

Development of an IP cryptographic function for the HF radio environment

DE STREICHER

 [orcid.org/ 0000-0002-8290-2037](https://orcid.org/0000-0002-8290-2037)

Dissertation accepted in fulfilment of the requirements for the
degree *Master of Engineering in Computer and Electronic
Engineering* at the North-West University

Supervisor: Prof ASJ Helberg

Co-supervisor: Dr M Ferreira

Graduation: April 2024

Declaration

I, David Ekhard Streicher hereby declare that the project report entitled “Development of an IP cryptographic function for the HF radio environment” is my own original work and has not already been submitted to any other university or institution for examination.

D.E. Streicher

Student number: 24167738

Signed on the 5th day of December 2023 in Pretoria.

Acknowledgements

*The earth is the LORD's, and everything in it.
The world and all its people belong to Him.
For He laid the earth's foundation on the seas
and built it on the ocean depths.*

- Psalm 24:1-2, New Living Translation

Firstly, I would like to thank my Lord and Saviour, Jesus Christ. His constant presence and guidance is more valuable than any earthly treasure. Psalm 24 serves as a reminder that at the end of the day, everything I am and everything I do is ultimately for Him and belongs to Him.

Second, to my supervisors, Professor Albert Helberg and Doctor Melvin Ferreira, thank you for your constant support since the inception of this research study. It would not be what it is today without your feedback and guidance to keep things on track over the last few years.

To the special lady in my life, Marinique Roos, thank you for every sacrifice you have made over the last months, for every bit of support and encouragement along the way, and for being patient with me in the tough times - it's been an adventure!

To my housemates, Braam Visser and Leslie de Jager, thank you for picking up a lot of life admin on my behalf, especially towards the end.

To my work colleagues at Rapid Mobile,

Jaco Boshoff, thank you for your leadership and technical and academic guidance throughout the course of this study, and every iteration of it before this final work;

Rainer Oellermann, thank you for your invaluable insights into the workings of the STANAG 5066 protocol stack;

and to every colleague who supported me throughout, especially in the times when part-time studies became full-time work - thank you!

Lastly, to Rapid Mobile, thank you for giving me the opportunity to further my studies, and for every bit of support along the way, financial and otherwise. I am truly blessed to work at a company like this.

Abstract

Provisioning Internet Protocol (IP) services over military High Frequency (HF) radio networks has been of interest from at least as early as 1999. Historically, this has only been achieved with limited success for User Datagram Protocol (UDP) based IP services due to the high latency and low bandwidth of HF channels. However, emerging Wideband High Frequency (WBHF) technology holds promise as a key enabler for this application domain, creating a renewed drive to develop a workable IP over HF system.

An aspect to consider in this system design is that of network security. While Annex E of the North Atlantic Treaty Organization (NATO) Standardization Agreement (STANAG) 5070 mandates the provisioning of security at the IP layer, little else is disclosed in the public domain. This presents a unique opportunity to develop an IP cryptographic function for the HF radio environment.

This is approached by firstly investigating related subject matter on topics such as the HF transmission medium in general, traditional data transmission over an HF channel, the history of IP data communication over HF, traditional security mechanisms in the HF military environment, and security in traditional IP networks.

A design and implementation of an IP cryptographic function, or IP crypto for short, is presented in the context of an envisioned IP over HF system. The design draws on information obtained in the literature study and from existing HF industry standards, and when necessary, compliance with HF industry standards is given preference when making design decisions. The implementation is done using the Linux Transform (XFRM) framework. It is suitable for the evaluation purposes of this research study and is not intended to be a fully developed IP crypto product ready for commercial use in the HF military environment.

This IP crypto is evaluated using an IP over WBHF capable test bench consisting of Commercial Off-The-Shelf (COTS) equipment, including WBHF modems and an HF channel simulator. This process consists of developing throughput- and latency tests by modifying the open source IPERF3 network performance evaluation tool, using these tests to establish a performance baseline of the test bench, viz. without the IP crypto included in the system, repeating the tests with the IP crypto included in the system, and lastly, comparing the IP crypto results to the baseline test results.

This comparative analysis quantifies the extent of the IP crypto's impact on the IP over WBHF test bench system's performance, and finds that this is more pronounced for the throughput rate than for the latency characteristics of the system. Additionally, while the presence of the IP crypto may influence the viability of some IP services over HF, it in no way renders the system completely unusable for IP services.

Keywords: *Cryptography, HF, HF-IP, Internet Protocol (IP), IPsec*

Contents

List of Figures	ix
List of Tables	xi
List of Acronyms	xii
1 Introduction	1
1.1 The landscape of IP services in the HF military industry	1
1.2 The case for developing an IP cryptographic function for the HF radio environment	1
1.3 Research study goals	2
1.4 Research method and dissertation overview	3
1.5 Concluding remarks	4
2 Literature study	5
2.1 Motivation for the structure of the literature study	5
2.2 Overview of the High Frequency (HF) transmission medium	6
2.2.1 Usage rationale and Application domain	6
2.2.2 Typical components of an HF system	6
2.2.3 Characteristics of an HF channel	8
2.2.3.1 Surface-wave propagation characteristics	8
2.2.3.2 Skywave propagation characteristics	8
2.2.3.3 Channel models used for simulation and benchmark testing	9
2.3 Transmitting data over an HF channel	10
2.3.1 Definition of an HF waveform	10
2.3.2 Relation to the Open Systems Interconnect (OSI) model	10
2.3.3 Conceptual data flow at the physical layer	11
2.3.4 Coding	11
2.3.5 Interleaving	13
2.3.6 Modulation	14
2.3.6.1 Disambiguation of terminology	14
2.3.6.2 Theoretical discussion	14
2.3.7 Higher layer protocols	17
2.3.8 Narrowband and Wideband High Frequency (HF)	19
2.4 Transmitting IP data over HF	21
2.4.1 History of IP services in the HF industry	21
2.4.1.1 Driving research factors	21
2.4.1.2 Problematic characteristics of HF for IP data transmission	21
2.4.1.3 Typical solution domains considered in IP over HF work	22
2.4.1.4 Security considerations	23
2.5 Security for HF communication	23
2.5.1 Security terminology used in the HF domain	23
2.5.2 COMSEC in an HF network	24
2.5.2.1 Scarcity of public information regarding COMSEC in an HF network	24
2.5.2.2 The partial exception of STANAG 5066 Edition 4	25
2.5.2.2.1 Definitions of service interfaces	26

2.5.2.2.2	Crypto mapping and counter mode	27
2.5.2.2.3	Crypto initialisation	28
2.5.2.2.4	Crypto layer protocol	29
2.5.3	TRANSEC in an HF network	31
2.5.4	A note on Linking Protection (LP)	31
2.6	Security for IP networks	32
2.6.1	Common security protocols	32
2.6.2	Internet Protocol Security (IPsec)	32
2.6.2.1	Overview of the IPsec protocol suite	32
2.6.2.2	Modes of operation	33
2.6.2.3	Base IPsec architecture	35
2.6.2.3.1	Outbound IP traffic IPsec processing	35
2.6.2.3.2	Inbound IP traffic IPsec processing	38
2.6.2.3.3	Structure of SPD and SAD entries	41
2.6.2.3.4	A note on the PAD	45
2.6.2.4	IPsec security protocols	46
2.6.2.4.1	Encapsulating Security Payload (ESP) header format and location	47
2.6.2.4.2	Outbound ESP processing	50
2.6.2.4.3	Inbound ESP processing	52
2.6.2.5	IPsec SA management methods and protocols	53
2.6.2.6	IPsec extensions	54
2.6.3	Other IP packet encryption options	54
2.7	Concluding remarks	55
3	Design and implementation of the IP cryptographic function	56
3.1	The case for an IP cryptographic function for the HF environment	56
3.2	A potential IP over HF system including a IP cryptographic function	56
3.2.1	IP over HF system design goals	56
3.2.2	IP over HF system architecture discussion	57
3.2.3	IP cryptographic function boundary	61
3.2.4	A note on the functionality provided by the various Performance Enhancing Proxys (PEPs)	61
3.3	IP crypto function requirements and design	61
3.3.1	Design inputs and requirement sources	61
3.3.2	Notable design exclusions	62
3.3.3	Analysis of STANAG 5070 Annex E	62
3.3.4	Functional design decisions	63
3.3.4.1	Limit support to Internet Protocol Version 4 (IPv4)	63
3.3.4.2	No packet queues or buffering	63
3.3.4.3	Red side LAN support	64
3.3.4.4	ESP protocol use	64
3.3.4.5	Static SA use	64
3.3.4.6	No form of IP crypto bypass	64
3.3.4.7	No Internet Control Message Protocol (ICMP) traffic processing	64
3.3.4.8	No fragmentation support	65
3.3.4.9	No Peer Authorization Database (PAD) support	65
3.3.4.10	No IPsec multicast extension support	65
3.3.4.11	Last remarks	65

3.4	IP crypto implementation options	65
3.4.1	Primary implementation goal	65
3.4.2	Selecting a suitable implementation option	66
3.5	Using the Linux XFRM framework to implement the IP crypto function	66
3.5.1	Preventing automatic SA establishment	67
3.5.2	Static Security Association Database (SAD) configuration	67
3.5.3	Static Security Policy Database (SPD) configuration	68
3.5.4	Linking back to the research study implementation goal	70
3.6	Concluding remarks	71
4	Development of the test bench setup	72
4.1	Overview of the test development process	72
4.2	Primary requirements of the testing setup	73
4.3	Test setup architecture and configuration	73
4.3.1	Possible testing approaches	73
4.3.2	Test bench architecture	74
4.3.3	Test bench configuration	76
4.3.3.1	Baseline RapidM hardware configuration	76
4.3.3.2	Motivation for baseline configuration choices	77
4.3.4	Test bench IP data flow	79
4.3.5	Verification of the IP crypto implementation	82
4.3.5.1	Normal ping test without any XFRM configuration	83
4.3.5.2	Basic XFRM configuration tests	83
4.3.5.3	XFRM configuration test with block all rule	84
4.3.5.4	XFRM configuration test with the final set of SPD entries	84
4.4	Test data generation and result measurement	85
4.4.1	Measurement goals and test design decisions	85
4.4.1.1	Testing goal and performance metrics	85
4.4.1.2	Transport layer protocol	85
4.4.1.3	Effect on packet loss	86
4.4.2	Selecting a suitable TCP/IP test traffic tool	86
4.4.3	High-level overview of iperf3 operation	87
4.4.4	General modifications to enable iperf3 use over an HF channel	87
4.4.5	iperf3 TCP-based latency test development	88
4.4.5.1	Modifications related to the TCP RTT measurements	89
4.4.5.2	One-way latency and jitter test development	89
4.4.6	iperf3 TCP-based latency test validation	90
4.4.7	iperf3 Throughput test overview	91
4.4.8	A last note on packet loss	91
4.5	Concluding remarks	92
5	IP cryptographic function performance impact test results	93
5.1	Test execution overview	93
5.2	Test verification and baseline results	95
5.2.1	Expected Over The Air (OTA) throughput rates	95
5.2.2	Test bench hardware and iperf3 configuration parameters	96
5.2.3	Baseline throughput results	98
5.2.4	Baseline latency results	100
5.2.4.1	Initial HF link setup RTT results	100
5.2.4.2	One-way latency results	101

5.2.4.3	One-way jitter results	104
5.2.4.4	TCP RTT results	104
5.2.4.5	Clarification of the seeming TCP RTT results anomaly	106
5.2.5	Baseline result verification	108
5.2.6	Impact of the STANAG 5066 DRC algorithm	111
5.2.6.1	The 12 kHz “poor” channel throughput anomaly test	111
5.2.6.2	The Bad SNR versus Average SNR latency anomaly test	112
5.3	IP crypto results and analysis	113
5.3.1	Important notes of the IP crypto functionality	113
5.3.2	IP crypto throughput rate results	113
5.3.2.1	Result presentation and verification	113
5.3.2.2	Validation	114
5.3.3	IP crypto latency results	114
5.3.3.1	Initial HF link setup RTT results	115
5.3.3.1.1	Result presentation and verification	115
5.3.3.1.2	Validation	115
5.3.3.2	One-way latency and jitter results	116
5.3.3.2.1	Result presentation and verification	116
5.3.3.2.2	Validation	117
5.3.3.3	TCP RTT results	117
5.3.3.3.1	Result presentation and verification	117
5.3.3.3.2	Validation	118
5.3.4	Performance impact analysis of the IP cryptographic function	118
5.3.4.1	Throughput rate analysis	118
5.3.4.2	Initial HF link setup RTT analysis	120
5.3.4.3	One-way latency and jitter analysis	120
5.3.4.4	TCP RTT analysis	122
5.3.5	Clarification for IP crypto result anomalies	123
5.3.5.1	STANAG 5066 DRC effects on throughput results	123
5.3.5.2	STANAG 5066 behaviour for small payloads	123
5.4	Concluding remarks	126
6	Conclusion	127
6.1	Overview of the research study	127
6.1.1	Motivation for an IP cryptographic function in the HF domain	127
6.1.2	Research study goals	127
6.1.3	Research methodology	128
6.2	Addressing the research study goals and IP crypto performance impact findings	129
6.3	Research Contributions	130
6.4	Potential future work	130
6.5	Closing remarks	131
	Bibliography	132
	Appendices	
A	Baseline latency result sample analysis and visualisation	139
A.1	Overview	139
A.2	One-way latency	139
A.3	TCP RTT	143

B	IP crypto latency result sample analysis and visualisation	147
B.1	Overview	147
B.2	One-way latency	147
B.3	TCP RTT	151

List of Figures

2.1	Typical HF Transmitter Components (adapted from [13])	7
2.2	Typical HF Receiver Components (adapted from [13])	8
2.3	Example of an HF House model (taken from STANAG 4539 Annex A [23])	10
2.4	Binary data flow in an HF system (adapted from [13])	11
2.5	Error Detection Concept (adapted from [29])	12
2.6	Examples of constellation diagrams	16
2.7	Inter-relationships between the primary ARCS constituent processes [26]	17
2.8	STANAG 5066 Crypto Architecture (adapted from [5])	26
2.9	Crypto Sync Protocol Data Unit (PDU) as specified in STANAG 5066 Annex T	29
2.10	Extended Crypto Sync PDU as specified in STANAG 5066 Annex T	30
2.11	Crypto Sync PDU for Advanced Encryption Standard (AES) as specified in STANAG 5066 Annex T	30
2.12	Interrelationship of IPsec and IKE documents (adapted from RFC 6071 [54])	33
2.13	IPsec Transport Mode (adapted from [29])	34
2.14	IPsec Tunnel Mode (adapted from [29])	34
2.15	High level IPsec processing model (taken from section 3.1 in RFC 4301 [8])	35
2.16	Outbound IP traffic IPsec processing model (adapted from section 5.1 in RFC 4301 [8])	36
2.17	Inbound IP traffic IPsec processing model (adapted from section 5.2 in RFC 4301 [8])	39
2.18	ESP Header location in IPv4 packet - IPsec Transport mode (adapted from section 3.1 in Request for Comments (RFC) 4303 and relevant errata [56])	47
2.19	ESP Header location in IPv4 packet - IPsec Tunnel mode (adapted from section 3.1 in RFC 4303 [56])	48
2.20	ESP Packet Format (adapted from section 2 of RFC 4303 [56])	49
3.1	High-level IP over HF system diagram	57
3.2	High-level IP service node architecture	58
3.3	High-level Linux PC setup for XFRM framework evaluation	67
4.1	High-level test bench block diagram	74
4.2	High-level Linux PC setup	80
4.3	STANAG 5066 Edition (Ed) 4 Layer Overview (optional layers omitted; adapted from [5])	81
4.4	High-level Linux PC setup for XFRM framework evaluation	82
4.5	iperf3 Test Connections	87
4.6	iperf3 TCP based latency test payload	89
5.1	General test execution flow	93
5.2	Test initialisation sequence - expanded	94
5.3	Latency test run flow	94
5.4	Throughput test run flow	94
5.5	One-way latency sample distribution for 12 kHz, AWGN channel, Good SNR test conditions	103
5.6	One-way latency sample distribution for 12 kHz, "Poor" channel, Bad SNR test conditions	103

A.1	Cumulative sample count: Baseline One-way latency, 12 kHz, AWGN channel model	140
A.2	Cumulative sample count: Baseline One-way latency, 12 kHz, "Poor" channel model	141
A.3	Cumulative sample count: Baseline One-way latency, 18 kHz, AWGN channel model	141
A.4	Cumulative sample count: Baseline One-way latency, 18 kHz, "Poor" channel model	142
A.5	Cumulative sample count: Baseline TCP RTT, 12 kHz, AWGN channel model . . .	144
A.6	Cumulative sample count: Baseline TCP RTT, 12 kHz, "Poor" channel model . . .	145
A.7	Cumulative sample count: Baseline TCP RTT, 18 kHz, AWGN channel model . . .	145
A.8	Cumulative sample count: Baseline TCP RTT, 18 kHz, "Poor" channel model . . .	146
B.1	Cumulative sample count: IP crypto One-way latency, 12 kHz, AWGN channel model	148
B.2	Cumulative sample count: IP crypto One-way latency, 12 kHz, "Poor" channel model	149
B.3	Cumulative sample count: IP crypto One-way latency, 18 kHz, AWGN channel model	149
B.4	Cumulative sample count: IP crypto One-way latency, 18 kHz, "Poor" channel model	150
B.5	Cumulative sample count: IP crypto TCP RTT, 12 kHz, AWGN channel model . . .	152
B.6	Cumulative sample count: IP crypto TCP RTT, 12 kHz, "Poor" channel model . . .	153
B.7	Cumulative sample count: IP crypto TCP RTT, 18 kHz, AWGN channel model . . .	153
B.8	Cumulative sample count: IP crypto TCP RTT, 18 kHz, "Poor" channel model . . .	154

List of Tables

2.1	Modulation used to obtain each data rate in STANAG 4539 [23]	15
2.2	Description of various STANAG 4538 ARCS processes [26]	18
2.3	Subset of WBHF data rates (<i>bps</i>) according to bandwidth and modulation scheme from MIL-STD-188-110D [24]	20
2.4	Structure of an SPD entry (based on section 4.4.1 of RFC 4301 [8])	42
2.5	Structure of an SAD entry (based on section 4.4.2.1 of RFC 4301 [8])	44
3.1	Description of various components in the IP service node architecture	59
4.1	Description of each component in the test bench setup	75
4.2	Software version information for each hardware device in the test bench	76
4.3	iperf3 Modification Verification - Throughput (<i>Mbps</i>) measurements	88
5.1	Version information of testing tools	95
5.2	Expected OTA Throughput rates (derived from MIL-STD-188-110C Annex D [40]; colloquial Signal to Noise Ratio (SNR) condition terms added)	95
5.3	RapidM Hardware Parameters	96
5.4	Throughput Test Parameters	97
5.5	Latency Test Parameters	98
5.6	Throughput Results (<i>bps</i>) - baseline (no IP crypto)	98
5.7	Test Bench Verification - Initial HF Link Setup RTT (seconds)	101
5.8	Test Bench Verification - One-way Latency (seconds)	102
5.9	Test Bench Verification - One-way Jitter (seconds)	104
5.10	Test Bench Verification - TCP RTT (seconds)	105
5.11	Expected OTA Throughput rates - next higher rate	108
5.12	Baseline Result Verification - Throughput rate (<i>bps</i>) comparison	109
5.13	Baseline Result Verification - Throughput rate difference and Latency comparison	110
5.14	Throughput (<i>bps</i>) investigation - DRC disabled, 1200 bps waveform	111
5.15	Latency investigation: One-way Latency (seconds) DRC disabled, 18 kHz "Poor" channel	112
5.16	Latency investigation: One-way Jitter (seconds) DRC disabled, 18 kHz "Poor" channel	112
5.17	Latency investigation: TCP RTT (seconds) DRC disabled, 18 kHz "Poor" channel	113
5.18	Throughput Results (<i>bps</i>) - with IP crypto	114
5.19	Initial HF Link Setup RTT (seconds) results - with IP crypto	115
5.20	One-way Latency (seconds) results - with IP crypto	116
5.21	One-way Jitter (seconds) results - with IP crypto	117
5.22	TCP RTT (seconds) results - with IP crypto	118
5.23	Throughput rate (<i>bps</i>) comparison: With/Without IP crypto	119
5.24	One-way Latency (seconds) comparison: With/Without IP crypto	120
5.25	One-way Jitter (seconds) comparison: With/Without IP crypto	121
5.26	TCP RTT (seconds) comparison: With/Without IP crypto	122
5.27	Throughput (<i>bps</i>) anomaly investigation: With/Without IP crypto DRC disabled, expected OTA rate waveform (2400 <i>bps</i>) used	123

5.28	One-way Latency (seconds) investigation: With/Without IP crypto DRC disabled, expected OTA rate waveforms used	124
5.29	TCP RTT (seconds) investigation: With/Without IP crypto DRC disabled, expected OTA rate waveforms used	124
5.30	One-way Latency (seconds) investigation: With/Without IP crypto 5K payload, DRC disabled, expected OTA rate waveforms used	125
5.31	One-way Jitter (seconds) investigation: With/Without IP crypto 5K payload, DRC disabled, expected OTA rate waveforms used	125
5.32	TCP RTT (seconds) investigation: With/Without IP crypto 5K payload, DRC disabled, expected OTA rate waveforms used	125
A.1	Baseline results - One-way latency (seconds) five-number summary with SD . . .	139
A.2	Baseline results - One-way latency (seconds) percentile values	140
A.3	Baseline results - TCP RTT (seconds) five-number summary with SD	143
A.4	Baseline results - Transmission Control Protocol (TCP) Round-Trip Time (RTT) (seconds) percentile values	144
B.1	IP crypto results - One-way latency (seconds) five-number summary with SD . . .	147
B.2	IP crypto results - One-way latency (seconds) percentile values	148
B.3	IP crypto results - TCP RTT (seconds) five-number summary with SD	151
B.4	IP crypto results - TCP RTT (seconds) percentile values	152

List of Acronyms

2G Second Generation

3G Third Generation

4G Fourth Generation

AAD Additional Authenticated Data

ACK Acknowledgement

ACL Access List

ACS Automatic Channel Selection

AES Advanced Encryption Standard

AH Authentication Header

ALE Automatic Link Establishment

ALM Automatic Link Maintenance

AM Amplitude Modulation

APP Application

ARCS Automatic Radio Control System

ARQ Automatic Repeat reQuest

ASCII American Standard Code For Information Interchange

ASYN Asynchronous

ATU Antenna Tuning Unit

AWGN Additive White Gaussian Noise

BER Bit Error Rate

BLOS Beyond Line-of-Sight

BPSK Binary Phase-Shift Keying

BRASS Broadcast maritime Rear link and Ship to Shore

BRIPES BRASS IP Enhanced System

BW Bandwidth

CAS Channel Access Sublayer

CCIR International Radio Consultative Committee (Comité Consultatif International pour la Radio)

CFTP Compressed File Transfer Protocol

COMSEC Communications Security

CONF Conference

COTS Commercial Off-The-Shelf

CRC Cyclic Redundancy Check

CSMA Carrier Sense Multiple Access

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

CTR Counter

DCE Data Circuit Terminating Equipment

DF Don't Fragment

DNS Domain Name System

DoS Denial of Service

DRC Data Rate Change

DSCP Differentiated Services Code Point

DSP Digital Signal Processor

DSPProxy Delay- and Disruption-Tolerant SOAP Proxy

DTE Data Terminal Equipment

DTS Data Transfer Sublayer

ECN Explicit Congestion Notification

Ed Edition

EIA Electronic Industries Alliance

EOM End of Message

EPM Electronic Protection Measures

ESN Extended Sequence Number

ESP Encapsulating Security Payload

ETH Ethernet

FEC Forward Error Correction

FF Fixed Frequency

FH Frequency Hopping

FIN Finish

FLSU Fast LSU

FQDN Fully Qualified Domain Name

FSK Frequency Shift Keying

GCM Galois/Counter Mode

HAIP High Assurance Internet Protocol Encryptor

HDL High-Throughput Data Link

HF High Frequency

HTTP Hypertext Transfer Protocol

ICMP Internet Control Message Protocol

ICV Integrity Check Value

ID Identifier

IFRB International Frequency Registration Board

IKE Internet Key Exchange

IP Internet Protocol

IPsec Internet Protocol Security

IPv4 Internet Protocol Version 4

IPv6 Internet Protocol Version 6

ISO International Standards Organisation

ITU International Telecommunication Union

ITU-R International Telecommunication Union Radiocommunication Sector

IV Initialization Vector

LAN Local Area Network

LDL Low-Latency Data Link

LFSR Linear Feedback Shift Register

LOS Line-of-Sight

LP Linking Protection

LPD Low Probability of Detection

LPI Low Probability of Intercept

LSU Link Setup

LTS Long Term Service

MAC Media Access Control

MCBSP Multichannel Buffered Serial Port

MH Mobility Header

MIL-STD Military Standard

MSS Maximum Segment Size

MTU Maximum Transmission Unit

NAT Network Address Translation

NATO North Atlantic Treaty Organization

NBHF Narrowband HF

NCW Network Centric Warfare

NetNS Network Namespace

NIC Network Interface Card

NINE Network and Information Infrastructure IP Network Encryption

NIST National Institute of Standards and Technology

NNEC NATO Network Enabled Capability

NSA National Security Agency

NVIS Near Vertical Incidence Skywave

OLSR Optimised Link State Routing

OS Operating System

OSI Open Systems Interconnect

OTA Over The Air

QoS Quality of Service

PA Power Amplifier

PAD Peer Authorization Database

PC Personal Computer

PDU Protocol Data Unit

PEP Performance Enhancing Proxy

PFP Populate from Packet

PKI Public Key Infrastructure

PMTU Path MTU

PU Participating Unit

PSK Phase-Shift Keying

PTM Point-To-Multipoint

PTP Point-To-Point

QAM Quadrature Amplitude Modulation

QPSK Quadrature Phase-Shift Keying

RapidM Rapid Mobile (Pty) Ltd.

RBG Random Bit Generator

RF Radio Frequency

RFC Request for Comments

RLSU Robust LSU

RS Recommended Standard

RTO Retransmission Timeout

RTT Round-Trip Time

Rx Receive

SA Security Association

SAD Security Association Database

SATCOM Satellite Communications

SDR Software Defined Radio

SIS Subnet Interface Sublayer

SNR Signal to Noise Ratio

SPD Security Policy Database

SPD-I SPD Inbound

SPD-ID SPD Identifier

SPD-O SPD Outbound

SPD-S SPD Secure Traffic

SPI Security Parameters Index

SSB Single Sideband

SSL Secure Socket Layer

STANAG Standardization Agreement

SD Standard Deviation

SMTP Simple Mail Transfer Protocol

SYN Synchronize

S/W Software

TCP Transmission Control Protocol

TFC Traffic Flow Confidentiality

TLS Transport Layer Security

TM Traffic Setup

TOD Time of Day

TRANSEC Transmission Security

TS Timestamp

Tx Transmit

UDP User Datagram Protocol

UHF Ultra High Frequency

US United States

VETH Virtual Ethernet

VHF Very High Frequency

VPN Virtual Private Network

WALE Wideband Automatic Link Establishment

WB Wideband

WBHF Wideband High Frequency

WBTS Wideband Transport Sublayer

WTRP Wireless Token Ring Protocol

XFRM Transform

XML Extensible Markup Language

XMPP Extensible Messaging and Presence Protocol

XOR Exclusive OR

1 Introduction

This chapter presents the introduction to the research study that explores the development of an IP cryptographic function for the HF radio environment. Section 1.1 presents an overview of the landscape of IP services within the HF military industry. Section 1.2 presents the motivation for developing an IP cryptographic function for the HF radio environment, followed by section 1.3 which presents the specific goals of this research study. The research method used in this study is discussed together with an overview of the rest of the dissertation structure in section 1.4. Lastly, some concluding remarks are presented in section 1.5.

1.1 The landscape of IP services in the HF military industry

Provisioning Internet Protocol (IP) services over High Frequency (HF) radio networks has been of interest from at least as early as 1999 [1], and continues to receive research interest, as evidenced by more recent work [2]–[4]. Historically, this has only been achieved with limited success, as is evidenced by the statements made in a fairly recent industry standard, viz. Annex U of the North Atlantic Treaty Organization (NATO) “Technical Standards for HF Radio Link Layer and Application Support Protocols”, Standardization Agreement (STANAG) 5066 Edition 4 [5] standard, which claim that the IP Client provisioned by this annexure is only suitable for Internet Control Message Protocol (ICMP) and some low volume User Datagram Protocol (UDP) based applications under most HF system conditions.

This is in part the result of the high latency, low bandwidth nature of HF channels, and partly due to shortcomings in some HF industry standards that fail to optimise IP traffic for HF transmission. For example, the above-mentioned IP Client simply encapsulates IP packets as-is within the STANAG 5066 protocol, without any consideration being given to improve the protocol behaviour for IP data transfer.

However, the advent of new Wideband High Frequency (WBHF) technology and industry standards, such as STANAG 5070, the “Interoperable Wideband Link Management” standard [6], holds promise as a key enabler for IP services in the HF domain. Recent years have also seen a renewed drive within the industry to realize a workable IP based system over military HF networks, as seen in a recent unclassified NATO tender document [7] which details requirements for a Broadcast maritime Rear link and Ship to Shore (BRASS) IP Enhanced System (BRIPES), among other things.

1.2 The case for developing an IP cryptographic function for the HF radio environment

Network security can be considered a key aspect of virtually any military communication system. Along with the drive for IP services in the HF environment, it is therefore not surprising to see requirements emerge for new IP security measures in the HF industry as well. Perhaps the most prevalent is that of Annex E in STANAG 5070 [6], which mandates the inclusion of an IP cryptographic function, or IP crypto for short, although, other than specifying a few restrictions,

it does not include a detailed specification for an IP crypto as such.

Note that the short-term *IP crypto* for *IP cryptographic function* is used throughout this dissertation for two reasons: use of the short-term *crypto* for *cryptographic* or *cryptographic function* is accepted in the HF industry, as explained in section 2.5.2.2, and the IP crypto designed in this research study is not fully compliant with traditional IP layer security Request for Commentss (RFCs), viz. the Internet Protocol Security (IPsec) RFC 4301 [8], as later summarised in section 3.3.4.11, i.e., the slightly more generic term *IP crypto* is more representative of the work done in this research study.

Other factors driving IP over HF research work also imply that the implementation of security at the IP layer is required. These and other factors are discussed as part of the literature investigation around general IP over HF data transmission (section 2.4), but are briefly stated here for continuity purposes:

- The general concept of inter-operation between communication systems in the military environment [2];
- The more specific concept of Network Centric Warfare (NCW) along with the view that IP will be the common protocol in military networks, stated in both older [9] and more recent research work [3]; and
- The need for a backup to Satellite Communications (SATCOM) based links, especially in a wartime situation [3], [10], [11].

Each of these factors applies to scenarios where an HF link is only one of the possible media used to transfer IP data. Therefore, traditional HF security measures cannot be considered as the only means of providing security in an IP over HF system.

Furthermore, very little is disclosed in public domain HF research about IP crypto design or security design in general. Papers such as [9], [12] include references to IP security, but these are terse at best.

More detailed IP crypto information is available within NATO, such as the tactical radio profile for an IP crypto module which was developed by the Network and Information Infrastructure IP Network Encryption (NINE) working group to address shortcomings of STANAG 4787, the NINE Interoperability Specification. However, both the radio profile and the specification itself are classified as NATO restricted, meaning that these are not accessible within for non-NATO countries such as South Africa.

Therefore, this provides a unique opportunity to develop an IP crypto function for the HF domain that is industry standards compliant and has a limited impact on IP over HF performance.

1.3 Research study goals

This research study aims to explore the development of an IP crypto function for the HF domain by achieving the following:

1. Presenting an IP crypto function design that is both industry standards compliant and considers the challenges presented by the nature of an HF communications channel;

2. Creating a representative implementation of this IP crypto function such that the evaluation goal (stated next) can be achieved; and
3. Evaluating the performance impact of this IP crypto function in an IP over HF system, viz. operating over WBHF channels.

The scope of each of these goals is expounded throughout the dissertation write-up, but some notable exclusions are worth stating here:

1. Neither the IP crypto design or implementation is intended to be exhaustive, such that the resulting IP crypto function is suitable as-is for deployment and integration within a military HF system, as further explained in sections 3.3.2 and 3.4.1;
2. Although the test system (test bench) used for the evaluation of the IP crypto function has to be representative of a practically workable IP over HF system, the aim is not to create and/or use a test system that is fully representative of the envisioned system in which the IP crypto function is expected to operate. This is addressed in greater depth in section 4.2.

1.4 Research method and dissertation overview

This research study aims to follow a predominantly empirical research approach, with a positivist-pragmatic philosophy. In other words, the study follows an objective analysis approach for the development of the IP crypto, and the evaluation of the IP crypto is done using experimental data obtained from a purpose-built test setup.

This is done by first compiling and analysing information from sources relevant to the research study goal, such as literature about the HF radio environment in general, and standards that govern both HF industry-and traditional IP network communication and security.

The results of this analysis are used in combination with the envisioned system in which the IP crypto is expected to function, to create a implementation that can be subjected to experimental evaluation, viz. performance testing that produces a data set usable for quantitative evaluation and analysis.

This performance testing is performed with the help of a hardware test bench consisting of Commercial Off-The-Shelf (COTS) components (HF protocol stacks, WBHF capable modems, and an HF/WBHF channel simulator) and test data generation and measurement tools that can provide performance metrics.

Finally, the performance impact of the IP crypto is quantified and analysed by comparing the performance results without the IP crypto present in the test system, viz. a baseline result set, to the performance results with the IP crypto included in the system.

This research method is reflected in the structure of the dissertation write-up, which is briefly summarised in the remainder of this section.

Chapter 2 contains a literature study that provides an overview of the HF medium, explores traditional data transmission over an HF channel, followed by a discussion of IP data transmission over HF, reviews existing HF security mechanisms, and discusses IP security with special focus placed on the IPsec protocol suite.

The information obtained from the literature study is combined with the IP crypto requirements presented in Annex E of the HF industry standard STANAG 5070 [6], to form the basis of the IP crypto design. This is presented as part of chapter 3, which discusses the high level architecture of a potential (envisioned) IP over HF system, followed by the IP crypto design done with the assistance of a functional analysis process. Thereafter, an IP crypto implementation, suitable for the purposes of this study, is presented which makes use of the existing IPsec implementation in the Linux kernel, known as the Transform (XFRM) framework.

Chapter 4 discusses the evaluation approach of the IP crypto's performance impact in a suitable hardware test bench system. This includes the test bench architecture, setup, and configuration, along with discussions of creating suitable performance tests by modifying an existing open source IP performance test tool, iperf3. The verification of the IP crypto implementation, as well as the verification and validation of the test tool modifications, are also included in this chapter.

The IP crypto test results and analysis is presented in chapter 5. This includes the presentation and verification of a baseline set of performance results, i.e. without the IP crypto included in the system. Subsequently, the test results with the IP crypto included in the system are presented and validated, followed by a comparative analysis of these results with the baseline results.

The conclusion of the research study is presented in chapter 6, which links the outcomes of the research study back to the original goals presented here in section 1.3. This includes a summary of the findings of the IP crypto performance impact evaluation, i.e. the comparative analysis of the IP crypto test results with the baseline test results, a discussion of the research contributions made by this research study, and presents possible research avenues that can be explored in future work.

1.5 Concluding remarks

This chapter provided a contextualization of the research study by providing an overview of the current landscape of IP services in the HF military industry in section 1.1, followed by the motivation for exploring the development of an IP cryptographic function for the HF radio environment in section 1.2. The specific research study goals are presented in section 1.3, and an overview of the structure of this dissertation along with the research method used in this study is presented in section 1.4.

2 Literature study

This chapter presents a literature study that investigates several fields related to the aim of developing an IP cryptographic function for the HF radio environment. For contextualization, the structure of the literature study is discussed briefly in section 2.1. Section 2.2 provides an overview of the HF transmission medium, which includes a discussion around the characteristics of an HF channel presented in section 2.2.3. Traditional data transfer over an HF channel is considered in section 2.3, followed by an investigation into IP data transfer over an HF channel in section 2.4. Section 2.5 covers traditional HF security mechanisms, and section 2.6 considers security in traditional IP networks, with a notable deep dive into the IPsec protocol suite presented in section 2.6.2. Lastly, some concluding remarks are given in section 2.7.

2.1 Motivation for the structure of the literature study

There are several knowledge domains that have a bearing on the development of an IP cryptographic function for the HF radio environment. This literature study is structured to present each of these domains in a sequence that allows subsequent discussions to build on each other as and where applicable.

To this end, an overview of the application domain, viz. the HF radio environment, is presented first (section 2.2) to provide the backdrop for the rest of the study. This discussion mainly uses textbook sources, along with some references to industry standards and published research papers.

The following discussion around traditional data transmission in the HF environment (section 2.3) aims to provide greater insight into the working of HF communication networks, including the limitations imposed by the HF medium. This discussion relies primarily on HF industry standards, but makes use of some textbook sources and published research papers as well.

The investigation of IP data transfer over an HF channel (section 2.4) starts narrowing the focus of the literature study to the problem domain at hand, by considering both historical and more recent IP over HF research work. It also highlights the gap in the literature related to network (and IP) security in the HF radio environment, which in turn provides the motivation for considering traditional HF security (section 2.5) and traditional IP security (section 2.6) separately, as is done in the remainder of the literature study.

The discussion regarding HF security mostly references HF industry standards, but also makes use of some National Institute of Standards and Technology (NIST) standards and textbook sources. Similarly, the IP network security discussion mostly references RFC standards but includes some textbook sources as well.

2.2 Overview of the High Frequency (HF) transmission medium

2.2.1 Usage rationale and Application domain

The High Frequency (HF) medium (electromagnetic radio waves propagating via the ionosphere) has been used in military strategic and tactical communication environments for more than 80 years [13]. Other users include emergency services, navigation services, and amateur radio operators [3].

The HF radio frequency range (alternatively known as the decametric band) extends from 3 to 30 MHz [14], and presents various usable communication ranges depending on which HF propagation method is leveraged. These include the following: Line-of-Sight (LOS) (< 30km), ground (surface) wave (up to 50 km on land; 300 km or more at sea), Beyond Line-of-Sight (BLOS) (up to 400 km using Near Vertical Incidence Skywave (NVIS) propagation), and long-range communications, also known as skywave (up to 4000 km and beyond; use of directional antennas is often required) [13], [15].

This versatility means that the HF medium is suitable for a large number of applications. For example, NVIS propagation can be used to provide coverage to a large geographical area, without being limited by terrain obstructions [13]. This is possible since the transmission is directed nearly vertically to the sky, meaning that the HF transmitter can be located in a valley and still realize this coverage, while another medium, such as Very High Frequency (VHF), would require the transmitters to be located on a high mountaintop to obtain the same coverage ([16] as cited by [17]).

Additionally, while the advent of Satellite Communications (SATCOM) initially caused declining interest in HF communications, the HF medium is currently the only known alternative (back-up) that can achieve ranges similar to SATCOM in the right conditions. This is especially of value in military scenarios where operating in satellite-denied environments (due to jamming and/or physical damage to satellite infrastructure) is a very real possibility, which has spurred renewed interest in HF technology over the last 2 decades or so [3], [13].

However, there are significant challenges and limitations that need to be considered when working in the HF environment. These are mentioned partly in the overview of the characteristics of an HF channel in section 2.2.3, and in the discussion of narrowband and wideband HF in section 2.3.8. In summary, these challenges prompt the description of an HF channel as a “high latency, low bandwidth” medium.

2.2.2 Typical components of an HF system

At a high level, an HF system typically consists of a transmitter, a receiver, and an antenna [13]. For the purposes of this discussion, the antenna is represented as part of the transmitter and receiver block diagrams.

Please note that most of the information in this section is based on [13] and [18], as well as the author’s own industry experience. For brevity, the citations for [13] and [18] are not repeated in each paragraph.

The basic components of a transmitter are depicted in Figure 2.1. While this may be an

oversimplification of the transmitter architecture, it is sufficient to explain the basic functions of the transmitter.

The source signal represents the input audio signal to the transmitter. It could, for example, be a normal analog voice signal from a microphone or headset connected to the transmitter, or a output baseband audio signal from an HF modem representing a data stream or coded digital voice. Note that, depending on the actual HF modem and transmitter configuration, the modem output may be transferred to the transmitter in digital form instead of analog form, as discussed in section 2.3.6.1. In this case, both the interface to and the information flow in the transmitter are somewhat different than shown here.

The exciter consists of an audio processor, modulator, and frequency converters. In short, it is responsible for generating a carrier signal, modulating it with the source signal, and converting it to the correct HF transmission frequency before sending it to the power amplifier.

The Power Amplifier (PA) is responsible for amplifying the transmission signal to a higher power level before it is sent to the antenna and radiated into the ionosphere.

Other notable components not included in Figure 2.1 include an antenna impedance matching circuit (often referred to as an Antenna Tuning Unit (ATU)) and various filters (to reduce noise, limit bandwidth, and to eliminate unwanted sidebands after transmitter modulation and/or frequency conversion).

The ATU is connected between the PA and the antenna to facilitate optimal power transfer from the PA to the antenna (the PA usually has an output impedance of $50\ \Omega$). It is normally programmatically tunable for a specific transmission frequency and is a necessity when the antenna's impedance varies with frequency and other environmental conditions. These antenna types typically include: dipoles, whips, and long-wire antennas.

However, if an antenna's impedance is not so sensitive to frequency, as is the case with log periodic antennas, often referred to as broadband antennas, it can be connected directly to the PA.

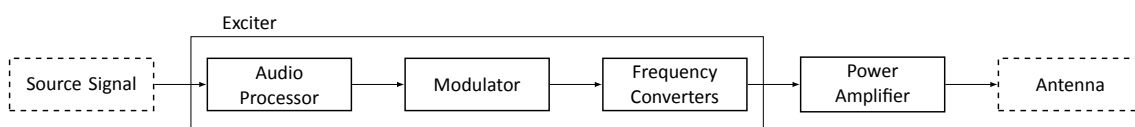


Figure 2.1: Typical HF Transmitter Components (adapted from [13])

A simplified representation of the receiver architecture is given in Figure 2.2. It is essentially the inverse of the transmitter architecture, with the exception of the Radio Frequency (RF) input filter.

If present, the RF input filter acts as a frequency or channel selector, sometimes termed a preselector, which is usually implemented as a tunable bandpass filter. It is primarily used to filter out noise and other unwanted signals before the signal recovery process is started.

The PA is then used to amplify the received signal before it is converted to another frequency for demodulation (this may occur via multiple frequency conversion stages). The demodulation process recovers the information-bearing signal from the received signal as a baseband audio signal.

After demodulation, additional audio processing can be done, or the signal can be fed to the appropriate output device, e.g. a speaker in the case of analog voice, or to the input of an HF modem.

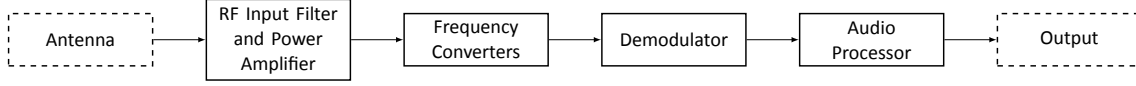


Figure 2.2: Typical HF Receiver Components (adapted from [13])

2.2.3 Characteristics of an HF channel

As mentioned in section 2.2.1, the HF medium can be utilized in several propagation methods, namely: LOS (< 30km), ground (surface) wave (up to 50 km on land; 300 km or more at sea), BLOS (up to 400 km using NVIS propagation), and Skywave (up to 4000 km and beyond). For the purposes of channel modelling (i.e. path loss modelling), it is sufficient to group these into two categories: surface-wave (including LOS) and skywave (including BLOS and NVIS), as is done in [15], [19].

The characteristics of each of these categories are discussed briefly in sections 2.2.3.1 and 2.2.3.2.

2.2.3.1 Surface-wave propagation characteristics

Surface-wave propagation is primarily subject to absorption and reflection path loss factors. According to [20], as quoted by [15], this results in an “attenuated, delayed, but otherwise undistorted version of the transmission signal”.

This results in fairly simple channel modelling considerations as outlined in [19]. For distances smaller than $10\lambda^{1/3}$ km, where λ is the wavelength in meters, the path loss can be represented by (2.1).

$$\text{Path Loss} \propto d^\delta \quad (2.1)$$

where d is the link distance in km, and δ is the path loss coefficient.

As the link distance increases, δ increases from 2 (close to the transmitter) to 4. However, for distances greater than $10\lambda^{1/3}$ km, the earth curvature is no longer negligible, and the path loss increases exponentially [20] as cited by [19].

Additionally, the surface-wave path loss is also dependent on the electrical characteristics of the terrain over which it travels (e.g., seawater, fresh water, dry ground, wet ground). This includes the conductivity and relative permittivity of the terrain [19].

2.2.3.2 Skywave propagation characteristics

Skywave propagation, which occurs via ionospheric refraction [15], is more complex than surface-wave propagation because it is subject to varying ionosphere conditions. This includes the following:

1. Hourly (and daily) variations in ionospheric propagation throughout the 11-year sunspot cycle [15]. (Sometimes referred to simply as time of day and day of year variations [2], [9], [19])
2. Geographical location [2], [9]
3. Longer fading conditions (seconds or minutes long) due to the presence of ionospheric motion, Faraday rotation, and other similar effects [15]
4. Short fading conditions (milliseconds long), i.e. Rayleigh or Rician fading, due to multipath spreading effects [15]. Sources of multipath include radio wave reflections from different layers of the ionosphere and multiple reflections between the ionosphere and the surface of the Earth [21].
5. Weather and/or seasonal changes [2]

These characteristics are summarized in older literature as “the HF channel is characterized as a multipath time-varying environment producing time and frequency spreads” [22].

The mathematical representation of a skywave channel model (or path loss model) is considered to fall outside the scope of this literary work (a simplified formula can be found in [19]). Typical models used for simulation purposes in the HF industry are discussed in section 2.2.3.3.

2.2.3.3 Channel models used for simulation and benchmark testing

HF modem industry standards, such as STANAG 4539 [23] and MIL-STD-188-110D [24] typically specify performance requirements for two separate channel models: a simple Additive White Gaussian Noise (AWGN) channel, and an International Telecommunication Union Radiocommunication Sector (ITU-R) ‘Poor’ channel with specific parameters given to match one of the ITU-R F.1487 channel models [25]. STANAG standards sometimes include performance requirements for an additional channel model, such as a Rician channel in the case of STANAG 4539, or an ITU-R ‘Good’ channel in the case of STANAG 4538 [26].

As a slight aside, the original ‘poor’ channel model was originally defined by the International Radio Consultative Committee (Comité Consultatif International pour la Radio) (CCIR) prior to its merge with the International Frequency Registration Board (IFRB) in 1992 to form the ITU-R [27]. As such, it is sometimes still referred to as the CCIR poor channel, which is synonymous with the ITU-R F.1487 Low Latitude Moderate conditions model [25].

Performance characteristics are normally specified for each data rate as a maximum Bit Error Rate (BER) metric that must be achieved under specific Signal to Noise Ratio (SNR) conditions, for each channel model. Particular emphasis is placed on the ITU-R ‘poor’ channel model, since it provides results comparable with Over The Air (OTA) measurements, by modelling the HF channel as a stationary channel over sufficiently short periods of time (i.e. approximately 10 minutes) [21], [28].

This is also evidenced by industry standards such as MIL-STD-188-110D which includes an Appendix E that contains specifications for baseband HF channel simulators (up to 24 kHz) that can be used for the performance testing of HF modems [24]. This specification mandates the implementation of a Watterson HF channel model, which was originally described in 1970 by Watterson et al. [28]. This is the same model specified in ITU-R F.1487, along with tunable parameter sets that model different channel conditions (one of which constitutes the ITU-R Poor model).

2.3 Transmitting data over an HF channel

2.3.1 Definition of an HF waveform

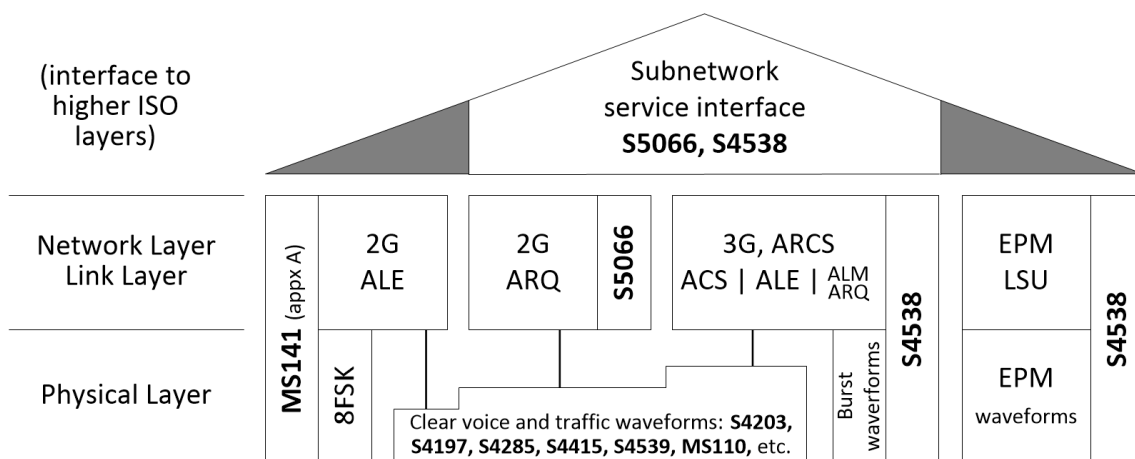
Before proceeding with the discussion around data transmission over an HF channel, it is useful to touch on the definition of the term 'HF waveform'. It is used somewhat colloquially in the HF industry, and the author has been unable to find a clear definition in introductory HF material (such as [13]), in published research papers, or in standards, such as the NATO STANAG standards or the United States (US) Military Standard (MIL-STD) standards.

It is normally used to refer to the physical layer characteristics of the HF signal such as: frame structure, modulation scheme, coding rate, and interleaver lengths (based on the waveforms defined STANAG 4539 [23]). However, this describes the HF signal in its baseband form and technically does not refer to the actual physical waveform radiated by an HF antenna.

2.3.2 Relation to the Open Systems Interconnect (OSI) model

Within the HF industry, data protocols and HF waveforms are typically categorized according to the HF House model, such as Annex A of STANAG 4539 (Technical standards for non-hopping HF communications waveforms) [23] for example (shown in Figure 2.3). This model includes a physical layer, and a combined data link and network layer, which aims to both mimic and be compatible with the International Standards Organisation (ISO) OSI model (although HF networks often do not include network layer functionality).

The multi-waveform concept in the HF House model (often repeated in NATO standards as well) is therefore, unsurprisingly, quite similar to advantages offered by the OSI model, namely, that different physical layer waveforms can often be used with the same data link layer protocols, allowing for the use of optimal waveforms depending on propagation and interference conditions, and that new waveforms can be developed without necessarily developing a new data link layer protocol (Annex A [23]).



