Setti, and K.-W. Wong, "A two-class information concealing system based on compressed sensing," in *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on.* IEEE, 2013, pp. 1356–1359.

[CMP+15a] V. Cambareri, M. Mangia, F. Pareschi, R. Rovatti, and G. Setti, "Low-Complexity Multiclass Encryption by Compressed Sensing,"

Signal Processing, IEEE Transactions on, vol. 63, no. 9, pp. 2183–2195, May 2015.

[CMP+15b] ——, "On Known-Plaintext Attacks to a Compressed Sensing-Based Encryption: A Quantitative Analysis," *Information Forensics and Security, IEEE Transactions on*, vol. 10, no. 10, pp. 2182–2195, Oct. 2015.

[CPB+12] S.-j. Chang, R. Perlner, W. E. Burr, M. Sonmez Turan, J. M. Kelsey, S. Paul, and L. E. Bassham, "Third-Round Report of the SHA-3 Cryptographic Hash Algorithm Competition," *NIST IR 7896*, Nov. 2012.

[CRT06] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, 2006.

[CT05] E. J. Candès and T. Tao, "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, no. 12, pp. 4203–4215, 2005.

[CT06] ——, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5406–5425, 2006.

[Dam90] I. B. Damgård, "A Design Principle for Hash Functions," in *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, Ed. Springer New York, 1990, pp. 416–427.

[Dev86] L. Devroye, *Non-Uniform Random Variate Generation*. Springer New York, 1986, ISBN: 978-1-4613-8645-2.

[DG14] N. C. Dwarakanath and S. D. Galbraith, "Sampling from discrete Gaussians for lattice-based cryptography on a constrained device," *Applicable Algebra in Engineering, Communication and Computing*, vol. 25, no. 3, pp. 159–180, Jun. 2014.

[Don04] D. L. Donoho, "For most large underdetermined systems of equations the minimal l1-norm near-solution approximates the sparsest near-solution," *Preprint, available online at http://statweb.stanford.edu/~donoho/Reports/2004/l1l0approx.pdf*, 2004, last accessed 27.09.2016.

[Don06] D. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, 2006.

[Dwo01] M. Dworkin, "NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation," 2001.

[Dwo07] M. J. Dworkin, "Recommendation for block cipher modes of operation :: GaloisCounter Mode (GCM) and GMAC," National Institute of Standards and Technology, Tech. Rep. NIST SP 800-38d, 2007.

[Fay16] R. Fay, "Introducing the counter mode of operation to Compressed Sensing based encryption," *Information Processing Letters*, vol. 116, no. 4, pp. 279 – 283, 2016.

[FBK+17] R. Fay, C. Bender, N. Klein, D. Krönert, F. Kußmaul, S. Merz, T. Pieper, J. Saßmannshausen, T. Steinbring, and C. Wurmbach, "Gefahrlos durch den Nebel - Ein Sicherheitskonzept für das Fog Computing," *15. Deutscher IT-Sicherheitskongress, BSI, Bonn*, May 2017.

[Fer05] N. Ferguson, "Authentication weaknesses in GCM," *Comments submitted to NIST Modes of Operation Process, available online http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/Ferguson2.pdf*, 2005, last accessed 02.02.2017.

[FLS+10] N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker, "The Skein hash function family," *Submission to NIST (round 3)*, vol. 7, no. 7.5, 2010.

[Fol14] J. Folláth, "Gaussian Sampling in Lattice Based Cryptography," *Tatra Mountains Mathematical Publications*, vol. 60, no. 1, pp. 1–23, Jan. 2014.

[FPRU10] S. Foucart, A. Pajor, H. Rauhut, and T. Ullrich, "The Gelfand widths of ℓ_p-balls for $0 < p \leq 1$," *Journal of Complexity*, vol. 26, no. 6, pp. 629–640, Dec. 2010.

[FR13] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*, ser. Applied and Numerical Harmonic Analysis. Springer New York, 2013.