KEY 2G: second generation 3G: third generation Snnnn: STANAG MSnnn: Mil-Std

Figure 2.3: Example of an HF House model (taken from STANAG 4539 Annex A [23])

2.3.3 Conceptual data flow at the physical layer

At the physical layer, the transfer of data is concerned with transmitting binary data (essentially a stream of 1's and 0's) from the transmitter- to the receiver station. For discussion purposes, this data could represent virtually any information, e.g. higher-layer protocol data, or simply ASCII characters represented in binary form.

A high-level block diagram indicating the processing flow of the binary data through an HF system is given in Figure 2.4.

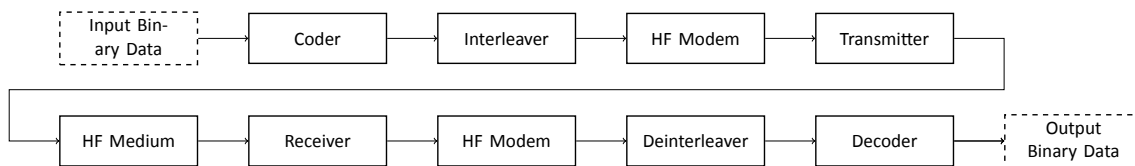


Figure 2.4: Binary data flow in an HF system (adapted from [13])

The coder and interleaver provide error detection and correction functionality at the cost of additional data overhead and increased processing and transmission time [13], [18]. The HF modem performs modulation and demodulation which, in the context of data transmission, is essentially the process of converting (or encoding) binary data into an audio (analog) signal and vice versa, as defined and discussed in various good textbooks [18], [29], [30].

It is important to note that the data modulation referred to in this context should not be confused with the modulation performed by the HF transmitter as mentioned in 2.2.2. See 2.3.6.1 for additional clarification.

Each of these three concepts are discussed in more detail in sections 2.3.4, 2.3.5, and 2.3.6. Please note that STANAG 4539 is often referred to for example or context purposes, since it provides waveform specifications that are usable without any data link layer protocol, i.e., it can be referenced without having to explain or account for higher layer protocol nuances.

2.3.4 Coding

In its simplest form, coding is concerned with adding additional bits (redundancy) to the original input data stream on the transmitting side to allow the receiving side to verify the validity of the data (error detection code). If the coding scheme makes it possible for the receiving side to correct errors in the data stream (without requiring retransmissions from the sender), it is known as an error correction scheme, or an Forward Error Correction (FEC) scheme [18].

Coding schemes can be broadly divided into block coding schemes and convolutional coding schemes [29]. Basic coding concepts are described in the block coding context here since it is simpler to follow than the complexities of convolution coding. However, it should be noted that HF waveform design typically makes use of FEC convolutional codes (most NATO standards, such as STANAG 4539 [23], simply state the requirements and specifications of the convolutional codes to use).

For explanatory purposes, it is still useful to consider the basic coding flow for an error detection code, as shown in Figure 2.5 (adapted from [29]). The discussion here is also largely

based on the information in [29].

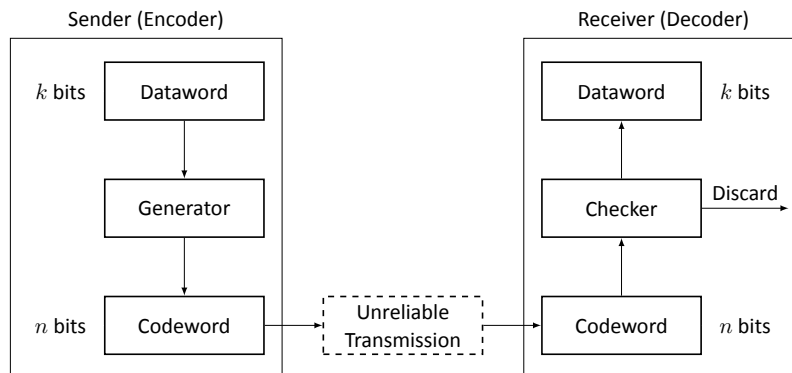


Figure 2.5: Error Detection Concept (adapted from [29])

The input binary data stream (not shown here) is processed in equal length datawords of k bits each (that is, a data block; hence the name “block coding”).

A codeword (of n bits) is generated from each dataword by adding redundancy bits. The complexity of this process (also known as encoding) depends on the block code used, but it could be as simple as adding a single parity bit to the end of the dataword.

The codeword is then transmitted to the receiver (note: the interleaver stage shown in Figure 2.4 is not considered in this discussion for clarity), which reverses the encoding process (also known as decoding).

This is done by splitting the codeword into the dataword and redundant bit(s) portions, and checking the validity of the dataword by using the redundant bits (e.g. calculating the parity of the received dataword and checking it against the redundant parity bit). If a dataword is determined to be in error, it is simply discarded. Higher-level protocols are then responsible for dealing with data loss.

Conceptually, an FEC scheme functions similarly to an error detection scheme. Redundancy added in the encoding process is simply designed such that it provides the receiver with enough information to correct errors in the dataword as well, reducing the number of discarded datawords.

There are, however, several limitations and considerations that affect the efficacy of a coding scheme:

- The possibility of false positives or false errors if bit error(s) occur in the redundant bits;
- The types of errors that can be detected and the ones that can be corrected.
For example, in the single parity bit example, all even-numbered bit errors will go undetected;
- The cost of the overhead introduced by the scheme.
Generally, smaller datawords and more redundancy bits will lead to a more effective coding scheme in terms of catching and correcting errors, but the larger overhead may be too inefficient in terms of channel utilization and bandwidth cost.

In the larger telecommunications industry, popular block FEC codes include the Hamming code and the Reed-Solomon code [18]. However, as stated before, block codes are typically not used in HF waveform design, in favour of convolutional FEC codes.

Convolutional codes differ from block codes in that they are a function of both current and past data bits and that the output of the coding process produces two data streams, neither of which contains the original data as-is. Both of these data streams are transmitted to the receiver [18].

According to [18], one of the main advantages of a convolutional coding scheme lies in the fact that each output code is different, meaning that the likelihood of recovering the original data is greater (i.e. it is more robust). This claim will likely make more sense with a more in-depth discussion of convolutional coding, but this is deemed unnecessary for the purposes of this overview.

2.3.5 Interleaving

One of the major vulnerabilities of virtually any FEC scheme is susceptibility to burst errors [18], [29], [31]. A burst error refers to sudden data loss due to a burst of noise in the transmission (common in wireless mediums, e.g. sudden deep channel fade).

Since an FEC scheme (block or convolutional) operates on a “per input dataword” basis, it cannot provide redundancy (protection) against the loss of an entire codeword in the transmission medium. In other words, if the codeword never reaches the receiver, the only recourse is for the sender to retransmit the data. Additionally, FEC schemes typically do not guard well against the corruption of sequential bits in a codeword [18].

The interleaving process aims to guard against this by mixing several codewords at a bit level (before transmission) so that the bits of any single codeword are no longer transmitted sequentially. This can be conceptually explained as follows:

1. On the sender side, the codewords are written into a table of fixed size $m \times n$ (each cell holds a single bit; assume that the codeword size is fixed (n) for explanatory purposes).
2. Once the table is full, data is read from the table per column (i.e. m bits at a time) and sent to the transmitter.
3. On the receiver side, this process is reversed.
This is called de-interleaving; data is written into table columns (m bits at a time) and read out row by row (n bits at a time) once the table is full.
4. The decoding can then take place as usual at the receiver.

In HF waveform design, a block interleaver scheme is usually used, which operates at a bit level, and is implemented in a one-dimensional array (such as in the high data rate waveforms of STANAG 4539 [23]). This interleaver block (the array) is characterized by two parameters: interleaver size (p ; the array length) and interleaver increment (e).

The interleaver size is chosen so that the input data block (called a packet, not to be confused with the term “packet” used to describe an OSI layer 3 Protocol Data Unit (PDU)) of size n can fit completely in the interleaver (data stored from 0 to $p - 1$). In the event that $n < p$, the

input packet is zero-padded to match the interleaver size.

The input packet is stored in the interleaver one bit at a time, starting at position 0, with each subsequent storage position calculated by adding the interleaver increment modulo the interleaver size ($e \bmod p$) to the last storage position.

A last note on HF waveform coding and interleaving design (based on the example of STANAG 4539): two additional techniques can be employed in these stages, namely: code puncturing and tailbiting. Code puncturing is used to improve the output data rate of the coder (at the cost of a slight reduction in redundancy), and tailbiting ensures that the ending state of the interleaver is the same as its starting state after each input data packet is processed (explanation of these techniques partly based on the definition in [32] and partly deduced from the standard specification).

2.3.6 Modulation

2.3.6.1 Disambiguation of terminology

In section 2.3.3, modulation, in the context of data transmission, is summarized as the conversion (or encoding) of binary data into an analog signal (and vice versa for demodulation), based on the definition and discussion from various good textbooks [18], [29], [30]. While this is mostly sufficient for the theoretical discussion, it is worth noting that this is not entirely true in practice, at least not in the HF industry.

Since an HF modem performs modulation and demodulation in the digital domain, with the aid of Digital Signal Processor (DSP) hardware, the output signal of the modulation process is also obtained in digital form, typically represented as complex mathematical values, which can be considered as a baseband digital signal. In some cases, the HF modem will proceed to convert this to a baseband analog signal, typically when the HF modem interfaces to legacy HF radio transmission equipment, which may only have an audio interface. However, the HF modem can also transmit the digital output directly to the radio via a digital interface (for example, via a EIA-422 or Multichannel Buffered Serial Port (MCBSP) interface).

In the first case, the radio (transmitter) will still perform its own analog modulation since it does not distinguish between e.g. an analog speech signal received from a handset and the analog output received from the HF modem. In the second case, the specific radio (transmission) equipment architecture will determine whether additional modulation is done or not. At a minimum, it is responsible for performing digital to analog conversion before transmitting the signal over HF.

To avoid confusion, the term modulation, as used in the discussion throughout the rest of section 2.3.6, is perhaps better understood as “the procedure of processing a signal (or input) for more efficient transmission” as generalized in [30].

2.3.6.2 Theoretical discussion

The basic principle of any modulation technique is as follows: an information signal is used to vary a property(ies) of a carrier signal which is suited for transmission over the medium in question, as is similarly defined and explained in any good textbook [18], [29], [30]. In the context of an HF modem, the information signal is a bit stream, and the carrier is normally a sinusoidal wave (in the digital domain) centred in the baseband channel, e.g. Annexure

B of STANAG 4539 mandates that the carrier is centred at 1800 Hz, in the 200 to 3400 Hz band [23].

For example, in a Binary Phase-Shift Keying (BPSK) modulation scheme, the carrier is transmitted at one phase for the duration of a binary 0 (e.g., at 0°) and at a 180° phase shift for the duration of a binary 1 [33]. Each of these phases can be considered to be a symbol representing 1 bit.

More complex modulation schemes, such as M -ary Phase-Shift Keying (PSK) or Quadrature Amplitude Modulation (QAM), can represent multiple bits per symbol (QAM modulation varies both the phase and the amplitude of the carrier). For example, 8-PSK encodes 3 bits per symbol (8 different carrier phases), thus increasing the transmission data rate over the same bandwidth. This characteristic is termed spectral efficiency and is measured in bits per second per hertz, that is, 8-PSK has a spectral efficiency of 3 bps/Hz , [33].

STANAG 4539 specifies the specific modulation schemes that are used to achieve each of the data rates covered in the standard. This is given in Table 2.1 for reference. Note that the 12 800 *bps* rate is achieved by making changes to the coding scheme used.

Table 2.1: Modulation used to obtain each data rate in STANAG 4539 [23]

Data rate (bps)	Modulation
3200	QPSK
4800	8PSK
6400	16QAM
8000	32QAM
9600	64QAM
12800	64QAM

However, higher data rates can only be realized under good SNR conditions. Or stated differently: the more symbols a modulation scheme supports, the more susceptible the modulated signal is to noise (i.e. the same amount of noise will cause more bit errors in more complex signals, since the symbols are closer together, resulting in a smaller margin of error at the receiver). This should become clearer in the context of constellation diagrams, as discussed below.

The symbols of a modulation scheme can be represented on a constellation diagram, which indicates the phase and amplitude of the carrier for each symbol [33]. Examples of QAM constellation diagrams are given in Figure 2.6.

The constellations in Figure 2.6 are perhaps best explained by considering how QAM modulation is achieved in practice. The QAM carrier signal is actually made up of two components (or subcarriers), namely the quadrature subcarrier (Q) and the in-phase subcarrier (I). By definition, the quadrature subcarrier has a 90° phase shift relative to the in-phase subcarrier [29], [30], [33]. This is also reflected in STANAG 4539, which expressly states that the quadrature subcarrier is mandated to have a phase relative to the in-phase subcarrier of 90° [23].

Each of the subcarriers amplitude's are varied separately based on the input binary data, and the resulting signals are then combined to generate a single output signal. This has the

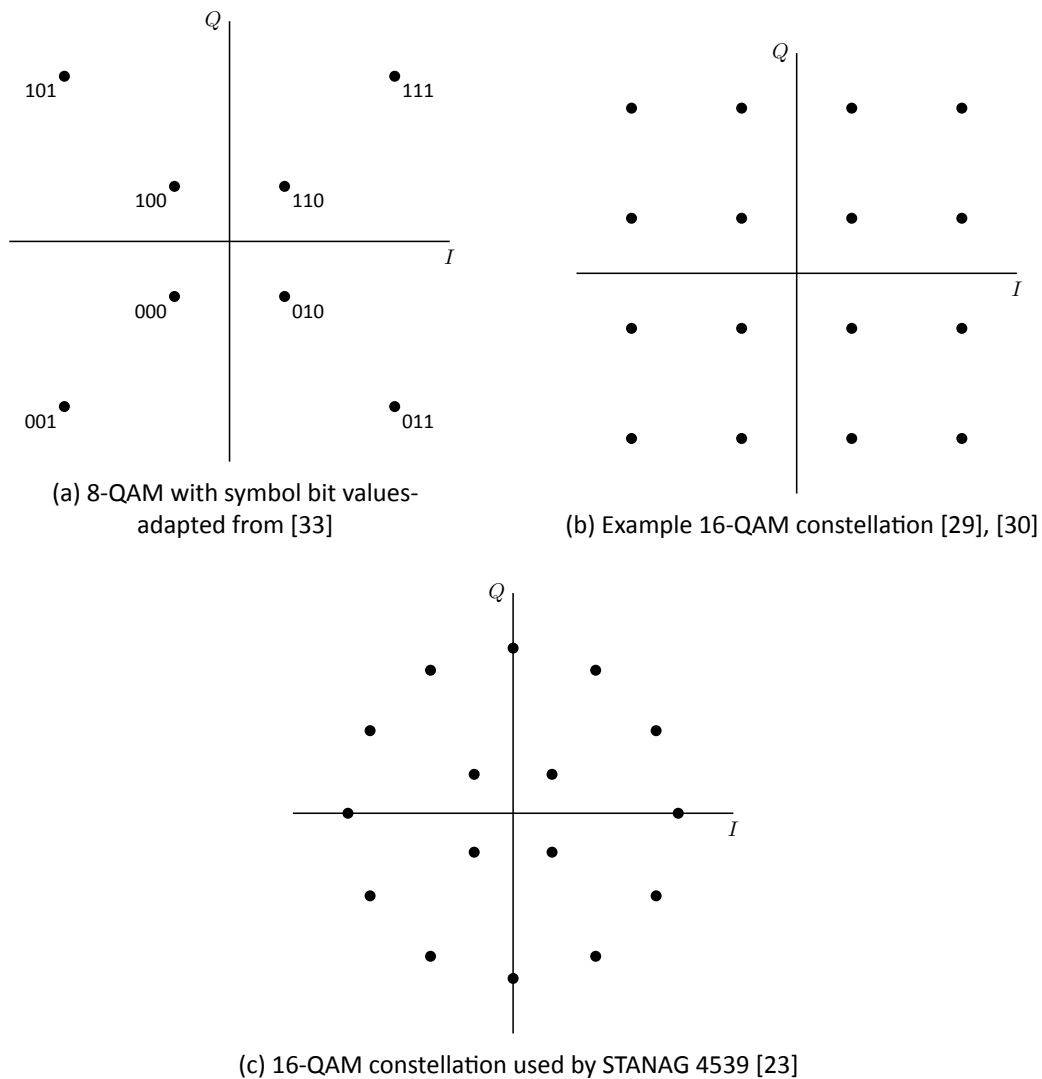


Figure 2.6: Examples of constellation diagrams

same net effect as varying the amplitude and phase of a single carrier signal, which matches the definition of QAM modulation given earlier.

With this in mind, the constellation diagrams can be read as a Cartesian plane, with the values on the axes indicating the amplitude of each subcarrier for a particular symbol. Additionally, Subfigure 2.6a includes example bit values or patterns (taken from [33]) that could be represented by each symbol in an 8-QAM modulation scheme.

Due to the signal distortion caused by noise (i.e., malformed amplitude and/or phase), it is possible that the receiver will demodulate transmitted symbols incorrectly (e.g., a symbol sent representing bit pattern 101 is received as a symbol representing bit pattern 100). As previously mentioned in this section, the likelihood of a demodulation error increases as the number of symbols in a modulation scheme increases (smaller margin for error at the receiver). When comparing the constellation diagrams, it is easy to see why this is the case.

2.3.7 Higher layer protocols

As indicated in section 2.3.2, the HF House model includes link layer protocols that provide various functionality within an HF system, such as Automatic Channel Selection (ACS), Automatic Link Establishment (ALE), Automatic Link Maintenance (ALM), and Automatic Repeat reQuest (ARQ). The historical progression of these functions is discussed as part of an overview of HF communication generation history in both [15] and [19].

For the purposes of this section, however, these functions are described as pertain to what is colloquially known as Third Generation (3G) HF networks in the HF industry (not be confused with cellular 3G networks), i.e. as defined by STANAG 4538 [26] and MIL-STD-188-141B [34] for Narrowband HF (NBHF) transmission.

Note: A updated revision of MIL-STD-188-141 (revision C [35]) added specifications for WBHF channels within a 3G HF network, while the current latest revision (revision D [36]) defines a new Wideband Automatic Link Establishment (WALE) system that constitutes the Fourth Generation (4G) of HF networks. NBHF and WBHF are discussed in section 2.3.8.

STANAG 4538 defines the 3G HF network as a NATO standard Automatic Radio Control System (ARCS). The three core processes of an ARCS network are ACS, ALE, and ALM, as represented in Figure 2.7 taken from STANAG 4538 [26]. A brief discussion of each of these processes is presented in Table 2.2.

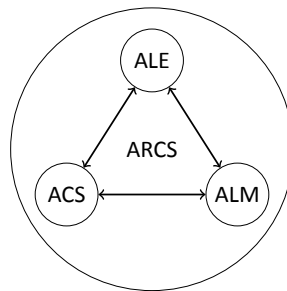


Figure 2.7: Inter-relationships between the primary ARCS constituent processes [26]

Table 2.2: Description of various STANAG 4538 ARCS processes [26]

ARCS process	Description
ACS	For ARCS and ACS purposes, a (simplex) channel is defined as “the upper or lower sideband related to a carrier frequency” (3 kHz channel spacing) [26]. ACS is concerned with the selection of channels “for calling and for traffic in a fashion that optimises overall network performance” [26]. The channels are classified according to their purpose as ‘search’ (call), ‘traffic’, or ‘both’ (search and traffic). Information considered during channel selection includes: propagation and occupancy of each channel, and the requirements of the traffic to be supported (e.g., analogue voice, digital voice, or data). This information can be obtained from channel measurements during system operation or other systems external to ARCS (e.g., propagation prediction system) [26].
ALE	As the name implies, Automatic Link Establishment (ALE) is concerned with establishing a link for traffic purposes using one of two LSU protocols (Fast LSU (FLSU) or Robust LSU (RLSU)). STANAG 4538 distinguishes between three types of links: physical, logical, and communication links. A physical link refers to the actual propagation of radio waves, i.e., can the signal physically reach another ARCS radio node (or Participating Unit (PU) according to STANAG 4538 terminology) in such a way that it can process the signal? Logical links are derived from physical link(s) as “a temporary association of one or more physical links with two or more PUs for the transfer of traffic”. This association is made using parameters such as channel frequency, PU addresses, Time of Day (TOD), and link access protocols, such as Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). Logical link types include: Point-To-Point (PTP), Point-To-Multipoint (PTM), and broadcast links. Communication links are formed using a logical link(s) to facilitate the transfer of traffic between PUs that need to communicate with each other. Communication link types include: unidirectional links (user data flows in one direction, acknowledgements - such as ARQ acknowledgements - flow in the reverse direction), and Conference (CONF) links. LSU protocols are primarily concerned with establishing logical links, although they do play a role in establishing communication links by interacting with the Traffic Setup (TM) protocol [26].
ALM	Automatic Link Maintenance (ALM) is described as being “responsible for monitoring the quality of an established link, and for initiating corrective action when necessary” in the STANAG 4538 standard [26]. Details about its working are considered to fall outside the scope of this study, although it is worth noting that ALM only comes into play after a link has been established (i.e., after LSU has completed). During data transfer, it monitors link conditions, and may interrupt a link if it deems it necessary for link parameters to be renegotiated, or even completely re-establish a new link.

Table 2.2: Description of various STANAG 4538 ARCS processes [26] (cont.)

ARCS process	Description
ARQ	ARQ functionality is implemented as a data link protocol, generically known as an ARQ protocol, which forms part of the error control measures within an ARCS network. Some protocol variations are employed in STANAG 4538: Low-Latency Data Link (LDL), which is preferred for short messages or low-SNR channels, and High-Throughput Data Link (HDL), which is preferred for longer messages (larger datagrams) under good HF channel conditions. It guarantees error-free delivery by segmenting a data stream into frames and ensuring that each frame is received error-free. The receiver determines if a received frame contains errors using a CRC check and requests retransmission if the frame was received in error, or sends an acknowledgement if it was received correctly. Other error control measures, spanning the physical layer and data link layer, include: 1. Forward Error Correction (FEC) 2. Repetition (repeat transmissions of the same frame) 3. Code combining (combine multiple received copies of the same frame; can potentially recover the correct frame from multiple erroneous frames) 4. Interleaving 5. Waveform selection Measures such as FEC and interleaving are performed at the physical layer, as has already been discussed in sections 2.3.4 and 2.3.5 respectively. However, it is worth noting that the ARQ protocol also employs some of these measures at the data link level (for example, interleaving within a frame) [26].

2.3.8 Narrowband and Wideband High Frequency (HF)

The spectrum in the HF range is traditionally allocated in 3 kHz channels. Historical reasons are cited in [15], while other sources [3], [13], [37] treat this as common knowledge within the HF environment. Although not always referred to as such, a 3 kHz channel is considered to be a Narrowband HF (NBHF) channel, since Wideband High Frequency (WBHF) channels are typically larger (wider) than 3 kHz (allocated in increments of 3 kHz up to 24 kHz in Annex D of MIL-STD-188-110C as cited by [38]; the upper limit of this range is increased to 48 kHz in the updated MIL-STD-188-110D [24]).

The main driving force for the development of WBHF was an increasing need for higher data rates from the HF user community [39]. To this end, the following design goals were set during the initial WBHF waveform design (i.e. the development of MIL-STD-188-110C [40]), as outlined in [39]:

1. Bandwidths extending from 3 kHz up to 24 kHz, in increments of 3 kHz;
2. As the data rate is decreased, the waveform becomes more robust;

3. Variable-length preamble (i.e. long for HF sky wave links, short for surface wave links);
4. Performance similar to that of the MIL-STD-188-110B 3 kHz waveforms;
5. Broadcast capability without “autobaud” (that is, the receiver modem must know all waveform parameters);
6. Tailbiting FEC codes (similar to 110B) to reduce coding overhead; and
7. “Nice” data rates.

This point is stated as-is in [39]. For clarification, this can be understood to simply mean ‘data rates that are notably higher than existing NBHF data rates’.

Note: additional design goals were set related to specific HF path propagation parameters (e.g. minimum multipath delay tolerance). Refer to [39] for additional details in this regard.

To illustrate the potential data transfer improvements achievable with WBHF, a subset of the data rates given in MIL-STD-188-110D (the current latest revision of the standard) is repeated in Table 2.3 for several modulation schemes.

Table 2.3: Subset of WBHF data rates (*bps*) according to bandwidth and modulation scheme from MIL-STD-188-110D [24]

Modulation scheme	QPSK	8PSK	16QAM	256QAM
Bandwidth (kHz)				
3	3 200	4 800	6 400	16 000
6	6 400	9 600	12 800	32 000
9	9 600	14 400	19 200	48 000
12	12 800	19 200	25 600	64 000
15	16 000	24 000	32 000	76 800
18	19 200	28 800	38 400	90 000
21	19 200	28 800	38 400	115 200
24	25 600	38 400	51 200	120 000
30	32 000	48 000	64 000	160 000
36	38 400	57 600	76 800	192 000
42	38 400	57 600	76 800	192 000
48	48 000	72 000	96 000	240 000

2.4 Transmitting IP data over HF

2.4.1 History of IP services in the HF industry

2.4.1.1 Driving research factors

Historically, the transmission of IP data over HF networks has not worked well in practice, in part due to the approach followed by HF industry standards. For example, the IP Client provisioned by Annexure F.12 in STANAG 5066 Edition 3 [41], as well as the IP Client in Annexure U of the updated STANAG 5066 Edition 4 [5], encapsulates IP packets as is, subjecting each packet to the latency and bandwidth constraints of the HF channel.

However, the use case for IP over HF has been of interest from at least as early as 1999 [1], and is still very prevalent as seen in more recent work [2]–[4]. Additionally, there is also a prominent drive within the industry to provision IP based services over HF networks, as is reflected in a recent NATO unclassified tender document [7] which details requirements for a BRASS IP Enhanced System (BRIPEs) amongst other things.

Some of the driving research factors include:

1. The concept of NCW along with the view that IP will be the common protocol in ‘future’ military networks (stated in 2004 [9]; also reflected in more recent work [3]). Similar concepts to NCW are discussed in other works, viz. NATO Network Enabled Capability (NNEC) in [12], and the general concept of inter-operation between communication systems in the military environment in [2]
2. An increasing demand for services such as email, file transfer, and web browsing in the tactical environment, including HF links, as indicated in research work [10]. This is also evidenced by the NATO BRIPEs tender specification, which includes these services, along with a multi-user chat service. Older research papers also noted an increase in IP use in military networks [11], [42]
3. The need for an alternative (or backup in a wartime situation) to satellite-based communication links (SATCOM links) [3], [10], [11].

2.4.1.2 Problematic characteristics of HF for IP data transmission

The challenges related to IP traffic transmission over an HF channel relate primarily to the high-latency, low-bandwidth nature of HF. This is the typical starting point (or at least a key consideration) in most IP over HF research work, both historically, such as [1], [9], [43] (ranging from the year 1999 to 2004), and in more recent work, such as [2]–[4] (ranging from the year 2017 to 2020).

This high-latency, low-bandwidth nature has also been alluded to in section 2.2.3 (which highlights the problematic propagation characteristics of an HF channel) and throughout 2.3 (low data rates indicated throughout the discussion of data transfer over HF, although WBHF can alleviate this somewhat as indicated in section 2.3.8).

For posterity, the problematic characteristics of HF, as highlighted in several IP over HF research papers, is summarized here:

1. Limited capacity (bandwidth) [1]–[3], [9]

2. Transmission delay (high latency) [3], [43]
3. Large link turn-around, or response time; partly caused by the half-duplex approach employed by lower-layer HF waveforms. For example, the MIL-STD-188-110B waveform is mentioned specifically by [43] (other sources just mention the constraint [9])
4. Unreliable nature of connections (high error rates, link failures, varying throughput and connectivity rates depending on physical conditions such as time of day and location) [2], [3], [9].

2.4.1.3 Typical solution domains considered in IP over HF work

While the solutions investigated in the papers cited in section 2.4.1.2 consider different aspects of- or approaches to solving the barriers posed by the HF medium, two themes are fairly common: increasing the link capacity (bandwidth), and optimising protocol(s) to better utilise the HF medium.

As indicated in section 2.3.8, WBHF was designed primarily to improve the data rates achievable over the HF medium. For this reason, WBHF is currently heralded as a key enabler for a practically workable IP over HF system within the HF industry. This view is also reflected in more recent published work [4], [10], [44].

As an aside, it is interesting to note that older published work made similar claims (or at least expressed similar hopes) regarding the advent of the STANAG 5066 standard [11], [42], [43], and again regarding narrowband 3G HF systems (i.e. STANAG 4538) [9]. Note that while STANAG 5066 is a data link protocol, usable with various HF waveforms - including WBHF, it was used with narrowband HF waveforms in these published works. This combination is typically termed Second Generation (2G) HF.

Solution approaches that focus on protocol optimisation aim to accomplish things such as:

1. Data compression (due to the bandwidth limitation of HF) [2], [3], [10]
2. Reducing protocol overhead [2], [3], [12]
This factor is perhaps not fully understood in [2]. Based on industry experience, the author's opinion is that a higher protocol overhead will likely be tolerated if it improves the probability of success of a transmission, since the cost of repeating a failed transmission is typically much higher in an HF network than dealing with more protocol overhead.
3. Grouping data into larger transmission frames where possible (longer transmissions reduce the effect of link turnaround time) [2], [10].
Another paper [12] very loosely addresses this aspect as part of its focus on "removing the dependence on end-to-end connections" of services such as Extensible Messaging and Presence Protocol (XMPP), Hypertext Transfer Protocol (HTTP), and Transmission Control Protocol (TCP).
4. Avoiding repeat transmissions when not needed [2], [3]
Although this point may seem obvious at first, it is worth mentioning since some standard IP protocols, such as TCP, are not necessarily designed with this in mind. For example: the standard TCP windowing and bulk Acknowledgement (ACK) concept may be detrimental to performance in an HF network, since it can cause the sender to resend data already received

by the recipient in certain scenarios, such as the recipient missing a single segment in the middle of a multi-segment transfer.

Some papers cited for the points mentioned above address several of these aspects as part of their recommendation for or implementation of a Performance Enhancing Proxy (PEP) (or similar mechanism) for protocols such as TCP [2] or XMPP [12]. The authors of [3] (and their subsequent work in [4]) take an alternative approach by designing a new TCP implementation (termed 'HF-TCP') which should still be compatible with normal TCP implementations.

The authors of [44] focus more on optimisation of TCP via a PEP without specific focus on the points mentioned above, although there is mention of the PEP's goal to "mitigate end-to-end performance impairments due to increased latency and packet loss". The basic function of a PEP in the HF environment is normally to spoof the working of a protocol to a client external to the HF network (e.g. a TCP PEP would spoof the expected TCP handshaking and data-flow ACKs), while repacking the data in a protocol (possibly proprietary) that is more suited to HF network conditions, i.e. that can achieve some or all of the protocol optimisation conditions stated earlier (also see section 3.2.4).

2.4.1.4 Security considerations

Very few papers cited throughout section 2.4.1 consider network security (or IP security) at all (only [9], [12]) and only [12] actually factor this into the design of their Delay- and Disruption-Tolerant SOAP Proxy (DSProxy). In summary, the DSProxy is described as "a proxy for Web services designed to handle all types of information and traffic flows that are suited to be implemented as Web services".

Based on industry experience, this gap in the literature is somewhat expected, since detailed information about the security aspect of any HF system is typically not published in the public domain. In lieu of this, it is worth considering general security within the HF domain (i.e. not IP specific), and general security within traditional IP networks. This is done in sections 2.5 and 2.6 respectively.

2.5 Security for HF communication

2.5.1 Security terminology used in the HF domain

Security within the HF domain is typically described or classified at the behest of two broad terms: Communications Security (COMSEC) and Transmission Security (TRANSEC). Similar to the term 'HF waveform' (discussed in section 2.3.1), these terms can be used within the industry with somewhat different colloquial meanings, and are consequently defined differently on occasion. For the purposes of this study, the terms should be understood as defined here.

COMSEC is the umbrella term used within the HF industry to refer to "methods that keep important communications secure" [13]. While this term is not unique to the HF industry (for example, used by the National Security Agency (NSA) of the United States [45], and in research related to security in IP-based satellite networks [46]) it is perhaps not commonly encountered within the world of traditional IP networking and communication, where the term "cryptography" is usually used.

TRANSEC can be defined as “schemes that make it difficult for someone to intercept or interfere with your communications” [13]. TRANSEC is sometimes viewed as a sub-section of COMSEC but, for the purposes of this study, it is treated as a separate security mechanism.

When viewed in light of a hostile party or attacker (i.e. the enemy trying to obtain the information being transmitted), COMSEC is concerned with protecting the information itself, e.g. encrypting the transmission, so that even if the attacker gets hold of the transmission, they still cannot access the information. TRANSEC is concerned with preventing the attacker from obtaining the transmission in the first place, and preventing hostile actions, such as jamming, from interfering with the transmission channel [13].

2.5.2 COMSEC in an HF network

2.5.2.1 Scarcity of public information regarding COMSEC in an HF network

At the time of writing, the author was unable to find much published research work specifically on COMSEC in the HF network environment. One old paper [47], which discusses a digital terminal for HF, VHF, Ultra High Frequency (UHF) radio communication and/or wireline use, at least indicates that a COMSEC module forms part of the unit, but this is a terse reference at best.

This gap is somewhat expected, since details around security systems in the HF military environment are normally treated as highly confidential within the industry. This is also reflected in unclassified NATO standards, such as STANAG 4538 [26] and STANAG 5066 Edition 3 [41] (referenced as part of the discussions in sections 2.3 and 2.4), which only include interoperability specifications stating that a compliant system shall use a NATO approved algorithm and/or cryptographic hardware module (such as the older KIV-7 High Assurance Internet Protocol Encryptor (HAIPE) device [48] referenced in [41].)

Appendix E of MIL-STD-188-141D [36], “Application Protocols for HF Radio Networks”, where “application protocols” include native HF applications and Internet applications or interoperability with Internet applications, contains some discussion of where COMSEC can be implemented in relation to the OSI model. This includes the following:

1. Application layer security, that is, encryption and decryption, takes place within each application “before it is delivered to the application-layer protocol for delivery”;
2. Transport layer security (reference made to the old Secure Socket Layer (SSL) protocol and the newer Transport Layer Security (TLS) protocol);
3. Network layer security (direct reference made to IPsec tunnels); and
4. Link level encryption, i.e. encryption at layer 2 or between layer 2 and layer 1 protocols.
Notably, this is treated rather generically, with no additional details or specification given.

However, beyond mentioning these options, no additional information is provided on the implementation of COMSEC within the HF environment. This does, however, enforce the idea that COMSEC is concerned with encrypting sensitive user data, but may also provide protection for address information (or other application information) depending at which layer(s) COMSEC is performed.

As a slight aside, MIL-STD-188-141D Appendix E also includes a note stating that COMSEC key management is evolving toward the use of Public Key Infrastructure (PKI). However, this note

has been present in the standard since the first notice of change was published for MIL-STD-188-141B in August 2001 [49] without any update since. In fact, newer NATO standards, such as the draft “Interoperable Wideband Link Management” standard STANAG 5070 [6], which also does not provide much information about COMSEC or cryptographic functions other than a few functionality related restrictions, specifically prohibit the use of any OTA mechanism to exchange and/or establish keys (implies static pre-configuration).

For this reason, and based on industry experience, the author is of the opinion that MIL-STD-188-141 Appendix E listed SSL and TLS as viable COMSEC options for an HF network, without fully considering the implications.

2.5.2.2 The partial exception of STANAG 5066 Edition 4

The scarcity of openly standardised or published COMSEC information is somewhat broken by the new draft of STANAG 5066, i.e. Edition (Ed) 4 [5] which is still in the process of ratification within NATO. While the primary aim of this edition is to add support for WBHF and ALE functions, it also includes a new Annexure, Annex T, which specifies an optional crypto sublayer for the 5066 protocol stack. Note that the use of the shorthand term *crypto* for *cryptographic* or *cryptographic function* is considered acceptable in the HF industry as evidenced by its use within the 5066 standard.

Please note that this annexure views this crypto sublayer as a TRANSEC function, since it defines COMSEC and TRANSEC somewhat differently (COMSEC limited to the protection of user data, TRANSEC as “protection applied to the lowest communication level”). As mentioned in section 2.5.1 this discrepancy is expected. In keeping with the definitions given in section 2.5.1 it is discussed as a COMSEC function in this study.

The crypto sublayer specifies a framework within which different encryption algorithms can be used in a STANAG 5066 system. Specific mention is made of the Advanced Encryption Standard (AES) encryption algorithm (as published in [50]), and mappings are provided for its use within the framework. The crypto sublayer specification also notes that AES is widely used for commercial, government, and military applications, and specifically within the US government as well.

This approach highlights a key factor typical of crypto design in the HF industry, i.e. the aim is normally not to ‘reinvent the wheel’ by designing proprietary encryption algorithms. Rather, the design aim is the integration and adaptation of well-established existing encryption algorithms (or mechanisms) within the HF environment.

To aid in a short discussion of the framework provided by the crypto sublayer, Figure 2.8 provides a very high-level block diagram representation of the crypto sublayer within a STANAG 5066 capable system. Please note that this discussion aims to summarize the key points of the crypto sublayer annexure [5].

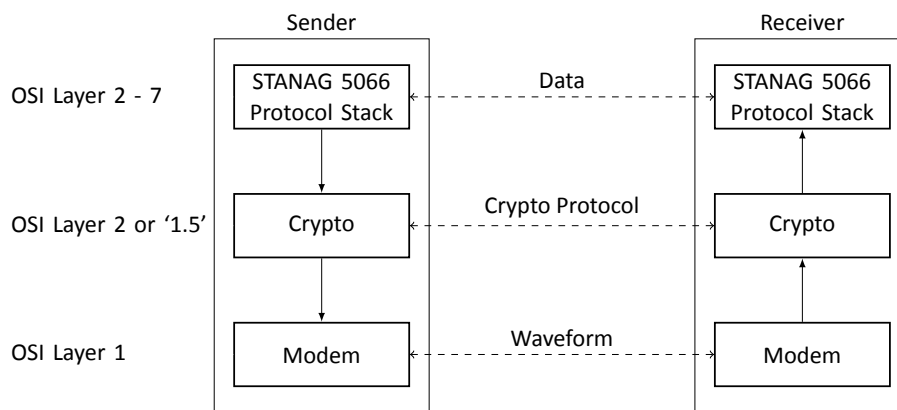


Figure 2.8: STANAG 5066 Crypto Architecture (adapted from [5])

The crypto sublayer framework includes provisioning for the following:

1. Definitions of service interfaces
2. Crypto mapping and counter mode
3. Crypto initialisation
4. Crypto layer protocol

In order to explain each of these provisions, it is important to note the characteristics of the normal transmission flow from the STANAG 5066 protocol stack to the modem. If the crypto sublayer is absent, the STANAG 5066 protocol stack provides data PDUs directly to the modem in a streaming fashion, known as a “bounded stream” (i.e., multiple data PDUs make up a single transmission stream). The transmission start is implicitly indicated by the start of the data, and the transmission end is indicated by the last data PDU. Specifically, the End of Message (EOM) marker is inserted into the data stream at the modem level when it receives the last data PDU. It is important to note that a STANAG 5066 data PDUs also includes an EOM field, but this is only used at the STANAG 5066 level and should not be confused with the modem EOM.

Furthermore, the control information (i.e., ‘transmission parameters’) is set on the modem by STANAG 5066. This includes: speed, interleaver parameters, waveform selection (such as STANAG 4285, 4529, or 4539), waveform-specific parameters, and bandwidth. In particular, this control information is not transmitted over-the-air. Rather, it simply facilitates the interaction of STANAG 5066 with the modem, and allows the use of features such as ALE and variable data rate.

2.5.2.2.1 Definitions of service interfaces

STANAG 5066 Annex T aims to provide a data interface to the bottom of the STANAG 5066 stack that is essentially the same as that provided by the modem when no crypto is present. Conversely, the crypto should provide an interface to the modem that is the same as the bottom of the STANAG 5066 stack (details around the interface are provided in STANAG 5066 Annex D).

However, the introduction of the crypto sublayer impacts both the data stream and the control information flow. The data stream can mostly be encrypted and decrypted as-is, although the crypto protocol will introduce some overhead (see the crypto layer protocol discussion in section 2.5.2.2.4). The STANAG 5066 stack must be aware of this overhead for transmission length

calculations and for (optional) alignment of data PDUs with modem block boundaries (i.e., matching PDU size plus crypto overhead with modem waveform transmission block size).

The main body of STANAG 5066 makes provision for the use of a 'Crypto Bypass' (in section 3.2) to facilitate the control information flow between STANAG 5066 and the modem. In principle, this bypass mechanism allows control information to reach the modem without being encrypted by the crypto sublayer and vice versa. However, details and/or specifications for the bypass itself are considered to be outside the scope of STANAG 5066.

2.5.2.2.2 Crypto mapping and counter mode

Since STANAG 5066 data transmission is handled in a stream fashion, the crypto needs to operate as such as well (as opposed to a pure block-oriented encryption algorithm). STANAG 5066 Annex T provides for the use of block-oriented ciphers in Counter (CTR) mode to achieve this (e.g., AES could be used in Galois/Counter Mode (GCM)).

The GCM mode for block ciphers is specified in the United States' NIST Special publication 800-38D [51]. The underlying symmetric key block cipher (e.g. AES) specifies two cipher functions, namely the forward and the reverse cipher function (inverses of each other; uses the same cipher key). GCM only makes use of the forward cipher function, but in doing so provides two functions of its own: authenticated encryption and authenticated decryption.

GCM uses the same cipher key (or key for short) that would be used by the block cipher (with additional requirements regarding key length - at least 128 bits - and key generation). In addition to the cipher and key, the authenticated encryption function requires the following input:

1. Plaintext;
2. Additional Authenticated Data (AAD)
Generally speaking this could be protocol header- or metadata, such as addresses, port numbers, and/or the protocol version. However, STANAG 5066 Annex T makes no provision for AAD requirements; and
3. an Initialization Vector (IV)

Both plaintext and AAD are authenticated by GCM, but only the plaintext is encrypted, or as [51] states it "GCM also protects the confidentiality of the plaintext".

Inclusion of the IV is the defining factor that enables the operation of the CTR mode. It has to be unique for each invocation of the authenticated encryption function with the same key, i.e. a nonce value for each block of plaintext that has to be encrypted. This ensures that the resulting ciphertext is unique even if repeating plaintext blocks are encrypted with the same key.

This uniqueness requirement is quantified in section 8 of [51] as follows: "The probability that the authenticated encryption function ever will be invoked with the same IV and the same key on two (or more) distinct sets of input data shall be no greater than 2^{-32} ".

In the simplest form of IV construction (termed 'deterministic construction' by [51]), the IV consists of a fixed field and an invocation field. The fixed field has to be unique for each device (i.e. every device that makes use of the same key), but remains constant for the duration that the key is used. The invocation field is changed after each IV use, that is, after each invocation of

the GCM authenticated encryption function, and could simply be a counter that is incremented (hence the term ‘CTR mode operation’) or something slightly more random, such as the output of a Linear Feedback Shift Register (LFSR).

A slightly more complex construction method (which employs Random Bit Generator (RBG) functions as discussed in section 8.2.2 of [51]) may be used only if the IV length is 96-bits or greater.

Generally speaking, a longer IV allows the use of the same key for a longer period of time, since the key must be changed once all possible IV values have been exhausted. NIST [51] mandates that the IV be a byte string (i.e., length should be multiple of 8) and not longer than $2^{64} - 1$ bits. However, for general interoperability and practical ease of implementation, [51] recommends using an IV length of 96-bits.

2.5.2.2.3 Crypto initialisation

STANAG 5066 Annex T considers two categories of crypto initialisation information. The first relates to information that has to be known prior to data transmission (e.g., pre-shared keys), and the second to information needed for each transmission, i.e., the IV in the case of AES-GCM.

The first group is not addressed in Annex T, except stating that the “distribution of AES keys and nonces is not specified in this annex”. The second group is addressed by the inclusion of a 2 byte Key/Nonce pair reference in the crypto sync PDU (see section 2.5.2.2.4), as well as the IV itself.

The specification around the IV is fairly straightforward: the length is fixed at 8 bytes (slightly shorter than the 96 bits recommended by [51], presumably to limit the OTA performance impact of crypto overhead) and the IV has to be unique for the key/ nonce pair.

In particular, there appears to be a partial gap in the STANAG 5066 specification regarding the uniqueness of the IV. While Annex T states that IV uniqueness can be achieved by incrementing the IV value or by use of a LFSR (similar to [51]), it does not provide guidance to ensure that the IV is unique if multiple (i.e., more than two) HF nodes share the same AES key.

In other words, no unique fixed portion of the IV field for each device is considered as set out in [51], allowing different node pairings to use the same IV if the same key/nonce pair is used. This is a security risk, since reuse of an IV value for the same key can be exploited by an attacker if both ciphertext data blocks are obtained and the attacker is aware that they were encrypted with the same key/nonce pair. The nature and extent of possible exploits fall outside the scope of this study, but the reader is referred to Appendix A in [51] for additional information in this regard.

This risk can be mitigated by proper management of the key/nonce pair reference across all devices in a radio network, but STANAG 5066 fails to mention the importance of this and management of the reference is specifically excluded from Annex T.

The last aspect of crypto initialisation addressed in STANAG 5066 Annex T relates to the possibility of corruption occurring during the OTA transfer of initialisation information. In short, this is achieved by providing for the use of an extended crypto sync PDU (see section 2.5.2.2.4).

This extended PDU includes a two-byte Maury-Styles pair, also used for normal data PDU sync as

specified in STANAG 5066 Annex C, for crypto stream synchronization purposes, and a counter which can be used to determine the amount of preceding data lost and the correct starting point at which the received data should be decrypted (relatively simple Exclusive OR (XOR) operation).

2.5.2.2.4 Crypto layer protocol

STANAG 5066 Annex T defines two generic Crypto Sync PDUs which constitute the crypto layer protocol. The normal crypto sync PDU format is given in Figure 2.9, and the extended crypto sync PDU is given in Figure 2.10.

The Version field is a 4-bit field that specifies the version of the crypto protocol used. The first version of this protocol (defined by STANAG 5066 Edition 4) is set at zero (0). The Algorithm field is a 4-bit field that specifies the crypto algorithm used. Annex T does not specify any fixed algorithm numbers, except to define AES as algorithm zero (0).

The Crypto Sync field is allocated a block of n bytes, details of which are to be determined for each crypto algorithm used.

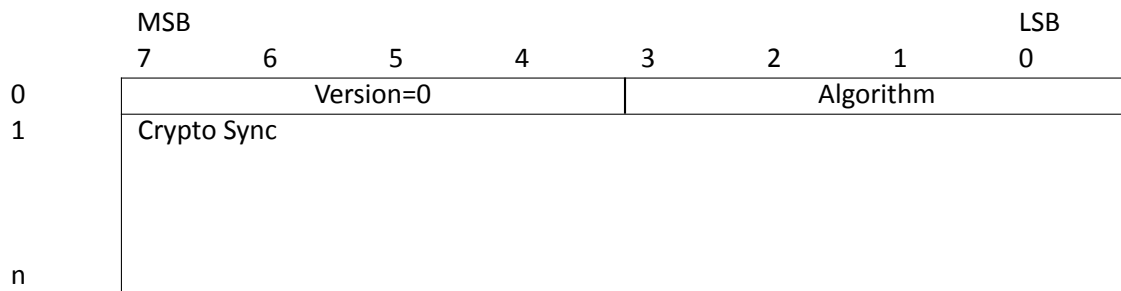


Figure 2.9: Crypto Sync PDU as specified in STANAG 5066 Annex T

The extended crypto sync PDU introduces two additional fields: the Maury-Styles field, and the sync counter field. Both of these fields are used for synchronisation purposes, as briefly discussed in section 2.5.2.2.3.

The Maury-Styles field (given at the start of the PDU) is a fixed 2-byte value (0x90EB), normally used for data PDU synchronisation as specified in STANAG 5066 Annex C, which is used to achieve crypto stream synchronisation (sync at start of crypto PDU to allow crypto stream decoding and unpacking if modem synchronisation fails).

The sync counter field is a 2-byte field that indicates the extended crypto PDU number for a given transmission (i.e., the first crypto sync PDU is numbered one (1), the second two (2) and so forth). This allows the receiver to determine if any data was lost during the transmission and makes it possible to calculate the correct point at which received data should be decrypted (i.e., at which point the decryption XOR operation should be started).

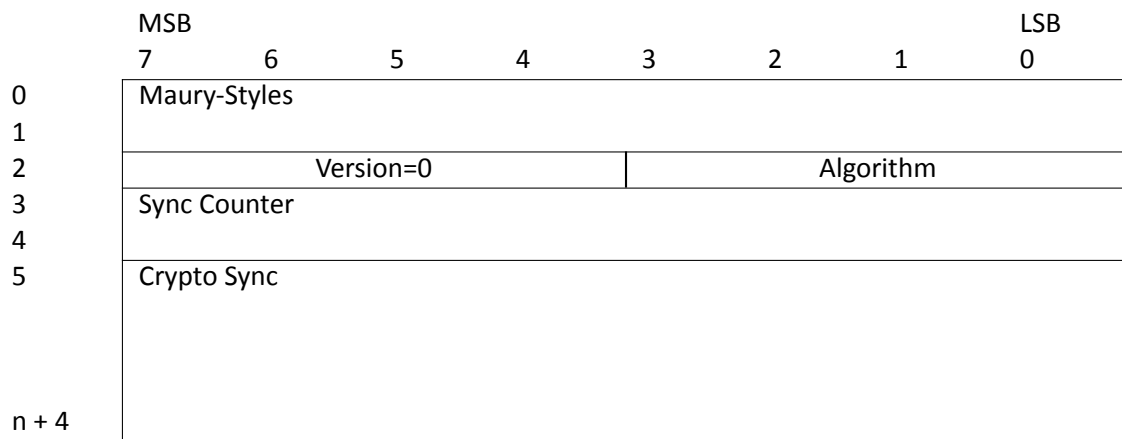


Figure 2.10: Extended Crypto Sync PDU as specified in STANAG 5066 Annex T

STANAG 5066 also specifies the PDU format (or more accurately the 'Crypto Sync' field format) for the AES algorithm. This is given in Figure 2.11.

The AES crypto sync field consists of two parts: the key/nonce reference and the IV field. The Key/Nonce reference is a 2-byte field used to uniquely identify the cipher key/IV pair used to encrypt the data following the sync PDU, and the IV field is an 8-byte field containing the IV itself.

The uniqueness aspect of both the IV and the key/nonce reference is discussed in section 2.5.2.2.2 and 2.5.2.2.3.