[FR16] R. Fay and C. Ruland, "Compressive Sensing Encryption Modes and their Security," *Proceedings of the 11th International Conference for Internet Technology and Secured Transactions (ICITST-2016), Barcelona, Spain*, pp. 119–126, 2016.

[FR17] ——, "Compressed Sampling and Authenticated-Encryption," *11th International ITG Conference on Systems, Communications and Coding (SCC-2017), Hamburg, Germany*, 2017.

[GGM85] O. Goldreich, S. Goldwasser, and S. Micali, "On the Cryptographic Applications of Random Functions (Extended Abstract)," in *Advances in Cryptology: Proceedings of CRYPTO 84*, G. R. Blakley and D. Chaum, Eds. Springer Berlin Heidelberg, 1985, pp. 276–288.

[GGM86] ——, "How to construct random functions," *Journal of the ACM*, vol. 33, no. 4, pp. 792–807, Aug. 1986.

[GJM+15] B. Guido, D. Joan, P. Michaël, V. A. Gilles, and V. K. Ronny, "Keyak v2," *Submission to CAESAR, http://keyak.noekeon.org/Keyak-2.1.pdf*, 2015, last accessed on 30.08.2016.

[GM82] S. Goldwasser and S. Micali, "Probabilistic encryption & how to play mental poker keeping secret all partial information," in *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*. ACM, 1982, pp. 365–377.

[Goo16a] Google, “Protocol Buffers,” https://developers.google.com/protocol-buffers/, 2016, last accessed 16.09.2016.

[Goo16b] ——, “Protocol Buffers, Language Guide,” https://developers.google.com/protocol-buffers/docs/proto, 2016, last accessed 16.09.2016.

[GPVD99] E. Guttman, C. Perkins, J. Veizades, and M. Day, “RFC 2608 - Service Location Protocol, Version 2,” *https://www.ietf.org/rfc/rfc2608.txt*, Jun. 1999, last accessed 14.09.2016.

[Gro97] L. K. Grover, “Quantum Mechanics Helps in Searching for a Needle in a Haystack,” *Physical Review Letters*, vol. 79, no. 2, pp. 325–328, Jul. 1997.

[Hod58] J. L. Hodges, “The significance probability of the smirnov two-sample test,” *Arkiv för matematik*, vol. 3, no. 5, pp. 469–486, Jan. 1958.

[HRU14] R. Huang, K. Rhee, and S. Uchida, “A parallel image encryption method based on compressive sensing,” *Multimedia Tools and Applications*, vol. 72, no. 1, pp. 71–93, 2014.

[IEE08] IEEE, “Standard for Floating-Point Arithmetic 754-2008,” Tech. Rep., 2008.

[ISO00] ISO, “ISO/IEC 10118-1:2000 Information technology - Security techniques - Hash-functions - Part 1: General,” Jun. 2000.

[ISO04] ——, “ISO/IEC 10118-3:2004 Information technology - Security techniques - Hash-functions - Part 3: Dedicated hash-functions,” Mar. 2004.

[ISO06] ——, “ISO/IEC 10116:2006 Information technology - Security techniques - Modes of operation for an n-bit block cipher,” Feb. 2006.

[ISO09] ——, “ISO/IEC 19772:2009 Information technology - Security techniques - Authenticated encryption,” Feb. 2009.

[ISO11] ——, "ISO/IEC 18031:2011 Information technology - Security techniques - Random bit generation," Nov. 2011.

[ISO16] ISO/IEC JTC1/SC27, "Standing Document (SD) 6 - Glossary of IT Security Terminology," Apr. 2016.

[JNP15] J. Jean, I. Nikolić, and T. Peyrin, "Deoxys v1.3," *Submission to CAESAR, http://competitions.cr.yp.to/round2/deoxysv13.pdf*, Aug. 2015, last accessed on 30.08.2016.