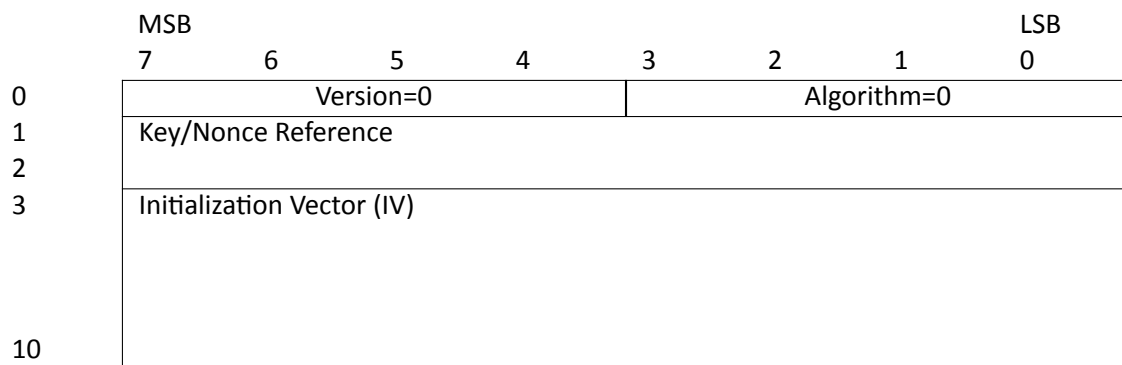


Figure 2.11: Crypto Sync PDU for AES as specified in STANAG 5066 Annex T

The crypto sync- and extended crypto sync PDUs are used to transfer encrypted data as follows:

1. During the first 4096 bytes of encrypted transmitted data, a (normal) crypto sync PDU is inserted before every 256 bytes of data. This repetition is necessary to protect against loss of the crypto sync PDU due to transmission corruption.
2. The last block of data may be less than 256 bytes if fewer than 4096 bytes in total are transmitted.
3. If more than 4096 bytes are transmitted, an extended crypto sync PDU is inserted before each data block of 4096 bytes after the initial 4096 bytes. In other words, the first extended crypto sync PDU is inserted in the data stream after the last 256 byte block of data (of the

initial 4096 bytes of data), and then only after every block of (subsequent) 4096 byte data blocks.

It is worth highlighting that the STANAG 5066 Annex T crypto protocol has to provide synchronisation capabilities (briefly mentioned throughout the protocol discussion), since STANAG 5066 itself functions in a streaming fashion and makes use of streaming modem waveforms. Therefore, if a receiving unit (radio) misses the start of the HF transmission but starts receiving the data halfway through the transmission (modem waveforms allow for this late entry possibility), some mechanism is required to synchronise the crypto stream at the receiver side for decoding and decryption purposes.

2.5.3 TRANSEC in an HF network

As mentioned in section 2.5.1, TRANSEC relates to security mechanisms that aim to prevent a potential attacker from obtaining the HF transmission, or from interfering with the transmission channel (e.g., jamming) [13], [52]. Since the aspects of TRANSEC security are not central to this study, a deep dive is considered unnecessary. However, some techniques are briefly mentioned for completeness sake.

The first group of TRANSEC mechanisms can be labelled as Low Probability of Intercept (LPI) schemes [13]. This relates to reducing transmission length and/or segmenting the transmission over multiple frequencies.

The second group can be labelled Low Probability of Detection (LPD) schemes [13]. This relates to reducing transmission power and or masking the signal with natural noise by spreading the transmission power over a wider bandwidth.

A common TRANSEC technique (which can probably be viewed as both an LPI and an LPD scheme) is Frequency Hopping (FH) [13], [52]. This technique functions at the physical layer by rapidly switching the transmission (carrier) frequency in a pseudo-random sequence, thereby using multiple channels and a larger bandwidth (as compared to continuous transmission on a single frequency).

Although using FH adds to system complexity, i.e., the transmitter and receiver need to synchronise channel switching according to the same hopping sequence, devices need to be able to switch frequencies and perform antenna impedance matching fast enough to keep up with the FH switching time, and so forth, it is traditionally deemed to provide a strong level of TRANSEC [52], thereby justifying the trade-off. However, emerging research, such as [52], proposes techniques that allow an attacker to intercept FH fairly quickly (within 20 hops) using new generations of Software Defined Radio (SDR) equipment.

Other TRANSEC options include burst encoding and spread spectrum techniques.

2.5.4 A note on Linking Protection (LP)

Another security mechanism commonly employed in ALE HF networks is known as Linking Protection (LP) [13], [26], [36].

The original aim of LP when it was first introduced in the 1990s was to provide an authentication mechanism to protect against vulnerabilities in the ALE protocol such as unauthorized net entry,

denial of access to legitimate users, and protection of address information [53]. This aim has remained largely unchanged, although LP has been improved as the corresponding HF ALE standards have been improved or new ones developed, for example, 3G and 4G HF [26], [36].

At its core, LP makes use of scrambling techniques to provide protection during the LSU phase of an HF link, that is, not directly applicable to the data transfer phase of an HF link. Notably, provisions for periodic LP key changes (in the form of protection intervals) enables LP to protect the HF network against tape replay attacks, that is, an attacker cannot simply replay a previously recorded link setup or link termination transmission to interfere with network communications [13].

Additional details around LP are deemed unnecessary for the purposes of this study.

2.6 Security for IP networks

2.6.1 Common security protocols

Perhaps the two most used security protocols throughout the Internet (i.e. the world's largest IP-based network) is Transport Layer Security (TLS) (the successor of Secure Socket Layer (SSL)) and the Internet Protocol Security (IPsec) protocol suite. These protocols provide security at the transport- and network layer respectively [18], [29]. Both of these attempt to provide data confidentiality, i.e. encryption and decryption of sensitive data, and authentication functionality, i.e. ensuring the data integrity can be trusted and that the sender and receiver are who they claim to be.

For completeness, it is worth noting that there are a host of other methods (or protocols) employed throughout the Internet to provide security in various ways. For example, user access control mechanisms are usually employed above and beyond lower layer security protocols whenever sensitive data needs to be protected, ranging anywhere from banking to social media websites [18].

However, these methods are considered too far out of scope of this research study's aim, viz. developing an IP cryptographic function for the HF radio environment, to warrant further consideration in this literature study. Note that the specific research study goals are presented in section 1.3.

For a similar reason, TLS is also not investigated further in this literature study. Since this research study focusses on developing a cryptographic function operating at the IP protocol level (OSI layer 3), TLS, or other security mechanisms operating at the transport layer are not considered applicable to this study.

2.6.2 Internet Protocol Security (IPsec)

2.6.2.1 Overview of the IPsec protocol suite

Although IPsec is often referred to as a protocol, it actually consists of a suite of protocols as outlined in the IPsec roadmap, RFC 6071 [54]. In other words, IPsec is not technically a protocol in the same sense as other common Internet protocols such as TCP or UDP. However, for ease of writing and in accordance with the base IPsec architecture, RFC 4301 [8], the singular term (or acronym) IPsec is used to refer to the protocol suite.

The main aim of IPsec is to provide security at the network layer for IP based systems. This typically translates to providing secure gateway-to-gateway (e.g., between two routers) and host-to-gateway (e.g., between an endpoint and a router) communication (common Virtual Private Network (VPN) use cases), but IPsec can also be used for host-to-host communication security [54].

The interrelationship diagram between IPsec and Internet Key Exchange (IKE), the main protocol used for automated key management in an IPsec system, given in RFC 6071, provides a good picture of the various components of the IPsec protocol suite [54]. It is repeated in Figure 2.12 to provide a departure point for the rest of the discussion around IPsec.

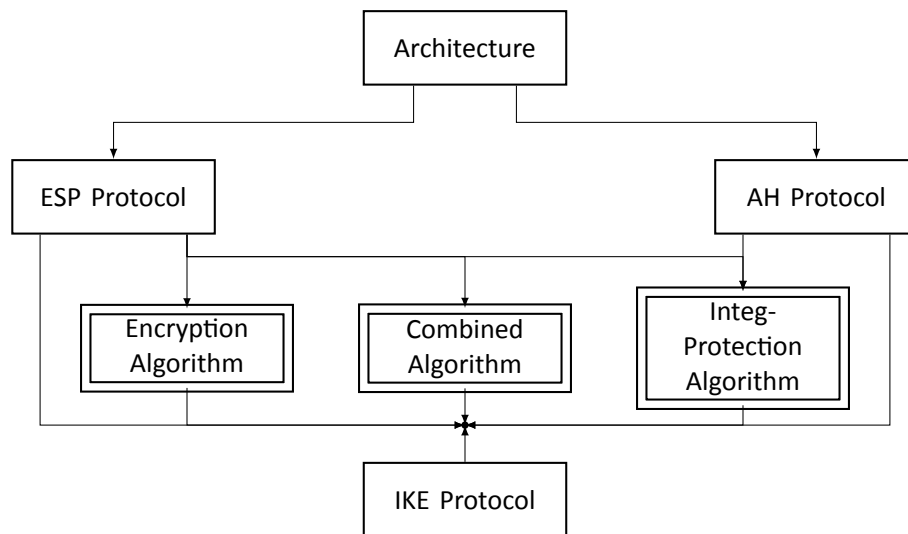


Figure 2.12: Interrelationship of IPsec and IKE documents (adapted from RFC 6071 [54])

2.6.2.2 Modes of operation

IPsec, or more specifically one of the two security protocols, Authentication Header (AH) or Encapsulating Security Payload (ESP), provides security at the network layer in one of two main operation modes: transport mode or tunnel mode [8], [29]. Both of these modes provide protection of the entire transport layer datagram (i.e., the entire IP packet payload is protected), but only tunnel mode provides protection for the original IP header.

It is important to note that strictly speaking, AH applies to the immutable parts of the IP header in transport mode as well. However, since AH does not provide encryption (confidentiality protection), this is limited to “integrity and data origin authentication” protection (section 1 of RFC 4302, [55]).

These modes of operation are illustrated in Figures 2.13 and 2.14. It is important to note that IPsec operates between the transport and network layer when in transport mode, but partitions the network layer when operating in tunnel mode.

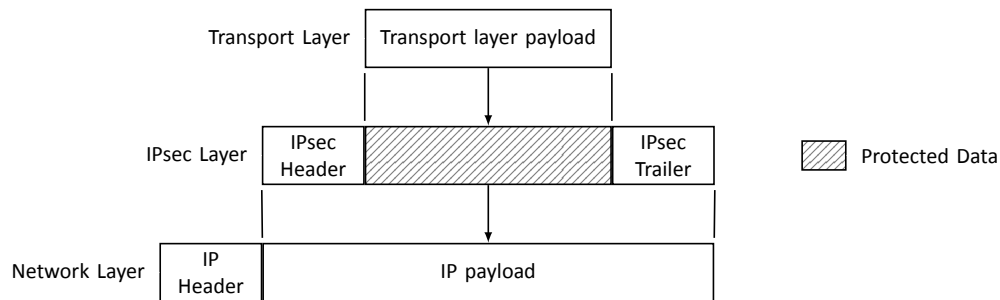


Figure 2.13: IPsec Transport Mode (adapted from [29])

Transport mode is typically used for end-to-end security, while tunnel mode is normally used for gateway-to-gateway security. However, there are cases where tunnel mode may be used between two hosts, and transport mode may be used between a host and a gateway, or between two gateways (see section 4 in [8] for applicable conditions and exceptions).

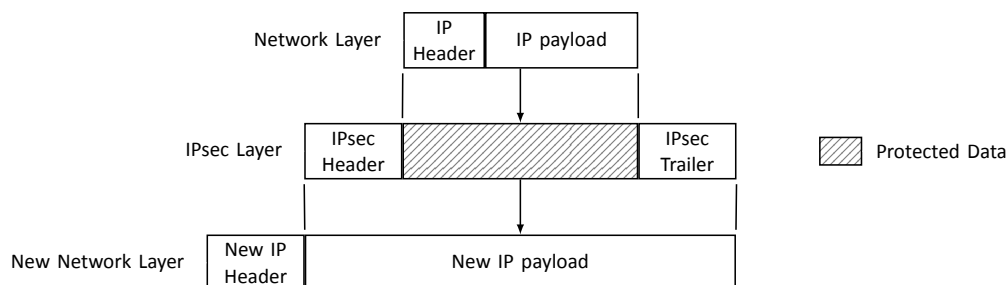


Figure 2.14: IPsec Tunnel Mode (adapted from [29])

2.6.2.3 Base IPsec architecture

At its core, IPsec functions by inspecting IP packets and performing processing actions based on IP packet and “next layer header” information (termed “selectors” in RFC 4301). This is done primarily by performing a lookup in the Security Policy Database (SPD) for each IP packet, both incoming and outgoing, and performing the action specified by the matching entry [8]. Therefore, it might be helpful to think of the SPD as similar to a normal IP Access List (ACL), at least from a high-level functional or usage point of view.

Each entry (or policy) in the SPD maps certain selector criteria to one of three actions: PROTECT, BYPASS, or DISCARD. Policies are determined (and/or managed) either directly by a system administrator or similar user, or by some application that performs the same function [8].

Figure 2.15 illustrates the top-level (conceptual) processing model employed by IPsec as given in section 3.1 of RFC 4301 [8].

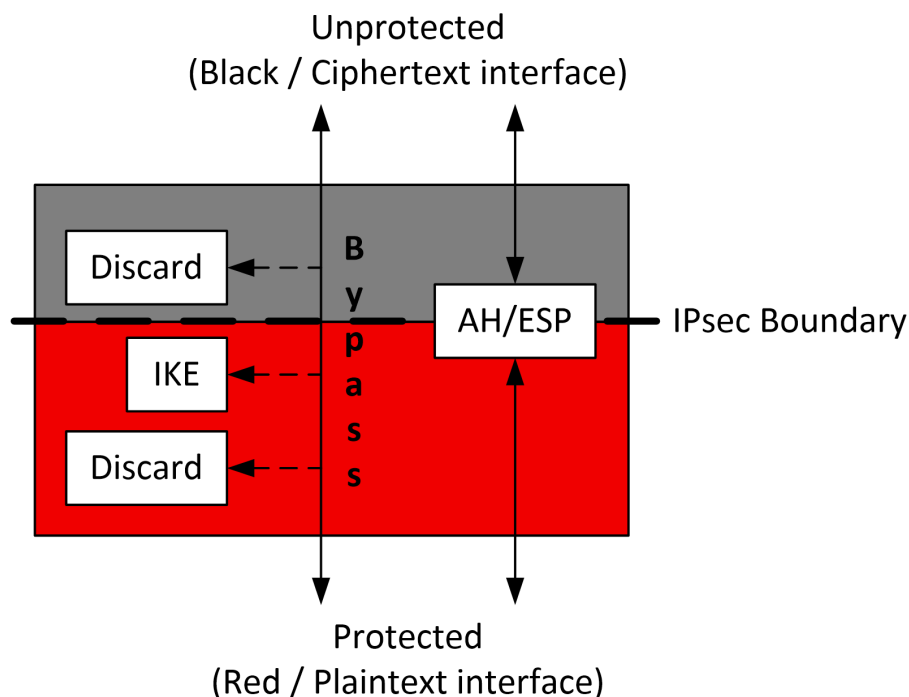


Figure 2.15: High level IPsec processing model (taken from section 3.1 in RFC 4301 [8])

The actual sequence of events is better explained by considering outbound- and inbound IP traffic processing separately, as is done in RFC 4301 section 5. Here, *outbound* represents traffic flow from the protected (red) interface to the unprotected (black) interface (section 2.6.2.3.1), and *inbound* represents traffic flow in the opposite direction (section 2.6.2.3.2).

2.6.2.3.1 Outbound IP traffic IPsec processing

The outbound traffic processing flow is given in Figure 2.16. Each processing block is described in a bottom-to-top sequence (packet flow sequence) in the following paragraphs, with the exception of the ICMP block, which is left till the end so as not to break the flow of the discussion. Note that this discussion draws from various sections of RFC 4301 [8], but primarily from the RFC discussion on outbound processing (section 5.1) and loosely follows the same outline and

approach used there.

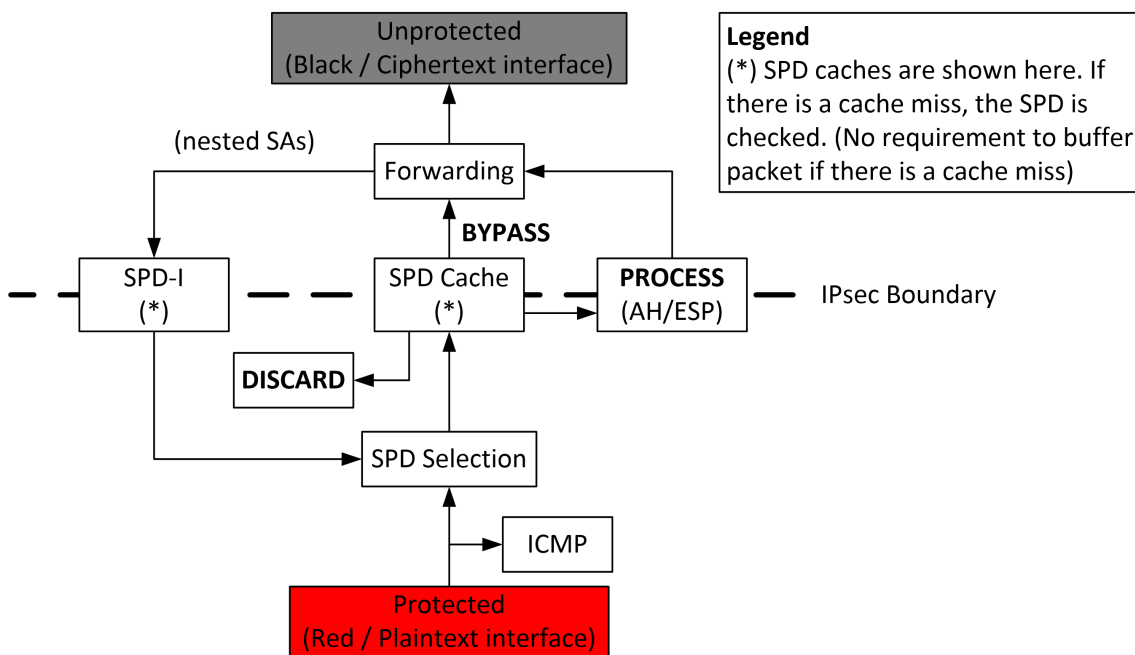


Figure 2.16: Outbound IP traffic IPsec processing model (adapted from section 5.1 in RFC 4301 [8])

SPD Selection If an IPsec implementation supports the use of multiple SPDs, the first step in outbound processing is to select the appropriate SPD that the packet should be checked against. Note that this can be a no-op if multiple SPDs are not supported. For example, multiple SPDs may be used if the IPsec implementation chooses to maintain separate SPDs for each network interface. The output of this selection function is an SPD Identifier (SPD-ID).

SPD Matching (SPD Cache) The appropriate SPD cache is selected based on the SPD-ID. Thereafter, a lookup is performed based on the packet header (i.e., based on the selector information) to determine the appropriate action. As mentioned in the legend in Figure 2.16, if a cache miss occurs, the SPD itself is checked.

At this point, it is also worth noting that the SPD is split (at least logically) into three parts: the SPD Inbound (SPD-I), SPD Outbound (SPD-O), and SPD Secure Traffic (SPD-S). The SPD-I contains policy entries for inbound traffic that should be discarded or bypassed, while the SPD-O contains entries for outbound traffic that should be discarded or bypassed. The SPD-S contains entries for packets (inbound and outbound) that are subjected to IPsec protection.

An IPsec implementation may choose to decorrelate these SPD parts for caching purposes, i.e., to improve lookup performance. As should be evident by now, the outbound processing flow discussed in this section assumes that these caches are in place.

SPD Matching - Cache Hit If a match is found in the cache, the appropriate processing action is taken: discard, bypass, or protect (using AH or ESP). A *protect* entry in the SPD cache implies that a valid IPsec Security Association (SA) already exists for the matching packet, which means that an AH or ESP packet can be generated without requiring any other external input or interaction,

such as interaction with the IKE protocol to first negotiate a valid SA.

The notion of a SA, and the accompanying role of the Security Association Database (SAD), has purposefully been omitted up until this point, with the aim of first providing some context of the basic IPsec processing model to better facilitate the explanation. SAs are discussed below as part of the 'SPD cache miss' condition and the 'Process' discussion.

SPD Matching - Cache Miss If no match is found in the cache, the SPD-S and SPD-O is searched. If the matching SPD entry specifies a bypass or discard processing action, the relevant entries are created in the SPD-O cache. According to section 4.4.1 of RFC 4301 [8], the expectation is that each SPD should have a final catch-all entry to discard all traffic that does not match any other SPD entry.

Additionally, for a bypass action, an entry is created in the SPD-I cache. The nature of this SPD-I entry is not explained clearly in section 5.1 of RFC 4301, but the discussion of SPD entries in section 4.4.1 explicitly states that separate inbound and outbound entries are supported for discarded and bypassed traffic (i.e., traffic flow can be uni-directional). However, it is unclear how this would be any different from simply requiring the system administrator to explicitly create the entries in the SPD, especially considering that there is no directionality field or flag in an SPD entry itself, in which case the author is of the opinion that no automatic creation of an SPD-I cache entry should occur.

Regardless, if the SPD-I cache entry is created as indicated in section 5.1 of RFC 4301, it either has to be an "inverted but matching" bypass entry, i.e., source and destination addresses and/or ports swapped from the SPD-O entry for bidirectional traffic flow, or an "inverted but matching" discard entry for unidirectional flow (i.e. only outbound traffic flow allowed).

Returning to the original SPD search action (after a cache miss), if the search matches an entry with a protect processing action, i.e. a match in the SPD-S, the SA creation process is started. This involves using the key management mechanism, usually the IKE protocol, to negotiate the SA parameters and instantiate the SAs used for this traffic flow between two IPsec implementations (IPsec peers), i.e., the local and remote node specified in the SPD-S entry. If the SA creation process is successful, entries are created in the SPD-S cache and the SAD. Both inbound and outbound SAD entries are created if the IKE protocol is used, since SAs are always created in pairs ("inverted but matching") by the IKE protocol (most common use cases require bidirectional IP traffic flow between nodes - e.g. TCP traffic).

Process (AH or ESP) Once the SPD lookup matches with a valid SA for the outbound IP packet, the packet is protected by the associated security protocol, i.e. an AH or ESP packet is generated from the original IP packet. This is perhaps better understood by explaining the notion of an SA in greater detail.

RFC 4301 [8], section 4.1, defines an SA as "a simplex 'connection' that affords security services to the traffic carried by it". In essence, an SA keeps track of the necessary parameters and meta-data needed by the specified security protocol (AH or ESP) to effect protection of the IP packet. This includes parameters such as the Security Parameters Index (SPI), sequence number counter, the authentication, encryption, integrity, or combined mode algorithm used, IV values, the cryptographic key, and so forth.

The SPI parameter is transmitted as part of the AH or ESP header and is used to identify the SA (especially needed at the receiving side when processing inbound AH/ESP packets). The rest of the working of the AH and ESP protocols are discussed in section 2.6.2.4.

Details of the SPD and SAD entries are presented in section 2.6.2.3.3, which should assist in understanding the role of the SPD and SAD, respectively. It may also prove helpful to the reader to cross-reference this section with section 2.6.2.3.3 before proceeding.

Forwarding The forwarding function represented here actually exists outside of the IPsec implementation. It encompasses the normal forwarding function of the device on which IPsec is implemented (e.g. packet forwarding implementation in a router or a host device's Operating System (OS) network stack).

As a slight aside, it should be noted that the specification of the forwarding function falls outside the scope of the IPsec processing model (i.e., outside the scope of RFC 4301). However, based on the author's reading of RFC 4301, and as directly assumed in Appendix B of RFC 4301 [8], it is assumed that the forwarding function gives effect to some sort of separation between handling protected- and unprotected side forwarding. This is probably achieved by either maintaining separate forwarding tables for the two sides but still using the same forwarding instance, or by making use of two completely separate forwarding function implementations and/or instances.

Regardless, the forwarding block in Figures 2.16 primarily serves to indicate that a packet is forwarded as normal after IPsec processing is completed. However, it is also possible that a forwarding rule may exist to send the outbound packet back to the IPsec implementation for additional processing.

This mechanism is primarily supported to allow for nesting of SAs, e.g. if both AH and ESP should be applied to a packet, or if a transport mode ESP packet should be encapsulated in a tunnel mode ESP packet. Additional information on nested SAs is considered unnecessary for the purposes of this discussion.

ICMP The ICMP processing block, located on the protected side of the IPsec boundary, represents the processing of ICMP messages directed to the IPsec implementation (node) that were generated within the protected network.

Section 6 of RFC 4301 deals with ICMP processing and the associated risks, such as a compromised host sending erroneous ICMP redirect error messages, in depth, along with specific requirements related to the management and/or configuration of an IPsec implementation's handling of ICMP messages. It is deemed unnecessary to explore this further for the purposes of this research study.

2.6.2.3.2 Inbound IP traffic IPsec processing

The inbound traffic processing flow is given in Figure 2.17. Each processing block is described in a top-to-bottom sequence (packet flow sequence) in the following paragraphs. Similar to the outbound processing discussion (section 2.6.2.3.1), this discussion draws from various sections of RFC 4301 [8], but primarily from the RFC discussion on inbound processing (section 5.2) and loosely follows the same outline and approach used there.

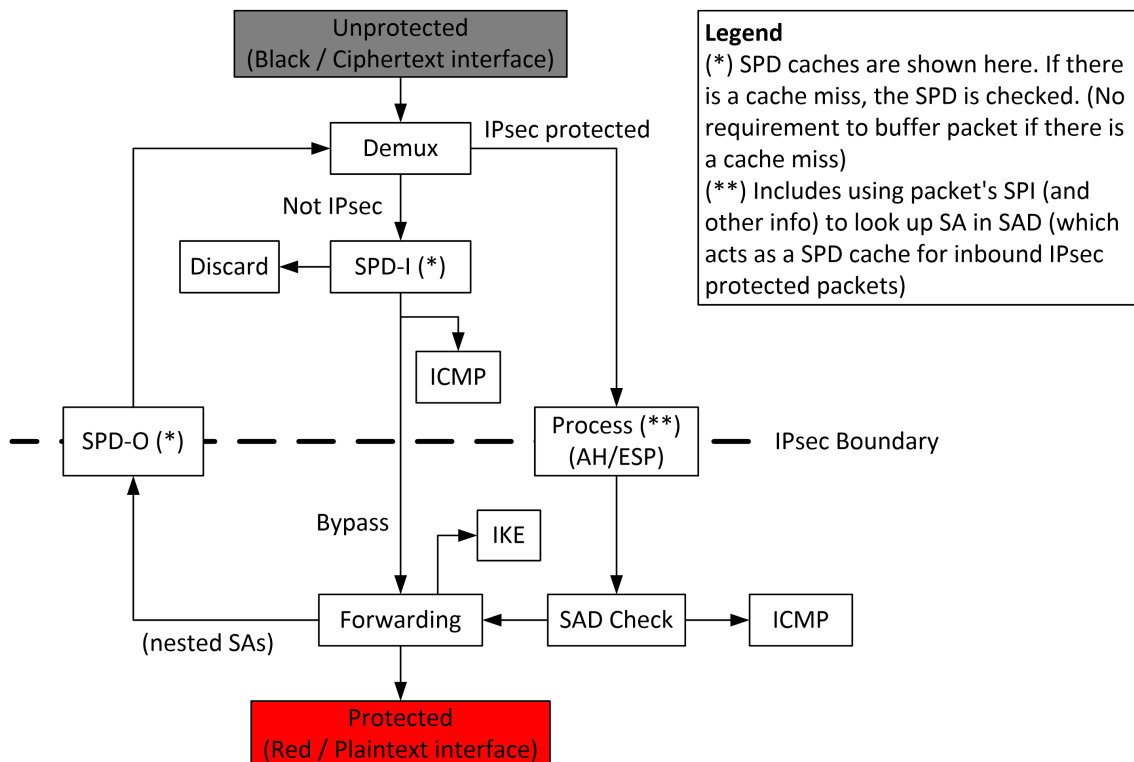


Figure 2.17: Inbound IP traffic IPsec processing model (adapted from section 5.2 in RFC 4301 [8])

Demux The first operation for an inbound IP packet is to determine if the packet is IPsec protected (i.e., AH or ESP) and addressed to the IPsec implementation (node or device). If so, it is subjected to IPsec protocol processing. If not, the packet is subjected to an SPD-I lookup. This includes both IPsec traffic, AH or ESP, *not* addressed to the IPsec device, and all other IP traffic.

IPsec traffic - Process (AH or ESP) Prior to the actual AH or ESP processing, the IPsec protected packet is subjected to a lookup in the SAD (not shown in Figure 2.17). For normal unicast IP packets, this lookup simply makes use of the SPI field. If no match is found (keeping in mind that the demux action has already determined that the IPsec packet is addressed to this IPsec implementation), the packet has to be discarded.

If a valid SA match is found, the packet is processed accordingly, i.e. the original IP packet is recovered from the AH or ESP packet. Thereafter, the recovered packet is subjected to the 'SAD check' processing block.

SAD Check The SAD check is used to verify that the recovered packet matches the selectors associated with the SAD. This serves as a sort of final validity check of the recovered IP packet, i.e., is the recovered packet allowed on the matched SA?. If there is a mismatch in any of the selector parameters, the packet has to be discarded.

Generally, the IPsec implementation should be able to notify the remote IPsec device (IPsec peer) of this failure. This is possible via a "invalid selectors" message from the IKE protocol, hence the IKE processing block on the protected side of the IPsec boundary.

Forwarding After a packet has passed the SAD check, it is sent to the normal forwarding function.

As discussed in the outbound processing discussion (section 2.6.2.3.1) the forwarding function actually falls outside the scope of the IPsec security function.

As with the outbound forwarding function, it is possible that a packet may be sent back to the IPsec implementation (in this case it will trigger a lookup in the SPD-O) in support of nested SAs. Please refer back to section 2.6.2.3.1 for more information on the forwarding function(s).

Protected-side ICMP processing When considering ICMP traffic received via an SA, i.e. the ICMP traffic itself was generated by a host or device in the protected-side network of the IPsec peer, there can be some notable exceptions to the inbound processing sequence. This mostly relates to the SAD check and subsequent forwarding decision of the ICMP traffic. (Hence the ICMP block on the protected side of the IPsec boundary).

Section 6.2 of RFC 4301 outlines two possible processing actions (which also has some bearing on the outbound processing flow at the IPsec peer): process the ICMP traffic as normal (which requires the SAD to have an entry which will match the ICMP packet header), or unpack the ICMP packet and check the header of the IP packet contained in the ICMP payload against the SAD.

The first action presents a significant security risk in certain IPsec implementation contexts, presumably any scenario where tunnel mode SAs are used and at least one of the IPsec peers is a gateway device, although other applicable contexts may also exist, especially pertaining to Denial of Service (DoS) attacks by misusing ICMP “destination host unreachable” messages. This specific attack vector would require the attacker to gain access to the protected side network of the IPsec peer in order to use or misuse the SA between the two IPsec peers.

The second action provides protection for these types of security risks, as the ICMP payload itself is inspected and verified against the SAD. However, it is not required in all IPsec contexts.

Regardless, section 6.2 in RFC 4301 mandates support for both processing actions and mandates that the second action be executed if no SAD entry exists for the ICMP header itself. Therefore, an IPsec administrator can control the ICMP processing actions taken by adding SPD entries for ICMP traffic (at the outbound IPsec peer) if desired.

The presence of these SPD entries will cause SA creation to take place for the ICMP traffic itself, eventually leading to the first processing action taking place at the receiver (inbound IPsec peer). In the absence of these SPD entries, the outbound IPsec peer will send the ICMP traffic on the SA that matches the IP packet header in the ICMP payload. This will effectively cause the second processing action to take place on the receiving side.

Non-IPsec traffic - SPD-I lookup As previously mentioned in section 2.6.2.3.1, the SPD-I contains entries for traffic that should be bypassed or discarded. If no match is found in the SPD-I cache, the SPD-I itself is checked. If a match is found, the appropriate action is taken and the relevant cache entries are created.

The SPD-I is expected to contain a final entry to discard all traffic that is otherwise unmatched. Note that, in contrast to the outbound processing flow, there is no condition that will cause the receiver to initiate IPsec SA negotiation and creation (i.e., no ‘protect’ entries exist in the SPD-I).

Unprotected side ICMP processing This primarily encompasses unencrypted ICMP traffic (re-

ceived from the unprotected network) addressed to the IPsec implementation (device or node) itself (not to be confused with ICMP traffic received via an SA).

RFC 4301 expects that this traffic is first processed on the unprotected side of the IPsec boundary. As with the case for ICMP traffic received on the protected side of the IPsec boundary (see section 2.6.2.3.1), it is deemed unnecessary to present an in-depth discussion of this for the purposes of this study.

Section 6 of RFC 4301 describes the potential security risks and performance impact associated with processing, partial processing, or rejecting these ICMP messages. The reader is referred to it for more information.

2.6.2.3.3 Structure of SPD and SAD entries

After the discussions on outbound and inbound IP traffic processing in sections 2.6.2.3.1 and 2.6.2.3.2, the high-level role of the SPD and SAs within the IPsec processing model should be clear. It is perhaps worth expressly stating that the SAD itself was not highlighted in these discussions, since it simply serves to contain SA information, i.e., each SA is represented as a single entry in the SAD.

However, some of the details around the relation between the SPD and accompanying SAD entries, especially pertaining to packet selectors and how these form part of an SPD entry and an SA, warrant further discussion. It may also prove helpful to cross-reference this section when working through the outbound and inbound IP processing model discussions in sections 2.6.2.3.1 and 2.6.2.3.2.

An SPD entry is essentially equivalent to a mapping of an IPsec action (discard, bypass, or protect) to a combination of selectors. The structure of an SPD entry is presented in Table 2.4.

It is important to note that the SPD is an ordered database, which allows IPsec administrators to control the sequence in which packets are matched against entries (once a entry is matched, the lookup is stopped and the specified IPsec action is applied). This is especially important when considering that several selectors support the use of value ranges and/or special values, including a wildcard value, as indicated in Table 2.4.

The two special values provided for in RFC 4301 are labelled ANY and OPAQUE. OPAQUE is used to indicate that a particular selector is not available for a packet, as would be the case for fragmented packets, encrypted packets (in the case of nested SAs), or if the particular selector does not exist in a certain next-layer protocol. ANY is the wildcard value, which will lead to a valid match for literally any value of a selector, or the absence of the selector, i.e., ANY includes OPAQUE.

Table 2.4: Structure of an SPD entry (based on section 4.4.1 of RFC 4301 [8])

Field	Description
Name	(Optional; quasi-selector) Identifier (ID) or list of IDs. This field is applicable for special use case scenarios during IKE negotiation. Several forms of names must be supported as outlined in section 4.4.1.1 of RFC 4301.
PFP flags	One flag per traffic selector. If set, the traffic selector value for a new SA entry is obtained from the packet that causes the initiation of the SA creation instead of the selector value specified in the SPD entry. Flags exist for: • Local address • Remote address • Next-Layer Protocol • Local port (or ICMP message type (code) or Mobility Header type) • Remote port (or ICMP message type (code) or Mobility Header type)
Selector sets (1 to N)	Each selector set corresponds to a condition that, if matched, indicates that the specified IPsec action must be applied to a packet (see “Processing info” below). Each selector set consists of: • Local address (single address, list of addresses, or range of addresses) • Remote address (single address, list of addresses, or range of addresses) • Next-Layer Protocol (single value, list of protocols, ANY, or OPAQUE) • Additional selectors depend on the next layer protocol. For the most common case, viz. two ports for protocols such as TCP or UDP: – Local port (single value, list of ranges or values, ANY, or OPAQUE) – Remote port (single value, list of ranges or values, ANY, or OPAQUE) • (Other expected next-layer protocols include IPv6 Mobility Header (MH) and ICMP messages. Please refer to section 4.4.1.1 of RFC 4301 for additional selector details for these protocols.)

Table 2.4: Structure of an SPD entry (based on section 4.4.1 of RFC 4301 [8]) (cont.)

Field	Description
Processing info	The required IPsec action - PROTECT, BYPASS, or DISCARD. This action applies to all the selector sets (i.e. a match on any of the selector sets will cause this action to be taken). If the action is PROTECT, this entry contains the following additional fields: • IPsec mode - tunnel or transport • Local tunnel address (tunnel mode only) • Remote tunnel address (tunnel mode only) • Extended sequence number (flag) - True or False • Stateful fragment checking (flag) - True or False • Bypass DF bit (flag; IPv4 tunnel mode only) - True or False • Bypass DSCP (flag) - True or False • IPsec protocol - AH or ESP • Algorithms (use for AH, ESP, or combined mode) - list of algorithms ordered in decreasing priority

For SPD entries specifying the protect action, i.e. SPD-S entries, there has to be a link between the SPD-S entry and the corresponding SAD entry, i.e., the corresponding SA. However, the details of this link (reference) are left up to the implementer, although section 4.4.2 of RFC 4301 provides some implementation guidance. For the purposes of the remainder of this discussion, the reader just needs to be aware that such a link (reference) exists.

The structure of a SAD entry is presented in Table 2.5. After consulting both the SPD and the SAD entry structures, it should be clear that there is quite a bit of overlap between the two. This is primarily a one-way relationship; i.e. values from an SPD entry end up in an SA to the extent that these parameters are required to apply IPsec protocol processing. For example, a SA entry needs to include the bypass DF and DSCP flags, since this information is required when creating an IPsec protected packet (especially when generating an AH or ESP packet in IPsec tunnel mode).

Table 2.5: Structure of an SAD entry (based on section 4.4.2.1 of RFC 4301 [8])

Field	Description
Security Parameters Index (SPI)	Value selected by the receiving end to uniquely identify an SA. (32-bit value; part of the AH or ESP header.)
Sequence Number Counter	Counter used to generate the sequence number field in AH/ESP headers. (64-bit)
Sequence Counter Overflow	Flag indicating if sequence number roll over is permitted or not. (True or False)
Anti-Replay Window	Counter and bitmap used to determine if an inbound AH/ESP packet is a replay packet. (64-bit counter; bitmap can be replaced by a suitable equivalent)
AH Authentication fields	(Optional; only required if AH supported) Fields include: algorithm, key, and so on. The exact list depends on the actual algorithm used.
ESP Encryption fields	(Optional; only required if combined mode is not used) Fields include: algorithm, key, mode, IV, and so on. The exact list depends on the actual algorithm used.
ESP Integrity fields	(Optional; only required if combined mode is not used) Fields include: algorithm, keys, and so on. The exact list depends on the actual algorithm used.
ESP Combined mode fields	(Optional; only required if a combined mode algorithm is used) Fields include: algorithm, key(s), and so on. The exact list depends on the actual algorithm used.
SA Lifetime	Time interval after which SA should be replaced or terminated (time value or byte count - i.e. amount of bytes which may still be processed by this SA; or both). This entry should also specify which of the two timeout actions should occur.

Table 2.5: Structure of an SAD entry (based on section 4.4.2.1 of RFC 4301 [8]) (cont.)

Field	Description
IPsec protocol mode	Tunnel or Transport
Stateful fragment checking	Flag - is stateful fragment checking applicable or not. (True or False)
Bypass DF bit	Flag - IPv4 tunnel mode only. (True or False)
DSCP values	Set of DSCP values allowed. (Used for outbound SAs only)
Bypass DSCP	Flag. (True or False)
Path MTU	Observed path MTU (and other ageing variables)
Tunnel header IP source and destination address	(Optional; only applicable for IPsec tunnel mode) Both addresses must be of the same IP version (4 or 6).
Original packet fields (i.e. packet protected by this SA)	This could include: • Local and/or remote address of packet protected by IPsec • Protocol(s) of the packet (more than one protocol may be allowed on an SA in certain circumstances) • Local and remote port values • MH type (IPv6 only) • ICMP type and code • “Name” field (if the name quasi-selector is used) (Also see the <i>Note on original packet fields</i> below this table)

Note on original packet fields: These fields are not expressly included in the “must” list of SAD entry fields (section 4.4.2.1 of RFC 4301), but are included in section 4.4.2.2 of RFC 4301 which deals with the relationship between a SPD entry, its Populate from Packet (PFP) flags, and the corresponding SAD entry. Based on the author’s understanding, these fields would only serve a purpose for inbound SAs as part of the ‘last packet validation check’ (see the discussion on inbound IP traffic processing in section 2.6.2.3.2). Therefore, although not expressly stated in RFC 4301, it is probably safe to assume that these fields will only be present for inbound SAs.

For additional information on the relationship between an SPD entry, its PFP flags, and the corresponding SAD entry, please refer directly to section 4.4.2.2 of RFC 4301.

2.6.2.3.4 A note on the PAD

Extensive details around the Peer Authorization Database (PAD) are presented in section 4.4.3 of RFC 4301 [8]. This discussion, however, while based on section 4.4.3 of RFC 4301, simply aims to highlight some of the key functionalities provided by the PAD.

As the name may imply, the PAD is required (used) whenever an IPsec peer needs to be authorized. This is normally required as part of an IPsec key exchange protocol such as IKE, which

requires that a IPsec peer be authorized before a child SA may be negotiated (see section 4.4.3.4 of RFC 4301).

Similarly to the SPD, the PAD is an ordered database that is populated and managed by an IPsec administrator (normally the same administrator). Each entry in the PAD represents (identifies) a peer, or group of peers, with whom the IPsec implementation is allowed to communicate. As such, it should go without saying that each entry in the SPD requires a matching or associated entry in the PAD as well (at least whenever automated SA creation and management is employed).

An IPsec peer may be identified by one of six ID types, as outlined in section 4.4.3.1 of RFC 4301 [8]:

- Domain Name System (DNS) name (specific, i.e. Fully Qualified Domain Name (FQDN), or partial)
- Distinguished name (complete or sub-tree constrained)
- E-mail address (RFC 822; complete or partially qualified)
- IPv4 address (single address, list of addresses (prefix), or arbitrary range of addresses)
- IPv6 address (single address, list of addresses (prefix), or arbitrary range of addresses)
- Key ID (octet byte string)

Note that the symbolic names, viz. the first three types of IDs, and the “Key ID” match the types of “names” supported by the SPD entry ‘Name’ quasi-selector.

In addition to peer identification, each PAD entry performs the following additional functions (section 4.4.3 of RFC 4301):

- Specifies the authentication protocol and method used when authorizing / authenticating this peer;
- Contains authentication data for the peer(s) (X.509 certificate or pre-shared secret);
- Limits the IDs (types; values) for which the peer(s) may be authorized for, i.e., which IDs, such as an IPv4 address in an SPD entry, may be represented by a peer; and
- (Optional) Peer gateway location info for peers located behind another security gateway. Note: presumably this will be the address info (IPv4 or DNS) of the security gateway itself (RFC 4301 is not really clear in this regard)

For more information on how the PAD functions or is used, please refer to section 4.4.3 of RFC 4301 (along with the relevant 4.4.3.x sub-sections).

2.6.2.4 IPsec security protocols

As mentioned previously, IPsec encompasses two security protocols: Authentication Header (AH) and Encapsulating Security Payload (ESP). The latest versions of these protocols (at the time of writing) are defined in RFC 4302 [55] and RFC 4303 [56] respectively.

The aim of this section is not to present a deep dive of either of these protocols. Instead, the aim is to provide an overview of outbound and inbound processing (in keeping with the outbound

and inbound IP processing discussions in sections 2.6.2.3.1 and 2.6.2.3.1) to explain the working of and services offered by an IPsec security protocol.

To this end, the discussion only considers the ESP protocol. In addition to the motivation given above, ESP is chosen for the following reasons:

- Support for the AH protocol is no longer a mandatory condition (“MUST”) for compliance with the IPsec architecture RFC (4301), since “experience has shown that there are very few contexts in which ESP cannot provide the requisite security services” (section 3.2 of RFC 4301 [8]).
- The basic outbound and inbound processing model outlined in the security protocols RFCs (4302 and 4303) seems to be fairly similar between AH and ESP.
- The AH RFC (4302) itself claims that the primary benefit of using AH integrity protection over that possible with ESP “is the extent of coverage” of IP header fields, i.e. ESP only provides protection for header fields encapsulated by it (section 1 of RFC 4302 [55]). This difference is not deemed sufficient to warrant a separate discussion of AH in this study.

The remainder of this section is structured as follows: the ESP header is considered in section 2.6.2.4.1, followed by outbound- and inbound processing flow discussions in sections 2.6.2.4.2 and 2.6.2.4.3. Similarly to the discussion of the base IPsec architecture, the ESP RFC serves as the primary source of information (RFC 4303 [56]).

2.6.2.4.1 ESP header format and location

As a starting point for the ESP header (and trailer) discussion, the location of the ESP header in IPv4 packets is given in Figures 2.18 and 2.19, for IPsec transport- and tunnel mode operation, respectively. For example purposes, the original IPv4 packet is presented as a TCP packet, but it is important to stress that ESP is usable on IP packets regardless of next-layer protocol type.

For IPsec transport mode operation, the ESP header and trailer is inserted such that it encapsulates the payload of the original IP packet. The form or level of protection (encryption and integrity) depends on the actual security algorithms and/or options used, with the possible scope of encryption and/or integrity as indicated in Figure 2.18.

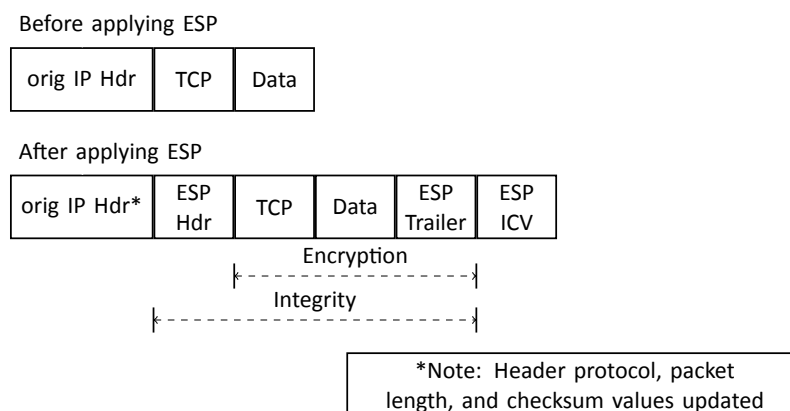


Figure 2.18: ESP Header location in IPv4 packet - IPsec Transport mode (adapted from section 3.1 in RFC 4303 and relevant errata [56])

The ESP Integrity Check Value (ICV) field is present only if integrity services are provided, i.e., if an integrity or combined mode algorithm is used. Details around this field and its use (i.e. length, calculation, and validation) are determined by the specific integrity or combined mode algorithm used.

Note: a combined mode algorithm refers to any algorithm that provides both encryption and integrity services.

For IPsec tunnel mode operation, the ESP header and trailer encompass the entire original IP packet, i.e. including the IP header, as shown in Figure 2.19. A new IP header is constructed to carry the ESP packet according to the information specified by the particular SPD and SAD entry (see the IPsec discussion in section 2.6.2.3).

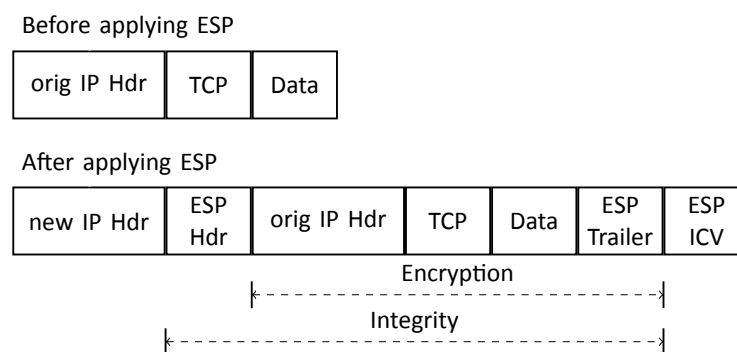


Figure 2.19: ESP Header location in IPv4 packet - IPsec Tunnel mode (adapted from section 3.1 in RFC 4303 [56])

As with IPsec transport mode operation, the possible scope for the integrity and encryption services effected by tunnel mode operation are shown in Figure 2.19. The same comment on the ESP ICV field also applies for tunnel mode operation.

At risk of re-stating the obvious, only tunnel mode therefore provides security services to the original IP header. However, some values from the original IP header, such as the DSCP values, may be reused if specified by the applicable SPD and SAD entry (see section 2.6.2.3.3).

The format of an entire ESP packet is given in Figure 2.20. The SPI and sequence number fields (4 bytes each) make up the ESP header. The IV, payload data, and Traffic Flow Confidentiality (TFC) fields (each of variable length) are all considered part of the payload, while the padding bytes (0 to 255 bytes), pad length (1 byte), and next header (1 byte) fields form the ESP trailer.

Note that while the ESP ICV value (if present; variable length) is transmitted as part of the packet, it is not considered to form part of the ESP trailer.

The role of the SPI value is discussed as part of the IPsec processing model and the structure of SAD entries (see sections 2.6.2.3.1, 2.6.2.3.2, and 2.6.2.3.3). Therefore, it is not discussed here again.

The sequence number can be used to effect anti-replay functionality when a unicast SA or single-sender multicast SA is used, although this is only possible if the SA includes an ESP integrity protection service. The transmitting side is responsible for incrementing the counter for each

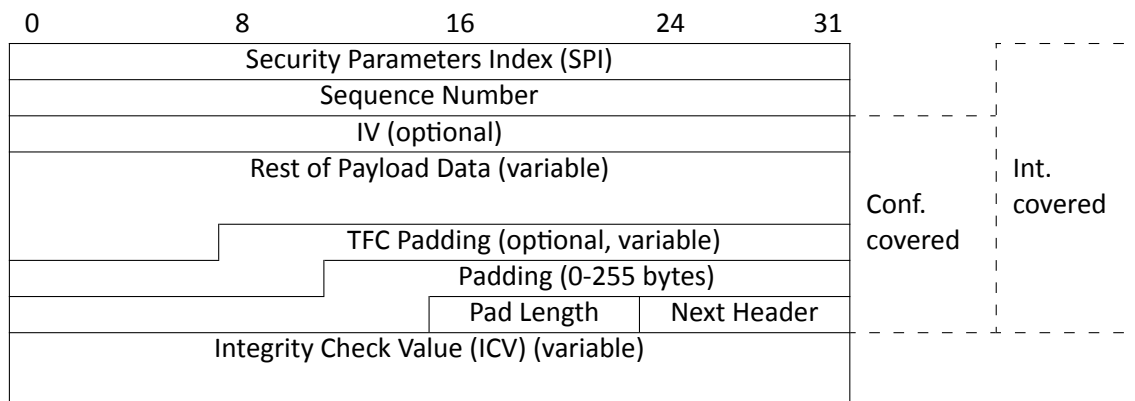


Figure 2.20: ESP Packet Format (adapted from section 2 of RFC 4303 [56])

packet sent, regardless of whether an ESP integrity service is used or not, but the receiver may choose to ignore it (although it should always account for the presence of the sequence number when processing the packet).

The presence of the IV and the TFC padding fields is optional (the two fields are completely unrelated), depending on the encryption algorithm used, and the availability of IP packet (payload) length data, respectively. The function of the IV has been discussed at length as part of the crypto counter mode discussion in section 2.5.2.2.2. It fulfils a similar role here.

The TFC padding field (not to be confused with the ESP trailer padding field) forms part of the (optional) TFC functionality/service offered by ESP. The purpose of this function (broadly) is to further obfuscate the true nature of the traffic flowing between two IPsec devices from potential eavesdroppers by including additional padding data.

For example, TCP handshakes (or the general flow of TCP data) can be hidden by padding each packet to a similar length (i.e. ESP packets carrying shorter TCP SYN or ACK payloads are indistinguishable from packets carrying larger data payloads). For additional information on TFC functionality, please refer to section 2.7 of RFC 4303 [56].

The ESP trailer padding field is intended for cryptographic use. RFC 4303 identifies two expected use cases: to facilitate plaintext size constraints for an encryption algorithm (e.g., filling a data block for a block cipher) and for byte alignment purposes.