[KL15] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, 2nd ed., ser. Chapman & Hall/CRC cryptography and network security. CRC Press, Taylor & Francis, 2015.

[KM03] H.-J. Kowalsky and G. O. Michler, *Lineare Algebra*. Berlin, New York: Walter de Gruyter, Jan. 2003.

[Kot33] V. A. Kotelnikov, *On the transmission capacity of "ether" and wire in electrocommunications*. Translated and reprinted in *Modern Sampling Theory: Mathematics and Applications*, Birkhauser Boston, Applied and Numerical Harmonic Analysis Series, 2001, 1933, pp. 27–47.

[KR11] T. Krovetz and P. Rogaway, "The Software Performance of Authenticated-Encryption Modes," in *Fast Software Encryption*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and A. Joux, Eds. Springer Berlin Heidelberg, 2011, vol. 6733, pp. 306–327.

[KR14] ——, "OCB (v1)," *Submission to CAESAR, https://competitions.cr.yp.to/round1/ocbv1.pdf*, 2014, last accessed on 30.08.2016.

[Li09] C. Li, "An efficient algorithm for total variation regularization with applications to the single pixel camera and compressive sensing," Ph.D. dissertation, Rice University, 2009.

[LLY+14] X. Li, X. Lan, M. Yang, J. Xue, and N. Zheng, "Efficient Lossy Compression for Compressive Sensing Acquisition of Images in Compressive Sensing Imaging Systems," *Sensors*, vol. 14, no. 12, pp. 23 398–23 418, Dec. 2014.

[MBZJ09] H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A Fast Approach for Overcomplete Sparse Decomposition Based on Smoothed ℓ^0 Norm," *IEEE Transactions on Signal Processing*, vol. 57, no. 1, pp. 289–301, Jan 2009.

[Mer90] R. C. Merkle, "A Certified Digital Signature," in *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, Ed. Springer New York, 1990, pp. 218–238.

[Min15] K. Minematsu, "AES-OTR v2," *Submission to CAESAR, https://competitions.cr.yp.to/round2/aesotrv2.pdf*, Aug. 2015, last accessed on 30.08.2016.

[MV04a] D. McGrew and J. Viega, "The Galois/counter mode of operation (GCM)," *Submission to NIST*, 2004.

[MV04b] D. A. McGrew and J. Viega, "The security and performance of the Galois/Counter Mode (GCM) of operation (full version)," *Cryptology ePrint Archive, Report 2004/193*, 2004.

[NIS14] NIST, "Permutation-Based Hash and Extendable-Output Functions," *NIST FIPS 202*, 2014.

[NRS14] C. Namprempre, P. Rogaway, and T. Shrimpton, "Reconsidering Generic Composition," in *Advances in Cryptology – EUROCRYPT 2014*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, A. Kobsa, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, D. Terzopoulos, D. Tygar, G. Weikum, P. Q. Nguyen, and E. Oswald, Eds. Springer Berlin Heidelberg, 2014, vol. 8441, pp. 257–274.

[NW16] Y. Naito and L. Wang, "Replacing SHA-2 with SHA-3 Enhances Generic Security of HMAC." in *Topics in Cryptology - CT-RSA*

2016, K. Sako, Ed. Springer International Publishing, 2016, vol. 9610, pp. 397–412.

[OASB08] A. Orsdemir, H. Altun, G. Sharma, and M. Bocko, "On the security and robustness of encryption via compressed sensing," in *Military Communications Conference, 2008. MILCOM 2008. IEEE*, 2008, pp. 1–7.

[Piv16] Pivotal Software, "RabbitMQ," http://www.rabbitmq.com/, 2016, last accessed 16.09.2016.

[RB08] Y. Rachlin and D. Baron, "The secrecy of compressed sensing measurements," in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, 2008, pp. 813–817.

[Rog02] P. Rogaway, "Authenticated-encryption with associated-data," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM, 2002, pp. 98–107.