The byte alignment use case centers on the need for the ICV to be aligned to a 4-byte boundary. For this reason, the pad length and next header fields are mandated to be “right-aligned within a 4-byte word” (section 2.4 of RFC 4303 [56]) as shown in Figure 2.20.

The role of the padding length and next header fields in the ESP trailer is self-explanatory. It is perhaps interesting to note that the ESP trailer is intended to be parsed from the back when determining the location of the padding length and next header fields. This allows for cases where payload length is not necessarily known, especially if the TFC functionality is employed.

Finally, it is worth touching on the scope of integrity- and encryption (confidentiality) protection. While it is possible to effect integrity coverage over all fields marked under ‘Int. covered’ in Figure 2.20, namely, SPI, sequence number, IV, payload data, TFC padding, ESP padding, padding length,

and the next header field, the actual scope is dependent on the actual integrity or combined mode algorithm used (some combined mode algorithms only provide integrity services for the encrypted data).

Encryption protection encompasses all fields marked under 'Conf. covered' in Figure 2.20, namely, IV, payload data, TFC padding, ESP padding, padding length, and the next header field. It is important to note that the IV itself is not encrypted, seeing as it is needed as an input into the decryption process, although it is typically considered to still form part of the ciphertext (hence the inclusion in the 'Conf. covered' scope, section 2 of RFC 4303 [56]).

2.6.2.4.2 Outbound ESP processing

This section aims to outline the processing flow for outbound IP packets that require ESP protection (after an SA has been established). It draws primarily from section 3.3 of RFC 4303 [56]. Please refer to section 2.6.2.3.1 for a discussion of the IPsec processing flow that leads up to this point, i.e. the point where the *ESP process* action is applied.

The outbound ESP processing flow is a fairly linear process. In keeping with the outline in RFC 4303, this process is discussed under three subtopics:

1. Packet encryption and ICV calculation;
2. Sequence number generation; and
3. Fragmentation

Note that sequence number generation actually takes place before the ICV calculation is done (if applicable).

Packet encryption and ICV calculation This step differs slightly depending on whether separate integrity and encryption algorithms are used, or if a combined mode algorithm is used. For the purposes of this discussion, it is sufficient to consider only the case of separate algorithms.

The processing flow is as follows:

1. Encapsulate IP data into the ESP payload.
 - (a) For transport mode, this is limited to the next-layer protocol data, i.e. the original IP payload.
 - (b) For tunnel mode, this encompasses the entire original IP packet.
2. Add padding if necessary (TFC and/or cryptographic padding). This includes populating the pad length and next header fields in the ESP trailer.
3. Apply the encryption algorithm to the result.
 - (a) The required meta-data for this, IV values, key to use, other synchronisation data or mechanism(s) to use etc., is obtained from the corresponding SAD entry (see Table 2.5 in section 2.6.2.3.3)
 - (b) If no encryption (confidentiality) is desired or chosen, the encryption algorithm will be indicated as a NULL cipher.
In this sense, the RFC 4303 processing flow remains linear in that an encryption algorithm is always applied (used), even if it is a no-op.

4. Implicit step: populate the ESP header, viz. the SPI value and sequence number, from the SAD entry
5. If an integrity algorithm is chosen, perform the necessary calculations and append the resulting ICV to the ESP packet.
 - (a) As discussed in section 2.6.2.4.1, the scope of integrity protected fields depends on the actual integrity algorithm used.
 - (b) Encrypted fields are verified and/or used in the ICV calculation in their ciphertext form.

It is worth noting that RFC 4303 deliberately mandates that encryption be performed before integrity is applied to facilitate receiver-side processing. Specifically, this allows processing optimisation and partial protection against the impact of DoS type attacks by:

1. Allowing for parallel processing of each packet (decryption and integrity checks at the receiver can be done at the same time); and/or
2. Allowing integrity checks, and/or an anti-replay check if applicable, to take place before decryption is attempted (or before decryption is completed), i.e. the computational resources used to decrypt bogus packets can be limited (reduced).

However, this approach means that the ICV itself is not protected (i.e., not encrypted). Therefore, RFC 4303 mandates that a keyed integrity algorithm be used when separate confidentiality and integrity algorithms are used.

Sequence number generation The sequence number is treated as a function (or property) of the SA. It is initialised to zero (0) during SA creation, and incremented to one (1) before the first packet is sent (sender increments the counter before inserting it into the ESP header).

As stated in section 2.6.2.4.1, the receiver can use the sequence number to effect anti-replay functionality (enabled by default). Therefore, the sender can only use the SA until the sequence number would roll over. If incrementing the sequence number causes it to roll over, the packet for which it rolled over must not be sent (in fact, the IPsec implementation is expected to establish a new SA before this occurs).

The Extended Sequence Number (ESN) feature can be used to increase the number of packets that can be sent using the SA (see RFC 4303 for more details around this functionality and how it impacts the processing flow and ICV calculation). It is worth noting that using this feature does not increase the ESP header size (extended counter bits are not sent with the packet).

If the anti-replay service is disabled or not enforceable, as is the case for multicast, multisender SAs, the sequence number is not monitored or used. The sender still increments the counter, but no special handling or event occurs if it overflows.

Fragmentation ESP does not support the processing of or operations on fragmented packets if an ESP packet itself is fragmented. Therefore, all fragmented packets have to be reassembled prior to ESP processing of any kind.

RFC 4303 makes special mention of potential DoS vulnerabilities if an ESP receiver must accept fragmented packets. As such, support for fragmentation is optional in ESP implementations. However, support for the generation of ICMP Path MTU (PMTU) messages is mandatory. Please refer to section 3.3.4 of RFC 4303 for more information [56].

2.6.2.4.3 Inbound ESP processing

This section aims to outline the processing flow for inbound ESP packets. Similar to the outbound processing discussion (section 2.6.2.4.2), this section draws primarily from RFC 4303, section 3.4. Please refer to section 2.6.2.3.2 for a discussion of the IPsec processing flow that leads up to this *Process ESP* action.

The inbound processing flow is nearly the inverse of the outbound processing flow. It is discussed under the following subtopics:

1. Reassembly
2. SA lookup
3. Sequence number Verification
4. ICV verification and packet decryption

Reassembly Packet reassembly is generally a function of the IP protocol itself. However, it is worth highlighting two things:

1. The ESP receiver will discard any fragments it receives (i.e. reassembly has to be done before ESP processing is started); and
2. ESP requires the *offset* field and the *more fragments* flag to be cleared and reset respectively in a reassembled IP packet (according to section 3.4.1. of RFC 4303 this is not mandated by the IPv4 specification)

SA lookup As mentioned in the inbound IPsec processing discussion (section 2.6.2.3.2) the SA lookup takes place after the IPsec implementation has determined that the ESP packet is, in fact, addressed to the IPsec implementation.

Aside from filtering packets that do not have a matching SA entry, this lookup also serves to determine the required metadata to process the ESP packet. This includes: whether the sequence number field should be checked (anti-replay enabled or disabled), if ESN is used or not (64-bit or 32-bit sequence number), the algorithm(s) and keys to use, and if the ICV field is expected to be present, along with its size if applicable.

Sequence number verification The purpose of sequence number verification is to effect the anti-replay service if enabled. If anti-replay is not enabled, this processing step becomes a no-op.

As stated in the outbound discussion (section 2.6.2.4.2), this service is enabled by default. However, it is worth clarifying this statement here, especially with reference to the receiver (inbound) side.

Although RFC 4303 mandates that an ESP implementation must support the anti-replay service, the receiver chooses to enable or disable the anti-replay service on a per-SA basis. The sender, on the other hand, should always assume that it is enabled, unless notified by the receiver that it is disabled during SA establishment (or configured as such when manual SAs are used). Additionally, the anti-replay service may only be used if an integrity algorithm (or combined mode algorithm) is used that encompasses the sequence number field.

Details around the anti-replay functionality is presented in section 3.4.3 of RFC 4303 (use of a sliding window to verify that a sequence number is not repeated and so on). It is, however, worth mentioning the preliminary check that is made available for implementations that cannot perform IPsec processing at line rate.

When using this check, an implementation first checks the validity of the sequence number before performing any cryptographic operations, viz. integrity checks or decryption. If this check fails, the packet can be discarded without committing additional resources to processing the packet. If the check passes, the ESP implementation first needs to complete the cryptographic operations before accepting the packet (and sequence number) as valid.

ICV verification and packet decryption As with the outbound ESP processing flow, this step differs slightly depending on whether separate integrity and encryption algorithms or a combined mode algorithm is used. The separate algorithm case is presented here to match the outbound discussion (section 2.6.2.4.2).

The inbound processing flow is presented linearly here. However, note that the receiver implementation may choose to perform integrity checks and decryption operations in parallel, although integrity checking must be completed before a decrypted packet is processed any further.

1. If an integrity algorithm is used, verify the ICV value. The receiver performs the ICV calculation on the ESP packet minus the received ICV value. If the calculated ICV value matches the received ICV value, the packet is treated as valid and processing is continued. If not, the packet is discarded. The full algorithm (or process) for this can be found in section 3.4.4.1 of RFC 4303.
2. The encrypted fields of the ESP packet are decrypted next (ESP payload data, padding, padding length, next header fields).
 - (a) As for the encryption process, the necessary meta-data is obtained from the matching SAD entry, viz. IV value details, encryption algorithm, other synchronisation data or mechanism(s) used, and so on.
 - (b) If necessary, the IV, or other synchronisation data, is recovered from the ESP payload before decryption is performed. This applies to cases where algorithms are used that do not make use of the explicit IV field after the sequence number, i.e. the IV may need to be calculated from metadata in some other way for example.
3. ESP padding is removed (processed) as required by the encryption algorithm.
4. Based on the next-layer header field (and the IPsec mode of operation), the original IP packet is recovered (reconstructed).
 - (a) RFC 4303 makes provision for ESP dummy packets as part of TFC methods. If an ESP dummy packet is received, it is simply discarded at this point.
 - (b) If applicable, TFC padding is usually removed during the IP packet recovery process.

2.6.2.5 IPsec SA management methods and protocols

The latest version of the IKE protocol, IKEv2, is documented in RFC 7296 [57]. This RFC summarizes IKE as “a component of IPsec used for performing mutual authentication and establishing and maintaining Security Associations (SAs)”. In other words, the IKE protocol enables the automatic

creation and maintenance of SAs between IPsec peers.

In traditional IP networks, this is the preferred mechanism for managing SAs in IPsec deployments, since manual SA management is not really scalable as stated in section 1 of RFC 7296 [57] and section 4.5.1. of RFC 4301 [8]. This is also reflected in section 4.5.2 of RFC 4301 [8] which names IKEv2 as the default key management protocol for IPsec implementations. However, it is important to note that this does not preclude the use of manual SA creation and management in an IPsec implementation in its entirety. On the contrary, section 4.5 of RFC 4301 requires all standard compliant IPsec implementations to provide support for both manual and automatic key and SA management.

The IKEv2 protocol is designed as a request and response type of protocol (section 1 of RFC 7296 [57]). In other words, for each IKE message (request) sent, a corresponding response is expected from the remote IPsec peer, termed an IKE exchange. In light of this, the general working of the IKEv2 protocol is briefly summarized as follows based on the information in RFC 7296 [57]:

- The first IKE exchange, IKE_SA_INIT, is responsible for establishing what is referred to as an IKE SA. This SA is used for all subsequent IKE exchanges between the two IPsec peers.
- The second IKE exchange, IKE_SA_AUTH, sets up a child SA between the two IPsec peers which is used for the actual traffic that is protected by IPsec between the peers. Therefore, the child SA will always be an ESP or AH SA.
- All IKE sessions are mandated to start with these two exchanges. Once they are completed, the IPsec protected traffic can start flowing between the peers via the child SA, and other IKE exchanges can be performed as the need arises, without any mandatory order. For example, the CREATE_CHILD_SA IKE exchange is used to re-key both the IKE SA and the child SA as and when necessary, and INFORMATIONAL requests can be used to perform housekeeping tasks such as deleting a SA or reporting error conditions.

Further discussion of IKEv2 functionality is deemed unnecessary for the purposes of this study of the literature. Note that RFC 7296 [57] is a fairly comprehensive source that can be consulted for additional information about IKEv2.

2.6.2.6 IPsec extensions

For completeness sake, it is worth noting that multiple IPsec extension RFCs exists. This includes extensions such as the IPsec multicast extensions described in RFC 5374 [58] which introduces a Group Key Management protocol and other IPsec modifications required for automatic multicast group SA establishment, and RFC 3947 [59] which presents a method for IKE packets to traverse Network Address Translation (NAT) devices.

Additional information about these extensions is considered to fall outside the scope of this literature study.

2.6.3 Other IP packet encryption options

RFC 8922 [60] titled “A Survey of the Interaction between Security Protocols and Transport Services”, mentions two alternative protocols to IPsec, namely, OpenVPN and WireGuard. These are considered to be packet-level security protocols in that they offer protection directly to IP packets.

However, both of these protocols operate by encapsulating IP packets into transport layer protocols, viz. UDP (WireGuard and OpenVPN) or TCP (OpenVPN only). In the HF environment, this is considered unnecessary additional overhead, and therefore neither of these protocols is considered any further in this research study.

2.7 Concluding remarks

This literature study presented an overview of the HF domain (section 2.2), investigated both traditional and IP data transmission over an HF channel (sections 2.3 and 2.4), and explored both traditional security for HF communication and security for traditional IP networks (sections 2.5 and 2.6), with a notable deep dive into the functionality of the IPsec protocol suite (section 2.6.2).

This information forms a good basis from which to transition into the design and implementation of a suitable IP cryptographic function for the HF radio environment. In particular, knowledge about the challenges related to IP communication in the HF domain, together with the context of traditional HF security and knowledge of the working of IPsec, are key sources used when making design decisions.

3 Design and implementation of the IP cryptographic function

This chapter discusses the design and implementation of the IP cryptographic function. Section 3.1 provides some contextualization, followed by section 3.2 which discusses a potential IP over HF system within which the IP crypto can be expected to operate. Section 3.3 presents the IP crypto requirements and design, and section 3.4 considers potential implementation options. The IP crypto implementation done using the Linux XFRM framework is presented in section 3.5. Lastly, some concluding remarks are given in section 3.6.

3.1 The case for an IP cryptographic function for the HF environment

As previously illustrated in section 2.4.1.1, there are compelling reasons to work towards provisioning IP services over HF, spurred on further by the advent of WBHF technology as indicated in section 2.4.1.3. Within this landscape of developing a workable IP over HF system, one aspect to consider is that of network security. As shown in section 2.4.1.4, network security rarely receives attention in public domain IP over HF research work, which presents an opportunity to explore the design and implementation of a suitable IP cryptographic function.

It would be possible to handle the security of an IP over HF system by simply employing traditional HF security measures. However, this would inherently limit the scope of such a system to pure HF networks, which is in contrast with one of the research drivers for IP over HF systems, namely communication system inter-operation in the military environment (see section 2.4.1.1).

Furthermore, Annex E of the “Interoperable Wideband Link Management” standard, STANAG 5070 [6], mandates a standard compliant system to provision an IP cryptographic function. Although the standard is still in the draft state and is currently undergoing ratification within NATO, there is no reason to expect this mandatory requirement to be relaxed. Therefore, an IP cryptographic function, or IP crypto for short, is considered a mandatory requirement for any IP over HF system deployed in a military context.

3.2 A potential IP over HF system including a IP cryptographic function

3.2.1 IP over HF system design goals

Before considering the design of the IP crypto itself, a high-level discussion of a potential IP over HF system is presented. This provides the context within which the IP crypto is expected to function and delimits the boundary of what is encompassed by the IP crypto.

The following design goals are considered as the primary drivers for the architecture of the potential IP over HF system:

- Provisioning IP based services desired within the HF military environment. This includes email, file transfer, web-based, and multi-user chat services, which reflects the services

requested in the NATO BRIPEs tender, as mentioned in section 2.4.1.1;

- Complying with industry standards as and where applicable to IP crypto requirements, i.e. system design aspects that are influenced by IP crypto requirements, notably including Annex E of STANAG 5070 [6];
- Interoperability with other IP based military communication systems, as reflected in section 2.4.1.1

3.2.2 IP over HF system architecture discussion

A high-level IP over HF system diagram is presented in Figure 3.1. This diagram introduces the concept of an IP service node, which refers to a node that provides IP services along with an IP crypto function. For clarity, only two IP service nodes, each at a different radio site, are indicated, but it is expected that each radio site will be able to support multiple IP service nodes.

Furthermore, each radio site can potentially provide access to multiple different radio networks and/or radio network types, such as HF, VHF, and/or UHF radio networks. To this end, the federating router functions as the intermediary between the IP service node and the various radio networks.

The IP crypto can be considered to partition the system into a red side- (protected) and black side- (unprotected) network domain, similar to that achieved by the IPsec protocol suite, as shown in Figure 2.15 in section 2.6.2.3. Based on the author's experience, the red/black terminology is more commonly used within the HF industry, and is therefore the terminology of choice going forward.

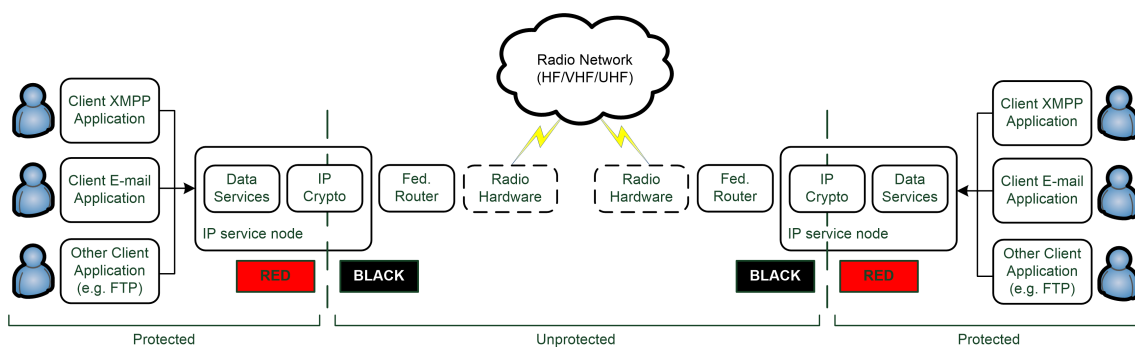


Figure 3.1: High-level IP over HF system diagram

As a short aside, the overall scalability of the system, both in terms of the number of IP service nodes supported by each radio site and the total number of radio sites that can be accommodated before network performance starts degrading, is dependent on a number of factors, some of which will be deployment dependent, e.g. the total users at each site, the number of available channels in the HF network, the number and type of radio networks available at each site, and so forth. However, since each IP service node includes its own IP crypto function, scalability is not considered to be a major factor in the design of the IP crypto itself.

The high-level architecture envisioned for a single IP service node is presented in Figure 3.2. This diagram includes reference as to which OSI layer applies to which components. External applications, the wireless networks federating router, and various IP networks are included for

context but do not expressly form part of the IP service node architecture.

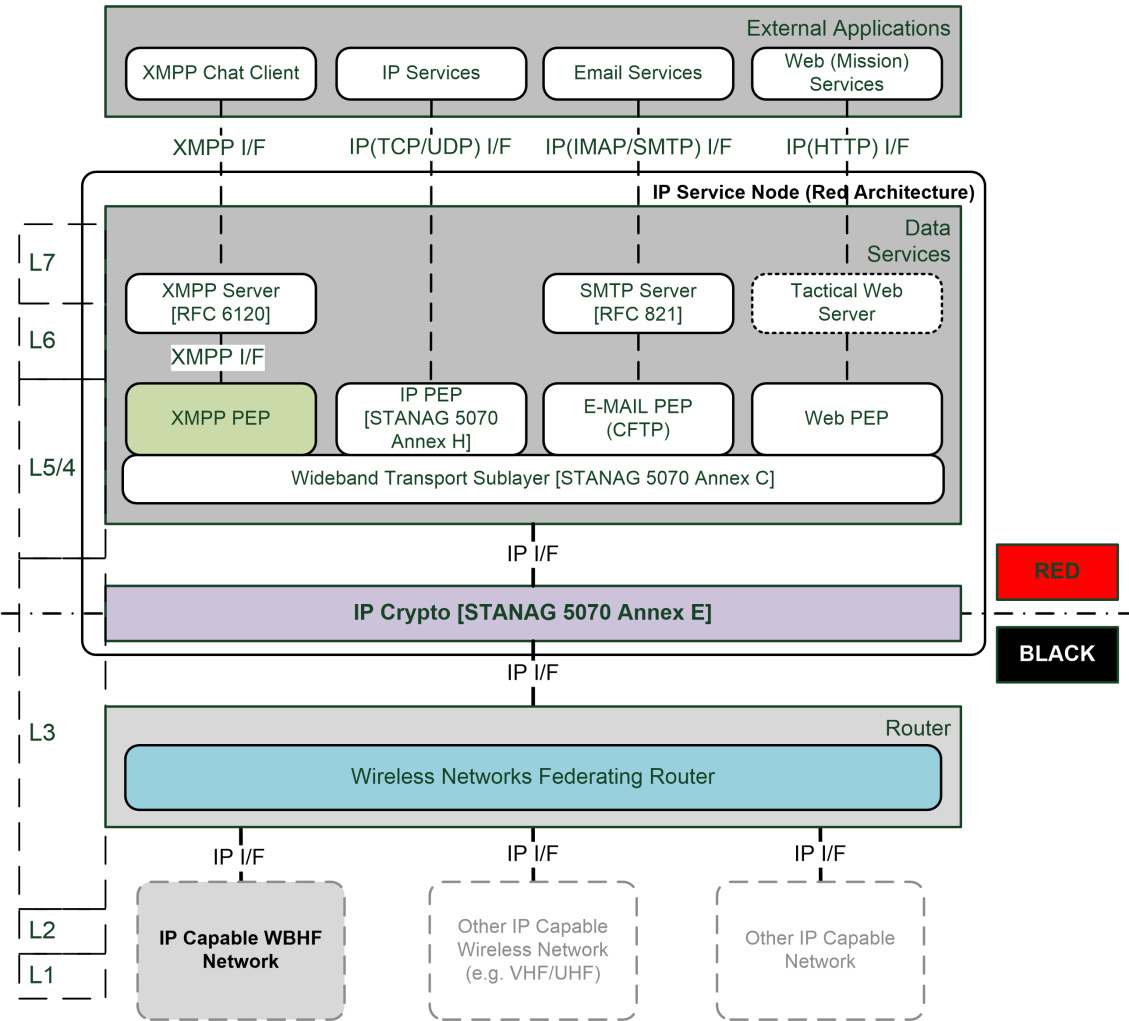


Figure 3.2: High-level IP service node architecture

Table 3.1 contains a brief description of each component in Figure 3.2, including the mention of relevant industry standards. It is also worth noting that some inspiration is taken from the system set out in the NATO BRIPEs specification [7], although achieving compliance with this specification is not a design goal, per say.

Table 3.1: Description of various components in the IP service node architecture

Component	Description	Relevant Standard(s)
XMPP server	Hosts XMPP services for client XMPP applications. Functions in conjunction with the XMPP PEP to provision these services over IP capable radio networks.	RFC 6120 [61]
XMPP PEP	Optimises XMPP data streams (XML-based) for transmission over a radio network. Functions in conjunction with the XMPP server to provision XMPP services over IP capable radio networks.	None (proprietary component)
IP PEP	Provides support for other TCP and UDP based IP services. Especially crucial for supporting TCP based services over an HF radio network, since it optimises TCP streams for low bandwidth, high latency networks.	STANAG 5070 Annex H [6]
SMTP server	Provides SMTP server functionality (mail delivery) for email clients, as specified in RFC 821 [62]. Note: Although RFC 821 has been obsoleted by RFC 2821, which in turn has been obsoleted by RFC 5321, it is still referenced here due to its reference by the specification for the CFTP client and protocol in Annex V of STANAG 5066 Edition 4 [5].	RFC 821 [62] RFC 2821 [63] RFC 5321 [64]
E-mail PEP	The intended purpose of the E-mail PEP is similar as for the other PEP components: optimisation of mail traffic for HF transfer. However, the CFTP protocol specified in Annex V of STANAG 5066 Edition 4 [5], is already optimised for this purpose. As such, the E-mail PEP could potentially fulfil its function by serving as a translator of sorts between SMTP and CFTP. It is also worth noting that the WBTS provides support for direct integration of STANAG 5066 clients [6], implying that the E-mail PEP could interface with the WBTS via a STANAG 5066 CFTP client (not shown).	None (proprietary component)
Tactical Web Server	Hosts tactical web (mission) services for client web applications. It is expected to always work in conjunction with a Web PEP. It is important to note that these services are expected to be designed specifically with the tactical HF environment in mind and should not be mistaken for web services similar to those provided via the Internet or other high-speed IP networks. This component is also not mandatory for every IP service node, since client web applications are expected to connect to web servers located at other IP service nodes as well.	None (proprietary component)

Table 3.1: Description of various components in the IP service node architecture (cont.)

Component	Description	Relevant Standard(s)
Web PEP	Similar to other PEPs, the Web PEP's goal is to optimise web traffic for transmission over HF. Functions in conjunction with the Tactical Web Server, but can also interface directly with client web applications.	None (proprietary component)
WBTS	Provides a common IP interface to the HF subnetwork in particular for IP applications (via client IP interface - not shown) and legacy HF applications (via STANAG 5066 Raw SIS interface - not shown). However, the same abstraction can be used for other IP capable network types such as VHF/UHF radio networks or traditional hard-line IP networks.	STANAG 5070 Annex C [6]
IP crypto	Provides encryption and decryption services and gives effect to the red/black separation. This IP crypto is the core focus of the research study.	STANAG 5070 Annex E [6]
Wireless Networks Federating Router	Provides several features such as multi-wireless network support for IP data, inter-network routing (IP based at layer 3, possibly an adaption of the OLSR protocol), inter-network routing via a traditional IP network link (if present), and simultaneous support for different services that require different radio network types.	None (proprietary component)
IP capable WBHF network	The primary network environment for which the system is designed. The HF environment is considered the most hostile towards IP services, as it has the highest latency and lowest bandwidth of the wireless network types indicated (HF, VHF, and UHF). Therefore, designing the system for this worst-case environment ensures that it will be functional for all types of wireless networks envisioned.	STANAG 5070 [6]; or STANAG 5066 Ed. 4 [5]

3.2.3 IP cryptographic function boundary

The IP crypto function boundary is delimited as shown in the IP service node architecture diagram in Figure 3.2 in section 3.2.2. A key assumption for the rest of the design discussion is that the IP crypto is designed as if the rest of the IP over HF system already exists.

The full extent of functionality included in the IP crypto is outlined in section 3.3. However, two notable exclusions are worth highlighting: all packet routing functionality is performed outside of the IP crypto, and the IP crypto is expected to be service-agnostic.

This means that the IP crypto operates expressly at the IP protocol level without any regard for the type of data included in the IP packet payload and without any knowledge of the current routing table state. Therefore, any system-wide Quality of Service (QoS) or service priority functionality has to occur prior to submitting IP packets to the IP crypto.

3.2.4 A note on the functionality provided by the various Performance Enhancing Proxys (PEPs)

The general purpose of PEPs and their basic function within the HF environment is summarized briefly as part of the discussion in section 2.4.1.3. This includes the brief statement made in [44] that the purpose of a PEP is to “mitigate end-to-end performance impairments due to increased latency and packet loss”.

Several of the requirements in section 2.2 of the NATO BRIPEs specification [7] also outline the purpose of a service-specific PEP to “improve the efficiency of information exchange” through the following techniques:

- Overhead reduction;
- Data compression; and/or
- The reduction of command and response transactions so that it reduces the total data transferred, as well as the number of transactions that occur over an HF link.

The implication of this is that the inclusion of a service-specific PEP will change or at least influence the OTA traffic profile of that service.

3.3 IP crypto function requirements and design

3.3.1 Design inputs and requirement sources

A functional analysis approach is used to determine the requirements for the IP crypto. From the outset, it is worth stating that this design discussion focusses on the factors that encompass the functionality of the IP crypto as pertains to its performance impact on the IP over HF system. This is in line with the goals of this research study, as set out in section 1.3.

The following inputs (sources) are used for the functional analysis of the IP crypto:

1. The IP over HF system design goals as outlined in section 3.2.1;

2. The architecture of the IP over HF system and the IP service node as presented in section 3.2.2. This can also be considered as the first-order functional analysis in the design process since it encompasses the use cases (services) that have to be supported by the IP crypto; and

3. HF industry standard compliance, primarily Annex E of STANAG 5070 [6], and RFC compliance, as applicable.

This also forms part of the IP over HF system design goals, but has to be clarified in terms of the extent of its application to the crypto design. STANAG standard compliance is seen as mandatory and always takes preference over RFC compliance if a contradiction occurs. Furthermore, RFC compliance is viewed more as a guiding principle than a hard requirement, since interoperability with normal RFC compliant cryptography implementations is not a requirement or design goal for the IP crypto. For example, if an RFC includes mandatory requirements that are not considered relevant for the IP crypto to fulfil its goal, these requirements are not considered in the functional analysis of the IP crypto.

3.3.2 Notable design exclusions

Supporting functionality of the IP crypto, that is, functionality that does not directly impact the performance of the IP crypto when it is operational, is excluded from the IP crypto design presented in the rest of this chapter. This includes things such as: configuration storage and loading, SA lifetime and key usage management, IV generation and management, and general configuration change handling in an operational deployment.

This decision is made for two primary reasons: each of these aspects has the potential of being complex enough to warrant investigation in a separate research study (or studies), and since this research study focusses on the performance impact of the IP crypto, their inclusion is not necessary to fulfil the research study goals set out in section 1.3. Therefore, the design of these supporting functions is left for future research work, as stated in section 6.4 of the concluding chapter of this study (chapter 6).

3.3.3 Analysis of STANAG 5070 Annex E

As was previously highlighted in section 2.5.2.2, the general approach to crypto design in the HF industry is not to ‘reinvent the wheel’, but to adapt well-established existing encryption algorithms (or mechanisms) for use within the HF environment. This is reflected in Annex E of STANAG 5070 [6] which does not specify a new IP crypto as such, but outlines a set of restrictions and features, termed a crypto profile, that a compliant IP crypto has to adhere to.

Specifically, the claim is made that the crypto profile matches features found in an IPsec crypto operating in tunnel mode. The nomenclature used throughout the crypto profile definition further enforces this claim.

The following key requirements arise from the crypto profile and are applicable as-is to the IP crypto design:

1. The IP crypto shall operate on a packet-per-packet basis.
Any data concatenation techniques to reduce overhead therefore have to be performed prior to submitting packets to the IP crypto (e.g. at PEP level);

2. No sensitive red-side information should ever be exposed to the black domain in the clear. Although this requirement is not stated verbatim in Annex E of STANAG 5070, it is strongly implied. Added to that, when considering that the IP crypto is intended for use in the military environment, this requirement becomes mandatory;
3. No OTA communication of any kind may occur between IP crypto's.
In other words, no tunneling, keying, or any other crypto-to-crypto protocols may be used.
4. All of the information required for the crypto to perform its functions has to be pre-stored. As a minimum, this includes: addressing information, the ciphering algorithm to use, and relevant key material.
This implies that a compromised or stolen node affects the whole system, but this static configuration approach is typical in most military HF scenarios, as evidenced by its inclusion in this standard. Based on industry experience, the risk is typically addressed by introducing other mitigating measures in the final design, such as a zeroise capability to expediently wipe all sensitive information from the crypto device;
5. The DSCP and Explicit Congestion Notification (ECN) header fields have to be preserved when a packet traverses the crypto boundary from red to black.
This allows for some service-level QoS in the red domain.
6. The IP crypto must be able to process packets with unicast, multicast, or broadcast IP addresses.

3.3.4 Functional design decisions

Aside from the claim that the Annex E STANAG 5070 [6] crypto profile matches features found in a IPsec crypto operating in tunnel mode (section 3.3.3), to the best of the author's knowledge no other IP security mechanism and/or protocol exists that can match all of the functionality outlined in section 3.3.3. Therefore, it is sensible to use IPsec operating in tunnel mode as the basis for the IP crypto design.

Section 2.6.2 in chapter 2 presents an in-depth discussion of the IPsec suite as a whole. The rest of the IP crypto design discussion assumes that the reader is familiar with this information.

The following subsections outline modifications to IPsec and other key functional design choices necessary to meet the requirements outlined in section 3.3.3 and to address the IP crypto design goals set out in section 3.3.1.

3.3.4.1 Limit support to Internet Protocol Version 4 (IPv4)

The IP crypto will only support IPv4 traffic, since IPv6 packet header overhead is considered excessive for the HF environment. If external applications require IPv6 support, IPv6 to IPv4 translation should be done either in the service specific PEPs, or by introducing another component into the IP over HF system.

3.3.4.2 No packet queues or buffering

The IP crypto shall not provide any form of packet queuing or buffering functionality beyond that required to prevent incoming packet drops while the crypto is busy processing other packets. Since packet routing and forwarding is handled outside the IP crypto boundary (section 3.2.3), it

does not make sense for the IP crypto to include this functionality. Therefore, this restriction is added to simplify the crypto design.

It is also worth noting that the WBTS is typically responsible for these aspects, as well as providing QoS functionality (Annex C of STANAG 5070 [6]).

3.3.4.3 Red side LAN support

The IP crypto shall only support a single red side Local Area Network (LAN). This prevents the need for red side routing functionality and simplifies network configuration by allowing the red side IP interface of the IP crypto to function as the default gateway for all red side hosts.

3.3.4.4 ESP protocol use

The IP crypto shall only support the use of the ESP protocol in tunnel mode. This choice is made since packet encryption is mandatory (section 3.3.3), and the AH protocol does not provide encryption capabilities (section 2.6.2.2).

3.3.4.5 Static SA use

The IP crypto shall only make use of statically preconfigured SAs. This is in keeping with the requirement from Annex E of STANAG 5070 [6] that no OTA communication is allowed between IP cryptos (see section 3.3.3).

This decision also implies that all crypto key material has to be pre-configured and stored in each IP crypto module.

3.3.4.6 No form of IP crypto bypass

To ensure that no traffic traverses the cryptographic boundary without being encrypted, the IP crypto function shall not support any bypass actions (see section 2.6.2.3), and shall not support the use of any NULL ciphers (see section 2.6.2.4.2).

This design decision can also be termed a pure red/black separation.

3.3.4.7 No Internet Control Message Protocol (ICMP) traffic processing

Due to the inherent security risks associated with ICMP traffic processing, for both red- and black side ICMP traffic, as outlined in sections 2.6.2.3.1 and 2.6.2.3.2, the IP crypto function shall not process any ICMP traffic.

Since this includes ICMP path MTU discovery traffic, this design decision places a burden on the various PEPs to limit the size of generated IP packets such that encryption by the IP crypto results in black side packets that still fit into the standard Ethernet MTU frame size of 1500 bytes.

This requirement is relaxed for the IP crypto implementation used in this study to allow the execution of functional ping tests (ICMP echo request/response) for the verification of the IP crypto (section 4.3.5), and to allow measurement of the initial HF link setup time using a single packet ping test as part of the performance evaluation of the IP crypto (section 5.1).

3.3.4.8 No fragmentation support

The IP crypto shall not support any form of IP fragmentation. This avoids the potential DoS vulnerability associated with fragmented packets, as outlined in section 2.6.2.4.2, and simplifies the overall IP over HF system design.

3.3.4.9 No Peer Authorization Database (PAD) support

The IP crypto shall not contain any form of PAD. Since the crypto may not use any form of OTA key exchange or SA establishment, the IPsec PAD becomes unnecessary (see section 2.6.2.3.4).

3.3.4.10 No IPsec multicast extension support

The IP crypto shall not support any IPsec multicast extensions. This has a two-fold motivation. Firstly, multicast IP packets can be handled by creating the appropriate static SAs meaning that the IPsec multicast extensions are not required to satisfy the requirement from STANAG 5070 Annex E [6] which mandates processing support for multicast packets (section 3.3.3). Second, some modifications introduced by the IPsec multicast extension RFC, namely RFC 5374 [58], can compromise red-side address information, viz. “Tunnel Mode with Address Preservation” in section 3.1 of RFC 5374 [58].

3.3.4.11 Last remarks

Several of the design decisions given throughout section 3.3.4, viz. section 3.3.4.5 to 3.3.4.9, modify the behaviour of an IPsec tunnel mode crypto such that it can no longer be fully compliant with RFC 4301 [8], and can therefore no longer be described as an IPsec crypto. For this reason, the crypto is referred to as an IP crypto throughout this dissertation, instead of an IPsec crypto.

3.4 IP crypto implementation options

3.4.1 Primary implementation goal

For the purposes of this research study, and due to time constraints, the primary research implementation goal is not to produce a fully functional military grade IP crypto product with all of the features and capabilities required for use in the potential IP over HF system presented in section 3.2. Rather, in keeping with the study goals set out in section 1.3, the goal is to implement a representative IP crypto, such that it can execute the main crypto functionality that will impact the performance of IP traffic within the IP over HF system.

In short, this includes:

1. All functionality related to packet encryption and decryption, in particular including a IPsec like framework and an ESP protocol implementation;
2. The capability to drop disallowed traffic such as ICMP packets (section 3.3.4.7); and
3. Sufficient configuration capability to setup the required SAs and load the necessary key material for testing purposes.

3.4.2 Selecting a suitable implementation option

To achieve the implementation goal set out in section 3.4.1, three options are available, namely:

1. Develop an IP crypto implementation from the ground up;
2. Create a simulation model that exhibits the performance impact effects of the IP crypto; or
3. Modify (if required), setup, and configure an off-the-shelf (open source) implementation of IPsec such that it resembles the envisioned IP crypto.

The first two options both require a large time investment, especially when considering the verification effort required. In the first case, this would involve verifying that the ESP protocol implementation is correct amongst other things, and in the second case some reference would be required to verify the accuracy of the simulation, which to the best of the author's knowledge, is not readily attainable.

A viable starting point for the third option is found in the Linux kernel which contains a fully functional IPsec implementation in the form of the Linux XFRM framework as documented in the IP-XFRM manual pages [65]. Additionally, the IP-XFRM user space tool allows direct control over the IPsec SPD and SAD, indicating that static configuration is possible without having to make any source code changes.

This implies that it may be possible to use the Linux XFRM framework without having to make any source code changes or other material modifications, i.e. it may be usable as-is, which is in fact the case as summarised later in section 3.5.4. Therefore, the third option is the chosen course of action.

3.5 Using the Linux XFRM framework to implement the IP crypto function

To achieve the implementation goal set out in section 3.4.1, three objectives have to be achieved in the Linux XFRM framework:

1. Disable all key exchange and automatic SA establishment functionality, that is, all functionality that would result in OTA communication taking place;
2. Ensure that it is possible to statically generate and configure key material and SA entries; and
3. Ensure that the SPD can be configured so that desired traffic is encrypted/decrypted and that all other traffic is dropped.

To this end, the Linux XFRM framework is evaluated using the same Linux Personal Computer (PC) used in the performance evaluation test bench, discussed at length in section 4.3.4, with one notable change: the RapidM hardware link between the two router Network Namespaces (NetNSs) is replaced with a direct Virtual Ethernet (VETH) connection, as shown in Figure 3.3.

Each of the three objectives presented in this section is addressed in sections 3.5.1 to 3.5.3. For continuity the static SAD and SPD configuration used to give effect to the IP crypto in the performance evaluation test bench is presented as part of sections 3.5.2 and 3.5.3 respectively.

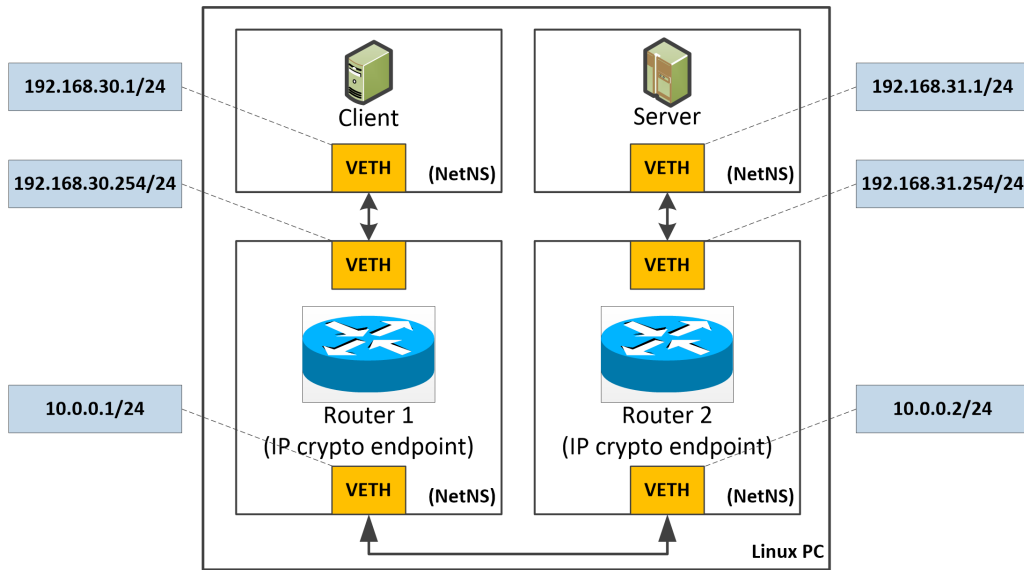


Figure 3.3: High-level Linux PC setup for XFRM framework evaluation

However, additional details of the test bench setup and PC configuration are required as a backdrop for the verification discussion of the IP crypto, notably regarding the role and function of the various NetNSs. Therefore, the verification of the XFRM framework SAD and SPD configuration, and by extension the IP crypto implementation, is presented at the end of the test bench architecture and configuration discussion in section 4.3.5.

3.5.1 Preventing automatic SA establishment

After further investigation, it turns out that the Linux XFRM framework does not include any automated key-exchange and/or SA setup mechanisms to begin with. Instead, popular open source IPsec based VPN solutions, such as strongSwan [66] and libreSwan [67], provide the IKE protocol implementation (see section 2.6.2.1) and other related features, but use the Linux XFRM framework to provide the core functionality of the IPsec protocol suite.

Therefore, to prevent any form of OTA key establishment or SA negotiation taking place, it is as simple as not including any IKE protocol implementations or similar in the eventual Linux XFRM framework based solution.

3.5.2 Static SAD configuration

The Linux manual pages for the IP-XFRM tool [65] make it clear that it is possible to statically configure valid SA entries, but are somewhat lacking in terms of explanation(s) around the expected sequence of commands required to create such an entry. As such, a more informal source, viz. [68], is consulted to determine if the static configuration options will be suitable for this study's implementation goal.

The tutorial presented in [68] provides a method for static SA generation using the IP-XFRM tool. This includes setting up the SA for tunnel mode use, specifying an AES GCM cipher option in accordance with IPsec related RFC 4106 [69], and providing keying material for the cipher.

This approach presents a potential security risk, as it does not include any mechanism to monitor and/or prevent IV rollover, or at least none that the author could determine. If an IV is reused (e.g., due to rollover), it opens up possible attack vectors if an attacker can obtain two packets encrypted with the same key/IV pair. This is explained in greater detail in section 2.5.2.2.3.

However, since the goal of this study is to evaluate the performance impact of the IP crypto, as stated in section 1.3, this risk is considered acceptable. Stated differently, whether or not the ciphertext produced by the IP crypto is actually secure is irrelevant for the purposes of evaluating the IP crypto's performance. Just as long as it executes the cipher algorithm correctly thereby accurately exhibiting the processing delay introduced by the IP crypto.

Two SA entries are generated in each router NetNS to allow tunnel mode IPsec operation on traffic exchanged between the two router NetNSs. All of these SAs are set to use the RFC 4106 compliant AES GCM cipher. The entries are as follows:

- In the Router 1 NetNS:
 - For use with outbound SPD entries: a tunnel mode SA with source address selector 10.0.0.1 and destination address selector 10.0.0.2 (cipher key and ESP SPI match that of the inbound SA in the Router 2 NetNS); and
 - For use with Inbound SPD entries: a tunnel mode SA with source address selector 10.0.0.2 and destination address selector 10.0.0.1 (cipher key and ESP SPI match that of the outbound SA in the Router 2 NetNS).
- In the Router 2 NetNS:
 - For use with outbound SPD entries: a tunnel mode SA with source address selector 10.0.0.2 and destination address selector 10.0.0.1 (cipher key and ESP SPI match that of the inbound SA in the Router 1 NetNS); and
 - For use with Inbound SPD entries: a tunnel mode SA with source address selector 10.0.0.1 and destination address selector 10.0.0.2 (cipher key and ESP SPI match that of the outbound SA in the Router 1 NetNS).

3.5.3 Static SPD configuration

Similarly to the static SAD configuration (section 3.5.2), informal sources are consulted for assistance with the static SPD configuration. This includes [68], as well as two additional sources, viz. [70] and [71].

The creation of static tunnel mode SPD entries is fairly simple, but creating the entries (rules) such that the desired traffic behaviour is obtained, i.e. encrypt and decrypt traffic from the Client to the Server NetNS and vice versa, but drop all other packets, proved challenging. In addition to specifying the selectors (section 2.6.2.3) that cause a packet to match a rule, this rests on the following three properties of XFRM SPD entries: the directionality of the rule (OUT, FWD, or IN), the priority of the rule, and the rule action (apply SA, ALLOW, or BLOCK).

Specifically, determining the necessary SPD entries (rules) to give effect to the desired traffic drop behaviour was more difficult than expected due to the rule directionality behaviour. No clear explanation of the directionality specifiers is given in the IP-XFRM tool manual pages [65], but information in the informal sources [68] and [70], claims that:

1. SPD entries with a OUT direction should only be searched when the XFRM framework decides which outgoing packets to encrypt, bypass, or block;
2. SPD entries with a IN direction should only be searched for incoming packets that have been decrypted and decapsulated and are addressed to the local system; and
3. SPD entries with a FWD direction should only be searched for incoming packets that have been decrypted and decapsulated and need to be forwarded (routed) to another system.

However, initial testing showed that the OUT rules seem to be applied to incoming traffic as well, after it has been processed against the FWD and IN rules. Subsequent verification testing, specifically the test in section 4.3.5.3, found that this is actually due to the XFRM rules being searched again as part of the forwarding (routing) process of the incoming traffic, i.e. when the incoming traffic from one interface becomes outgoing traffic on another interface, since the XFRM SPD rules are applied to all interfaces in the NetNS.

This unexpected OUT rule behaviour created a scenario where a ‘drop all network traffic that did not match another SPD entry’ OUT rule caused valid decrypted and decapsulated incoming traffic to be dropped instead of forwarded as expected (see observations of the verification test performed in section 4.3.5.3). This was corrected by adding an additional, higher-priority OUT rule with an ALLOW action for this valid traffic.

As a short aside, the IP-XFRM tool does include an optional parameter for both SAD and SPD entries referred to as an interface ID specifier in the tool manual [65], but the author only realised this after the completion of the IP crypto performance evaluation test work presented in chapter 5. This specifier can be used to limit the application of an SPD rule to a specific network interface, which offers finer control over the SPD rule design.

This finer grain control could have been used to limit the scope of the additional work around OUT rule design, and could possibly have given rise to slightly better overall SPD rule design. However, this would not make any material difference to the IP crypto functionality or its performance impact (see the rule priority discussion below), at least not as pertains to the test bench setup used in this study.

One last design consideration in the SPD rule setup relates to the potential impact of rule priority on packet processing speed. Each packet that triggers an SPD lookup in a particular SPD rule set (OUT, FWD, or IN), will cause a sequential search of the rule set according to rule priority, until the first matching rule is found (section 2.6.2.3.3).

Therefore, rules for valid (expected) test traffic need to have higher priority than block rules to prevent additional delays due to the lookup process. However, this point is moot in the context of the test bench used in this study, since the only rule set that needs more than one entry is the OUT set (see rule sets at the end of this section), and the block rule has to have a lower priority than the valid traffic rules to fulfil its function correctly in any case.

Furthermore, even if more rules were required for the IP crypto implementation and these were prioritised inefficiently, the additional delay due to the inefficient lookup is expected to be incredibly small - in the order of nanoseconds or microseconds at worst. Since the typical measured one-way latency of an HF link is in the order of 4 seconds (see results in section 5.2.4.2), this additional delay would not make any material difference to the test result measurements.

The process of determining the required SPD rules, and verifying their impact on the system, is presented as part of the overall IP crypto verification discussion in section 4.3.5. For continuity, the final set of SPD rules for each router NetNS is given here, sorted by rule priority, and then by rule directionality. The outbound and inbound SAs refer to the SA entries given in section 3.5.2 (each router NetNS only has one outbound and one inbound SA).

- In the Router 1 NetNS:
 - OUT rule: apply outbound SA for packets from source subnet 192.168.30.0/24 to destination subnet 192.168.31.0/24, priority 100;
 - OUT rule: allow packets from source subnet 192.168.31.0/24 to destination subnet 192.168.30.0/24, priority 100 (additional OUT rule to address unexpected behaviour);
 - FWD rule: allow incoming packets received on inbound SA from source subnet 192.168.31.0/24 to destination subnet 192.168.30.0/24, priority 100;
 - IN rule: allow incoming packets received on inbound SA from source subnet 192.168.31.0/24 to destination subnet 192.168.30.0/24, priority 100 (identical to previous rule, just for the IN rule set);
 - OUT rule: block all packets, priority 200.
- In the Router 2 NetNS:
 - OUT rule: apply outbound SA for packets from source subnet 192.168.31.0/24 to destination subnet 192.168.30.0/24, priority 100;
 - OUT rule: allow packets from source subnet 192.168.30.0/24 to destination subnet 192.168.31.0/24, priority 100 (additional OUT rule to address unexpected behaviour);
 - FWD rule: allow incoming packets received on inbound SA from source subnet 192.168.30.0/24 to destination subnet 192.168.31.0/24, priority 100;
 - IN rule: allow incoming packets received on inbound SA from source subnet 192.168.30.0/24 to destination subnet 192.168.31.0/24, priority 100 (identical to previous rule, just for the IN rule set);
 - OUT rule: block all packets, priority 200.

3.5.4 Linking back to the research study implementation goal

As should be evident from sections 3.5.1 to 3.5.3, the implementation goal as set out in 3.4.1 can be achieved by using the Linux XFRM framework as-is, together with the appropriate static configuration, hereafter termed the IP crypto implementation.

Therefore, verifying the IP crypto implementation is actually just a case of verifying that the XFRM configuration, viz. the SAD and SPD entries, give effect to the desired IP crypto behaviour. This is presented in section 4.3.5, after the rest of the test bench setup architecture and configuration is discussed.

3.6 Concluding remarks

This chapter presented the requirements and design of the IP crypto function (section 3.3) within the context of a potential IP over HF system (section 3.2). Possible implementation options are considered (section 3.4), and an IP crypto implementation suitable for the purposes of this research study is created using the Linux XFRM framework (section 3.5).

This implementation can be used to evaluate the performance impact of the IP crypto in a suitable test bench setup. In other words, the nature of the IP crypto implementation serves as an important input in the test bench development process.

4 Development of the test bench setup

This chapter discusses the development of a suitable test setup for the performance evaluation of the IP crypto. Section 4.1 provides an overview of the approach followed to conduct this development. The primary requirements of the testing setup are listed in section 4.2. Section 4.3 considers different testing approaches and discusses the architecture and configuration of a physical test bench setup using COTS equipment. Test data generation and result measurement are discussed in section 4.4, which notably includes the development of suitable latency- and throughput tests by modifying the open-source iperf3 test traffic tool as documented in sections 4.4.5 and 4.4.7. Lastly, some concluding remarks are presented in section 4.5.

4.1 Overview of the test development process

As a starting point in the test development process, the requirements of the testing setup are identified (section 4.2). These requirements are used as the main input in the decision about which testing approach to follow.

As mentioned briefly in section 4.3.1, a hardware setup consisting of COTS elements is used in favour of a simulation approach, due to the complexity of the various components required to adequately represent a functional IP over HF system, along with the characteristics of an HF/WBHF channel. Stated differently, there are simply too many variables to account for in order to develop a usable simulation model for the purposes of this research study in a reasonable amount of time.

For the same reason, viz. system complexity, there may be factors present in the system behaviour that could lead to unexpected anomalies or outliers in the test results. If this occurs, each anomaly will be investigated and clarified as necessary. This approach is taken as opposed to performing a characterisation exercise of each component or element in the test system, since such an undertaking will likely be complex enough to warrant dedicated research.

The architecture and configuration of this hardware setup is discussed at length in section 4.3, which includes motivation for the test setup configuration choices in light of the test setup requirements. This discussion concludes with the verification testing of the IP crypto implementation.

The last part of the test development process considers the generation of suitable test data, and the metrics which have to be measured in order to evaluate the performance impact of the IP crypto (section 4.4). This includes the selection and modification of a TCP/IP traffic generation tool, the development and validation of a suitable latency test by adding additional modifications to the tool, and the discussion of a throughput test which can be performed without additional modifications to the tool.

4.2 Primary requirements of the testing setup

In order to provide some context regarding the decisions made around the creation of a suitable testing setup, it is worth considering both what the test setup is *not* expected to achieve and what it should achieve.

The following is expressly *excluded* from the test setup requirements:

1. **Representing an HF system optimized for IP-based traffic**
The topic of optimizing IP data over HF (or WBHF) is certainly research worthy, as should be evident from the background given in section 2.4. However, it does not constitute part of the research goals of this study.
2. **Representing the characteristics of the potential HF system outlined in section 3.2**
While the IP crypto is designed for deployment in this system, the system, or a comparable equivalent, does not fully exist yet, to the best of the author's knowledge at the time of writing.

On the other hand, the test setup aims to achieve the following:

1. **Provide an accurate (enough) representation of the HF/WBHF environment**
The term "accurate enough" is perhaps somewhat subjective. However, a real-world setup, i.e. WBHF capable radios set up at different physical sites that can accurately provide HF/WBHF channels of varying conditions, is unfortunately not available to the author for this study.
2. **Allow for the transmission of IP data over WBHF**
Although not necessarily optimized for IP traffic, there are existing WBHF protocols which include provision for IP data transfer.
3. **Allow for a sufficiently controlled environment to provide repeatable and reliable results**
This can perhaps be considered obvious, but is stated here for completeness sake.

4.3 Test setup architecture and configuration

4.3.1 Possible testing approaches

To achieve the requirements (goals) set out in section 4.2, there are two main avenues to consider, bearing in mind that a physical (real-world) setup is not available or reasonably obtainable as stated in 4.2. These are: simulating the effects of an IP-capable HF protocol stack and the HF / WBHF channel itself on test IP data, or using existing Commercial Off-The-Shelf (COTS) equipment to create a physical test bench setup.

The simulation route presents significant risk, since additional work is required both to develop such a simulation and to prove that the simulation accurately represents the relevant HF effects and conditions. In other words, the testing and verification of the simulation model may very well prove to be more cumbersome than the actual and verification required for this study.

Fortunately, the COTS test bench option is a viable alternative, as the required hardware is readily available to the author, along with support from the manufacturer (Rapid Mobile (Pty) Ltd. (RapidM)) should the need arise. Therefore, this is the route taken to verify the viability of the IP crypto in an HF environment.

4.3.2 Test bench architecture

The test bench is represented in block diagram format in Figure 4.1. Accompanying descriptions of each block (i.e., hardware device) are given in Table 4.1. Note that the descriptions are purposefully kept brief since a detailed technical breakdown of each device is deemed unnecessary for the purposes of this dissertation.

Additional information regarding the test bench configuration is given in section 4.3.3 and details around the flow of IP traffic through the test bench is given in section 4.3.4.

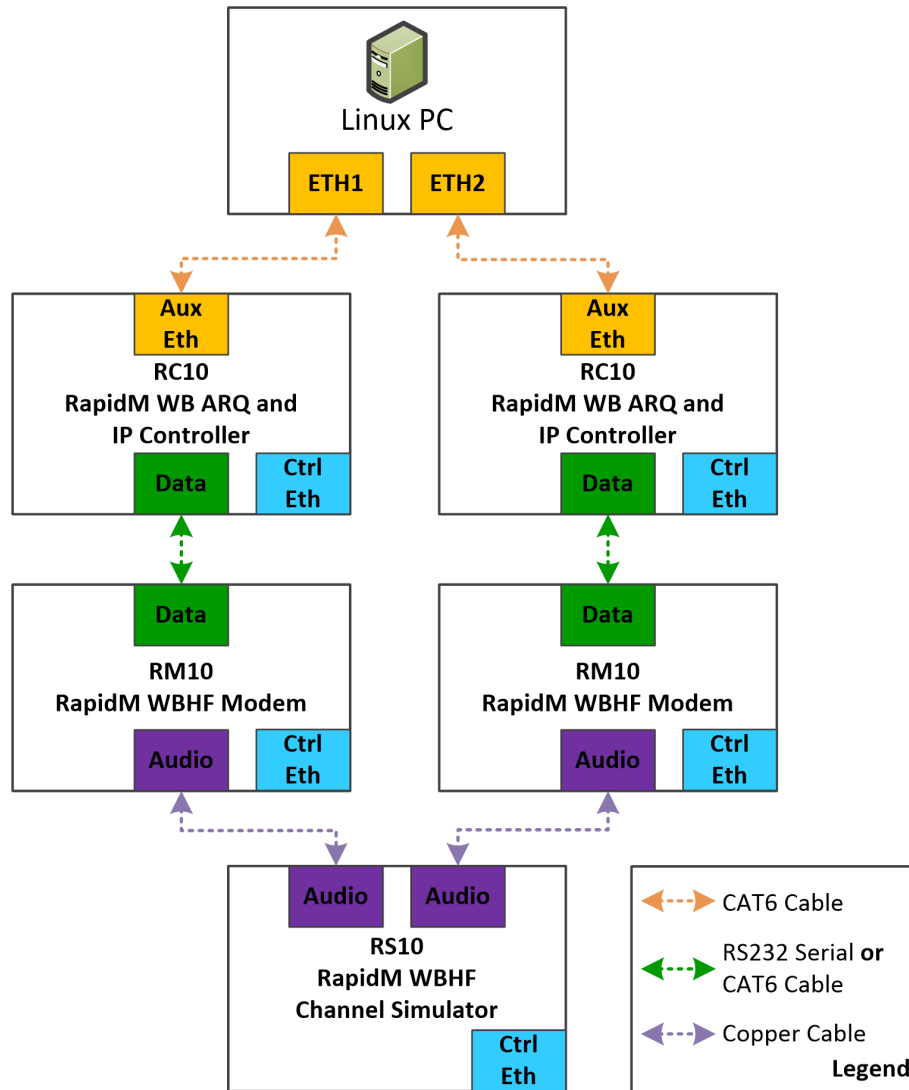


Figure 4.1: High-level test bench block diagram

Table 4.1: Description of each component in the test bench setup

Component	Description
Linux PC	A single Linux-based PC with multiple Ethernet NICs is used to host the IP crypto implementation, and for test traffic handling (data generation and result measurement). A more detailed PC setup diagram and discussion is presented in section 4.3.4. The test traffic handling is discussed in section 4.4.
RapidM RC10	The RC10 is described as a ‘ARQ server and IP controller’ in its datasheet [72]. It functions as a companion product to the RM10 (RapidM WBHF modem) and can provide IP data transfer capabilities over NBHF and WBHF via the embedded STANAG 5066 data stack and its associated IP Client, also embedded in the RC10. It is worth noting that the STANAG 5066 stack is Edition 4 compatible (i.e. WBHF compatible) even though Edition 4 is technically still in a draft state, as mentioned in section 2.5.2.2. This is common in the HF industry, since the ratification of a STANAG standard can be an extremely time-intensive process. Other RC10 features are omitted here for brevity’s sake.
RapidM RM10	The RM10 is labelled as a “Wideband HF Modem and ALE” product, although it should be noted that it also provides NBHF capabilities [73]. It provides a MIL-STD-188-110C Appendix D [40] compliant modem, with a waveform set that allows for bandwidths varying from 3 kHz to 24 kHz (in 3 kHz increments) and associated OTA data rates varying from 75 bps to 120 kbps. Note that the MIL-STD-188-110D revision of the standard [24] (i.e. for use of waveforms with bandwidths up to 48 kHz), discussed in section 2.3.8, is not supported by the RM10. The RM10 also provides WALE capabilities, along with other features which are omitted here for brevity’s sake.
RapidM RS10	The RS10 is RapidM’s “Wideband HF and V/UHF Channel Simulator” [74]. It provides an HF Watterson-Coon channel model and can be used to simulate various channel conditions, including the CCIR/ITU channels mentioned in 2.2.3.3.
RC10/RM10 Data connection	Data transfer between an RC10 and RM10 can either be via serial (synchronous or asynchronous; RS232 or RS422) or a TCP/IP connection (not to be confused with the IP packets that are actually sent OTA).
RM10/RS10 Audio connection	Since no radios are used in the test bench, the RM10 modem input and output are transferred as a baseband analog (audio) signal to or from the RS10. The connection is made using normal copper cable.
Control Ethernet	All of the RapidM devices provide a separate TCP/IP capable Ethernet port for control functions. This is mostly used to allow for RC10 modem control functions (see section 4.3.3), but can potentially be used for automated test control as well.

4.3.3 Test bench configuration

4.3.3.1 Baseline RapidM hardware configuration

As a starting point for the test bench configuration details, it is worth noting the Software (S/W) version information used for all of the equipment in the test bench. This is presented in Table 4.2.

Note that the Linux PC version details are included here for the sake of grouping the version information. Details of the PC configuration are presented in the IP data flow discussion in section 4.3.4.

Table 4.2: Software version information for each hardware device in the test bench

Component	S/W Version Info
Linux PC	OS: Ubuntu 22.04.3 LTS Kernel: Linux 5.15.0-1045-gke Architecture: x86-64
RapidM RC10	Function S/W version: 1.4.2.21 DSP APP: 4.4.1.18
RapidM RM10	Function S/W version: v1.3.5.41 DSP APP: v6.7.5.16.1.24
RapidM RS10	Function S/W version: 1.1.3.1

Both the RC10 and the RM10 provide various preset configurations for different features and/or operation modes. To configure the units for IP data over WBHF operation, a RC10 preset configuration is adapted, and the RM10 is configured for HF mode operation, i.e., it is not necessary to use a preset on the RM10, since WALE functionality is not used, as motivated in section 4.3.3.2.

The baseline RC10 configuration is done as follows:

1. Preset used: ‘STANAG 5066 WB, Fixed Frequency (FF), 120K, Asynchronous (ASYN)’ (with differing STANAG 5066 addresses for the two RC10s)
2. Data interface configuration changed to synchronous serial (external clock source)
3. Modem control enabled
This helps to ensure consistency of the RM10 modem state, since it allows the RC10 to set certain settings on the RM10 (such as the selected waveform) when the RC10 is set to the system online state, and allows for the use of the STANAG 5066 Data Rate Change (DRC) algorithm. (Note that the RC10 configuration is typically done when it is in a system offline state, during which the STANAG 5066 data stack is not operational.)
4. Enable the STANAG 5066 DRC algorithm
This allows the STANAG 5066 to dynamically adjust the data rate (which translates to a waveform change on the RM10) based on the current HF channel conditions (SNR measurements performed by the RM10). While not strictly speaking necessary for ideal HF channels, this functionality is crucial when testing with varying (non-ideal) HF conditions, e.g., using the CCIR poor channel model on the RS10.

5. Media Access Control (MAC) mode set to “BASIC”
This is essentially “MAC disabled”. This configuration choice is clarified in section 4.3.3.2.
6. STANAG 5066 IP Client enabled with ARQ delivery
(local node address set to match the physical Ethernet interface IP address in the corresponding router NetNS - see Figure 4.2 in section 4.3.4)

The baseline RM10 configuration is comparatively much simpler:

1. No preset used
2. RM10 operational mode set to “HF modem”
This operational mode allows the use of MIL-STD-188-110C WBHF waveforms, but disables other functionality such as WALE. This choice is motivated in section 4.3.3.2.
3. Data interface configuration set to synchronous serial (internal clock source)

It is important to note that the test run specific parameters are purposefully omitted here. For example, the HF channel bandwidth which is settable via the RC10 since modem control is enabled. Additionally, the RS10 configuration is not mentioned here, since the various channel models are configured simply by selecting the applicable preset.

4.3.3.2 Motivation for baseline configuration choices

As stated in section 4.2, one of the test setup aims is to provide an accurate (enough) representation of the HF/WBHF environment. This is partly achieved by the inclusion of the RS10 channel simulator and partly achieved by the STANAG 5066 configuration on the RC10, i.e., representing a real HF network’s configuration.

It is worth reiterating that STANAG 5066 is not designed for optimum IP over HF data transfer, as should be evident from the following discussion, and previously discussed in section 2.4.1. However, it is used in the test bench setup since it is currently, at least to the author’s knowledge, the only existing fully implemented and commercially available protocol that is widely adapted within the HF industry that is used for IP data transfer over HF, and WBHF with Edition 4 of the stack.

There are two sources to draw from when quantifying what constitutes a real STANAG 5066 network configuration for IP data transfer: requirements imposed by STANAG 5066 itself and deployments within the HF industry.

The IP Client specification in Annex U of STANAG 5066 Ed 4 [5] gives some guidance on how the IP Client can be implemented, but imposes virtually no hard restrictions on the STANAG 5066 network configuration. However, section 6.1. in Annex U indicates that the use of ARQ delivery between two nodes (point to point) is “preferred for efficiency in the IP and higher-layer protocols” [5].

Drawing from industry experience and support from Rapid Mobile’s STANAG 5066 expert [75], various types of STANAG 5066 deployments (i.e. network configurations) can be considered. This includes the following:

1. Fixed frequency, with channel access governed by the Carrier Sense Multiple Access (CSMA) MAC (Annex K of STANAG 5066 [5]);

2. Fixed frequency, with channel access managed by the Wireless Token Ring Protocol (WTRP) (Annex L of STANAG 5066 [5]); and
3. Scanning network (i.e. multiple frequency), with WALE use.
STANAG 5066 Ed 4 does not provide ALE directly, but supports interfacing with MIL-STD-188-141D ALE mechanisms via its Channel Access Sublayer (CAS), as outlined in section 3.4 in the main body of the standard [5]

Typically fixed frequency, or “single shared channel”, networks are subject to fairly long forwarding delays (e.g. long one-way latency times) due to the nature of the CSMA MAC and WTRP. For example, initial one-way latency can range from 30 to 40 seconds when the CSMA MAC is used, and subsequent latency at each link turn-around easily reaches the 30 second mark, as observed from some exploratory testing with the test bench hardware.

Data passed to the STANAG 5066 stack is concatenated by the stack’s Data Transfer Sublayer (DTS) to form larger HF payloads, so not all IP packets will experience the same delay. However, the type of IP services that can be successfully used with this type of latency is still severely limited.

Although not directly attributed to the latency effects, the limitation of viable IP services when using STANAG 5066 is expressed in the IP Client specification (section 3 of Annex U [5]). Notably, TCP and HTTP are not expected to work for most deployments, due to the interaction between TCP and the IP Client subnet (i.e. the STANAG 5066 stack).

However, when used in a deployment with WALE functionality, the link turn-around time is expected to be much quicker after the initial link-setup process concludes between two nodes. This holds true for point-to-point links, since ALE will ensure that only two nodes will access the channel in this case, thereby eliminating the need for CSMA-type mechanisms. For this reason, the test bench configuration aims to resemble the functionality of a scanning network with WALE use, i.e., the best case STANAG 5066 deployment for IP traffic.

The above statement may seem slightly contradictory, since the RC10 configuration is based on a fixed frequency preset (section 4.3.3.1), and WALE functionality is not enabled on the RM10 (requires a different operational mode on the RM10). This is done because part of the WALE functionality is to negotiate the bandwidth of the channel used. For example, if the network configuration includes channel(s) with a maximum allowable bandwidth of 24 kHz, WALE will negotiate the highest bandwidth (in 3 kHz increments) based on current channel conditions.

Therefore, it is not possible to guarantee the channel bandwidth used for a test if WALE is used. This makes test repeatability difficult, as well as test result comparison, for example, when repeating the same test with and without the IP crypto enabled in the test bench.

However, since there are only two STANAG 5066 nodes present in the test bench, both the WALE and STANAG 5066 MAC mechanisms can be disabled (only “basic” MAC used on RC10, which is essentially “MAC disabled”), to mimic the same system characteristics (especially latency) as if WALE was employed. This may cause a slight change in initial link setup time, which might be slightly faster than if WALE was employed, but since it does not affect the rest of the test flow, it is deemed a worthwhile trade-off to ensure test repeatability.

Note that STANAG 5066 goes through a link setup phase prior to starting data transfer, regardless of whether MAC mechanisms are enabled on the RC10, or if WALE functionality is enabled on the RM10.

4.3.4 Test bench IP data flow

The function of the PC in the test bench is two-fold, namely: hosting the IP crypto implementation, using the Linux kernel XFRM subsystem as discussed in section 3.5, and providing test traffic generation and measurement capabilities. A single PC is used to simplify test control and automation, as well as to ensure the same clock source is used for test traffic generation and result measurement, which is especially critical for latency test result measurement (see section 4.4.5.2).

This creates the need to ensure the following:

1. That generated test traffic (traffic for transmission) is clearly separated from measured test traffic (traffic received for analysis), thereby ensuring that it always traverses the HF bench and not the PC network stack back plane;
2. That traffic presented to the HF system is not contaminated with test automation- or other normal PC Internet IP traffic; and
3. That the IP crypto implementation can be added or removed from the data path as needed for different tests.

To achieve this, two features of the Linux kernel are employed: Network Namespaces (NetNSs) and Virtual Ethernet (VETH) devices. A NetNS is described by [76] as a logical “copy of the network stack, with its own routes, firewall rules, and network devices”, which includes the Linux XFRM subsystem used to provision the IP crypto implementation. As the acronym implies, VETHs are virtual Ethernet devices, which “can act as tunnels between NetNSs” among other uses [77].

Using these two features makes it possible to create a complete virtual testing network without the need to introduce additional virtualization (hypervisors) to host full virtual machines. However, this approach still provides the flexibility to add virtual machines if the need should arise, since a hypervisor can be run within a NetNS.

A block diagram depicting the setup of the Linux PC is given in Figure 4.2. Separate NetNS are created for client (traffic generation) and server (traffic measurement and analysis) purposes. Two router NetNS are created to host the IP crypto implementation, or to simply forward (route) the network traffic if the IP crypto’s are disabled. A physical Ethernet NIC is assigned to each router NetNS to connect to the HF hardware in the test bench, i.e. to the RC10s.

It is important to note that all the offload functionality of the Linux kernel is disabled for all network interfaces, that is, VETHs and physical interfaces, in the PC setup via simple commands from the ETHTOOL user space tool, i.e., no kernel source modification is required. This includes TCP segmentation offloading and the generic segmentation and -receive offloading, amongst other things. These offload mechanisms are intended to speed up network traffic processing by leveraging the capabilities of certain NICs [78], but the author has observed that this has strange side effects during IP crypto verification testing (section 4.3.5) even on VETH interfaces, such as the TCP segmentation size not adhering to the limits set via normal socket configuration.

Since these offload mechanisms are definitely not required to improve performance for IP packets transferred over an HF channel, i.e. the bandwidth is much too low for the PC side processing to become the bottleneck, they are disabled to remove unknowns from the test setup, especially as relates to TCP test traffic generation.

The general network traffic flow is described here to better illustrate the function of the NetNSs and VETH pairs. Traffic (IP packets) generated in the client NetNS flows through the 192.168.30.0/24 subnet VETH pair to the Router 1 NetNS. If the IP crypto is enabled, ESP encryption and encapsulation is performed (see section 2.6.2.4) prior to transmission via the first physical Ethernet NIC (10.0.0.1/24). Otherwise, the packets are simply forwarded as-is via the Ethernet NIC.

After traversing the HF test bench (which can be viewed as a transparent point-to-point link from the PC perspective), the packets are received by the second physical Ethernet NIC (10.0.0.2/24) in the Router 2 NetNS. If the IP crypto is enabled, ESP decapsulation and decryption is performed, prior to forwarding the packets to the server NetNS via the 192.168.31.0/24 subnet VETH pair. Otherwise, the packets are simply forwarded as-is via the VETH pair.

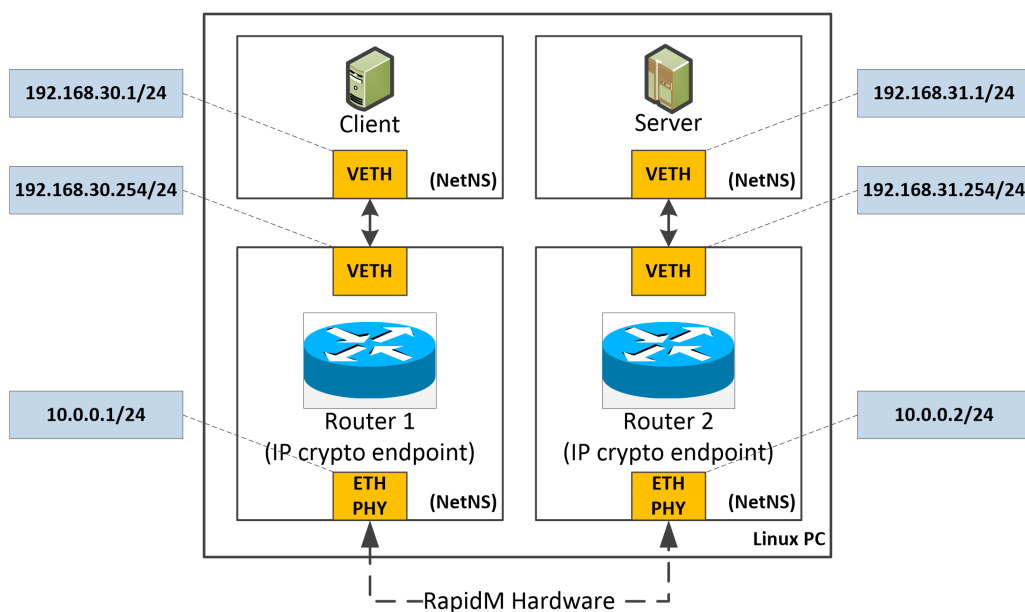


Figure 4.2: High-level Linux PC setup

Before discussing the data flow through the rest of the test bench, a quick overview of the STANAG 5066 internals is required. Figure 4.3 shows a high-level view of the STANAG 5066 data stack sublayers. Both the application and the crypto, modem, and radio blocks are included to provide context but do not form part of the STANAG 5066 stack.

Similarly to the OSI model approach to protocol design, each layer (or sublayer according to the STANAG 5066 terminology) defines a PDU format for layer-to-layer communication, which typically encapsulates the PDU from the higher layer as its data payload and adds a new header.

However, this does not apply directly to M_PDUs. The term M_PDU is used ambiguously in the standard [5], and can refer to management PDUs which are internal to the stack, or to MAC

layer PDUs (as is the case here), which are actually D_PDUs with different type field values used for MAC algorithm functions. In other words, data D_PDUs are typically not encapsulated in M_PDUs, but are used directly in transmissions to/via the modem.

While this is somewhat an simplification of the STANAG 5066 stack (e.g. the interface between the CAS and the modem for ALE purposes is not shown), it is deemed sufficient to provide context for the remainder of this section.

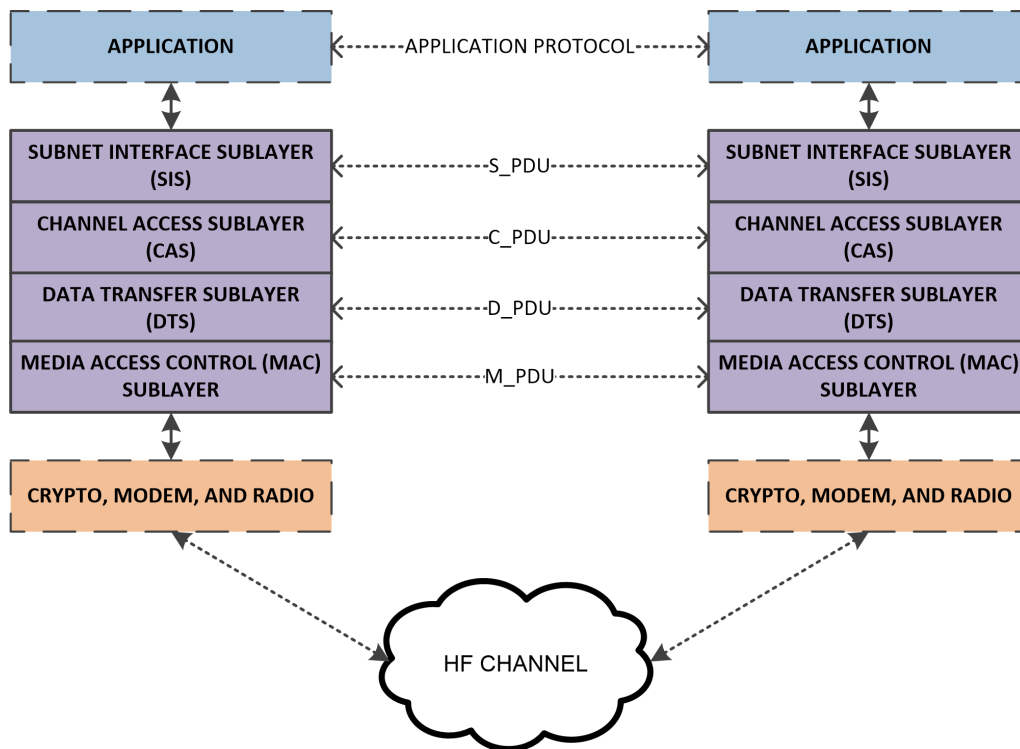


Figure 4.3: STANAG 5066 Ed 4 Layer Overview (optional layers omitted; adapted from [5])

From the perspective of the HF portion of the test bench (RC10/RM10/RS10 hardware), IP packets traverse dedicated Ethernet links between the physical Ethernet interfaces of the PC and the corresponding RC10. Once received by the RC10, the packets are handled by the embedded IP Client in the RC10. The IP Client in turn interfaces with the STANAG 5066 stack via the Subnet Interface Sublayer (SIS), and IP packets are submitted to the SIS as-is, i.e. each IP packet is encapsulated fully in an S_PDU by the SIS.

If no STANAG 5066 link has been established yet, the SIS will request the CAS to establish the required link, i.e. link to the intended STANAG 5066 destination node (not to be confused with modem level ALE, although modem ALE - since it is enabled in the test bench configuration - will be executed during this linking process). While not identical, this linking process is similar (at least in concept) to that of a TCP handshake (STANAG 5066 can be viewed as a streaming protocol in the same sense that TCP is a streaming protocol).

Once the link is established, the SIS submits S_PDUs to the CAS. The CAS encapsulates each S_PDU into a data C_PDU, which only adds 4 bits of overhead to distinguish it from other C_PDU types, and submits these to the DTS. The DTS layer functionality is slightly more complex, notably since functionality such as data segmentation and node-to-node ARQ is provided by the DTS, but

ultimately the same principle applies: C_PDUs are encapsulated in D_PDUs.

There are several requirements governing D_PDUs transmission, as laid out in section 7.5, Annex C of [5], although there is some room for implementation-specific decisions. Amongst other things, this includes: determining the number of D_PDUs to send in a single transmission, transmission speed, and interleaver size.

The maximum allowable length of a single transmission is 127.5 seconds. The RapidM implementation purposefully skews the transmission length decision by allowing the link master (usually the node which initiated the link) to send full 127.5 second transmissions, but only allowing the receiving node a responding transmission time of 10 seconds maximum. This behaviour is enforced until the link master no longer has any data to send, at which point the receiving node will become the link master and the restrictions are swapped [75].

Once the creation of the transmission data stream is complete, it is transmitted to the sender RM10 for HF transmission. If the serial link between the RC10 and RM10 is used in synchronous mode, the RM10 matches the serial port rate with the waveform OTA rate. As the RM10 receives the data stream, it is transmitted via the channel simulator (“OTA”) to the receiver RM10, with some additional latency due to interleaver block effects (see section 2.3.5).

On the receiving side, the entire data flow occurs in reverse. The transmission stream is sent to the receiver RC10 where it is handled by the DTS. As DTS overhead is removed and C_PDUs are reassembled, they are passed back to the CAS, from the CAS to the SIS, to the receiving IP Client, and then back to the PC via the other physical Ethernet interface.

4.3.5 Verification of the IP crypto implementation

As mentioned at the start of section 3.5, the IP crypto evaluation, i.e. Linux XFRM framework evaluation, is done using the test bench PC, but with the link between the two router NetNS replaced with a direct VETH connection. This is represented in Figure 4.4, which is identical to Figure 3.3, but it is repeated here for convenience.

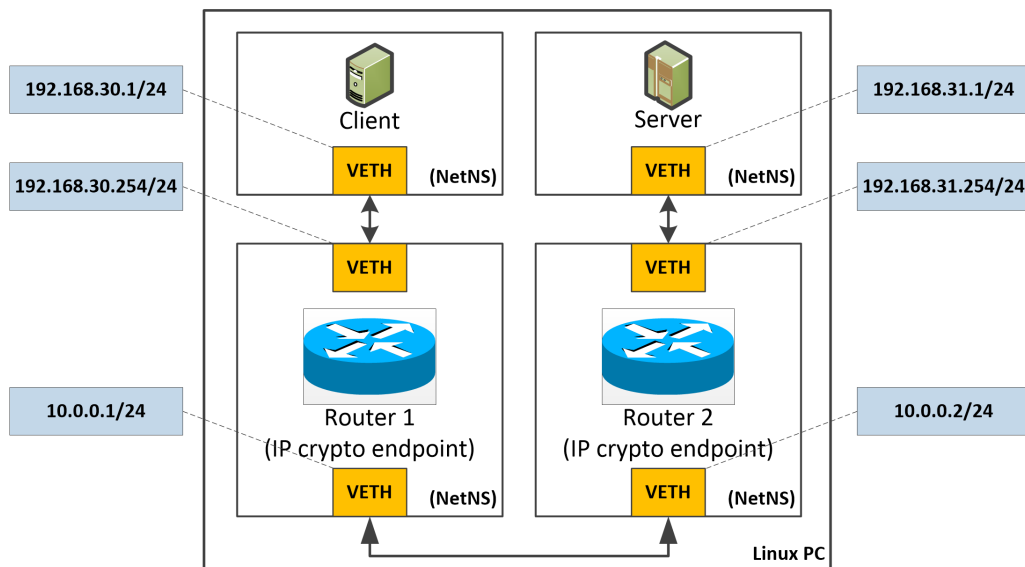


Figure 4.4: High-level Linux PC setup for XFRM framework evaluation

The same PC setup is used to verify the IP crypto functionality, viz. the Linux XFRM framework SAD and SPD configuration. This is done by running a series of tests using either the normal Linux PING utility (in accordance with the exception in section 3.3.4.7) or the standard iperf3 tool (section 4.4.3), and monitoring both the test result (from a execution perspective, i.e. did the test execute successfully or not), and the packet flow in the router NetNSs via packet captures.

The various tests and their results are discussed in sections 4.3.5.1 to 4.3.5.4. Each section presents a short conclusion as to what the specific test verifies. Taken together, these tests verify the functionality of the IP crypto.

4.3.5.1 Normal ping test without any XFRM configuration

The first test is a simple ping test from the client to the server NetNS, without any XFRM configuration set up but with the necessary routing table entries configured in the router NetNS to forward packets correctly.

The ping test was executed successfully, and the packet captures showed that ICMP echo requests and responses were forwarded in the clear. This verifies that the basic NetNS configuration is correct and functional.

4.3.5.2 Basic XFRM configuration tests

For this test, the SAD configuration (section 3.5.2) is applied in each router NetNS, and all of the SPD rules that have actions related to an SA are added (section 3.5.3), for a total of three rules in each router NetNS. In other words, no rules with BLOCK or ALLOW actions are added for this test.

A simple ping test from the client to the server NetNS was successfully executed with this configuration. The packet captures indicate that tunnel mode ESP packets are exchanged between the two routers, that is, packets with a new IP header using the router 10.0.0.1 and 10.0.0.2 IP addresses, and carrying an encrypted ESP payload.

This verifies that the SAD entries are correctly configured, that the SPD entries related to packet encryption and encapsulation, and decryption and decapsulation are correctly set up, and that no form of SA negotiation or key exchange takes place before the IP crypto starts generating encrypted packets. Additionally, it is also observed from the packet captures that the total overhead added by the XFRM framework, including the new IP header, amounts to 56 bytes.

A second test is performed with this XFRM configuration, using the standard iperf3 testing tool, to test if the ICMP PMTU discovery is functional. The iperf3 client connection, run from the client NetNS, is set to transfer 2000 bytes to the iperf3 server, run from the server NetNS, and forced to use an initial TCP Maximum Segment Size (MSS) size of 1460 bytes, using the --SET-MSS configuration option.

This leads to the client generating at least one TCP packet with maximum payload, i.e. results in a full 1500 byte IP packet. Upon receiving the packet, the router 1 NetNS generates an ICMP Fragmentation Needed message, since it cannot apply tunnel mode ESP encryption without generating an IP packet that is larger than the standard 1500 byte Ethernet MTU limit. The cli-

ent responds by re-sending the data in smaller TCP packets, and the test is successfully completed.

This verifies that ICMP PMTU discovery is functional if no BLOCK rules are present in the XFRM configuration, which is relevant when considering the result of the last verification test (section 4.3.5.4). Additionally, it is observed from the packet captures that the total overhead added by the XFRM framework amounts to 56 bytes for small packets, e.g. TCP Synchronize (SYN), which matches what is seen with the ping test but is constant at 54 bytes for larger packets, i.e. a IP packet with a total size of 1446 bytes, is always encrypted (ESP tunnel mode) to yield a new IP packet of 1500 bytes.

4.3.5.3 XFRM configuration test with block all rule

The same XFRM configuration is used for this test as for the basic XFRM configuration test (section 4.3.5.2), with one addition: a block all OUT rule is added in each router NetNS, with a lower priority than the SA related rules.

Once again a simple ping test is executed from the client to the server NetNS. This ping test fails, i.e. no ICMP echo responses are received at the client NetNS. Looking at the packet captures, the ICMP echo requests from the client NetNS are traced as follows:

- ICMP echo requests are successfully sent out from the client NetNS;
- The requests are correctly encrypted and encapsulated into tunnel mode ESP packets in the router 1 NetNS;
- The requests are received by the router 2 NetNS and correctly decapsulated and decrypted at the 10.0.0.2 VETH;
- However, the requests are never sent from the 192.168.31.254 VETH.

This observation led to the workaround of adding an additional ALLOW action OUT rule to prevent valid test traffic from being dropped as discussed in section 3.5.3.

4.3.5.4 XFRM configuration test with the final set of SPD entries

The last verification tests are performed with the full SAD entry set, and with the final set of SPD entries in each router NetNS as given in section 3.5.3. As in previous cases, a simple ping test is performed from the client to the server NetNS.

This ping test executed successfully, and packet capture inspection verified that the ICMP echo request and response traffic exchanged between the router NetNSs is encrypted and encapsulated correctly in tunnel mode ESP packets and decapsulated and decrypted as expected.

Secondly, the iperf3 test performed in section 4.3.5.2 is repeated here to determine if the BLOCK all rule still has the desired effect with the workaround OUT rule added (see the observations from the previous verification test in section 4.3.5.3).

The iperf3 test fails, as expected, and packet capture inspection verifies that the initial 1460 byte payload TCP packets are received by the router 1 NetNS, but no ICMP Fragmentation Needed packets are sent back to the client NetNS. This verifies that the BLOCK action OUT rule still causes

unexpected traffic to be dropped, thereby verifying that the IP crypto does not bypass traffic that does not match encryption rules (see section 3.3.4.6).

As an aside, this point could have been verified with ping tests executed directly to or from the router NetNSs, but the iperf3 approach was taken here to serve a dual purpose of verifying the IP crypto, and providing justification for one of the iperf3 modifications made in section 4.4.4.

4.4 Test data generation and result measurement

4.4.1 Measurement goals and test design decisions

4.4.1.1 Testing goal and performance metrics

The primary testing goal is to measure the impact of the IP crypto function on the performance of IP data transmitted over WBHF. This encompasses answering the slightly more fundamental question: Is IP data transmission via WBHF still viable if the IP crypto function is introduced? To this end, several network performance metrics are considered in the test design, including: throughput, packet loss, latency, and jitter.

These metrics, along with several others, are formally defined in the benchmarking RFCs [79] and [80]. The definitions include technical references as to how these metrics should be measured, either at a frame- or bit level as is the case in [79], or at a packet level, as is mostly the case in [80].

However, for the purposes of this study, measurements are performed based on application layer data and, as such the formal technical definitions of the performance metrics are not fully applicable. Rather, the manner in which each metric is measured is laid out in sections 4.4.5 and 4.4.7.

The decision to measure results based on application layer data is made based on a number of factors: the likelihood of hiding the impact of the overhead introduced by the IP crypto if measurements are done at the IP packet- or Ethernet frame level; the possibility of hiding transport protocol effects if measurements are done at a packet level; and the benefit of- versus the development effort required to create or modify test tools for measurements at an IP packet level or at an application data level.

4.4.1.2 Transport layer protocol

Since measurements are performed on application layer data (section 4.4.1.1) another key consideration is the choice of transport layer protocol used. As explained in the test bench configuration discussion, specifically in section 4.3.3.2, TCP is typically not a protocol of choice in real life HF deployments. However, when used with a STANAG 5066 deployment such as that created with the test bench configuration, i.e. with WBHF waveforms and WALE like functionality, which leads to much shorter link turn-around times as motivated in section 4.3.3.2, the use of TCP, while not optimal, is expected to at least become viable.

Additionally, the expectation within the HF industry is that a mixture of UDP and TCP services will be supported by IP over WBHF systems, although TCP services are expected to always make use of some form of PEP. Both of these claims are reflected in the IP over HF system design presented in section 3.2.2, and is specifically mentioned in the short description of the IP PEP in Table 3.1.

This implies that TCP without any form of PEP (“pure” TCP), while not an expected use case in a typical IP over HF system, can be viewed as a type of worst-case scenario for IP services over HF. Therefore, if the performance of the IP crypto is evaluated with “pure” TCP based test traffic, the results should represent the worst-case performance impact that the IP crypto can have on an IP over HF system. Stated differently, if “pure” TCP is viable with the IP crypto included in the system, all other IP services (UDP based, TCP based with a standard IP PEP, and TCP based with a service-specific PEP) should be viable with the IP crypto included in the system. As such, “pure” TCP is chosen as the transport layer protocol to use.

4.4.1.3 Effect on packet loss

Performing measurements on application layer data in combination with the use of TCP has at least one notable consequence: packet loss is not directly measurable. However, it is possible to obtain the number of TCP retransmits via the metadata tracked per TCP connection by the Linux network stack in the `STRUCT TCP_INFO`, which is obtainable in user space via a call to `GETSOCKOPT()` [81]. Although this is not a direct replacement for the packet loss metric, it provides a basic indication of whether packet loss (dropped or timeout) occurred at the TCP level during a test or not, which can be used during test result analysis if required.

4.4.2 Selecting a suitable TCP/IP test traffic tool

There are two options to consider regarding the selection and/or development of suitable test traffic tool(s):

1. Creating proprietary testing tool(s) from the ground up; or
2. Using and/or adapting existing open-source testing tool(s)

Given that viable open-source tools exist, as discussed below, and the time and effort that would be required to develop proprietary tool(s), the first option is not really worth entertaining any further.

Existing open-source tools which can be considered for the second option include:

1. `iperf3` [82]
2. `nuttcp` [83]
3. `hurl` (primarily for throughput) [84]
4. `wrk2` (primarily for latency) [85]

Both `hurl` and `wrk2` are HTTP based testing tools which are intended for use with a separate web server. This implies that extensive modifications will be required to remove the HTTP protocol implementation and implement ‘TCP only’ based tests.

The `iperf3` and `nuttcp` tools, on the other hand, are much closer to the desired goal. Both tools implement test data transfer directly over TCP, both provide a client and server implementation within the same tool, and both include performance metric result measurement features.

However, `iperf3` appears to have a much larger support base, as evidenced by the project’s GitHub community and active development [86]. Additionally, the author has previous experience with the use of `iperf3`, as well as a basic understanding of its internal workings. Therefore, `iperf3` becomes the natural choice to use.

4.4.3 High-level overview of iperf3 operation

A conceptual view of the connections made during an iperf3 test is given in Figure 4.5. This diagram and the following discussions are created based on the use of the iperf3 tool and an investigation of the source code [86].

A single TCP control connection is established at the start of a test, which is used by the client and server to exchange test control commands and to transfer test result summaries, or full server-side results depending on the test options used, from the server to the client.

After the initial test commands and server information is exchanged between the client and the server, the client starts establishing data connections to the server in a one-after-the-other fashion. The number of data connections is determined by passing the `--PARALLEL` parameter to the client at start-up.

Once all data connections are established, the test is started via the control connection, and the client starts generating and sending data via the data connections. The test ends if the user terminates the client, or if a specific test end condition is hit, which needs to be specified at client start-up.

Possible test end conditions include: a time limit, a total byte transferred limit, or a total number of blocks transferred limit. A block is essentially a buffer of data of configurable length. The iperf3 client will always generate data as complete blocks and attempts to divide the blocks equally between each of the data connections for transmission to the server.

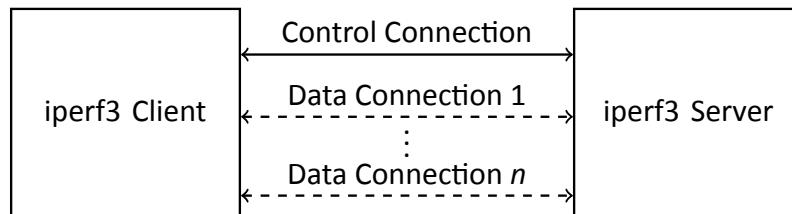


Figure 4.5: iperf3 Test Connections

4.4.4 General modifications to enable iperf3 use over an HF channel

Some general modifications to the iperf3 tool are required to stabilize iperf3 operation over an HF channel. Most of these modifications are made to address issues observed during exploratory tests with iperf3 over the WBHF test bench. The changes can be summarized as follows:

1. Modify error handling to ensure that server test results are logged before the test state is flushed (reset) on the server.
This is especially necessary if the grace period timer (an iperf3 application timer) times out during test closure before the client terminates the control connection. The timeout causes the server to drop and reset the control connection, presumably as a safety- and/or clean-up mechanism to ensure the control connection does not remain open indefinitely.
2. Extend the grace period timer timeout length to account for control traffic exchange delays due to HF channel conditions.

3. Prevent the client from prematurely ending a test that is byte- or block-count limited if there is still data in the kernel TCP socket buffers of the data connections.
This ensures that all client-generated data is actually transferred to the server before the test is terminated, since the kernel network stack only removes data from the TCP socket buffer after the corresponding TCP ACK has been received.
4. Limit the control TCP socket's MSS to a maximum of 1300 bytes.
Due to the crypto configuration, the ICMP PMTU discovery mechanisms are not functional (see section 4.3.5.4). Therefore, the control socket packet size has to be limited to ensure that packets do not exceed the path MTU of 1500 bytes if crypto overhead is added.

To verify that these changes do not hinder or alter the behaviour of the iperf3 tool such that it no longer fulfills its original function, a simple back-to-back test is performed to compare the functionality of the unmodified iperf3 tool with the modified version. This test is performed on the Linux PC by connecting the two Ethernet NICs in the two Router NetNS directly to each other, ensuring that the IP crypto is disabled, and running the iperf3 tool in the Client and Server NetNSs.

The iperf3 server is started in the Server NetNS with all default configuration parameters, and the iperf3 client is run from the Client NetNS without any parameter changes beyond specifying the server IP address. In this default mode, the iperf3 client opens a single data connection to the iperf3 server, sends as much data as possible to the server for 10 seconds, calculates the resulting throughput, and terminates automatically.

This default test is repeated 5 times with the unmodified tool, and 5 times with the modified tool. When comparing the results presented in Table 4.3, it is clear that there is no significant difference in the throughput measured by the unmodified and modified tool, verifying that the modified tool still performs the original iperf3 function correctly.

Table 4.3: iperf3 Modification Verification - Throughput (*Mbps*) measurements

Test Run	iperf3 Tool	
	Unmodified	Modified
1	943.397	943.501
2	942.764	943.505
3	943.397	943.922
4	943.082	942.872
5	942.663	941.403
Mean ($\bar{x}$)	943.061	943.041

4.4.5 iperf3 TCP-based latency test development

By default iperf3 does not provide latency or jitter metrics (results) for TCP based tests. However, it already records the TCP Round-Trip Time (RTT) metadata for data connections, and the source code contains re-usable timing functions. This, along with the iperf3 result collection infrastructure, provides a good basis to develop a TCP based latency test by making some additions to the iperf3 tool.

The basic premise of the latency test is fairly simple: use the existing client-side TCP RTT measurements, obtained from the kernel-tracked TCP metadata in `STRUCT TCP_INFO`, as a round-trip latency metric; and add timestamp data to the client generated block data, along with analysis logic on the server side to obtain one-way latency and jitter statistics. This approach makes it possible to measure one-way latency and jitter, as well as an RTT metric using the same TCP connection, viz. the same `IPERF3` data connection, although the measurement methodologies and time sources differ.

4.4.5.1 Modifications related to the TCP RTT measurements

The existing TCP RTT measurements are nearly usable as-is, but some changes are required regarding how the client calculates the mean RTT. Querying the kernel `STRUCT TCP_INFO` metadata will always return the last calculated RTT value, which the kernel handles according to RFC 6298 [87], with no indication as to whether the measurement has been updated since it was last read. The `iperf3` assumption, which is sensible for normal IP networks, seems to be that querying the RTT value for every result reporting interval (roughly every second) is sufficient to ensure that the RTT value would have changed.

However, this assumption does not hold for an HF/WBHF network, or any high latency network really. Therefore, some additional logic is added to only include an RTT sample in the mean calculation if it differs from the previous measured RTT value. Theoretically, this leaves a small margin for error if the RTT value happens to stay the same between updates, but since the kernel tracks it with a microsecond resolution, it is highly unlikely that this will occur.

It is also worth noting that a modification was made to store the actual RTT samples used in the mean calculation so that additional statistical analysis, such as calculating the Standard Deviation (SD), is possible for the RTT measurement data.

4.4.5.2 One-way latency and jitter test development

The payload modification premise, relevant for one-way latency and jitter measurements only, is illustrated in Figure 4.6. The timestamp size (8 bytes) is based on the existing `iperf3` timing structures, which provides a microsecond time resolution. Some of the random test data at the start and end of a data block is replaced with timestamp data if the block size is sufficiently large. If the configured data block size is too small to accommodate two timestamps, only the start-block data is replaced with a timestamp. If this fails, no timestamp data is included in the data block.

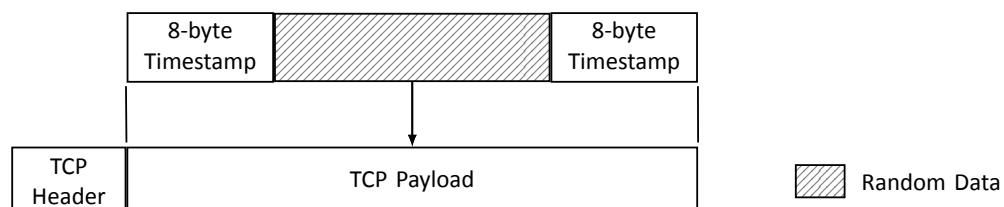


Figure 4.6: `iperf3` TCP based latency test payload

The end-block timestamp is included in order to improve the potential test utility. This may be slightly over-engineered, but there could be some value in tracking two different timestamps if

the data block size is such that it spans multiple TCP datagrams.

The timestamp value is obtained from the monotonic system clock, and is set when the client generates the data block immediately prior to writing the data to a TCP data connection socket. When the server receives a data block, the timestamp is processed to obtain a one-way latency and jitter sample, for both the start-block and end-block timestamps.

The one-way latency sample is obtained by taking the time at which a data block is Rx'ed (*BlkRxTime*) and subtracting the Timestamp (TS) value within the Rx data block (*BlkTS*), as is represented in (4.1). The *BlkRxTime* value is obtained from the same clock source (monotonic system clock) as the *BlkTS*, which ensures that no error is introduced due to clock source discrepancies.

$$OneWayLatency_{\mu sec} = BlkRxTime - BlkTS \quad (4.1)$$

The jitter sample is obtained by calculating the absolute value of the difference in one-way latency measurements for the current data block (n) and the previous data block ($n - 1$), as represented in (4.2). This implies that there will always be one less jitter than latency sample.

$$OneWayJitter_{\mu sec} = |OneWayLatency_n - OneWayLatency_{n-1}| \quad (4.2)$$

4.4.6 iperf3 TCP-based latency test validation

Due to the interaction between the client side (user space) data generation and handling, and the kernel TCP stack, as well as the interaction between the kernel TCP stack and the entire WBHF data path, there are some restrictions that have to be imposed on the latency test to obtain reliable results.

The primary issue stems from two factors: the data block timestamps are generated in the iperf3 client (i.e. user space), and not at the actual time that the resulting TCP packet(s) are transmitted from the client NetNS VETH to the client router NetNS VETH; and there is no guarantee that data is transmitted as soon as it is submitted to the TCP socket by the iperf3 client.

The actual transmission time is dependent on the state of the TCP flow, i.e. is there sufficient space in the current TCP window to transmit more data onto the wire or not? Given the high latency, low bandwidth characteristic of an HF/WBHF channel, it is highly likely that iperf3 client data can lie idle on the TCP socket send buffer for quite some time, implying that the timestamps can age prematurely.

To address this, a configuration option, titled `--TCP-BASED-LATENCY-TEST`, is added for the iperf3 client, which introduces a restriction that limits the iperf3 client to sending a single data block at a time, and only generating and sending another data block after the last sent data block is ACKed at the TCP level.

This check is done in user space by querying the number of bytes in the TCP socket write buffer using the `TIOCOUTQ` option with `IOCTL` system calls [88]. If the TCP write buffer is empty and the TCP connection is still active, this implies that the previous data in the buffer has been ACKed at the TCP level.

The implication of this modification is that one-way latency and jitter tests must be performed separately from throughput tests. It also limits the scope of one-way latency and jitter testing to lightly loaded network conditions. However, it still provides a usable and reliable latency test that can be used to explore the impact of the IP crypto on the latency characteristics of the network under different WBHF channel conditions.