[Rog04] ——, "Nonce-Based Symmetric Encryption," in *Fast Software Encryption*, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, B. Roy, and W. Meier, Eds. Springer Berlin Heidelberg, 2004, vol. 3017, pp. 348–358.

[Rog11] ——, "Evaluation of some blockcipher modes of operation," *Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan*, 2011.

[RS07] P. Rogaway and T. Shrimpton, "The SIV mode of operation for deterministic authenticated-encryption (key wrap) and misuse-resistant nonce-based authenticated-encryption," *available online http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/siv/siv.pdf*, Aug. 2007, last accessed 20.02.2017.

[Sha49a] C. Shannon, "Communication in the Presence of Noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, Jan. 1949.

[Sha49b] C. E. Shannon, "Communication theory of secrecy systems*," *Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949.

[Sho04] V. Shoup, "Sequences of games: A tool for taming complexity in security proofs." *Cryptology ePrint Archive, Report 2004/332*, 2004.

[SRVV14] S. Sinha Roy, F. Vercauteren, and I. Verbauwhede, "High Precision Discrete Gaussian Sampling on FPGAs," in *Selected Areas in Cryptography – SAC 2013: 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, T. Lange, K. Lauter, and P. Lisoněk, Eds. Springer Berlin Heidelberg, 2014, pp. 383–401.

[TG07] J. A. Tropp and A. C. Gilbert, "Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[TP14] A. M. Tillmann and M. E. Pfetsch, "The Computational Complexity of the Restricted Isometry Property, the Nullspace Property, and Related Concepts in Compressed Sensing," *IEEE Transactions on Information Theory*, vol. 60, no. 2, pp. 1248–1259, Feb. 2014.

[Tsu92] G. Tsudik, "Message authentication with one-way hash functions," in *INFOCOM '92. Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE*, May 1992, pp. 2055–2059.

[Tur08] J. M. Turner, "The keyed-hash message authentication code (HMAC)," *Federal Information Processing Standards Publication (FIPS) 198-1*, 2008.

[TZ12] S. A. H. A. E. Tabatabaei and N. Zivic, "Security analysis of the joint encryption and compressed sensing," in *20th Telecommunications Forum (TELFOR)*, 2012, pp. 799–802.

[TZ14] ——, "Revisiting a primitive: Analysis of approximate message authentication codes," in *, International Conference on Communications (ICC), IEEE*, Jun. 2014, pp. 743–748.

[Vaa00] Vaart, A.W. van der, *Asymptotic Statistics*, 1st ed., ser. Cambridge series in statistical and probabilistic mathematics. Cambridge Univ. Press, 2000.

[WFH03] D. Whiting, N. Ferguson, and R. Housley, "Counter with CBC-MAC (CCM)," *RFC3610, https://tools.ietf.org/rfc/rfc3610.txt*, Sep. 2003, last accessed 02.02.2017.

[Whi35] J. M. Whittaker, "Interpolatory function theory," *Cambridge Tracts in Mathematics and Mathematical Physics vol. 33*, 1935.

[WLD+06] M. B. Wakin, J. N. Laska, M. F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "An architecture for compressive imaging," in *Image Processing, 2006 IEEE International Conference on*. IEEE, 2006, pp. 1273–1276.

[ZWLZ14] L. Y. Zhang, K.-W. Wong, C. Li, and Y. Zhang, "Towards Secure Compressive Sampling Scheme," *arXiv preprint arXiv:1406.1725*, 2014.

[ZWX+14] Y. Zhang, K.-W. Wong, D. Xiao, L. Y. Zhang, and M. Li, "Embedding Cryptographic Features in Compressive Sensing," *arXiv preprint arXiv:1403.6213*, 2014.

[ZZZ+16] Y. Zhang, L. Y. Zhang, J. Zhou, L. Liu, F. Chen, and X. He, "A Review of Compressive Sensing in Information Security Field," *IEEE Access*, vol. 4, pp. 2507–2519, 2016.