4.4.7 iperf3 Throughput test overview

As briefly mentioned in section 4.4.3 the available configuration parameters make it possible to set a test end condition based on the amount of data that is transferred. This is a more reliable / predictable test end condition than a time limit, since a time limit condition simply stops iperf3 dead in its tracks regardless of the current data transfer state.

That being said, one modification is required to ensure that the iperf3 client always sends the exact same amount of data when it is byte limited. The iperf3 client contains an internal logic concept labelled as *multisend* in the source code. This was presumably introduced to allow the client to send more data per iperf3 processing cycle, which could present some benefit for high-speed IP network testing.

However, this has the side effect that some data variation can occur if the test byte limit is hit in the middle of a multisend cycle. Therefore, the iperf3 client is modified to allow the byte limit condition check to take effect regardless of the multisend cycle state.

It is worth highlighting the timing interaction between the iperf3 client and the server. The “total test time” on the client stops as soon as the last data ACK is received, but the server timer runs until the client indicates that it is done sending data (via a control message).

Therefore, throughput measurements are performed based on client result data to ensure that the throughput results are not affected by delays in control message processing. The resulting throughput calculation formula is given in (4.3).

$$Throughput_{bps} = \frac{TotalDataSent_{bits}}{ClientTotalTestTime_{sec}} \quad (4.3)$$

It is important to note that this calculation defines throughput as a data rate metric. Therefore, the term should be understood as such throughout the dissertation.

4.4.8 A last note on packet loss

As mentioned in section 4.4.1.3, the test design decision to take test measurements on application data transferred via TCP has the consequence that a packet loss metric is not directly measurable. Fortunately, iperf3 already tracks the TCP metadata (STRUCT TCP_INFO) of each data connection on the client side, which includes the number of TCP retransmits that occurs over the lifetime of the data connection, as discussed in section 4.4.1.3.

The TCP retransmits value includes all retransmits that occur during the lifetime of the connection, i.e. including retransmits during handshaking and connection closure (TCP SYN, Finish (FIN), ACK). Therefore, it is not possible to reliably use the TCP retransmit data as a direct indication of data packet loss.

However, the retransmit data can potentially be used as a type of sanity or gatekeeping check when analysing throughput results to determine if excessive packet loss occurred during the test or not. In other words, if a throughput test yields much lower than expected results, the retransmit data can be used as a clarifying factor in the result analysis.

4.5 Concluding remarks

This chapter presented a test bench setup that is suitable for the performance evaluation of the IP crypto. The test setup architecture and configuration (section 4.3) is discussed in light of the test setup requirements that are presented in section 4.2. Similarly, the development of suitable latency and throughput tests is presented in section 4.4 in light of the test measurement goals and design decisions presented in section 4.4.1.

This test setup, along with the IP crypto implementation discussed in section 3.5 of chapter 3, constitutes everything that is required to start evaluating the performance impact of the IP crypto.

5 IP cryptographic function performance impact test results

This chapter presents the evaluation of the performance impact of the IP crypto. Section 5.1 documents the test execution flow for both the throughput- and latency tests and defines the sequence of events in each test run. The baseline test results (tests without the IP crypto in the test bench) are presented section 5.2, along with the verification of these test results. The IP crypto test results are presented in section 5.3, which includes the verification and validation of the IP crypto test results. Notably, section 5.3.4 evaluates the performance impact of the IP crypto by comparing the IP crypto test results to the baseline test results. Lastly, some concluding remarks are presented in section 5.4.

5.1 Test execution overview

The performance impact of the IP crypto function is evaluated for different HF channel bandwidths and channel conditions. Each of these instances is evaluated with the IP crypto disabled and enabled, by performing a latency- and a throughput test, the details of which are discussed in sections 4.4.5 and 4.4.7 respectively. In other words, a total of four (4) tests are performed for each HF channel bandwidth, HF channel model, and HF channel SNR condition pairing.

The tested pairings consist of the following parameters: a 12 kHz and 18 kHz bandwidth, an AWGN and a “poor” HF channel model, and three sets of SNR conditions, namely “Good”, “Average”, and “Bad”, for a total of twelve (12) test condition pairings. These pairings are discussed in greater detail as part of section 5.2.1.

The general test execution flow is shown in Figure 5.1. The flow consists of three (3) primary phases: test bench configuration and initialisation, execution of test run(s), and result parsing.

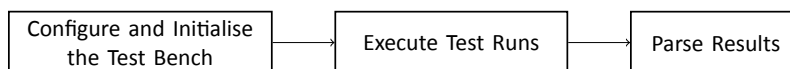


Figure 5.1: General test execution flow

The “test bench configuration and initialisation” phase is expanded in Figure 5.2. It consists of configuring the test bench for the required HF bandwidth and channel conditions, starting the STANAG 5066 stacks and IP Clients by placing the RC10 units in the online operational state, and configuring the IP XFRM subsystem in the router 1 and 2 NetNSs for the required IP crypto state, viz. enabled or disabled.

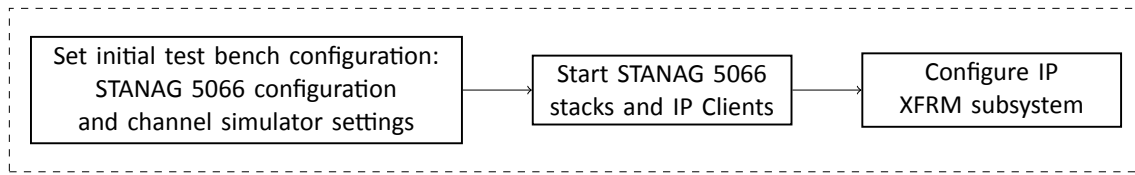
Configure and Initialise the Test Bench

Figure 5.2: Test initialisation sequence - expanded

The latency- and throughput test run flows are given in Figures 5.3 and 5.4, respectively. As should be evident from the figures, the general test run flow is identical for both test types, with the exception of the additional single packet ping test in the latency test run (in accordance with the exception in section 3.3.4.7).

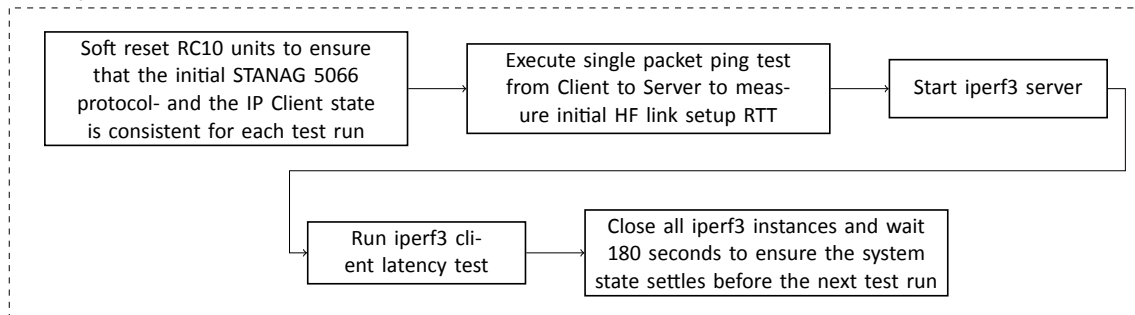
Latency test run flow

Figure 5.3: Latency test run flow

The additional ping test is performed using the standard Linux ping utility. Its purpose is to measure the initial IP packet RTT time when the transfer is subject to the initial HF link setup time. It is necessary to perform this separately from the iperf3 latency test since the initial HF link setup is hidden from the iperf3 test measurement due to the setup of the iperf3 control connection prior to the test start (see section 4.4.3).

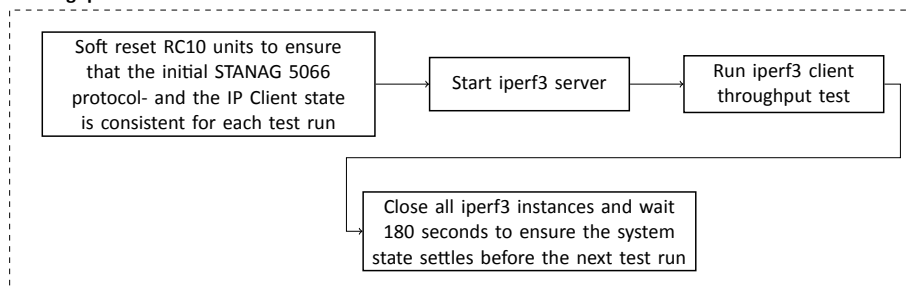
Throughput test run flow

Figure 5.4: Throughput test run flow

The soft reset of the RC10 units is done by placing them in the offline operational mode and returning them to the online operational mode. This amounts to a full reset of the STANAG 5066 stack, along with all RM10 settings controlled by the RC10, which is essentially the equivalent of resetting the RM10's as well.

The 180 second wait time to ensure that the system state settles (i.e. all RM10 transmissions are completed, and the STANAG 5066 stack returns to an idle, unlinked state on both RC10s) is determined by observation during exploratory testing. However, if some oddity does occur during test automation execution which renders the 180 second wait insufficient, the soft reset of the RC10 units serves as an additional safeguard to ensure system state consistency at the start of each test.

The version information of the ping utility and the base iperf3 tool to which the modifications outlined in section 4.4.4 to 4.4.7 were made is given in Table 5.1 for reference.

Table 5.1: Version information of testing tools

Tool	Version
ping utility	version info output: ping from iputils 20211215
iperf3 (modified)	base iperf3: iperf 3.14 (cJSON 1.7.15); changes added to the master branch on commit ID 208a19a6fade42dec22b29277171200101971226

5.2 Test verification and baseline results

5.2.1 Expected Over The Air (OTA) throughput rates

As a starting point for verifying the baseline TCP performance tests, the expected instantaneous OTA data rates for different bandwidths, SNR conditions, and channel models are given in Table 5.2. Each of the OTA data rates represents an HF waveform designed with physical layer characteristics (e.g., coding, interleaving, and modulation scheme) such that it can transfer data at the specified rate. This is discussed in greater detail throughout section 2.3.

Table 5.2: Expected OTA Throughput rates
(derived from MIL-STD-188-110C Annex D [40]; colloquial SNR condition terms added)

Bandwidth (kHz)	Channel SNR condition	Average SNR (dB) level for BER $\leq 10^{-5}$		OTA rate (bps)
		AWGN channel model	“Poor” channel model	
12	“Good”	21	31	38400
	“Average”	6	11	6400
	“Bad”	0	5	1200
18	“Good”	21	31	57600
	“Average”	6	11	9600
	“Bad”	0	5	2400

The given OTA rates are expected in the sense that the corresponding MIL-STD-188-110C Annex D [40] compliant waveform should perform at a BER level less than or equal to 10^{-5} for the given channel model and SNR level. Therefore, these rates serve as the upper bound for the expected throughput measurements. Stated differently: the measured throughput rates should always be lower than the OTA data rates in a statistical sense, i.e. after enough measurements are taken to reach convergence, due to protocol (e.g. overhead, flow control) and latency effects.

Note that, within the HF industry, an HF waveform is often referred to by the data rate it provides, for example, the “38 400 bps waveform”. This nomenclature is used interchangeably with the term “OTA rate” throughout this discussion.

It should also be noted that MIL-STD-188-110C Annex D [40] provides a much larger range of bandwidths, SNR conditions, and expected waveform OTA rates, but it is deemed unnecessary to evaluate the IP crypto for each of these combinations. Instead, the entries in Table 5.2 represent a sensible subset of these combinations for which the IP crypto is evaluated. This subset is chosen to ensure that the evaluation is performed at different WBHF bandwidths and under a sufficiently wide variety of SNR conditions for each HF channel model.

The chosen WBHF bandwidths, viz. 12 kHz and 18 kHz, is deemed a suitable representation of the bandwidth range in MIL-STD-188-110C Annex D [40], viz. 3 kHz to 24 kHz in 3 kHz increments, for the following reasons: the lower end of the bandwidth range, 3 kHz, which is actually NBHF (i.e., not part of the research study goals - section 1.3), and 6 kHz, the first WBHF waveform, do not provide data rates that are deemed suitable for IP over HF, with the exception of the 6 kHz channel under “Good” SNR conditions; and at the top end of the range, the 21 kHz channel provides rates identical to the 18 kHz channel under most SNR conditions.

The qualitative terms “Good”, “Average”, and “Bad” that represent the different SNR conditions are subjectively chosen by the author. Their only purpose is to allow easier reference to different test conditions during the discussion of test results and should not be considered to convey anything more than the SNR levels they represent.

5.2.2 Test bench hardware and iperf3 configuration parameters

The RapidM hardware configuration parameters that have a bearing on the baseline TCP performance test results are given in Table 5.3. The RC10 parameters are kept constant for all tests run at a specific bandwidth, but the channel simulator (RS10) parameters are updated for each throughput and latency test run.

Table 5.3: RapidM Hardware Parameters

Device	Bandwidth (kHz)	Parameters
RC10	12	STANAG 5066 DRC algorithm Min and Max limits: 300 bps to 38400 bps Modem bandwidth: 12 kHz Initial DRC Tx and Rx rate (set when RC10 goes online): 2400 bps
	18	STANAG 5066 DRC algorithm Min and Max limits: 600 bps to 57600 bps Modem bandwidth: 18 kHz Initial DRC Tx and Rx rate (set when RC10 goes online): 4800 bps
RS10	All	Channel model: AWGN or ITU-R F.1487 HF Mid-Latitude Disturbed (the “poor” model specified by MIL-STD-188-110C Annex D [40]) Noise bandwidth: Match modem bandwidth Noise center frequency: Default for noise bandwidth Target (desired) SNR level: Set for the applicable channel SNR condition (see Table 5.2)

Special note should be taken of the STANAG 5066 DRC algorithm limits, since this places a hard limit on the waveforms (rates) that the RC10/RM10 will attempt to use for data transmission. The maximum is chosen to match the expected OTA rate under “Good” SNR conditions (see Table 5.2) and the minimum is chosen as the lowest OTA rate supported in the bandwidth under test.

The iperf3 parameters for the throughput tests are given in Table 5.4. The client parameters are determined by a trial-and-error tuning process under 12 kHz AWGN conditions. The tuning goal is to ensure that the iperf3 client generates enough data to utilize the HF link at saturation and produce a stable throughput measurement, but does not flood the system with so much data that the TCP data connections start hitting the maximum Retransmission Timeout (RTO) before receiving ACKs for the data sent.

Table 5.4: Throughput Test Parameters

Component	Channel SNR condition	Parameters
iperf3 client	All	General Parameters TCP MSS: 1460 bytes TCP window size: 65535 bytes Data block size: 4096 bytes
	“Good”	Total data to transmit: 2 MiB Number of parallel TCP data connections: 10
	“Average”	Total data to transmit: 500 KiB Number of parallel TCP data connections: 4
	“Bad”	Total data to transmit: 100 KiB Number of parallel TCP data connections: 1
iperf3 server	All	Server receive timeout: 600 seconds

Note that the MiB and KiB units are standard units used to disambiguate the data size, 1 MiB is 1024 KiB, and 1 KiB is 1024 bytes [89].

In this context, *stable* means that enough data is generated such that the measured mean throughput rate will not change significantly if the client had been tuned to generate more data, which would in turn cause the test to run for a longer period of time. This ensures that start-up effects, such as the data connections’ initial TCP window growth, do not skew the throughput measurements.

The flooding scenario can occur, especially at lower data rates, since the maximum STANAG 5066 link master transmission length (127.5 seconds - see section 4.3.4) exceeds the maximum TCP RTO timer in the Linux kernel (hardcoded as 120 seconds [90, include/net/tcp.h, Line 141]).

The iperf3 client parameters are varied for each SNR condition, but are kept identical for both the AWGN- and the HF Mid-Latitude-Disturbed channel model, and the 12 kHz and 18 kHz channel bandwidth. Additional fine tuning for each bandwidth, channel model, and SNR pairing could yield (marginally) better throughput results, but since the goal of the study is not to optimize IP over HF traffic, but rather to compare the throughput performance with- and without the IP crypto included under stable loaded channel conditions (in keeping with the study goals set in section 1.3), the parameters determined by the trial-and-error approach are deemed sufficient.

The iperf3 parameters for the latency tests are given in Table 5.5. Due to the single data block nature of the TCP based latency test (see section 4.4.6), no specific tuning is required for different SNR levels or channel conditions. It is important to note that this tuning ensures that the data block and the resulting TCP datagram always fit into a single IP packet. Therefore, only the start-block timestamp data (see section 4.4.5.2) is used for one-way latency and jitter test measurements.

Table 5.5: Latency Test Parameters

Component	Channel SNR condition	Parameters
iperf3 client	All	TCP MSS: 1000 bytes TCP window size: 100 bytes Data block size: 32 bytes Total number of data blocks to transmit: 60 Set the --TCP-BASED-LATENCY-TEST option
iperf3 server	All	Server receive timeout: 600 seconds

Based on the iperf3 source code [86], and verification via packet captures, the iperf3 client makes use of the TCP header timestamp option, which adds an additional 12 bytes to the TCP header. Therefore, the data block size is chosen as 32 bytes to ensure that the resulting TCP datagram will form a 64 byte IP packet payload, for a total IP packet size of 84 bytes. This is done to match the payload size with what is typically expected in an ICMP echo request (i.e. ping test).

5.2.3 Baseline throughput results

The baseline throughput test results are given in Table 5.6. The baseline classification indicates that the tests were executed without adding the IP crypto to the system. These results are obtained by repeating the throughput test run for each combination of channel conditions five (5) times and performing basic statistical calculations on the resulting data set.

Table 5.6: Throughput Results (*bps*) - baseline (no IP crypto)

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	5 of 5	27295	27329	27365	24.65
	Average	5 of 5	9119	9230	9478	150.47
	Bad	5 of 5	1425	1541	1586	66.39
12 kHz "Poor"	Good	5 of 5	19261	20397	21107	749.92
	Average	3 of 5	5061	5709	6334	636.79
	Bad	3 of 5	1490	1684	1812	170.94
18 kHz AWGN	Good	5 of 5	34135	35519	36709	1032.70
	Average	5 of 5	11780	12269	12955	440.57
	Bad	5 of 5	3067	3271	3344	115.43
18 kHz "Poor"	Good	5 of 5	21615	23612	24946	1264.79
	Average	3 of 5	6062	7213	7836	998.20
	Bad	5 of 5	1754	1956	2211	169.43

Some of the test runs for the “poor” channel model failed in such a way that no throughput measurement was produced. This mostly occurs if the interaction of the TCP/IP stack with the STANAG 5066 IP Client and protocol stack, along with the interaction of the STANAG 5066 DRC algorithm with the “poor” channel model causes the setup of the iperf3 TCP control connection to fail, or if it causes the control socket to break down in such a way that control information cannot be exchanged between the client and the server at the end of the test.

The five (5) run throughput tests were repeated to ensure that these failures were not one-off occurrences. While the number of failures varied in these reruns, failures occurred frequently enough to report as part of the test results.

Occasionally, failures were also observed due to stability issues in the RapidM hardware, such as the RC10 temporarily losing data path communication to the RM10, or the RM10 perpetually failing to lock (sync) correctly on a received waveform. Both of these failure states will result in a failed test run which, similar to the above, will result in no throughput measurement being produced. The hardware failure conditions are cleared when the RC10 units are given a soft reset (section 5.1), thereby ensuring that it only causes the test run in which it occurred to fail. These issues have been reported to RapidM for investigation.

Failed test runs are reflected in the column “Successful Run Count”, but are not included in the statistical analysis. For example, if a test only had three (3) successful runs, the mean throughput rate value is calculated as the sum of the throughput measurement of the three runs divided by three, instead of five (5) as would be the case for five successful runs. As explained in section 5.2.2, each throughput test runs for a sufficiently long enough period of time to produce a stable result, i.e. to reach statistical convergence, which indicates that making this exclusion for tests with failed runs will still lead to valid statistical analysis.

Apart from verifying the throughput results against the expected OTA rates, which is done in section 5.2.5, some result validation is also possible by examining the measured results in reference to one another. This yields the following observations that contribute to verifying the integrity of both the test results and the test bench setup:

- The mean throughput rate always decreases as the SNR conditions deteriorate.
This verifies the impact of the SNR conditions on the throughput results.
- The mean throughput rate for “poor” channel conditions is always lower than for AWGN conditions, under the same SNR conditions, with the exception of the 12 kHz Bad SNR conditions case.
Although higher SNR levels for the “poor” channel condition tests may be expected to compensate for the differences in the AWGN vs. the “poor” channel model, the STANAG 5066 DRC interaction with the “poor” channel model generally yields lower throughput rates - see section 5.2.5.
The 12 kHz, Bad SNR exception is fairly minor, since the AWGN mean lies within one standard deviation of the “poor” channel mean (when considering the “poor” channel’s standard deviation). However, it warrants further investigation to ensure that it is not caused by a testing setup or configuration error. Section 5.2.6.1 presents this investigation, which shows that it is caused by the behaviour of the STANAG 5066 DRC algorithm under “poor” channel conditions;
- The mean throughput rate is consistently higher for 18 kHz tests than for 12 kHz tests, under

otherwise the same channel conditions.

This verifies that larger bandwidth yields increased throughput rates;

- The SD is always very small under AWGN channel conditions, less than 5% of the mean throughput rate when converted to a percentage.

This indicates that test results are consistent between test runs when no varying effects are present in the system, such as the changing conditions simulated when the “poor” channel model is used. Stated differently, this verifies that the throughput tests are consistent and yield repeatable results;

- The SD is slightly higher under “poor” channel conditions as expected, but still relatively small, with the highest SD being approximately 13.8 % of the mean throughput for the 18 kHz bandwidth, “poor” channel model, Average SNR conditions test case.

This verifies that the “poor” channel model produces more realistic HF conditions than the AWGN model, but the test results are still reasonably consistent between test runs;

5.2.4 Baseline latency results

The baseline latency test results, consisting of the initial HF link setup RTT measurements, one-way latency, one-way jitter, and TCP RTT measurements, are given in Tables 5.7 to 5.10, in sections 5.2.4.1 to 5.2.4.4. As with the throughput results (see section 5.2.3), the latency results are obtained by repeating the latency test run for each channel condition combination five (5) times and performing basic statistical analysis on the resulting data set.

The test run failure that occurred during the 12 kHz, “poor” channel model, Bad SNR conditions test is attributed to the same reasons given for throughput test failure under “poor” channel conditions. Please refer to section 5.2.3 for the full discussion.

Similarly to the throughput tests, the latency test results can be validated to some extent by examining the results for different test conditions in reference to each other. The latency results are also included in the verification discussion against the expected OTA rates in section 5.2.5.

5.2.4.1 Initial HF link setup RTT results

The baseline initial HF link setup RTT results is presented in Table 5.7.

Some observations on these results:

- For the 12 kHz tests:
 - The mean setup time increases as the SNR conditions deteriorate.
This verifies the impact of the SNR conditions on the initial HF link setup time.
 - The mean setup time is fairly consistent, with a very low SD for Good SNR conditions.
This verifies the repeatability of the initial HF link setup measurements.
 - The maximum initial setup time exhibits a notable increase for an AWGN channel with Bad SNR conditions, and for a “poor” channel with Average or Bad SNR conditions.
This verifies the impact of deteriorating HF channel conditions on the initial link setup time.

Table 5.7: Test Bench Verification - Initial HF Link Setup RTT (seconds)

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	5 of 5	17.742	17.831	17.869	0.053
	Average	5 of 5	17.765	17.852	17.923	0.070
	Bad	5 of 5	17.920	19.855	27.000	3.995
12 kHz "Poor"	Good	5 of 5	17.862	17.964	18.061	0.074
	Average	5 of 5	17.952	19.864	27.286	4.149
	Bad	4 of 5	18.081	20.946	27.433	4.426
18 kHz AWGN	Good	5 of 5	19.773	20.078	20.365	0.210
	Average	5 of 5	20.021	20.313	20.505	0.191
	Bad	5 of 5	20.385	20.439	20.509	0.050
18 kHz "Poor"	Good	5 of 5	20.261	20.357	20.426	0.064
	Average	5 of 5	20.193	20.299	20.346	0.062
	Bad	5 of 5	20.361	20.381	20.404	0.020

- For the 18 kHz tests:
 - There is virtually no impact on the setup time for any change in the channel model or SNR condition.
Although this is unexpected, since the same trends seen in the 12 kHz results are expected here, it is not cause for concern, since the other latency test results clearly show the impact of channel model- and SNR changes in the 18 kHz bandwidth. Additionally, one possible explanation could be that the initial waveform used in the 18 kHz tests (i.e. initial STANAG 5066 DRC rate - see Table 5.3 in section 5.2.2) is more robust than the initial waveform used in the 12 kHz tests.

5.2.4.2 One-way latency results

The initial analysis of the one-way latency results, presented in Table 5.8 yields the following:

- For AWGN channel model results:
 - In both the 12 kHz and 18 kHz cases, the mean one-way latency is nearly identical at approximately 4.3 seconds under Good and Average SNR conditions, with a very small SD. The largest SD is 102 ms for the 18 kHz, Good case, which has a slightly higher maximum latency than the other Good and Average SNR cases. This verifies that the latency test is consistent and yields repeatable results.
 - In both the 12 kHz and 18 kHz cases, the mean one-way latency increases under Bad SNR conditions, and there is a significant increase in the maximum one-way latency. Correspondingly, the SD also increases, indicating that the latency to which each data block is subjected becomes increasingly volatile as the conditions of the HF channel deteriorate. This verifies that degrading SNR conditions are correctly reflected in the latency test results.

- For “Poor” channel model results:
 - Even under Good SNR conditions the effects of the “Poor” HF channel are already evident in the higher mean latency measurements, and the much higher maximum latency measurements, compared to the AWGN channel, Good SNR case for the 12 kHz and 18 kHz case, respectively;
 - Generally, the same trend is observed as for the AWGN results: as the SNR conditions deteriorate, the latency measurements become increasingly volatile, as evidenced by the larger maximum latency measurements and higher SD values.
This verifies that the change in the HF channel model and the degrading SNR conditions are correctly reflected in the latency test results;
 - One anomaly is present in both the 12 kHz and 18 kHz case: the Bad SNR test yields a lower mean latency than the Average SNR test, and in the 12 kHz AWGN case the Average SNR case yields a lower mean than the Good SNR case.
As with the 12 kHz “poor” channel anomaly in the throughput results (see section 5.2.3), this warrants further investigation to ensure the anomaly is not caused by a test setup or -configuration error. Section 5.2.6.2 presents this investigation, which shows that the anomaly is caused by the STANAG 5066 DRC algorithm’s behaviour under “poor” channel conditions.

Table 5.8: Test Bench Verification - One-way Latency (seconds)

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	5 of 5	4.139	4.306	4.501	0.066
	Average	5 of 5	4.150	4.244	4.437	0.045
	Bad	5 of 5	4.005	4.799	21.250	2.425
12 kHz “Poor”	Good	5 of 5	3.963	4.670	15.241	1.604
	Average	5 of 5	4.008	13.402	91.773	16.165
	Bad	4 of 5	3.972	10.369	149.219	15.027
18 kHz AWGN	Good	5 of 5	3.934	4.342	5.858	0.102
	Average	5 of 5	4.153	4.301	4.558	0.055
	Bad	5 of 5	5.104	5.277	16.502	0.660
18 kHz “Poor”	Good	5 of 5	4.191	9.082	72.058	9.414
	Average	5 of 5	3.897	13.074	78.656	15.312
	Bad	5 of 5	4.177	11.568	130.149	14.695

To help visualize the increase in volatility of one-way latency measurements when HF channel and SNR conditions deteriorate, the sample distribution of the latency results for the 12 kHz, AWGN channel, Good SNR test case and the sample distribution of the latency results for the 12 kHz, “poor” channel, Bad SNR test case are presented in Figures 5.5 and 5.6 respectively. Note that the total number of samples in these graphs differ, since the 12 kHz “poor” channel case only has 4 successful test runs.

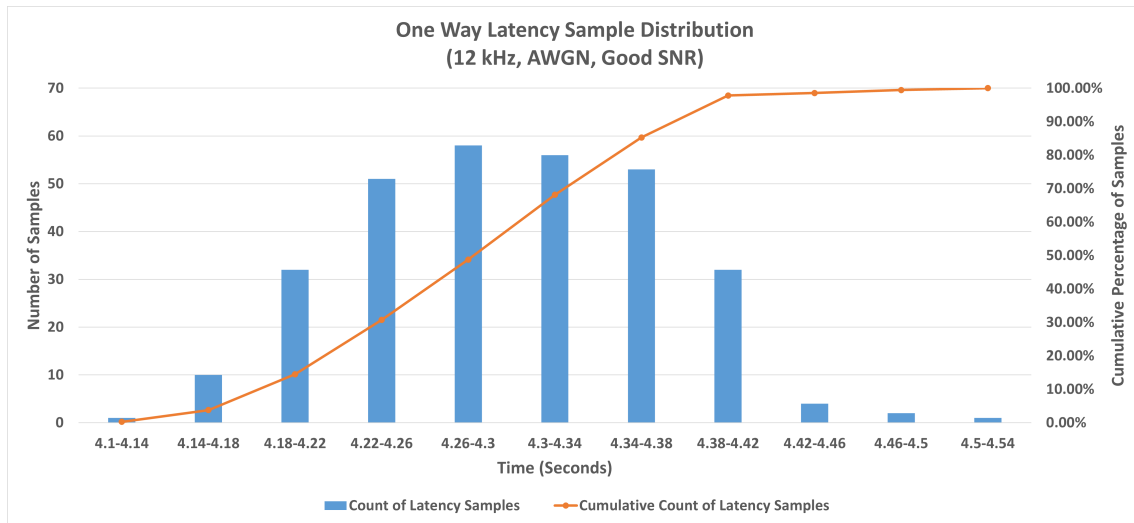


Figure 5.5: One-way latency sample distribution for 12 kHz, AWGN channel, Good SNR test conditions

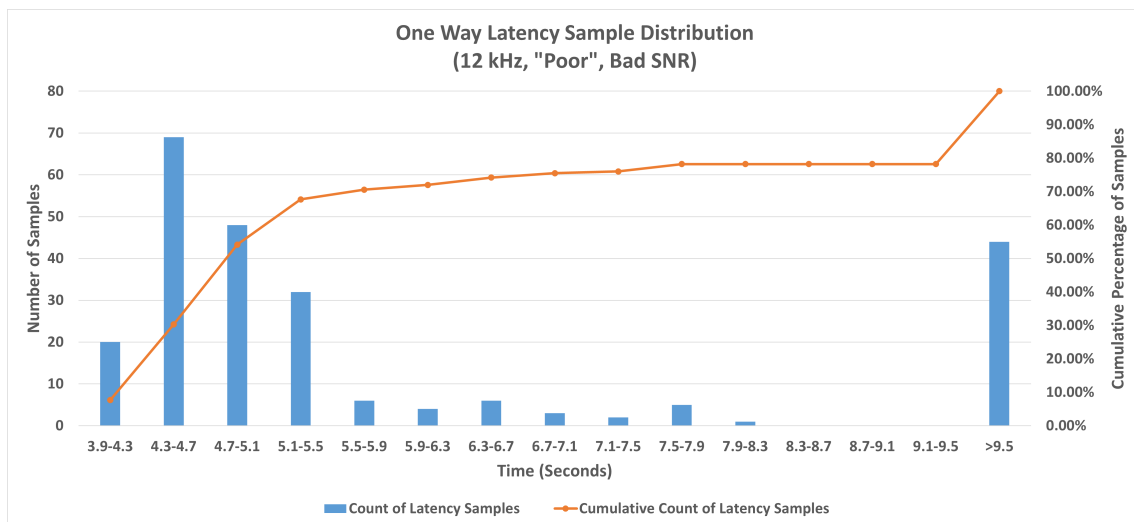


Figure 5.6: One-way latency sample distribution for 12 kHz, "Poor" channel, Bad SNR test conditions

The cumulative sample count graphs presented in Appendix A show similar trends for the other channel bandwidth and SNR conditions as well, viz. when comparing the 12 kHz AWGN channel graphs, Figure A.1, with the 12 kHz "poor" channel graphs, Figure A.2, and when comparing the 18 kHz AWGN channel graphs, Figure A.3, with the 18 kHz "poor" channel graphs, Figure A.4.

It is interesting to note that the bulk of the samples, approximately 70%, for the 12 kHz "Poor" channel, Bad SNR conditions fall within the 3.9 second to 5.5 second range. This indicates that, although the latency measurements are more spread out, and there are several high latency measurements, the bulk of the measurements remain in the same range as for 12 kHz AWGN channel, Good SNR conditions.

Furthermore, this indicates that the "poor" channel model introduces some effects that can cause severe delays at times, which introduces some skewness into the data. This skewness, i.e. increas-

ing difference between the mean and the median (see Table A.1 in Appendix A for specific median values), can also be inferred from the mean and SD measurements reported in Table 5.8.

5.2.4.3 One-way jitter results

The one-way jitter results, presented in Table 5.9, exhibit the same trends as the one-way latency results (section 5.2.4.2). This includes the increase in mean and maximum jitter as the HF channel and SNR conditions deteriorate, as well as the anomaly where the “Poor” channel, Bad SNR condition test results yield a lower mean than the “Poor” channel, Average SNR condition test results.

Since these trends are consistent in the jitter results, it verifies that the jitter results are calculated correctly from the one-way latency measurements.

Table 5.9: Test Bench Verification - One-way Jitter (seconds)

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	5 of 5	0.000025	0.075	0.246	0.059
	Average	5 of 5	0.000047	0.045	0.166	0.037
	Bad	5 of 5	0.000004	1.158	17.121	2.915
12 kHz “Poor”	Good	5 of 5	0.000023	0.617	10.960	2.238
	Average	5 of 5	0.000274	12.880	87.559	17.962
	Bad	4 of 5	0.000092	7.985	144.609	17.549
18 kHz AWGN	Good	5 of 5	0.000024	0.067	1.600	0.135
	Average	5 of 5	0.000006	0.063	0.322	0.050
	Bad	5 of 5	0.000010	0.144	11.246	0.923
18 kHz “Poor”	Good	5 of 5	0.000044	7.195	65.785	10.881
	Average	5 of 5	0.000261	10.537	72.841	15.112
	Bad	5 of 5	0.000077	8.533	125.726	15.205

5.2.4.4 TCP RTT results

The last baseline results related to latency to consider are the TCP RTT measurements, given in Table 5.10. When considering the following analysis, it is important to remember that the TCP RTT measurements are taken from the kernel-tracked TCP metadata for the iperf3 data connection in the latency test (see section 4.4.5). The implication is that the TCP RTT measurements cannot be compared on a one-to-one basis with the one-way latency measurements, since they are measured using different time sources and methodologies. This point is better explained in section 5.2.4.5.

It is also worth mentioning that since the HF channel is utilized in a half-duplex fashion and HF channel conditions vary with time (viz. the “poor” channel model), forward and reverse transmissions on the HF channel are not expected to always undergo the exact same delay, especially in “poor” HF channel conditions. Therefore, a precise mathematical check cannot, for example, be used to verify that the TCP RTT measurement is twice the one-way latency measurement for the same test conditions. However, some comparison of the mean values is

still sensible. For example, it is still expected that the mean TCP RTT will be roughly two times the one-way latency mean, at least under AWGN conditions.

Table 5.10: Test Bench Verification - TCP RTT (seconds)

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	5 of 5	8.589	8.624	8.790	0.025
	Average	5 of 5	8.554	10.048	26.987	2.776
	Bad	5 of 5	8.339	9.271	12.196	0.649
12 kHz "Poor"	Good	5 of 5	8.739	9.293	12.114	0.655
	Average	5 of 5	8.878	27.060	45.601	9.034
	Bad	4 of 5	9.173	20.347	79.037	12.188
18 kHz AWGN	Good	5 of 5	8.665	8.732	9.057	0.076
	Average	5 of 5	8.590	9.857	18.281	2.188
	Bad	5 of 5	10.287	10.579	15.825	0.767
18 kHz "Poor"	Good	5 of 5	8.968	17.208	43.349	4.620
	Average	5 of 5	10.500	26.809	79.444	10.077
	Bad	5 of 5	11.717	26.232	49.690	9.291

The following observations are worth noting:

- When comparing the mean TCP RTT with the mean one-way latency results for the same test conditions, the mean RTT is always approximately equal to double the mean one-way latency. Considering that the iperf3 latency test only sends a single data block at a time, which translates to a single TCP packet for the chosen iperf3 client parameters, this is the expected relation between the means of the two metrics;
- "Poor" channel conditions generally yield higher maximum and higher SD values, for both the 12 kHz and 18 kHz channels. Similar to the one-way latency results, this verifies that the effect of the "poor" channel model is correctly reflected in the TCP RTT results;
- Generally, Average and Bad SNR conditions yield higher mean and maximum TCP RTT values than Good SNR conditions. This verifies that the effect of deteriorating SNR conditions is reflected correctly in the RTT results;
- Two seeming anomalies exist in the result set:
 - The mean RTT under Bad SNR conditions is slightly lower than for Average SNR conditions in the 12 kHz AWGN channel and 18 kHz "poor" channel case, and notably lower in the 12 kHz "poor" channel case.
The same investigation which showed that the STANAG 5066 DRC algorithm behaviour is the cause of the anomalies in the one-way latency results also shows that the DRC algorithm behaviour is the cause for this anomaly. This investigation is presented in section 5.2.6.2;
 - The maximum RTT value is often much lower than the maximum one-way latency value (Table 5.8) for the same channel conditions.
This anomaly is addressed in detail in section 5.2.4.5 by first showing that the maximum measurement values can be considered statistical outliers and, consequently,

that the mean and SD values can still be considered reliable test results. Second, the manner in which one-way latency and TCP RTT measurements differ with regard to TCP packet retries is highlighted as one of the factors responsible for the differences in maximum values, indicating that the maximum values are valid, but cannot actually be compared verbatim between the one-way latency and TCP RTT result sets.

5.2.4.5 Clarification of the seeming TCP RTT results anomaly

The maximum measured one-way latency and TCP RTT values are considered as outliers, since they are always larger than the mean by at least two SDs, and under some “poor” channel conditions, such as the 18 kHz “poor” channel Bad SNR test case, by more than three SDs. The sample analysis data presented in Appendix A is also helpful to consider here, as explained in the following paragraphs.

When comparing each of the cumulative sample count one-way latency diagrams, viz. Figures A.1 to A.4, to the mean and SD values presented in Table 5.8 on a test-by-test basis, at least 85% of the sample values lie within one SD of the mean, with the exception of the 12 kHz AWGN channel Good SNR case which has slightly less than 80% of sample values within one SD of the mean. This exception is not cause for concern, since the difference between the mean and maximum values is not even 200 milliseconds in this test case, which is almost inconsequential when considering that the latency statistics (minimum, mean, maximum) are all in the order of 4 seconds in this test case.

Note that the cumulative sample graph scale, necessitated by the magnitude of some outlier samples, makes it difficult to read specific values from the graphs in some cases. Therefore, the supplementary percentile ($P_{\%}$) data presented in Table A.2 were used to provide specific values for the above comparison.

Similarly, when comparing each of the cumulative sample count TCP RTT diagrams, viz. Figures A.5 to A.8, to the mean and SD values presented in Table 5.10 - with the help of the supplementary percentile data in Table A.4, at least 90% of the sample values lie within one SD of the mean, with the exception of the 12 kHz “poor” channel Bad SNR case, and the 18 kHz “poor” channel Bad SNR case, which have between 80% and 85%, and between 75% and 80% of sample values within one SD of the mean, respectively.

These exceptions are not cause for concern, since the STANAG 5066 DRC investigation in section 5.2.6.2 can also be used to verify that the DRC algorithm is responsible, at least in large, for the wider sample spread in “poor” channel Bad SNR conditions. Specifically, this statement is supported by comparing the investigation results for the 18 kHz “poor” channel Bad SNR test case in Table 5.17 with the baseline results in Table 5.10. This comparison shows that the DRC disabled case yields a much smaller SD (295 milliseconds) than the baseline test case (9.291 seconds), as well as a smaller difference between the maximum and mean values, approximately 2.5 seconds in the DRC disabled case, versus approximately 23.5 seconds in the baseline result case.

The implication of these observations is that the mean and SD values are still reliable, although they are somewhat influenced by the outlier values. It is also important to note that these outliers are not excluded in the rest of the analysis in this research study, i.e., they are always included in the mean and SD calculations, since they represent factors that have an effect on the performance of IP data transmission over the HF channel.

Although not all mechanisms in the system that are responsible for these outliers are definitively identified at this stage, it is worth reiterating that the effects of the STANAG 5066 DRC algorithm are at least partly responsible, as previously illustrated with the 18 kHz “poor” channel Bad SNR exception in section 5.2.4.4. Furthermore, the “poor” HF channel model has an impact on the severity of the outliers, as has been discussed in section 5.2.4.2.

However, regardless of the root cause, the magnitude of these outliers is indicative that packet retransmits due to a packet drop or timeout occur during at least the extreme (maximum) outlier measurements. Note that the TCP retry count data (section 4.4.8) is not reported or referenced here, since there is no way to relate the retry count to a specific one-way latency or TCP RTT measurement sample.

These retries can occur at either the TCP level or the STANAG 5066 DTS level. If TCP retransmits do occur, it is important to note that this will have different effects on the one-way latency and TCP RTT measurements as explained next, even though the measurements are taken as part of the same iperf3 latency test for the same test traffic.

Recall from section 4.4.5.1 that the TCP RTT is measured using existing mechanisms in the Linux kernel, which is handled according to RFC 6298 [87] according to the kernel source code [90, `net/ipv4/tcp_input.c`], and that the one-way latency is measured using timestamps placed in the actual data transferred from the iperf3 client to the server. Furthermore, as verified by packet capture (see section 5.2.2), the iperf3 client makes use of the TCP header timestamp option.

Therefore, when a TCP retransmit occurs, the kernel will adjust the RTT measurement for the retransmit (see section 3 in RFC 6298 [87]) such that the RTT is only taken for the packet that is successfully ACKed, which effectively discards the time from when the data was first transmitted until the data was sent again in the retried TCP packet. On the other hand, the one-way latency measurement does not account for TCP retransmits, in the sense that the measurement is not adjusted if a retransmit does occur, since it calculates the time difference using the timestamps present in the actual data payload (section 4.4.5.2).

Returning to the original observed anomaly, i.e. that the TCP RTT maximum value is often lower than the corresponding one-way latency maximum value, this shows that the observed anomaly is not caused by a problem in the test setup or its configuration, but that both the one-way latency and TCP RTT measurements are reliable in their own right. This also serves as an explanation for the point made at the beginning of section 5.2.4.4 that the TCP RTT measurements cannot be compared on a one-to-one basis with the one-way latency measurements.

However, some comparison between the mean values of the one-way latency and TCP RTT measurements is still sensible, as can be concluded from the statistical analysis of the outlier data. Specifically, since the outlier data at most only make up about 15% of one-way latency samples and about 10% of the TCP RTT samples, the retry handling difference between the one-way latency and TCP RTT measurements only has a notable influence on 15% of the samples at most.

5.2.5 Baseline result verification

Before performing a mathematical comparison between the measured throughput rates (Table 5.6) and the expected OTA rates (Table 5.2), an adjustment needs to be made to the expected OTA rates as discussed below.

When comparing the mean throughput rates in Table 5.6 with the expected instantaneous OTA rates in Table 5.2, it is evident that the measured rates for Average and Bad SNR conditions for the AWGN channel model tests and some “poor” channel model tests are higher than the expected OTA rates. This occurs due to waveform changes made by the STANAG 5066 DRC algorithm.

The STANAG 5066 DRC algorithm is constantly active during the lifetime of a STANAG 5066 link, and makes data rate (i.e. waveform selection) changes as channel condition measurements are updated, along with other metrics such as the successful delivery of STANAG 5066 data PDUs (STANAG 5066 Annex C [5]). For AWGN channel conditions, the algorithm logic is observed to result in a near-constant use of the next faster waveform, but in “poor” channel conditions this results in more frequent waveform changes.

The next higher rates are given in Table 5.11 for reference. No next higher rate is given for Good SNR conditions, since the DRC algorithm maximum rate is limited to the expected Good OTA rate as stated in Table 5.3.

Table 5.11: Expected OTA Throughput rates - next higher rate

Bandwidth (kHz)	Channel SNR condition	Expected OTA rate (bps)	Next higher OTA rate (bps)
12	Good	38400	N/A
	Average	6400	12800
	Bad	1200	2400
18	Good	57600	N/A
	Average	9600	19200
	Bad	2400	4800

The measured throughput rates are therefore verified against the next higher rate for the Average and Bad SNR conditions. Table 5.12 contains a comparison of the mean throughput results with the expected OTA rates. The percentage difference is expressed for the measured throughput values adjusted for TCP/IP overhead and STANAG 5066 overhead. This adjustment is necessary to make a sensible comparison with the OTA rate (i.e. the physical layer data rate).

The TCP/IP overhead equates to roughly 3.59 % per 1500 byte packet as shown in (5.1), and the STANAG 5066 overhead can be approximated to 20 % per IP packet [75].

$$\begin{aligned}
 TCP/IP \text{ overhead} &= TCP \text{ Timestamps} + TCP \text{ header} + IP \text{ header} \\
 &= 52 \text{ bytes} \\
 &\approx 3.59\% \text{ per packet}
 \end{aligned} \tag{5.1}$$

Table 5.12: Baseline Result Verification - Throughput rate (*bps*) comparison

Channel BW & Model	SNR	Expected OTA Rate	Measured Mean ($\bar{x}$)	Overhead Adj. Mean ($\bar{x}$)	Percentage Difference (%)
12 kHz AWGN	Good	38400	27329	33973	-11.53
	Average	12800	9230	11474	-10.35
	Bad	2400	1541	1916	-20.15
12 kHz "Poor"	Good	38400	20397	25355	-33.97
	Average	12800	5709	7097	-44.55
	Bad	2400	1684	2093	-12.76
18 kHz AWGN	Good	57600	35519	44153	-23.34
	Average	19200	12269	15252	-20.56
	Bad	4800	3271	4066	-15.28
18 kHz "Poor"	Good	57600	23612	29352	-49.04
	Average	19200	7213	8967	-53.30
	Bad	4800	1956	2432	-49.32

As expected, the measured throughput rate, after adjustment for protocol overhead, is lower than the OTA rate in all test cases. The difference can be attributed to protocol dynamics (e.g., both TCP and STANAG 5066 employ flow control mechanisms), the fact that the HF channel is utilized in a half-duplex fashion, and variations in HF channel conditions (more prevalent for the "Poor" HF channel model).

All of these effects are included in the latency metrics. Stated differently, given the difference between the measured and the expected throughput rates, one would expect to see longer one-way latency and TCP RTT times, and this is what the measured latency results show. Therefore, the difference in throughput rates also verifies the latency results.

To illustrate the above observation, the percentage difference calculated from the adjusted throughput results, the mean one-way latency, one-way latency SD, and mean TCP RTT results are presented together in Table 5.13 for ease of comparison.

When considering the 12 kHz, AWGN channel test cases, which are also the channel conditions used to tune the throughput test parameters (see section 5.2.2), the throughput difference is fairly small under Good and Average SNR conditions, but increases under Bad SNR conditions. This increase is accompanied by deteriorating latency characteristics, as evidenced by a slight increase in the mean one-way latency and a notable increase in the one-way latency SD.

This serves as an illustration of how the throughput results contribute to verifying the latency results. Similar analysis can be done for the other result sets as well.

Table 5.13: Baseline Result Verification - Throughput rate difference and Latency comparison

Channel BW & Model	SNR	Thpt. Percentage Difference (%)	One-way Latency (sec)		RTT (sec)
			Mean ($\bar{x}$)	SD	Mean ($\bar{x}$)
12 kHz AWGN	Good	-11.53	4.306	0.066	8.624
	Average	-10.35	4.244	0.045	10.048
	Bad	-20.15	4.799	2.425	9.271
12 kHz "Poor"	Good	-33.97	4.670	1.604	9.293
	Average	-44.55	13.402	16.165	27.060
	Bad	-12.76	10.369	15.027	20.347
18 kHz AWGN	Good	-23.34	4.342	0.102	8.732
	Average	-20.56	4.301	0.055	9.857
	Bad	-15.28	5.277	0.660	10.579
18 kHz "Poor"	Good	-49.04	9.082	9.414	17.208
	Average	-53.30	13.074	15.312	26.809
	Bad	-49.32	11.568	14.695	26.232

Some other notable observations for the baseline result verification:

- The fact that the 12 kHz "Poor" channel, Bad SNR throughput test yields better results than the results of the 12 kHz AWGN channel, Bad SNR throughput test is addressed in the initial analysis of the throughput results in section 5.2.3;
- The results of the "poor" channel model tests generally yield larger differences from the expected OTA rate than their AWGN counterparts, a trend which has been highlighted as part of the throughput and latency result analysis. It is worth noting here that, in addition to the deteriorating one-way latency results seen for these tests, there is also a notable increase in the mean TCP RTT which further corroborates the lower throughput results and vice versa;
- The 18 kHz results generally yield larger differences from the OTA rate than the 12 kHz results. This can mostly be ascribed to the fact that the throughput tests were not re-tuned for the 18 kHz tests, especially for the Good and Average SNR conditions. As discussed in section 5.2.2, this was deemed unnecessary, since the goal of this study is not to optimize IP traffic transmission over HF.

5.2.6 Impact of the STANAG 5066 DRC algorithm

Throughout the test result verification discussion, two notable anomalies are identified: the 12 kHz Bad SNR condition throughput test yields better results for the “poor” channel model than the AWGN channel model, and the latency result anomaly where testing with Bad SNR conditions yields lower one-way latency and TCP RTT measurements than when testing with Average SNR conditions under “poor” channel conditions.

When observing the waveform changes due to the STANAG 5066 DRC algorithm (visible on the RC10 and RM10 front panel displays), the DRC appears to change waveforms more frequently when testing with the “poor” channel model versus the AWGN channel model. This is an indication that the DRC algorithm is responsible for the measured anomalies. This is verified as discussed below.

Two tests, one for each anomaly, are performed with the DRC algorithm disabled to determine if the anomalies can indeed be attributed to the DRC algorithm’s interaction with the different HF channel models. The throughput test is discussed in section 5.2.6.1, and the latency test in section 5.2.6.2.

5.2.6.1 The 12 kHz “poor” channel throughput anomaly test

The 12 kHz “poor” channel throughput anomaly is investigated by repeating the throughput test runs for the 12 kHz channel under Bad SNR conditions. For these test runs, the STANAG 5066 DRC algorithm is disabled, and the waveform selection is fixed to the 12 kHz 1200 bps waveform, which corresponds to the expected OTA rate for a 12 kHz channel under Bad SNR conditions (see Table 5.2).

The results are presented in Table 5.14. With the DRC disabled, the mean throughput rate is lower for “poor” HF channel conditions than for AWGN channel conditions. This is the expected relation between the two measurements, and verifies that the anomaly seen in the baseline throughput results (Table 5.6) is caused by the DRC algorithm’s behaviour for the different channel conditions.

Table 5.14: Throughput (*bps*) investigation - DRC disabled, 1200 bps waveform

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz, AWGN	Bad	5 of 5	772	796	814	18.11
12 kHz, “Poor”	Bad	4 of 5	764	778	801	15.98

Identifying and/or quantifying the specific factors within the DRC algorithm logic that is responsible for this behaviour is left for future research work. Note that the end-goal of this study is to evaluate the performance of the IP crypto in a representative (realistic enough) IP over WBHF system, which does not include performing a deep-dive investigation into the interaction of IP traffic with a WBHF system (section 1.3 and 4.2).

As explained in the latency anomaly investigation in section 5.2.6.2, the effects of the DRC algorithm are included in the analysis throughout the rest of this dissertation, for the same reason, i.e., when considering the purpose of this study as highlighted above.

5.2.6.2 The Bad SNR versus Average SNR latency anomaly test

Similarly to the 12 kHz Bad SNR throughput anomaly, the latency anomaly is investigated by repeating some of the latency test runs with the DRC disabled and fixed waveform rates selected. Specifically, the 18 kHz “poor” channel latency tests are repeated for Average and Bad SNR conditions, with the waveform set for the corresponding expected OTA rate.

The latency tests are not repeated for the 12 kHz channel since the anomaly’s characteristic is the same for both the 12 kHz and 18 kHz channels. In other words, the anomalous behaviour is not affected by the channel bandwidth.

The one-way latency and jitter results are presented in Tables 5.15 and 5.16 respectively. When comparing the latency and jitter results, it is clear that Bad SNR conditions yield worse (longer) values than Average SNR conditions for all statistics (minimum, mean, maximum). This is the expected relation, which verifies that the anomaly in the one-way latency and jitter baseline test results (Tables 5.8 and 5.9) is caused by the behaviour of the DRC algorithm for the different SNR conditions in a “poor” HF channel.

Table 5.15: Latency investigation: One-way Latency (seconds)
DRC disabled, 18 kHz “Poor” channel

Waveform (OTA Rate)	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
9600 bps	Average	5 of 5	4.326	4.820	5.282	0.114
2400 bps	Bad	5 of 5	4.899	5.179	8.143	0.299

It is interesting to note that the mean and maximum values of the latency and jitter measurements, as well as the SD, for both Average and Bad SNR conditions are considerably lower than the corresponding baseline test results. This indicates that the usage of the DRC algorithm causes the latency and jitter characteristics of the system to deteriorate for small packet transmissions.

However, the DRC capability is invaluable for a real-world STANAG 5066 deployment, since its usage adds to the system’s capability to automatically adjust for changing HF conditions and consequently yields better throughput metrics (see the next higher OTA rate segment of the baseline result verification discussion in section 5.2.5). Therefore, the analysis in the rest of this dissertation is done with DRC enabled (also see the comment related to study goals in the throughput anomaly DRC test discussion, section 5.2.6.1).

Table 5.16: Latency investigation: One-way Jitter (seconds)
DRC disabled, 18 kHz “Poor” channel

Waveform (OTA Rate)	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
9600 bps	Average	5 of 5	0.000021	0.079	0.518	0.121
2400 bps	Bad	5 of 5	0.000056	0.207	3.203	0.344

The TCP RTT results for the latency test reruns are given in Table 5.17. As with the one-way latency results, it is clear that Bad SNR conditions yield worse statistics (minimum, mean, maximum) than Average SNR conditions. Consequently, this verifies that the behaviour of the DRC algorithm under the different SNR conditions in a “poor” HF channel is also the cause of the anomaly in the baseline TCP RTT results (Table 5.10).

Table 5.17: Latency investigation: TCP RTT (seconds)
DRC disabled, 18 kHz “Poor” channel

Waveform (OTA Rate)	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
9600 bps	Average	5 of 5	8.446	9.600	10.788	0.266
2400 bps	Bad	5 of 5	10.295	10.464	12.957	0.295

As with the throughput anomaly investigation in section 5.2.6.1, further investigation into the specific factors within the DRC algorithm logic that are responsible for this anomaly is left for future research work.

5.3 IP crypto results and analysis

5.3.1 Important notes of the IP crypto functionality

Before proceeding with the presentation and discussion of the IP crypto test results, it is worth highlighting some of the observations made during IP crypto verification testing (section 4.3.5). This includes the following:

- Both the Client and Server NetNS only generate and receive plaintext packets, i.e. normal TCP/IP packets.
- All outgoing traffic from the Router Ethernet (ETH) interfaces is ESP tunnel mode packets, using the Router interface addressing for source- and destination IP addresses.
- No form of handshaking or crypto negotiation takes place at the start of a test, i.e. the very first packet generated by the client NetNS is already encrypted in an ESP packet before being transmitted from the router 1 to router 2 NetNS, viz. via the STANAG 5066 IP Client (section 4.3.5.2).
- The total overhead added by the ESP protocol ranges from 54 bytes for large packets to 56 bytes for smaller packets (section 4.3.5.2).

5.3.2 IP crypto throughput rate results

5.3.2.1 Result presentation and verification

The throughput results with the IP crypto enabled in the test bench is presented in Table 5.18. These results are obtained by repeating the throughput test runs with the same RapidM hardware parameters and the same iperf3 configuration parameters used for the baseline results (see section 5.2.2), with one notable difference: the iperf 3 client TCP MSS value is limited to 1406 bytes instead of 1460 bytes, to allow for the 54 byte ESP protocol overhead for the full data

payload packets (section 4.3.5.2). This configuration difference ensures that the generated ESP tunnel mode IP packets do not exceed the 1500 byte Ethernet MTU limit, which is in accordance with the IP crypto ICMP traffic design choice as stated in section 3.3.4.7.

As with the baseline throughput results, some of the test runs failed under “poor” HF channel conditions. These failures are reflected in the test run count in Table 5.18. Please refer to section 5.2.3 for a full discussion of the reasons for these failures.

Table 5.18: Throughput Results (*bps*) - with IP crypto

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	5 of 5	24630	24978	25582	444.43
	Average	5 of 5	8105	8420	8566	187.17
	Bad	5 of 5	1311	1395	1501	75.77
12 kHz “Poor”	Good	5 of 5	19156	19401	19579	168.43
	Average	4 of 5	5276	5535	5755	231.15
	Bad	4 of 5	1347	1508	1734	188.02
18 kHz AWGN	Good	5 of 5	32478	33703	35477	1242.73
	Average	5 of 5	11189	11477	12017	317.79
	Bad	5 of 5	2970	3163	3271	120.99
18 kHz “Poor”	Good	5 of 5	18399	20409	23770	2009.54
	Average	3 of 5	4857	6076	7432	1293.12
	Bad	4 of 5	1894	2096	2376	211.25

Inspection of the results indicates that the IP crypto throughput results exhibit the same characteristics and trends as the baseline throughput results (section 5.2.3), including the anomaly addressed in section 5.2.6.1, which verifies the integrity of the results.

5.3.2.2 Validation

When comparing the IP crypto throughput rate results with the baseline results, it is evident that including the IP crypto in the system yields lower mean throughput rate results. This validates that the throughput performance impact of the IP crypto is measurable with these tests.

A detailed comparison and analysis of the baseline throughput rate results versus the IP crypto results is presented in section 5.3.4.1.

5.3.3 IP crypto latency results

The IP crypto latency results, including the initial HF link setup RTT, one-way latency, one-way jitter, and TCP RTT measurements, are presented in Tables 5.19 to 5.22, in section 5.3.3.1 to 5.3.3.3.

The RapidM hardware parameters and the iperf3 configuration parameters used for these latency tests are identical to the parameters used for the baseline latency tests. No adjustment is needed to account for the ESP protocol overhead, since the latency test payload size (iperf3

block size) of 32 bytes result in IP packets well below the 1500 byte MTU limit.

As in previous cases, test run failures are reflected in the successful run count in each result table. The “poor” HF channel failures are attributed to the same reasons expounded in section 5.2.3. The single test run failure under 12 kHz, AWGN, Good SNR conditions, and the two test run failures under 12 kHz, AWGN Bad SNR conditions are attributed to RapidM hardware failures, which are also discussed in section 5.2.3.

5.3.3.1 Initial HF link setup RTT results

5.3.3.1.1 Result presentation and verification

Regarding the initial HF link setup RTT measurements, nearly all of the observations made for the baseline results (section 5.2.4.1) are applicable to the IP crypto results. In particular, the maximum link setup RTT with the IP crypto is lower for the 12 kHz AWGN channel, Bad SNR, and the 12 kHz “poor” channel, Average SNR case when compared with the baseline results. However, since the mean link setup RTT is in the same range, the higher baseline maximum values can be attributed to a retransmit occurring at the STANAG 5066 DTS layer.

This retransmission claim is not verified definitively, but given that the magnitude of the difference (approximately 8.5 seconds) is close to the mean TCP RTT measurement, it is sensible to attribute this difference to a retransmission of the single ICMP echo request packet. Furthermore, since the initial link setup is measured using a single packet ping test, no mechanism from the PC side would trigger a retry, meaning that the retry would have to be caused by the ARQ functionality of the STANAG 5066 DTS layer.

Table 5.19: Initial HF Link Setup RTT (seconds) results - with IP crypto

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	4 of 5	17.989	18.052	18.107	0.049
	Average	5 of 5	18.209	18.298	18.423	0.089
	Bad	3 of 5	18.438	18.495	18.555	0.059
12 kHz “Poor”	Good	5 of 5	17.942	18.102	18.225	0.140
	Average	5 of 5	17.964	18.303	18.505	0.250
	Bad	4 of 5	18.445	21.336	29.828	5.661
18 kHz AWGN	Good	5 of 5	19.891	20.046	20.127	0.100
	Average	5 of 5	20.164	20.360	20.570	0.157
	Bad	5 of 5	20.188	20.387	20.501	0.120
18 kHz “Poor”	Good	5 of 5	20.093	20.187	20.282	0.074
	Average	3 of 5	20.161	20.263	20.466	0.176
	Bad	5 of 5	20.177	20.403	20.700	0.218

5.3.3.1.2 Validation

In general, the initial link setup RTT result statistics (minimum, mean, maximum) are slightly higher with the IP crypto included, with two exceptions, viz. the mean values for 12 kHz AWGN

channel, Bad SNR and 12 kHz “poor” channel, Average SNR cases are lower with the IP crypto included. However, these exceptions are small enough, approximately 1.5 seconds in both cases - roughly a 8% difference, that they can be attributed to delay effects caused by the Bad SNR conditions and the “poor” HF channel conditions, respectively (see the discussion at the end of section 5.2.4.2).

This validates that the initial HF link setup RTT tests reveal some of the latency impact caused by the IP crypto, albeit minor.

5.3.3.2 One-way latency and jitter results

5.3.3.2.1 Result presentation and verification

The one-way latency (Table 5.20) and one-way jitter (Table 5.21) results exhibit the same trends as the baseline results (section 5.2.3), including the same latency anomaly addressed in section 5.2.6.2, with some differences in the maximum statistics trends.

Table 5.20: One-way Latency (seconds) results - with IP crypto

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	4 of 5	4.132	4.301	4.459	0.068
	Average	5 of 5	4.271	4.679	16.187	0.669
	Bad	3 of 5	4.205	5.654	16.205	2.892
12 kHz “Poor”	Good	5 of 5	4.002	4.799	24.116	2.185
	Average	5 of 5	4.112	11.295	74.963	13.750
	Bad	4 of 5	4.240	6.749	52.125	5.558
18 kHz AWGN	Good	5 of 5	4.248	4.347	5.934	0.136
	Average	5 of 5	4.428	4.553	4.796	0.059
	Bad	5 of 5	5.124	5.205	5.918	0.089
18 kHz “Poor”	Good	5 of 5	3.632	8.023	78.249	8.661
	Average	3 of 5	4.095	11.312	77.087	14.946
	Bad	5 of 5	4.248	10.022	96.327	12.244

The differences in the maximum statistics do not follow a definitive trend. In some cases, such as the 12 kHz AWGN channel Average SNR case, the maximum one-way latency with the IP crypto is larger than the baseline maximum (by approximately 12 seconds), but in other cases, such as the 12 kHz “poor” channel Average SNR case, the baseline maximum is larger than the IP crypto case (by approximately 16 seconds).

As stated in section 5.2.4.2, the “poor” HF channel model introduces some effects that can cause severe delays at times. Therefore, the maximum value differences between the baseline and IP crypto results are attributed to differences in length of these severe delays. A similar inference can be made for the AWGN channel, Bad SNR cases, based on observation of the mean and SD values, i.e. larger mean and SD indicate that there are some temporary (varying) delay effects as well.

Table 5.21: One-way Jitter (seconds) results - with IP crypto

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	4 of 5	0.000019	0.073	0.204	0.052
	Average	5 of 5	0.000042	0.136	11.596	0.947
	Bad	3 of 5	0.000061	1.502	11.637	3.129
12 kHz "Poor"	Good	5 of 5	0.000032	0.882	19.839	3.033
	Average	5 of 5	0.000214	9.157	60.762	15.141
	Bad	4 of 5	0.000040	3.800	47.284	6.824
18 kHz AWGN	Good	5 of 5	0.000007	0.069	1.685	0.182
	Average	5 of 5	0.000012	0.060	0.285	0.058
	Bad	5 of 5	0.000004	0.050	0.751	0.086
18 kHz "Poor"	Good	5 of 5	0.000493	5.935	74.008	10.405
	Average	3 of 5	0.000423	9.440	72.928	16.468
	Bad	5 of 5	0.000022	7.443	80.560	13.263

Note that cumulative sample count graphs and supporting sample analysis data is presented in section B.2 of Appendix B, similarly to that done for the baseline one-way latency results in section A.2 of Appendix A. This analysis did not yield any information that makes a material difference to the result presentation and verification given in this section or the performance impact analysis done in section 5.3.4.3, but is included for the sake of completeness.

5.3.3.2.2 Validation

The mean values of the baseline- and IP crypto results have the same order of magnitude. However, in most cases the IP crypto tests yield slightly lower mean one-way latency and jitter results than the baseline tests, which is unexpected. This comparison is done in greater detail in section 5.3.4.3, and additional investigative tests are presented in section 5.3.5.2 which show that it is the result of behaviour within the STANAG 5066 stack.

5.3.3.3 TCP RTT results

5.3.3.3.1 Result presentation and verification

Similarly to the other latency test results, the TCP RTT measurements with the IP crypto included exhibit the same characteristics outlined for the baseline TCP RTT results in section 5.2.4.4. This includes the same anomaly where the mean TCP RTT under "poor" channel, Bad SNR conditions is lower than for "poor" channel, Average SNR conditions, which is addressed in section 5.2.6.2.

Note that cumulative sample count graphs and supporting sample analysis data is presented in section B.3 of Appendix B, similarly to that done for the baseline TCP RTT results in section A.3 of Appendix A. This analysis did not yield any information that makes a material difference to the result presentation and verification given in this section or the performance impact analysis done in section 5.3.4.4, but is included for the sake of completeness.

Table 5.22: TCP RTT (seconds) results - with IP crypto

Channel BW & Model	SNR	Successful Run Count	Minimum	Mean ($\bar{x}$)	Maximum	SD (σ)
12 kHz AWGN	Good	4 of 5	8.591	8.628	8.790	0.026
	Average	5 of 5	9.115	11.191	32.375	3.847
	Bad	3 of 5	9.264	11.740	20.340	2.244
12 kHz "Poor"	Good	5 of 5	8.662	9.395	12.313	0.727
	Average	5 of 5	9.275	23.928	66.757	10.021
	Bad	4 of 5	10.304	14.554	63.212	6.776
18 kHz AWGN	Good	5 of 5	8.659	8.730	9.046	0.074
	Average	5 of 5	9.091	10.539	27.420	2.703
	Bad	5 of 5	10.387	10.461	11.551	0.180
18 kHz "Poor"	Good	5 of 5	8.888	15.590	24.608	3.397
	Average	3 of 5	9.894	31.130	82.219	13.466
	Bad	5 of 5	10.360	23.471	88.395	13.408

5.3.3.3.2 Validation

When comparing the IP crypto TCP RTT results with the baseline results, the TCP RTT result statistics are nearly identical for the AWGN channel, Good SNR conditions, and slightly higher under AWGN channel, Average- and Bad SNR conditions, which reflects the expected impact of the IP crypto.

However, the results of the "poor" channel show mainly that the RTT measurements are lower with the IP crypto included in the system as compared to the baseline results (Table 5.10). This matches the trend seen in the one-way latency and -jitter results, and the same investigation in section 5.3.5.2 shows that the behaviour of the STANAG 5066 stack for small data payloads causes this anomaly in the one-way latency and TCP RTT results.

A more detailed comparison of the IP crypto RTT measurements with the baseline RTT results is made in section 5.3.4.4.

5.3.4 Performance impact analysis of the IP cryptographic function

5.3.4.1 Throughput rate analysis

The impact of the IP crypto on the throughput metrics of the HF system is best shown by comparing the mean throughput rate statistics. This comparison is presented in Table 5.23, which presents the mean throughput rate measurements with- and without the IP crypto side-by-side for convenience sake, and indicates the percentage difference of the IP crypto results relative to the baseline (without IP crypto) results.

Table 5.23: Throughput rate (*bps*) comparison: With/Without IP crypto

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto		Percentage
		Mean ($\bar{x}$)	SD (σ)	Mean ($\bar{x}$)	SD (σ)	Diff. (%)
12 kHz AWGN	Good	27329	24.65	24978	444.43	−8.60
	Average	9230	150.47	8420	187.17	−8.78
	Bad	1541	66.39	1395	75.77	−9.46
12 kHz “Poor”	Good	20397	749.92	19401	168.43	−4.88
	Average	5709	636.79	5535	231.15	−3.05
	Bad	1684	170.94	1508	188.02	−10.42
18 kHz AWGN	Good	35519	1032.70	33703	1242.73	−5.11
	Average	12269	440.57	11477	317.79	−6.46
	Bad	3271	115.43	3163	120.99	−3.29
18 kHz “Poor”	Good	23612	1264.79	20409	2009.54	−13.56
	Average	7213	998.20	6076	1293.12	−15.76
	Bad	1956	169.43	2096	211.25	+7.12

As stated in section 5.3.1, the IP crypto adds 54 bytes of overhead for each large IP packet. This amounts to approximately 3.73 % of overhead, as shown in (5.2), for a resulting 1500 byte packet. Note that the slightly larger overhead observed for small packets (56 bytes) is not applicable to the throughput rate overhead, since the throughput rate measurement only considers application data, which is always large packets (see section 4.4.7 and 5.2.2).

$$\begin{aligned}
 IP \text{ crypto overhead} &= \text{new IP header} + ESP \text{ packet overhead} \\
 &= 54 \text{ bytes} \\
 &\approx 3.73\% \text{ per packet}
 \end{aligned} \tag{5.2}$$

The measured impact of the IP crypto on the throughput results for most test cases is higher than this lower bound of 3.73 %. This is expected since the added overhead increases the total amount of data that has to be transferred, implying that more transmissions are required at the HF level. This in turn implies that the high latency nature of the HF system has an increased effect on the total time required to transfer the total application data payload, which further reduces the mean throughput rate.

The two cases which yield throughput differences less than 3.73 %, and the one case that yields better throughput results with the IP crypto enabled than without (18 kHz, “poor” channel, Bad SNR conditions), can be attributed to the effects of the STANAG 5066 DRC algorithm. This is shown with an additional investigative test in section 5.3.5.1.

These results indicate that the IP crypto can reduce the throughput rate of the HF system anywhere between 3 % and 15 % depending on the current channel conditions. Although this may impact the types of IP services that can successfully be deployed over the HF system, it by no means renders the system unusable for IP services as a whole.

As a last remark, the difference of the impact of the IP crypto on the throughput rate between the AWGN channel and the “poor” channel is greater for the 18 kHz bandwidth tests than the

12 kHz bandwidth tests. Investigating this is left for future research work.

A hypothesis for this future work is that it could simply be a side-effect of the fact that the throughput tests were not re-tuned for the 18 kHz tests (see section 5.2.2).

5.3.4.2 Initial HF link setup RTT analysis

As stated in section 5.3.3.1.2, the initial link setup RTT statistics (minimum, mean, maximum) are generally slightly higher with the IP crypto included. However, since these differences are quite small - mostly in the range of a few hundreds or tens of milliseconds, it is not possible to conclude anything more from this comparison other than stating that the presence of the IP crypto results in a minor increase of the initial HF link setup time.

5.3.4.3 One-way latency and jitter analysis

The one-way latency and jitter IP crypto results are compared with the baseline results in Tables 5.24 and 5.25. Care has to be taken when analysing these results, since most of the differences between the baseline and IP crypto results are a result of effects in the STANAG 5066 stack when transmitting small data payloads, as shown with additional tests in section 5.3.5.2.

Stated differently, the IP crypto causes differences in the latency metrics not because it itself adds notable additional latency, but because the larger payloads used in the IP crypto latency tests cause the STANAG 5066 stack to behave differently. The 56 bytes of crypto overhead (small packet overhead, section 4.3.5.2) lead to a total IP packet size of 140 bytes, which is approximately 67% larger than the 84 byte packet size used in the baseline tests (see section 5.2.2). Note that this packet size dynamic is not present in the throughput rate tests, since the tests always used the maximum packet size of 1500 bytes (see section 5.2.2 and 5.3.2).

Table 5.24: One-way Latency (seconds) comparison: With/Without IP crypto

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto		Percentage
		Mean ($\bar{x}$)	SD (σ)	Mean ($\bar{x}$)	SD (σ)	Diff. (%)
12 kHz AWGN	Good	4.306	0.066	4.301	0.068	-0.12
	Average	4.244	0.045	4.679	0.669	+10.25
	Bad	4.799	2.425	5.654	2.892	+17.80
12 kHz "Poor"	Good	4.670	1.604	4.799	2.185	+2.76
	Average	13.402	16.165	11.295	13.750	-15.72
	Bad	10.369	15.027	6.749	5.558	-34.92
18 kHz AWGN	Good	4.342	0.102	4.347	0.136	+0.12
	Average	4.301	0.055	4.553	0.059	+5.84
	Bad	5.277	0.660	5.205	0.089	-1.37
18 kHz "Poor"	Good	9.082	9.414	8.023	8.661	-11.67
	Average	13.074	15.312	11.312	14.946	-13.47
	Bad	11.568	14.695	10.022	12.244	-13.36

Table 5.25: One-way Jitter (seconds) comparison: With/Without IP crypto

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto		Percentage
		Mean ($\bar{x}$)	SD (σ)	Mean ($\bar{x}$)	SD (σ)	Diff. (%)
12 kHz AWGN	Good	0.075	0.059	0.073	0.052	-2.73
	Average	0.045	0.037	0.136	0.947	+203.42
	Bad	1.158	2.915	1.502	3.129	+29.65
12 kHz "Poor"	Good	0.617	2.238	0.882	3.033	+42.94
	Average	12.880	17.962	9.157	15.141	-28.91
	Bad	7.985	17.549	3.800	6.824	-52.41
18 kHz AWGN	Good	0.067	0.135	0.069	0.182	+2.77
	Average	0.063	0.050	0.060	0.058	-4.26
	Bad	0.144	0.923	0.050	0.086	-65.05
18 kHz "Poor"	Good	7.195	10.881	5.935	10.405	-17.51
	Average	10.537	15.112	9.440	16.468	-10.41
	Bad	8.533	15.205	7.443	13.263	-12.77

The results in (section 5.3.5.2) also indicate that the difference in one-way latency and jitter becomes much smaller for larger data payloads. This amounts to differences in the order of tens (10) of milliseconds for one-way latency and a few hundred (100) milliseconds for one-way jitter. Therefore, the impact of the IP crypto on the one-way latency and jitter characteristics is interpreted to range from zero, to the maximum increase in one-way latency and jitter as seen from the respective comparisons of the IP crypto test results with the baseline test results.

The largest one-way latency increase (percentage-wise) is seen for the 12 kHz, AWGN channel, Bad SNR conditions case, at 17.80 % (855 ms). The largest one-way jitter increase is interpreted based on magnitude, since this is deemed more useful than considering the largest percentage-wise difference (203.42%), which only amounts to a 100 ms increase. The largest jitter increase is therefore considered to be the 12 kHz, AWGN channel, Bad SNR conditions case as well, with an increase of approximately 350 ms.

When considering the order of magnitude of the impact of the IP crypto on the HF system's latency characteristics, the following is concluded: the one-way latency impact of the IP crypto may have an effect on which IP services can be supported over the HF system, but this is unlikely given the already high latency of the system without the IP crypto.

However, IP services which predominantly have a small packet traffic profile (e.g. short chat messages exchanged via IP over HF) may be impacted more adversely by the addition of the IP crypto. That being said, this impact may very well be mitigated by the inclusion of a IP traffic PEP or a service-specific PEP, which will likely be included in the system to improve the service quality regardless of the IP crypto's presence, as discussed in section 3.2.2 in chapter 3.

The degree of mitigation achieved by the PEP for a specific IP service will ultimately depend on the design and function of the specific PEP used with the service. However, when considering the primary goals of all PEPs (section 3.2.4), the author is confident that the inclusion of a PEP will at the very least reduce the number of small packets submitted to the IP crypto and subsequently transmitted OTA.

5.3.4.4 TCP RTT analysis

The last performance impact comparison to consider is that of the TCP RTT results given in Table 5.26. This is discussed separately from the one-way latency results, since the measurement dynamic differs for these metrics (see section 4.4.5 and section 5.2.4.5).

Table 5.26: TCP RTT (seconds) comparison: With/Without IP crypto

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto		Percentage
		Mean ($\bar{x}$)	SD (σ)	Mean ($\bar{x}$)	SD (σ)	Diff. (%)
12 kHz AWGN	Good	8.624	0.025	8.628	0.026	+0.04
	Average	10.048	2.776	11.191	3.847	+11.37
	Bad	9.271	0.649	11.740	2.244	+26.64
12 kHz "Poor"	Good	9.293	0.655	9.395	0.727	+1.09
	Average	27.060	9.034	23.928	10.021	-11.57
	Bad	20.347	12.188	14.554	6.776	-28.47
18 kHz AWGN	Good	8.732	0.076	8.730	0.074	-0.02
	Average	9.857	2.188	10.539	2.703	+6.93
	Bad	10.579	0.767	10.461	0.180	-1.12
18 kHz "Poor"	Good	17.208	4.620	15.590	3.397	-9.40
	Average	26.809	10.077	31.130	13.466	+16.12
	Bad	26.232	9.291	23.471	13.408	-10.52

As with the one-way latency results, care has to be taken when analysing the TCP RTT results. The additional tests in section 5.3.5.2 also show that the STANAG 5066 stack effects for small data payloads are responsible for most of the differences between the baseline (no IP crypto) and the IP crypto results.

Therefore, in analysing the TCP RTT results, the same approach is taken as for the one-way latency metrics: the lower limit of the IP crypto's impact on the TCP RTT results is taken as zero, i.e. the negative percentage difference cases in Table 5.26 are ignored, since these are caused by the small data payload effects (section 5.3.5.2) and not the IP crypto itself. The upper limit is taken as the worst-case increase measured for a IP crypto test case, i.e. the positive percentage difference cases in Table 5.26.

For the TCP RTT results this is between the 12 kHz AWGN channel, Bad SNR conditions test case with a 26.64 % (2.47 second) increase (worst-case percentage-wise) and the 18 kHz, "poor" channel, Average SNR conditions test case with a 16.12 % (4.32 second) increase (worst-case magnitude wise).

A similar conclusion is made as for the one-way latency and jitter results: the worst-case order of magnitude (seconds) impact caused by the IP crypto to the TCP RTT results is considered small given the already high TCP RTT metrics (tens of seconds) without the IP crypto in the system. Therefore, the degradation in the TCP RTT characteristic due to the IP crypto's effect on the system is unlikely to place additional restrictions on the IP services that can be supported by the HF system.

The same observation regarding IP services with a small payload traffic profile also holds for the TCP RTT characteristics. These services may be affected more adversely by the degradation of the TCP RTT characteristics, but this should be compensated for by including PEPs in the system. Investigating this hypothesis is left for future research work.

As a final remark on the performance impact of the IP crypto, it is worth reiterating that the STANAG 5066 protocol stack is not optimised for IP traffic (section 4.3.3.2). Therefore, it is likely that the performance impact of the IP crypto can be smaller and/or more consistent (especially latency wise) when used with an HF protocol stack purpose built for IP traffic, although this will have to be evaluated once such a protocol stack is available.

5.3.5 Clarification for IP crypto result anomalies

5.3.5.1 STANAG 5066 DRC effects on throughput results

One notable anomaly is present in the IP crypto throughput results: for the 18 kHz, “poor” channel, Bad SNR conditions test case, introducing the IP crypto into the system yields slightly higher throughput results, approximately 7 % more than compared to the baseline results (see Table 5.23 in section 5.3.4.1). Additionally, two smaller anomalies are present where IP crypto results show a smaller difference in throughput than expected given the overhead added by the IP crypto.

An investigative test is performed with the STANAG 5066 DRC algorithm disabled to determine if this is the cause of the anomaly, since the DRC was the primary cause of anomalies in the baseline results (section 5.2.6). The 18 kHz, “poor” channel, Bad SNR conditions test case with and without the IP crypto enabled is rerun with the DRC disabled as it exhibits the largest anomaly.

The results of this rerun, given in Table 5.27, indicate that the IP crypto reduces the throughput by 5.54 % if the DRC is disabled. This is in line with the rest of the IP crypto throughput results discussed in section 5.3.4.1, proving that the anomaly can be attributed to the STANAG 5066 DRC algorithm.

Table 5.27: Throughput (*bps*) anomaly investigation: With/Without IP crypto
DRC disabled, expected OTA rate waveform (2400 *bps*) used

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto		Percentage
		Mean ($\bar{x}$)	StdDev (σ)	Mean ($\bar{x}$)	StdDev (σ)	Diff. (%)
18 kHz, “Poor”	Bad	1664	71.58	1571	19.41	−5.54

As with the baseline result anomalies (section 5.2.6), identifying and/or quantifying the exact factors within the STANAG 5066 DRC algorithm logic that cause this anomaly is left for future research work.

5.3.5.2 STANAG 5066 behaviour for small payloads

The discussion of the latency results with the IP crypto enabled in section 5.3.3 highlights an anomaly where both the one-way latency and jitter, and the TCP RTT tests yield better results (lower values) when compared to the baseline results. This occurs mainly under “poor” HF channel conditions, but the one-way tests also yield better results for some cases of the AWGN channel tests.

Since the STANAG 5066 DRC was the primary cause for anomalies in the baseline results (section 5.2.6), and again for the IP crypto result throughput anomaly (section 5.3.5.1), the first course of action is to disable the DRC and rerun some of the latency tests for which the anomaly occurs with and without the IP crypto enabled. These rerun results for the one-way latency and TCP RTT tests are presented in Tables 5.28 and 5.29. The waveforms used correspond to the expected OTA rates presented in Table 5.2 in section 5.2.1.

Table 5.28: One-way Latency (seconds) investigation: With/Without IP crypto
DRC disabled, expected OTA rate waveforms used

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto	
		Mean ($\bar{x}$)	StdDev (σ)	Mean ($\bar{x}$)	StdDev (σ)
12 kHz, "Poor"	Bad	5.147	1.504	5.624	1.800
18 kHz	Average	4.820	0.114	4.141	0.865
"Poor"	Bad	5.179	0.299	5.418	1.528

Table 5.29: TCP RTT (seconds) investigation: With/Without IP crypto
DRC disabled, expected OTA rate waveforms used

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto	
		Mean ($\bar{x}$)	StdDev (σ)	Mean ($\bar{x}$)	StdDev (σ)
12 kHz, "Poor"	Bad	10.636	0.993	11.745	1.314
18 kHz	Average	9.600	0.266	9.084	0.568
"Poor"	Bad	10.464	0.295	10.673	0.430

The rerun results indicate that the DRC effects account for the anomaly in the 12 kHz "poor" channel Bad SNR conditions case, and for the anomaly in the 18 kHz "poor" channel Bad SNR conditions case. However, the 18 kHz "poor" channel Average SNR case still exhibits the anomaly, albeit with a smaller margin (less than one second) than for the original tests (several seconds) with the DRC algorithm enabled.

RapidM's STANAG 5066 expert [91], mentioned that the anomaly could occur due to nuances in the STANAG 5066 DTS layer, and/or the interaction between the DTS and the modem waveform interleaver block size. A full description of the DTS behaviour can be found in Annex C of STANAG 5066 [5], but in short: the DTS will try to send data such that it matches an exact number of modem waveform interleaver blocks. The DTS does this by repeating D_PDUs to fill up empty space in a STANAG 5066 transmission or by adding padding D_PDUs.

The hypothesis is that this repetition logic results in minor differences in STANAG 5066 transmission size, such that the transmission for the small latency payload with the added IP crypto overhead is actually slightly shorter than the transmission without the overhead. Confirming this at the STANAG 5066 level can be quite an arduous task. Therefore, a slightly different approach is taken.

Another investigative test is done for the 18 kHz bandwidth Average SNR condition case. The latency tests are repeated for the AWGN channel, which serves as a control test case, and the

“poor” channel, with the payload size (iperf3 block size) increased to 5120 bytes. A packet capture on the Client NetNS VETH is used to verify that the TCP packets encompassing the entire 5120 byte block is sent out on the wire at virtually the same time.

The one-way latency, one-way jitter, and TCP RTT results of this test is presented in Tables 5.30, 5.31, and 5.32.

Table 5.30: One-way Latency (seconds) investigation: With/Without IP crypto
5K payload, DRC disabled, expected OTA rate waveforms used

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto	
		Mean ($\bar{x}$)	StdDev (σ)	Mean ($\bar{x}$)	StdDev (σ)
18 kHz, AWGN	Average	5.537	0.085	5.595	0.105
18 kHz, “Poor”	Average	7.338	5.364	7.356	5.167

Table 5.31: One-way Jitter (seconds) investigation: With/Without IP crypto
5K payload, DRC disabled, expected OTA rate waveforms used

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto	
		Mean ($\bar{x}$)	StdDev (σ)	Mean ($\bar{x}$)	StdDev (σ)
18 kHz, AWGN	Average	0.084	0.071	0.124	0.095
18 kHz, “Poor”	Average	3.235	6.313	2.907	6.316

Table 5.32: TCP RTT (seconds) investigation: With/Without IP crypto
5K payload, DRC disabled, expected OTA rate waveforms used

Channel BW & Model	SNR	Without IP Crypto		With IP Crypto	
		Mean ($\bar{x}$)	StdDev (σ)	Mean ($\bar{x}$)	StdDev (σ)
18 kHz, AWGN	Average	13.308	0.650	13.216	0.578
18 kHz, “Poor”	Average	15.546	2.488	15.621	2.445

With the larger payload, there is virtually no difference in the measurement statistics of the one way latency, or TCP RTT results with- and without the IP crypto enabled, in both the AWGN channel and “poor” channel case. There is a minor difference in the one-way jitter results for the “poor” channel tests, with the IP crypto enabled case yielding a slightly lower mean than without the IP crypto. However, this difference simply indicates that the one-way transfer time is slightly more consistent with the IP crypto enabled.

Although this is not definitive proof that the cause for the anomaly, with the DRC disabled, can be attributed to the STANAG 5066 DTS logic, or its interaction with the modem interleaver block size, it does show that the anomaly is a result of effects caused by the STANAG 5066 stack and/or other factors within the HF system.

5.4 Concluding remarks

This chapter discussed the performance evaluation of the IP crypto by first discussing the details of both the throughput and latency test execution flow (section 5.1), followed by presenting the baseline throughput- and latency test results (section 5.2), viz. results measured without the IP crypto included in the test bench, and verifying these results via analysis, comparison with expected OTA throughput rates, and investigation into anomalies caused by the STANAG 5066 DRC algorithm. A similar undertaking was performed for the test results with the IP crypto included in the test bench (section 5.3), and the performance impact of the IP crypto was analysed by comparing these results with the baseline results (section 5.3.4).

This addresses the performance impact evaluation research study goal (section 1.3) and forms the basis for the research study conclusion presented in chapter 6.

6 Conclusion

This chapter presents the conclusion of the research study. A short overview of the research study is presented in section 6.1, including the study goals in section 6.1.2 and the research methodology followed in section 6.1.3. The completion of the study goals, along with the findings of the performance impact tests of the IP crypto function, is presented in section 6.2. The research contributions are given in section 6.3 and potential future research work is identified in section 6.4. Lastly, section 6.5 offers some closing remarks.

6.1 Overview of the research study

6.1.1 Motivation for an IP cryptographic function in the HF domain

Provisioning IP services over HF radio networks has only been achieved with limited success. In part this is due to the inherent challenges present in HF radio communication, notably the high latency, low bandwidth nature of an HF channel. However, shortcomings in some HF industry standards also contribute to the problem. For example, the STANAG 5066 protocol [5], transmits IP data packets by simply encapsulating each packet as-is within its own PDUs.

In recent years the goal to realize a workable IP based system over military HF networks has received renewed interest both within the research domain [2]–[4] and within the military HF industry [7]. In combination with this, emerging WBHF technology, such as the STANAG 5070 “Interoperable Wideband Link Management” draft standard [6], is heralded as a key enabler for IP services in the HF domain, as it begins to address the low bandwidth constraint of an HF channel.

Against this backdrop, it is necessary to consider the network security of such an IP over HF system. A case could be made to use traditional HF security mechanisms, but this would inherently limit the system to operate within pure HF radio networks. Therefore, to leverage the full benefit of IP communications, which, amongst other things, notably includes the possibility of facilitating the interoperation of communication systems in the military environment (section 2.4.1.1), it is pertinent to consider IP security (cryptography) mechanisms.

Industry standards, such as Annex E of STANAG 5070 [6], also mandate the use of an IP cryptographic function, or IP crypto for short, within a standard compliant system. However, as shown in section 2.4.1.4, very little information about network security is published in public domain IP over HF research work, which presents an opportunity to explore the development of a suitable IP cryptographic function for the HF domain.

6.1.2 Research study goals

As stated in section 1.3, this research study explores the development of an IP crypto function for the HF domain by achieving the following:

1. Presenting an IP crypto function design that is both industry standards compliant and considers the challenges presented by the nature of an HF communications channel;

2. Creating a representative implementation of this IP crypto function such that the evaluation goal (stated next) can be achieved; and
3. Evaluating the performance impact of this IP crypto function in an IP over HF system, viz. operating over WBHF channels.

The exclusions presented in section 1.3 are also restated here for continuity, namely:

1. Neither the IP crypto design or implementation is intended to be exhaustive, such that the resulting IP crypto function is suitable as-is for deployment and/or integration within a military HF system, as further explained in sections 3.3.2 and 3.4.1;
2. Although the test system (test bench) used for the evaluation of the IP crypto function has to be representative of a practically workable IP over HF system, the aim is not to create and/or use a test system that is fully representative of the envisioned system in which the IP crypto function is expected to operate. This is addressed in greater depth in section 4.2.

6.1.3 Research methodology

The research study goals presented in section 6.1.2, are addressed by conducting a literature study of various knowledge domains with bearing on development of a suitable IP crypto for the HF domain, performing a design of both a potential IP over HF system and the IP crypto, implementing a representative version of the IP crypto, creating a test bench representative of the HF environment and modifying test tools for the evaluation of the IP crypto, and finally, evaluating the performance of the IP crypto using the test bench and test tools. Each of these actions is briefly summarized below.

The literature study, presented in chapter 2, provided an overview of the HF medium, explored traditional data transmission over an HF channel followed by a discussion of IP data transmission over HF, reviewed existing HF security mechanisms, and discussed IP security with special focus placed on the IPsec protocol suite.

The information obtained in the literature study and the IP crypto requirements presented in Annex E of HF industry standard STANAG 5070 [6], are combined to form the basis of both a potential IP over HF system and the IP crypto design. Chapter 3 contains the high level architecture of the envisioned IP over HF system, and presented the design of the IP crypto. The last part of chapter 3 discussed the IP crypto implementation, created using the existing IPsec implementation in the Linux kernel, known as the XFRM framework, which is suitable for the purposes of this research study.

The approach taken to evaluate the IP crypto's performance impact, along with the test bench setup and test tool modifications, are discussed in chapter 4. This included the verification of the IP crypto implementation, and the verification and validation of the test tool modifications.

The performance impact of the IP crypto is evaluated in chapter 5 by comparing the throughput rate and latency performance test results with the IP crypto included in the test bench system, to a baseline test result set, i.e., with the IP crypto excluded from the system. This included the verification of both the baseline- and IP crypto results, along with the validation and analysis of the IP crypto results. The findings of these results, along with concluding remarks about the research study goals, are presented in section 6.2.

6.2 Addressing the research study goals and IP crypto performance impact findings

The IP crypto design and implementation goals of this study, repeated in section 6.1.2, are addressed as follows:

1. The IP crypto function design presented in chapter 3 considers the challenges imposed by the nature of an HF communications channel, and is compliant with Annex E of industry standard STANAG 5070 [6];
2. A representative IP crypto implementation, suitable for the evaluation goal of this study, is documented in chapter 3 and verified in section 4.3.5 of chapter 4;

The analysis of the IP crypto's performance impact in an IP over WBHF system, as presented in section 5.3.4 of chapter 5, leads to the following findings, which addresses the last research goal (evaluating the performance impact of the IP crypto function in an IP over HF system, operating over WBHF channels):

- The IP crypto function can reduce the throughput rate of the IP over HF system anywhere from 3% up to 15%, depending on the current HF channel conditions;
- The IP crypto function impacts the latency characteristics of the IP over HF system as follows:
 - The IP crypto has a minor impact on the initial HF link setup RTT time, in the order of a few hundred milliseconds at most;
 - The IP crypto's impact on one-way latency is quantified percentage wise as ranging from 0 % (no measurable impact) to 17.8%, and its impact on one-way jitter is quantified magnitude wise to range from 0 ms (no measurable impact) to 350 ms;
 - The IP crypto's impact on RTT delay, as measured by the TCP RTT tests, is quantified in two ways: percentage wise, ranging from 0% to 26.64% (12 kHz, AWGN channel, Bad SNR conditions) and magnitude wise as ranging from 0 ms to a worst case 4.32 seconds (18 kHz, "poor" channel, Average SNR conditions);
 - The IP crypto's impact on latency metrics in general is dependent on the data payload size, at least in a STANAG 5066 based system as used to perform the performance impact tests, such that its effects are more pronounced for small data payloads;
- The impact of the IP crypto on the type of IP services that can be supported by the IP over HF system will ultimately depend on the performance requirements of each IP service. However, when considering the order of magnitude of both the throughput rate and latency characteristics of the IP over HF system without the IP crypto included, the impact of the IP crypto is more pronounced on the throughput rate than on the latency metrics. So much so that the degradation of the latency metrics will likely not place any additional limitations on the types of IP services that can be supported by the IP over HF system (section 5.3.4.3 and 5.3.4.4).

6.3 Research Contributions

To the best of the author's knowledge, this research study is the first to provide a quantitative investigation into the performance impact of an IP cryptographic function in an IP over HF system. This performance information can be helpful in both the design of an IP over HF system and the design of IP services and/or service-specific PEPs for use in the HF radio environment.

This study can contribute to future HF industry standard creation and/or ratification, since the performance impact results are applicable to a STANAG 5070 Annex E [6] compliant IP crypto. For example, if an exercise is undertaken to update the STANAG 5070 standard with the goal of improving the overall system performance, the results of this study can be used to determine if it is necessary and/or worthwhile to update the Annex E specification in an attempt to reduce the performance impact of the IP crypto.

Furthermore, the functional IP crypto design decisions presented in section 3.3.4 can provide a basis for future IP crypto design and/or optimisation work. Similarly, the modifications made to the iperf3 testing tool (section 4.4.4) can serve to inform future TCP test tool design for HF use and as a potential design input source for general IP over HF system design.

6.4 Potential future work

The following is identified as potential avenues for future research work related to the performance impact of the IP crypto, and the broader IP over HF research domain:

- Evaluating the performance impact of the IP crypto in an IP over WBHF system that includes PEP functionality. As mentioned in section 3.2.4, the inclusion of a PEP should improve the overall performance of IP traffic over HF. The hypothesis is that this improvement will also lead to a reduction of the IP crypto's impact on the system performance. This is mentioned specifically for small packet payloads in the one-way latency and jitter analysis presented in section 5.3.4.3, and the TCP RTT analysis presented in section 5.3.4.4;
- Evaluating the performance impact of the IP crypto in an IP over WBHF system that uses an HF protocol stack that is purpose-built for IP data transmission over HF (section 5.3.4.4). Similar to the hypothesis related to the inclusion of a PEP, the overall performance impact of the IP crypto should be lessened when using a purpose-built HF protocol stack;
- Investigating why the difference of the impact of the IP crypto between the AWGN channel and the "poor" channel is higher for larger HF bandwidths, specifically for 18 kHz channels compared to 12 kHz channels (section 5.3.4.1); The hypothesis is that this is simply a result of test bench tuning, but this is not necessarily the case.

The following work (design aspects) can be explored as part of the general field of IP crypto design for the HF environment (arising from the design work presented in chapter 3, specifically section 3.3.2):

- configuration management, including configuration storage and loading, as well as configuration change management in an operational military HF environment; and
- SA lifetime and key management, together with IV generation and management;

Lastly, the following future work can be explored in the more general field of HF and/or WBHF research:

- Identifying the factors in the STANAG 5066 DRC algorithm that lead to better waveform selection choices under certain “poor” HF channel model conditions (section 5.2.6);
- Identifying the factors in the STANAG 5066 stack that are responsible for small data payloads exhibiting a longer latency characteristics than comparatively larger payloads (fully explained in section 5.3.5.2, along with a hypothesis related to STANAG 5066 DTS layer repetition and padding behaviour);
- Performing a functional characterisation exercise of different components in the STANAG 5066 protocol stack and different elements used in the test bench setup (section 4.1). The results of such an exercise could potentially be used to better inform (clarify) and/or predict test result anomalies that may occur due to factors inherent in the STANAG 5066 based test system.

6.5 Closing remarks

There is a clear case for providing IP services in the military HF radio network environment, as evidenced by public domain research work, and trends within the HF industry. This research study considered the network security of an IP over HF system and presented an HF industry standards compliant IP crypto design. A representative implementation of this design was created, along with a suitable test bench setup and test tools to evaluate the performance impact of this IP crypto on an IP over HF system.

The performance evaluation results of the IP crypto implementation in this test bench system, viz. throughput rate and latency results, showed that the IP crypto has a notable effect on the throughput rate, and a comparatively smaller effect on the latency metrics.

Bibliography

- [1] A. Ahrens and G. Greiner, "A radio protocol for TCP/IP application," in *IEE Colloquium on Frequency Selection and Management Techniques for HF Communications*, Mar. 1999, pp. 11/1–11/6. DOI: 10.1049/ic:19990078.
- [2] E. G. Elsadig, "Design of IP over HF channels under sudan ionospheric transmissions conditions," in *2017 Joint International Conference on Information and Communication Technologies for Education and Training and International Conference on Computing in Arabic (ICCA-TICET)*, Aug. 2017, pp. 1–7. DOI: 10.1109/ICCA-TICET.2017.8095287.
- [3] A. DMello, E. Foo and J. Reid, "Characterizing TCP/IP for High Frequency Communication Systems," in *2018 Military Communications and Information Systems Conference (MilCIS)*, Nov. 2018, pp. 1–7. DOI: 10.1109/MilCIS.2018.8574111.
- [4] A. DMello, E. Foo, J. Reid and D. Jayalath, "On the enhancement of TCP/IP protocol for use in High Frequency Communication Systems," in *2020 Military Communications and Information Systems Conference (MilCIS)*, Nov. 2020, pp. 1–7. DOI: 10.1109/MilCIS49828.2020.9282375.
- [5] *Technical Standards for HF Radio Link Layer and Application Support Protocols*. NATO STANDARD 5066 Edition 4 Draft Version 1.5 Unclassified, Allied Communications Publication, North Atlantic Treaty Organization Standardization Office, 6th Apr. 2021.
- [6] *Interoperable Wideband Link Management*. NATO STANDARD 5070 Edition 0.5 Unclassified, Allied Communications Publication, North Atlantic Treaty Organization Standardization Office, 3rd Jul. 2020.
- [7] NATO, *BRASS IP Enhanced System - System Requirements Specification*. IFB-CO-14887-BRE1TA Contract Part IV Annex A, NATO Unclassified, Communications and Information Agency, North Atlantic Treaty Organization, 12th Dec. 2018.
- [8] K. Seo and S. Kent, "Security Architecture for the Internet Protocol," Internet Engineering Task Force, Request for Comments RFC 4301, Dec. 2005, 101 pp. DOI: 10.17487/RFC4301. [Online]. Available: <https://datatracker.ietf.org/doc/rfc4301> (visited on 15/09/2022).
- [9] V. Jodalén, A. Eggen, B. Solberg and O. Gronnerud, "Military messaging in IP networks using HF links," *IEEE Communications Magazine*, vol. 42, no. 11, pp. 98–104, Nov. 2004, ISSN: 1558-1896. DOI: 10.1109/MCOM.2004.1362552.
- [10] I. Icart, J.-Y. Bernier, R. Elmostadi and R. Koch, "Design and demonstration of a very high data rate multimedia HF communication system," in *12th IET International Conference on Ionospheric Radio Systems and Techniques (IRST 2012)*, May 2012, pp. 1–5. DOI: 10.1049/cp.2012.0380.
- [11] S. R. Kariyawasam, B. M. Heravi, B. Honary, L. Barclay and G. Vongas, "HF-IP: Propagation analysis," in *The Institution of Engineering and Technology 11th International Conference on Ionospheric Radio Systems and Techniques (IRST 2009)*, Apr. 2009, pp. 1–5. DOI: 10.1049/cp.2009.0049.
- [12] K. Lund, E. Skjervold, F. T. Johnsen, T. Hafsøe and A. Eggen, "Robust web services in heterogeneous military networks," *IEEE Communications Magazine*, vol. 48, no. 10, pp. 78–83, Oct. 2010, ISSN: 1558-1896. DOI: 10.1109/MCOM.2010.5594680.

-
- [13] Harris Corporation, RF Communications Division, *Radio Communications In the Digital Age*, 2nd ed. Oct. 2005, vol. 1.
 - [14] *Nomenclature of the Frequency and Wavelength Bands Used in Telecommunications*. Rec. ITU-R V.431-8, International Telecommunications Union, Geneva, Switzerland, Aug. 2015.
 - [15] J. Wang, G. Ding and H. Wang, "HF communications: Past, present, and future," *China Communications*, vol. 15, no. 9, pp. 1–9, Sep. 2018, ISSN: 1673-5447. DOI: 10.1109/CC.2018.8456447.
 - [16] H. L. Silver, *The ARRL Handbook for Radio Communications*, 88th ed. Newington, CT: American Radio Relay League, Jan. 2011, ISBN: 978-0-87259-096-0.
 - [17] *Skywave*, in *Wikipedia*, Nov. 2, 2021. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=Skywave&oldid=1053142672> (visited on 11/12/2021).
 - [18] L. E. Frenzel Jr., *Principles of Electronic Communication Systems*, 4th edition. New York, NY: McGraw-Hill Education, 2016, 944 pp., ISBN: 978-0-07-337385-0.
 - [19] M. Uysal and M. R. Heidarpour, "Cooperative communication techniques for future-generation HF radios," *IEEE Communications Magazine*, vol. 50, no. 10, pp. 56–63, Oct. 2012, ISSN: 1558-1896. DOI: 10.1109/MCOM.2012.6316776.
 - [20] N. M. Maslin, *HF Communications: A Systems Approach*. Plenum Publishers, 1987, 248 pp.
 - [21] W. N. Furman and J. W. Nieto, "Understanding HF channel simulator requirements in order to reduce HF modem performance measurement variability," in *In Proc. 6th Nordic Short-wave Conf. HF, Fårö*, 2001.
 - [22] E. Eleftheriou and D. Falconer, "Adaptive Equalization Techniques for HF Channels," *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 2, pp. 238–247, Feb. 1987, ISSN: 1558-0008. DOI: 10.1109/JSAC.1987.1146531.
 - [23] *Technical Standards for Non-Hopping HF Communications Waveforms*. NATO STANDARD AComP-4539 Edition A Version 3 Unclassified, Allied Communications Publication, North Atlantic Treaty Organization Standardization Office, Mar. 2020.
 - [24] MIL-STD-188-110D, *Interoperability and Performance Standards for Data Modems*. US Dept of Defense, 29th Dec. 2017.
 - [25] *Testing of HF Modems with Bandwidths up to about 12 kHz Using Ionospheric Channel Simulators*. Rec. ITU-R F.1487, International Telecommunications Union, Geneva, Switzerland, 2000. [Online]. Available: https://www.itu.int/dms_pubrec/itu-r/rec/f/R-REC-F.1487-0-200005-I!!PDF-E.pdf (visited on 28/02/2022).
 - [26] *Technical Standards for an Automatic Radio Control System (ARCS) for HF Communication Links*. NATO STANDARD 4538 Edition 1 Unclassified, Allied Communications Publication, North Atlantic Treaty Organization Standardization Office, 24th Feb. 2009.
 - [27] "CCIR / ITU-R Study Groups celebrate 90 years," ITU. (2017), [Online]. Available: <https://www.itu.int:443/en/ITU-R/CCIR90/Pages/default.aspx> (visited on 02/28/2022).
 - [28] C. Watterson, J. Juroshek and W. Bensema, "Experimental Confirmation of an HF Channel Model," *IEEE Transactions on Communication Technology*, vol. 18, no. 6, pp. 792–803, Dec. 1970, ISSN: 2162-2175. DOI: 10.1109/TCOM.1970.1090438.
 - [29] B. A. Forouzan, *Data Communications and Networking, Global Edition*, 5th Edition. New York, NY: McGraw-Hill Education, 2013, Textbook on Data communications and networking, ISBN: 978-981-4577-51-9.
-

-
- [30] M. Schwartz, *Information Transmission, Modulation, and Noise*, 3rd Edition. New York, NY: McGraw-Hill Book Company, 1980, ISBN: 0-07-055782-9.
- [31] K. Andrews, C. Heegard and D. Kozen, "A Theory of Interleavers," 18th Jul. 1997.
- [32] The MathWorks, Inc. "Tail-Biting Convolutional Coding (r2021b)." (2021), [Online]. Available: <https://www.mathworks.com/help/comm/ug/tail-biting-convolutional-coding.html> (visited on 28/11/2021).
- [33] L. E. Frenzel Jr., "Digital Data Transmission", in *Principles of Electronic Communication Systems*, 4th edition, New York, NY: McGraw-Hill Education, 2016, pp. 374–403, ISBN: 978-0-07-337385-0.
- [34] MIL-STD-188-141B, *Interoperability and Performance Standards for Medium and High Frequency Radio Systems*. US Dept of Defense, 1st Mar. 1999.
- [35] MIL-STD-188-141C, *Interoperability and Performance Standards for Medium and High Frequency Radio Systems*. US Dept of Defense, 27th Dec. 2011.
- [36] MIL-STD-188-141D, *Interoperability and Performance Standards for Medium and High Frequency Radio Systems*. US Dept of Defense, 22nd Dec. 2017.
- [37] M. Hervás, P. Bergadà, and R. M. Alsina-Pagès, "Ionospheric Narrowband and Wideband HF Soundings for Communications Purposes: A Review", *Sensors*, vol. 20, no. 9, p. 2486, 9 Jan. 2020. DOI: 10.3390/s20092486. [Online]. Available: <https://www.mdpi.com/1424-8220/20/9/2486> (visited on 11/16/2021).
- [38] W. Furman, E. Koski and J. Nieto, "Design concepts for a wideband HF ale capability," in *12th IET International Conference on Ionospheric Radio Systems and Techniques (IRST 2012)*, May 2012, pp. 1–5. DOI: 10.1049/cp.2012.0379.
- [39] W. N. Furman, E. N. Koski and J. W. Nieto, "Design and System Implications of a Family of Wideband HF Data Waveforms," presented at the IST Symposium RTO-MP-IST-092: Military Communications and Networks, Wroclaw, Poland: NATO Research and Technology Organisation, Sep. 2010. [Online]. Available: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1018.9698&rep=rep1&type=pdf> (visited on 12/03/2022).
- [40] MIL-STD-188-110C with change 1, *Interoperability and Performance Standards for Data Modems*. US Dept of Defense, 3rd Jan. 2012.
- [41] *Profile for HF Radio Data Communications*. NATO STANDARD 5066 Edition 3 Unclassified, Allied Communications Publication, North Atlantic Treaty Organization Standardization Office, Dec. 2010.
- [42] D. Brown, S. Trinder and A. Gillespie, "An analysis of HF for IP sub-networks," in *2000 Eighth International Conference on HF Radio Systems and Techniques*, Jul. 2000, pp. 281–284. DOI: 10.1049/cp:20000188.
- [43] L. Soyer and W. Cottavoz, "IP traffic over STANAG5066," in *2003 Ninth International Conference on HF Radio Systems and Techniques*, Jun. 2003, pp. 234–239. DOI: 10.1049/cp:20030462.
- [44] E. Koski and J. Weston, "Efficient high-fidelity simulation of HF communications systems and networks," in *MILCOM 2015 - 2015 IEEE Military Communications Conference*, Oct. 2015, pp. 1460–1466. DOI: 10.1109/MILCOM.2015.7357650.
- [45] D. G. Boak, *A History of U.S. Communications Security (U)*. Fort George G. Meade, Maryland 20755: National Security Agency, Jul. 1981, vol. 2. [Online]. Available: https://www.nsa.gov/portals/75/documents/news-features/declassified-documents/cryptologic-histories/history_comsec_ii.pdf (visited on 07/06/2022).
-

-
- [46] J. M. Rodriguez Bejarano, A. Yun and B. De La Cuesta, "Security in IP satellite networks: COMSEC and TRANSEC integration aspects," in *2012 6th Advanced Satellite Multimedia Systems Conference (ASMS) and 12th Signal Processing for Space Communications Workshop (SPSC)*, Sep. 2012, pp. 281–288. DOI: 10.1109/ASMS-SPSC.2012.6333089.
- [47] D. Caprioni and P. Muscianesi, "ANDVT MINTERM (digital radiocommunication)," in *MILCOM 91 - Conference Record*, Nov. 1991, 1197–1203 vol.3. DOI: 10.1109/MILCOM.1991.258458.
- [48] "KIV-7." (), [Online]. Available: <https://www.cryptomuseum.com/crypto/usa/kiv7/index.htm> (visited on 10/06/2022).
- [49] MIL-STD-188-141B Notice 1, *Interoperability and Performance Standards for Medium and High Frequency Radio Systems*. US Dept of Defense, 31st Aug. 2001.
- [50] *Specification for the ADVANCED ENCRYPTION STANDARD (AES)*. Federal Information Processing Standards Publications (FIPS) Publication 197, Nov. 2001.
- [51] M. Dworkin, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", National Institute of Standards and Technology, NIST Special Publication (SP) 800-38D, Nov. 28, 2007. DOI: 10.6028/NIST.SP.800-38D. [Online]. Available: <https://csrc.nist.gov/publications/detail/sp/800-38d/final> (visited on 06/25/2022).
- [52] C. Lavaud, R. Gerzaguet, M. Gautier, O. Berder, E. Nogues and S. Molton, "Towards Real Time Interception of Frequency Hopping Signals," in *2020 IEEE Workshop on Signal Processing Systems (SiPS)*, Oct. 2020, pp. 1–6. DOI: 10.1109/SiPS50750.2020.9195246.
- [53] C. Redding and E. Johnson, "Linking protection for HF radio automatic link establishment," in *MILCOM 91 - Conference Record*, Nov. 1991, 1133–1137 vol.3. DOI: 10.1109/MILCOM.1991.258446.
- [54] S. Frankel and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap," Internet Engineering Task Force, Request for Comments RFC 6071, Feb. 2011, 63 pp. DOI: 10.17487/RFC6071. [Online]. Available: <https://datatracker.ietf.org/doc/rfc6071> (visited on 15/09/2022).
- [55] S. Kent, "IP Authentication Header," Internet Engineering Task Force, Request for Comments RFC 4302, Dec. 2005, 34 pp. DOI: 10.17487/RFC4302. [Online]. Available: <https://datatracker.ietf.org/doc/rfc4302> (visited on 14/10/2022).
- [56] S. Kent, "IP Encapsulating Security Payload (ESP)," Internet Engineering Task Force, Request for Comments RFC 4303, Dec. 2005, 44 pp. DOI: 10.17487/RFC4303. [Online]. Available: <https://datatracker.ietf.org/doc/rfc4303> (visited on 09/11/2022).
- [57] C. Kaufman, P. E. Hoffman, Y. Nir, P. Eronen and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)," Internet Engineering Task Force, Request for Comments RFC 7296, Oct. 2014, 142 pp. DOI: 10.17487/RFC7296. [Online]. Available: <https://datatracker.ietf.org/doc/rfc7296> (visited on 17/01/2023).
- [58] G. Gross, B. Weis and a. Ignjatic, "Multicast Extensions to the Security Architecture for the Internet Protocol," Internet Engineering Task Force, Request for Comments RFC 5374, Nov. 2008, 38 pp. DOI: 10.17487/RFC5374. [Online]. Available: <https://datatracker.ietf.org/doc/rfc5374> (visited on 19/11/2023).
- [59] T. Kivinen, A. Huttunen, B. Swander and V. Volpe, "Negotiation of NAT-Traversal in the IKE," Internet Engineering Task Force, Request for Comments RFC 3947, Jan. 2005, 16 pp. DOI: 10.17487/RFC3947. [Online]. Available: <https://datatracker.ietf.org/doc/rfc3947> (visited on 01/12/2023).
-

-
- [60] R. Enghardt, T. Pauly, C. Perkins, K. Rose and C. A. Wood, "A Survey of the Interaction between Security Protocols and Transport Services," Internet Engineering Task Force, Request for Comments RFC 8922, Oct. 2020, 19 pp. DOI: 10.17487/RFC8922. [Online]. Available: <https://datatracker.ietf.org/doc/rfc8922> (visited on 01/12/2023).
- [61] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," Internet Engineering Task Force, Request for Comments RFC 6120, Mar. 2011, 211 pp. DOI: 10.17487/RFC6120. [Online]. Available: <https://datatracker.ietf.org/doc/rfc6120> (visited on 18/11/2023).
- [62] "Simple Mail Transfer Protocol," Internet Engineering Task Force, Request for Comments RFC 821, Aug. 1982, 72 pp. DOI: 10.17487/RFC0821. [Online]. Available: <https://datatracker.ietf.org/doc/rfc821> (visited on 18/11/2023).
- [63] J. C. Klensin, "Simple Mail Transfer Protocol," Internet Engineering Task Force, Request for Comments RFC 2821, 1st Apr. 2001, 79 pp. DOI: 10.17487/RFC2821. [Online]. Available: <https://datatracker.ietf.org/doc/rfc2821> (visited on 18/11/2023).
- [64] J. C. Klensin, "Simple Mail Transfer Protocol," Internet Engineering Task Force, Request for Comments RFC 5321, Oct. 2008, 95 pp. DOI: 10.17487/RFC5321. [Online]. Available: <https://datatracker.ietf.org/doc/rfc5321> (visited on 18/11/2023).
- [65] David Ward, Christophe Gouault and Nicolas Dichtel, *Ip-Xfrm(8) Linux Manual Page*. 20th Dec. 2011. [Online]. Available: <https://man7.org/linux/man-pages/man8/ip-xfrm.8.html> (visited on 21/11/2023).
- [66] A. Steffen and T. Brunner, *strongSwan - IPsec VPN for Linux, Android, FreeBSD, macOS, Windows*. secunet Security Networks AG, 2023. [Online]. Available: <https://www.strongswan.org/> (visited on 11/21/2023).
- [67] J. Gilmore and H. Daniel, *Libreswan*. The Libreswan Project, 2023. [Online]. Available: <https://libreswan.org/> (visited on 21/11/2023).
- [68] "Manually configure IPsec VPN using ip xfrm". (Oct. 8, 2022), [Online]. Available: <https://www.sobyte.net/post/2022-10/ipsec-ip-xfrm/> (visited on 06/10/2023).
- [69] J. Viega and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)," Internet Engineering Task Force, Request for Comments RFC 4106, Jun. 2005, 11 pp. DOI: 10.17487/RFC4106. [Online]. Available: <https://datatracker.ietf.org/doc/rfc4106> (visited on 21/11/2023).
- [70] dorian. "Why are only 3 ip xfrm policies needed for a IPsec tunnel?" Server Fault. (10th Oct. 2014), [Online]. Available: <https://serverfault.com/q/635012> (visited on 10/06/2023).
- [71] Andrej Stender. "Nftables - Netfilter and VPN/IPsec packet flow." (14th Aug. 2022), [Online]. Available: https://thermalcircle.de/doku.php?id=blog:linux:nftables_ipsec_packet_flow#the_xfrm_framework (visited on 06/06/2023).
- [72] *RC10 Wideband ARQ & IP Controller Datasheet*. Rapid Mobile (Pty) Ltd., 2019. [Online]. Available: https://www.rapidm.com/wp-content/uploads/2019/02/RC10_HF_S5066_EN.pdf (visited on 09/06/2023).
- [73] *RM10 Wideband SDM & ALE - 24 kHz Data Modem: NB & WBHF Datasheet*. Rapid Mobile (Pty) Ltd., 2021. [Online]. Available: https://www.rapidm.com/wp-content/uploads/2018/10/RM10_WBHF_EN.pdf (visited on 09/06/2023).
- [74] *RS10 Wideband HF & V/UHF Channel Simulator - 24 kHz Datasheet*. Rapid Mobile (Pty) Ltd., 2019. [Online]. Available: https://www.rapidm.com/wp-content/uploads/2019/03/RS10_HFM_EN.pdf (visited on 06/09/2023).
-

-
- [75] Rainer Oellermann, *Personal Correspondance - General STANAG 5066 behaviour, STANAG 5066 Expert*, Jun. 2023.
- [76] E. W. Biederman and N. Dichtel, *Ip-Netns(8) Linux Manual Page*. 16th Jan. 2013. [Online]. Available: <https://man7.org/linux/man-pages/man8/ip-netns.8.html> (visited on 19/09/2023).
- [77] Anon., *Veth(4) Linux Kernel Interfaces Manual*. 5th Feb. 2023. [Online]. Available: <https://man7.org/linux/man-pages/man4/veth.4.html> (visited on 19/09/2023).
- [78] Anon, *Segmentation Offloads — The Linux Kernel Documentation*. The kernel development community. [Online]. Available: <https://docs.kernel.org/networking/segmentation-offloads.html> (visited on 24/11/2023).
- [79] S. O. Bradner, "Benchmarking Terminology for Network Interconnection Devices," Internet Engineering Task Force, Request for Comments RFC 1242, Jul. 1991, 12 pp. DOI: 10.17487/RFC1242. [Online]. Available: <https://datatracker.ietf.org/doc/rfc1242> (visited on 03/08/2023).
- [80] S. Poretsky, S. Erramilli, J. Perser and S. Khurana, "Terminology for Benchmarking Network-layer Traffic Control Mechanisms," Internet Engineering Task Force, Request for Comments RFC 4689, Oct. 2006, 34 pp. DOI: 10.17487/RFC4689. [Online]. Available: <https://datatracker.ietf.org/doc/rfc4689> (visited on 03/08/2023).
- [81] Anon., *Tcp(7) Linux Miscellaneous Information Manual*. 21st Mar. 2023. [Online]. Available: <https://man7.org/linux/man-pages/man7/tcp.7.html> (visited on 20/09/2023).
- [82] Jon Dugan, Seth Elliot, Bruce A. Mah, Jeff Poskanzer and Kaustubh Prabhu, *iPerf - The TCP, UDP and SCTP network bandwidth measurement tool*. [Online]. Available: <https://iperf.fr/> (visited on 20/09/2023).
- [83] Anon, *Nuttcp*. [Online]. Available: <https://www.nuttcp.net/Welcome%20Page.html> (visited on 20/09/2023).
- [84] Edgio, *Hurl: Http(s)+h2 server load tester*. [Online]. Available: <https://github.com/Edgio/hurl> (visited on 20/09/2023).
- [85] G. Tene, *Wrk2*. [Online]. Available: <https://github.com/giltene/wrk2> (visited on 20/09/2023).
- [86] Jon Dugan, Seth Elliot, Bruce A. Mah, Jeff Poskanzer and Kaustubh Prabhu, *Iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool*, ESnet. [Online]. Available: <https://github.com/esnet/iperf> (visited on 20/09/2023).
- [87] M. Sargent, J. Chu, V. Paxson and M. Allman, "Computing TCP's Retransmission Timer," Internet Engineering Task Force, Request for Comments RFC 6298, Jun. 2011, 11 pp. DOI: 10.17487/RFC6298. [Online]. Available: <https://datatracker.ietf.org/doc/rfc6298> (visited on 22/09/2023).
- [88] Anon., *ioctl_tty(2) Linux System Calls Manual*. 5th Feb. 2023. [Online]. Available: https://man7.org/linux/man-pages/man2/ioctl_tty.2.html (visited on 22/09/2023).
- [89] *Letter Symbols to Be Used in Electrical Technology - Part 2: Telecommunications and Electronics*. IEC 60027-2, 8th Jan. 2019.
- [90] G. Kroah-Hartman and L. Torvalds, *Linux kernel stable tree*, version 5.15.122, 24th Jul. 2023. [Online]. Available: <https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git/tree/?id=v5.15.122> (visited on 26/10/2023).
- [91] Rainer Oellermann, *Personal Correspondance - STANAG 5066 small payload behaviour, STANAG 5066 Expert*, Nov. 2023.
-

Appendices

A Baseline latency result sample analysis and visualisation

A.1 Overview

The data presented in this appendix serves as supplementary information for the initial analysis of the baseline one-way latency, and TCP RTT measurements, discussed in chapter 5 in sections 5.2.4.2 and 5.2.4.4 respectively.

Specific observations related to the information presented here is made directly in chapter 5, notably in regard to clarifying the seeming anomaly related to the maximum TCP RTT and maximum one-way latency results which is discussed in section 5.2.4.4.

A.2 One-way latency

The five-number summary of the one-way latency baseline results, together with the SD for ease of reference, is presented in Table A.1.

Table A.1: Baseline results - One-way latency (seconds) five-number summary with SD

Channel BW & Model	SNR	Successful Run Count	Min	Q_1	Median (Q_2)	Q_3	Max	SD (σ)
12 kHz AWGN	Good	5 of 5	4.139	4.259	4.300	4.343	4.501	0.066
	Average	5 of 5	4.150	4.231	4.236	4.276	4.437	0.045
	Bad	5 of 5	4.005	4.088	4.131	4.174	21.250	2.425
12 kHz "Poor"	Good	5 of 5	3.963	4.362	4.404	4.456	15.241	1.604
	Average	5 of 5	4.008	4.374	4.648	15.811	91.773	16.165
	Bad	4 of 5	3.972	4.605	4.933	6.364	149.219	15.027
18 kHz AWGN	Good	5 of 5	3.934	4.299	4.338	4.378	5.858	0.102
	Average	5 of 5	4.153	4.274	4.314	4.318	4.558	0.055
	Bad	5 of 5	5.104	5.212	5.216	5.255	16.502	0.660
18 kHz "Poor"	Good	5 of 5	4.191	4.350	4.673	6.399	72.058	9.414
	Average	5 of 5	3.897	4.379	4.703	15.403	78.656	15.312
	Bad	5 of 5	4.177	4.741	5.053	7.315	130.149	14.695

Figures A.1 to A.4 present cumulative sample count graphs for each test case, grouped according to the channel bandwidth and channel model pairing. These graphs visualise the outlier data, such as the maximum one-way latency sample for each test, in relation to the rest of the measurement samples, and provide a way to determine what percentage of samples are contained within a time interval, e.g. the percentage of samples contained in one SD from the mean.

It is important to note that the magnitude of the outliers necessitate the use of an x-axis scale that can make it difficult to read specific values directly from the graph in some cases. To address

this, some specific percentile ($P_{\%}$) values are given in Table A.2 for each test case.

Table A.2: Baseline results - One-way latency (seconds) percentile values

Channel BW & Model	SNR	Successful Run Count	$P_{75\%}$	$P_{80\%}$	$P_{85\%}$	$P_{90\%}$	$P_{95\%}$	$P_{98\%}$
12 kHz AWGN	Good	5 of 5	4.343	4.377	4.379	4.381	4.418	4.421
	Average	5 of 5	4.276	4.277	4.278	4.315	4.318	4.320
	Bad	5 of 5	4.174	4.176	4.261	4.373	12.569	12.696
12 kHz "Poor"	Good	5 of 5	4.456	4.481	4.484	4.522	4.563	13.242
	Average	5 of 5	15.811	23.159	28.641	39.620	46.158	64.300
	Bad	4 of 5	6.364	7.791	17.588	24.657	42.862	47.399
18 kHz AWGN	Good	5 of 5	4.378	4.379	4.381	4.383	4.419	4.422
	Average	5 of 5	4.318	4.354	4.356	4.358	4.395	4.398
	Bad	5 of 5	5.255	5.258	5.259	5.261	5.375	5.897
18 kHz "Poor"	Good	5 of 5	6.399	7.990	16.842	23.754	26.746	36.767
	Average	5 of 5	15.403	23.146	25.098	42.626	46.795	53.726
	Bad	5 of 5	7.315	14.958	23.415	33.727	43.644	54.070

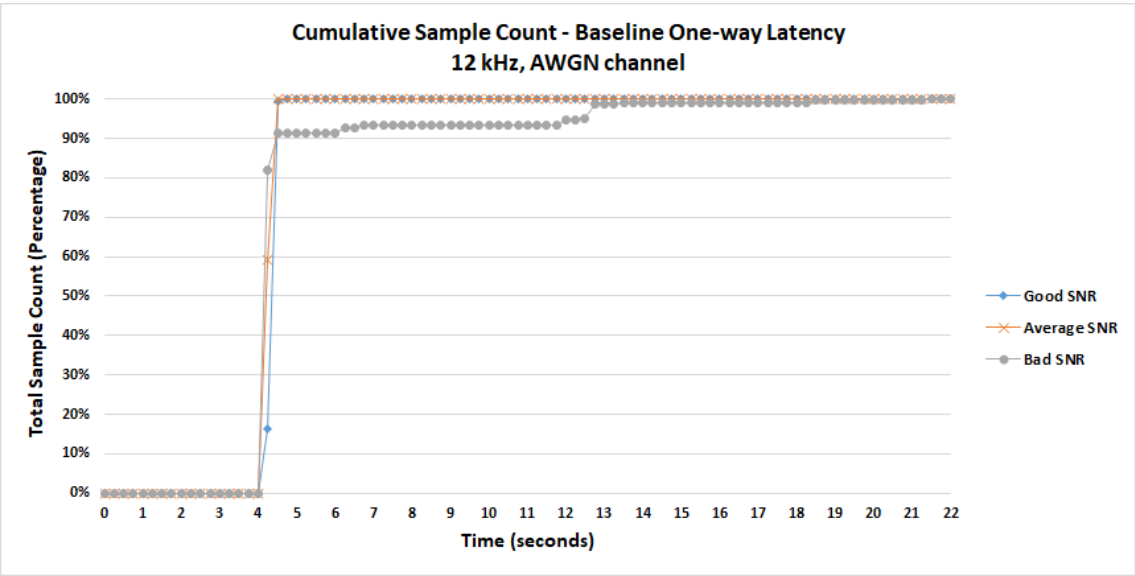


Figure A.1: Cumulative sample count: Baseline One-way latency, 12 kHz, AWGN channel model

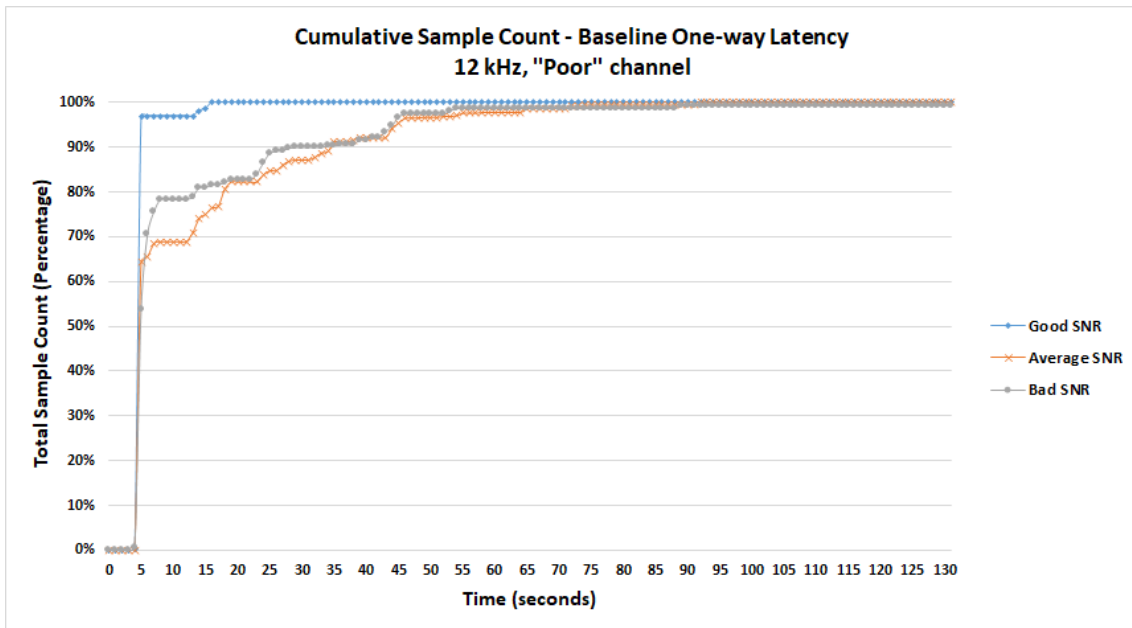


Figure A.2: Cumulative sample count: Baseline One-way latency, 12 kHz, "Poor" channel model

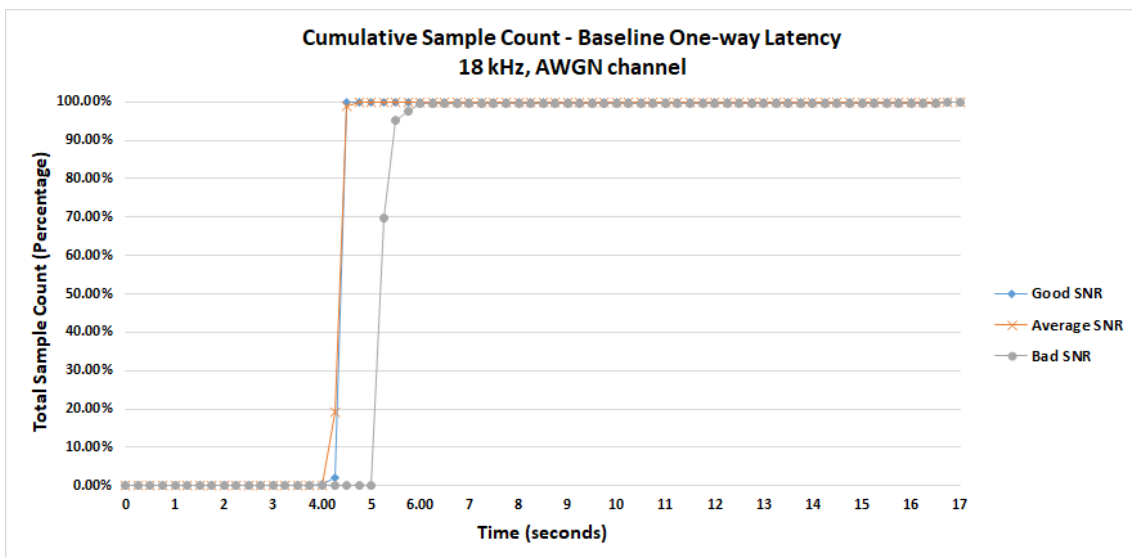


Figure A.3: Cumulative sample count: Baseline One-way latency, 18 kHz, AWGN channel model

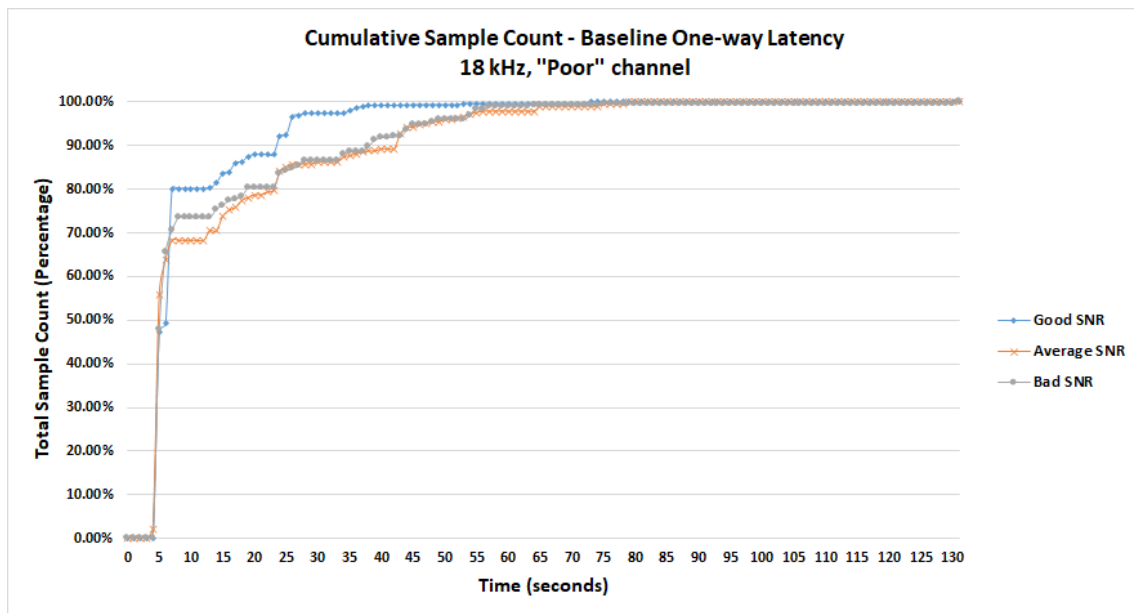


Figure A.4: Cumulative sample count: Baseline One-way latency, 18 kHz, “Poor” channel model

A.3 TCP RTT

The five-number summary of the TCP RTT baseline results, together with the SD for ease of reference, is presented in Table A.3.

Table A.3: Baseline results - TCP RTT (seconds) five-number summary with SD

Channel BW & Model	SNR	Successful Run Count	Min	Q_1	Median (Q_2)	Q_3	Max	SD (σ)
12 kHz AWGN	Good	5 of 5	8.589	8.611	8.619	8.628	8.790	0.025
	Average	5 of 5	8.554	8.591	8.761	10.077	26.987	2.776
	Bad	5 of 5	8.339	8.798	9.110	9.630	12.196	0.649
12 kHz "Poor"	Good	5 of 5	8.739	8.817	8.995	9.607	12.114	0.655
	Average	5 of 5	8.878	21.912	28.224	33.266	45.601	9.034
	Bad	4 of 5	9.173	11.198	15.087	27.789	79.037	12.188
18 kHz AWGN	Good	5 of 5	8.665	8.680	8.692	8.766	9.057	0.076
	Average	5 of 5	8.590	8.638	8.781	9.901	18.281	2.188
	Bad	5 of 5	10.287	10.410	10.415	10.419	15.825	0.767
18 kHz "Poor"	Good	5 of 5	8.968	14.696	16.532	18.505	43.349	4.620
	Average	5 of 5	10.500	20.179	24.665	31.224	79.444	10.077
	Bad	5 of 5	11.717	17.410	26.813	34.018	49.690	9.291

Figures A.5 to A.8 present cumulative sample count graphs for each test case, grouped according to the channel bandwidth and channel model pairing. These graphs visualise the outlier data, such as the maximum TCP RTT sample for each test, in relation to the rest of the measurement samples, and provide a way to determine what percentage of samples are contained within a time interval, e.g. the percentage of samples contained in one SD from the mean.

It is important to note that the magnitude of the outliers necessitate the use of an x-axis scale that can make it difficult to read specific values directly from the graph in some cases. To address this, some specific percentile ($P_{\%}$) values are given in Table A.4 for each test case.

Table A.4: Baseline results - TCP RTT (seconds) percentile values

Channel BW & Model	SNR	Successful Run Count	$P_{75\%}$	$P_{80\%}$	$P_{85\%}$	$P_{90\%}$	$P_{95\%}$	$P_{98\%}$
12 kHz AWGN	Good	5 of 5	8.628	8.631	8.634	8.641	8.665	8.704
	Average	5 of 5	10.077	10.849	11.997	13.708	16.277	18.511
	Bad	5 of 5	9.630	9.750	9.953	10.150	10.504	10.931
12 kHz "Poor"	Good	5 of 5	9.607	9.778	9.942	10.164	10.589	11.309
	Average	5 of 5	33.266	34.654	36.210	37.965	40.356	43.377
	Bad	4 of 5	27.789	30.157	34.205	37.829	40.940	51.323
18 kHz AWGN	Good	5 of 5	8.766	8.788	8.816	8.846	8.894	8.947
	Average	5 of 5	9.901	10.539	11.505	12.937	15.056	17.022
	Bad	5 of 5	10.419	10.420	10.422	10.424	11.438	13.853
18 kHz "Poor"	Good	5 of 5	18.505	19.337	20.180	21.260	23.375	33.840
	Average	5 of 5	31.224	33.076	34.424	36.928	46.745	54.844
	Bad	5 of 5	34.018	35.749	36.732	38.247	40.308	42.138

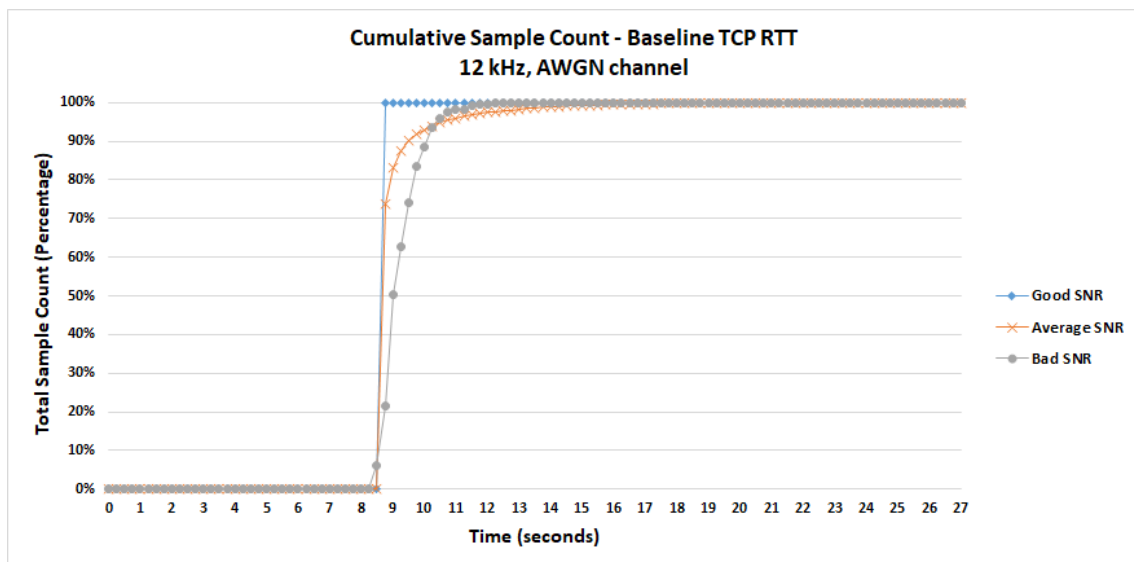


Figure A.5: Cumulative sample count: Baseline TCP RTT, 12 kHz, AWGN channel model

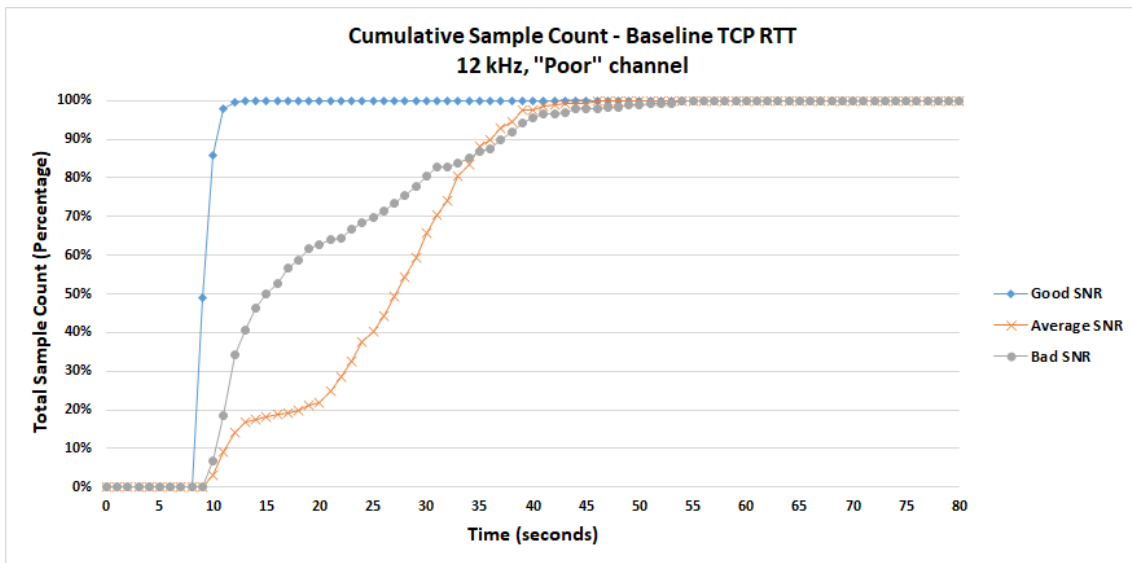


Figure A.6: Cumulative sample count: Baseline TCP RTT, 12 kHz, "Poor" channel model

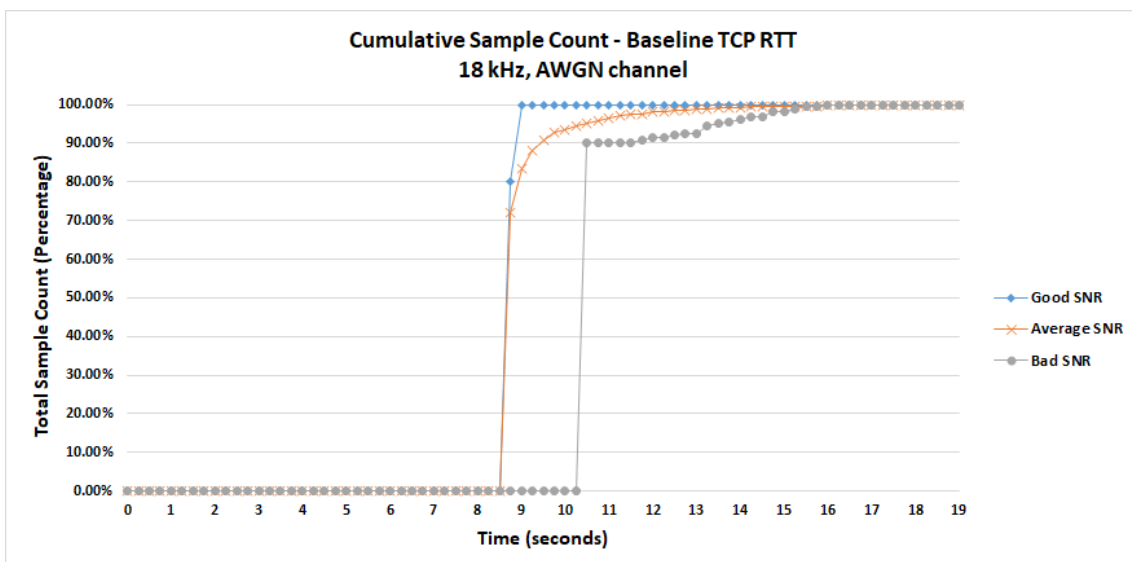


Figure A.7: Cumulative sample count: Baseline TCP RTT, 18 kHz, AWGN channel model

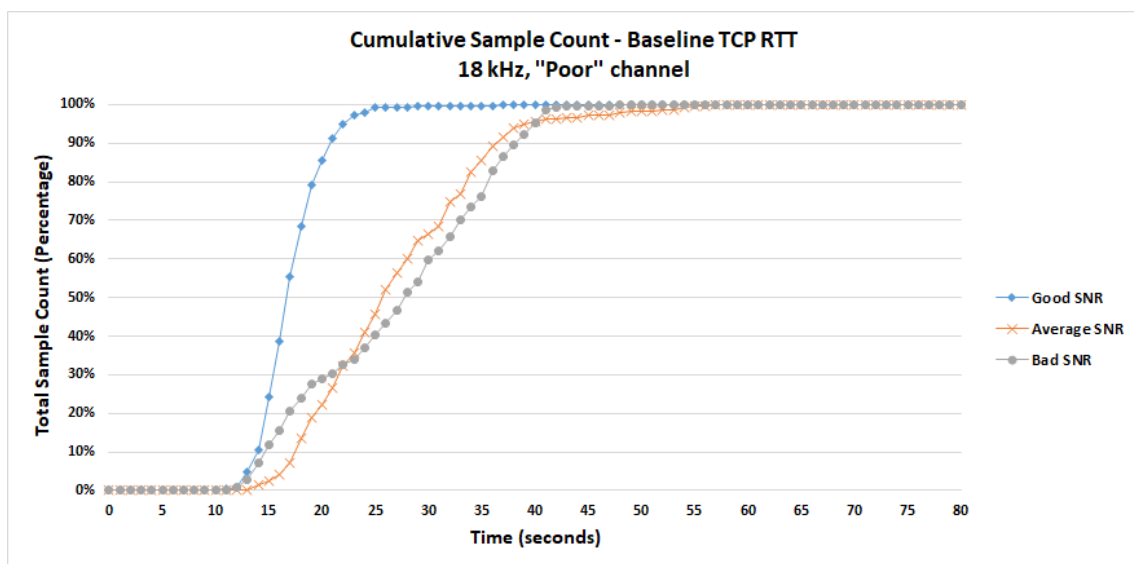


Figure A.8: Cumulative sample count: Baseline TCP RTT, 18 kHz, “Poor” channel model

B IP crypto latency result sample analysis and visualisation

B.1 Overview

The data presented in this appendix is the equivalent of the supplementary information in Appendix A, but for the IP crypto test result set. It is provided primarily for information purposes, since this additional data did not make any material contribution to the IP crypto latency performance impact analysis presented in sections 5.3.4.3 and 5.3.4.4 of chapter 5.

B.2 One-way latency

The five-number summary of the one-way latency IP crypto results, together with the SD for ease of reference, is presented in Table B.1.

Table B.1: IP crypto results - One-way latency (seconds) five-number summary with SD

Channel BW & Model	SNR	Successful Run Count	Min	Q_1	Median (Q_2)	Q_3	Max	SD (σ)
12 kHz AWGN	Good	4 of 5	4.132	4.254	4.294	4.337	4.459	0.068
	Average	5 of 5	4.271	4.626	4.649	4.671	16.187	0.669
	Bad	3 of 5	4.205	4.539	4.606	4.819	16.205	2.892
12 kHz "Poor"	Good	5 of 5	4.002	4.356	4.399	4.476	24.116	2.185
	Average	5 of 5	4.112	4.710	4.834	7.029	74.963	13.750
	Bad	4 of 5	4.240	4.564	4.724	5.365	52.125	5.558
18 kHz AWGN	Good	5 of 5	4.248	4.295	4.333	4.373	5.934	0.136
	Average	5 of 5	4.428	4.511	4.550	4.590	4.796	0.059
	Bad	5 of 5	5.124	5.168	5.207	5.211	5.918	0.089
18 kHz "Poor"	Good	5 of 5	3.632	4.325	4.406	6.244	78.249	8.661
	Average	3 of 5	4.095	4.532	4.831	6.567	77.087	14.946
	Bad	5 of 5	4.248	4.687	5.088	5.973	96.327	12.244

Figures B.1 to B.4 present cumulative sample count graphs for each IP crypto test case, grouped according to the channel bandwidth and channel model pairing. These graphs visualise the outlier data, such as the maximum one-way latency sample for each test, in relation to the rest of the measurement samples, and provide a way to determine what percentage of samples are contained within a time interval, e.g. the percentage of samples contained in one SD from the mean.

It is important to note that the magnitude of the outliers necessitate the use of an x-axis scale that can make it difficult to read specific values directly from the graph in some cases. To address this, some specific percentile ($P_{\%}$) values are given in Table B.2 for each test case.

Table B.2: IP crypto results - One-way latency (seconds) percentile values

Channel BW & Model	SNR	Successful Run Count	$P_{75\%}$	$P_{80\%}$	$P_{85\%}$	$P_{90\%}$	$P_{95\%}$	$P_{98\%}$
12 kHz AWGN	Good	4 of 5	4.337	4.373	4.374	4.381	4.415	4.418
	Average	5 of 5	4.671	4.672	4.707	4.710	4.712	4.747
	Bad	3 of 5	4.819	4.859	4.889	13.246	13.643	14.025
12 kHz "Poor"	Good	5 of 5	4.476	4.477	4.479	4.517	4.563	13.194
	Average	5 of 5	7.029	15.626	24.500	33.442	44.935	57.050
	Bad	4 of 5	5.365	6.246	7.128	13.772	17.007	24.175
18 kHz AWGN	Good	5 of 5	4.373	4.374	4.375	4.381	4.416	4.456
	Average	5 of 5	4.590	4.591	4.593	4.598	4.634	4.752
	Bad	5 of 5	5.211	5.211	5.214	5.250	5.253	5.338
18 kHz "Poor"	Good	5 of 5	6.244	6.329	14.820	16.693	23.977	34.845
	Average	3 of 5	6.567	12.924	23.196	34.239	44.198	63.433
	Bad	5 of 5	5.973	13.633	15.786	24.816	37.421	47.628

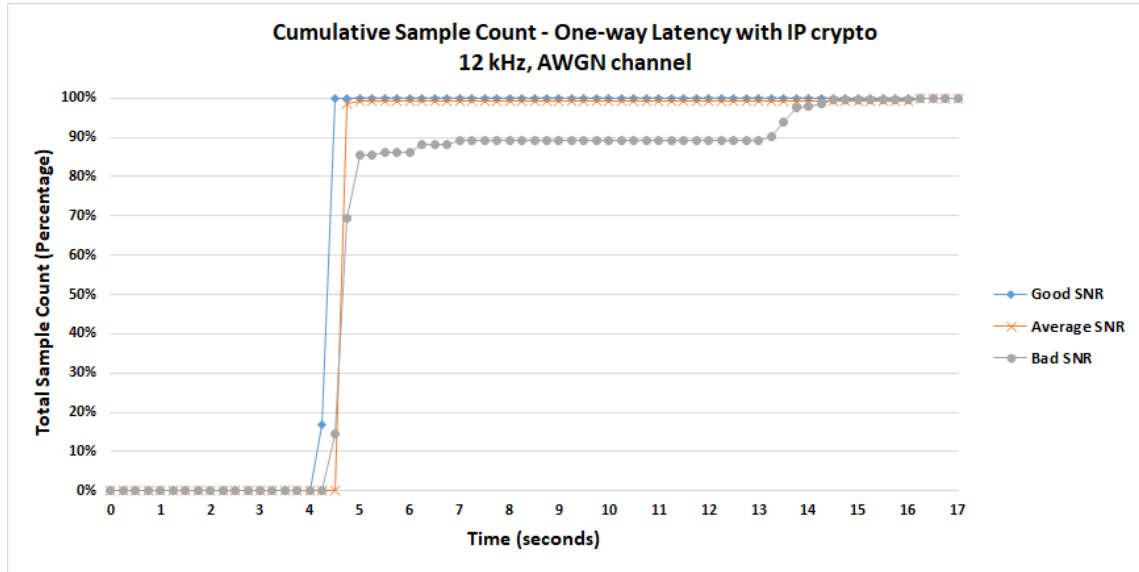


Figure B.1: Cumulative sample count: IP crypto One-way latency, 12 kHz, AWGN channel model

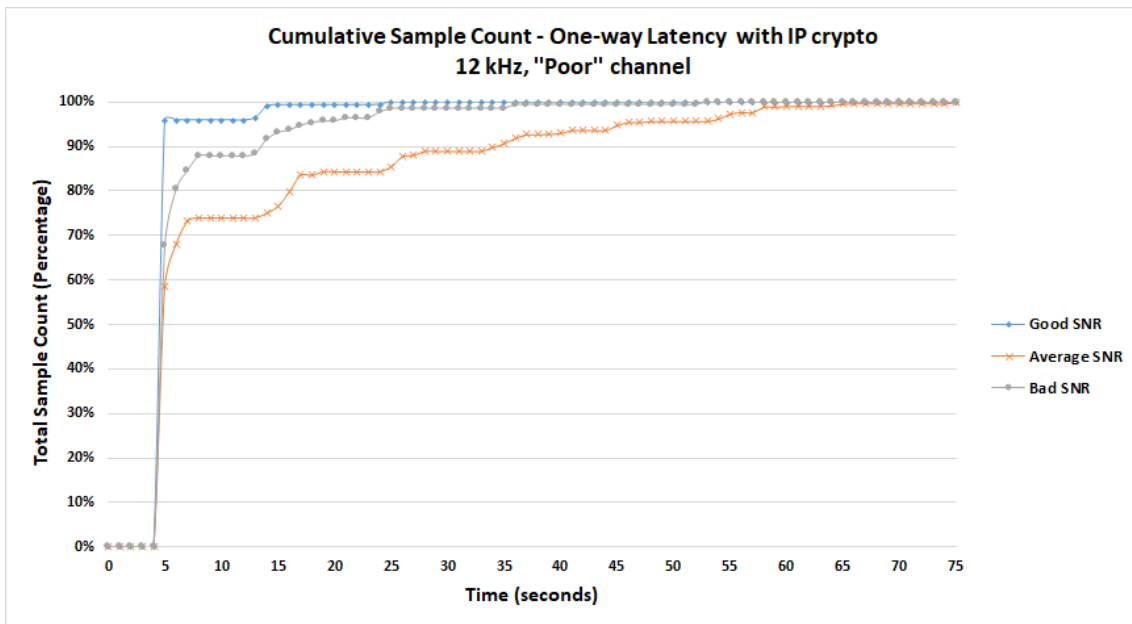


Figure B.2: Cumulative sample count: IP crypto One-way latency, 12 kHz, "Poor" channel model

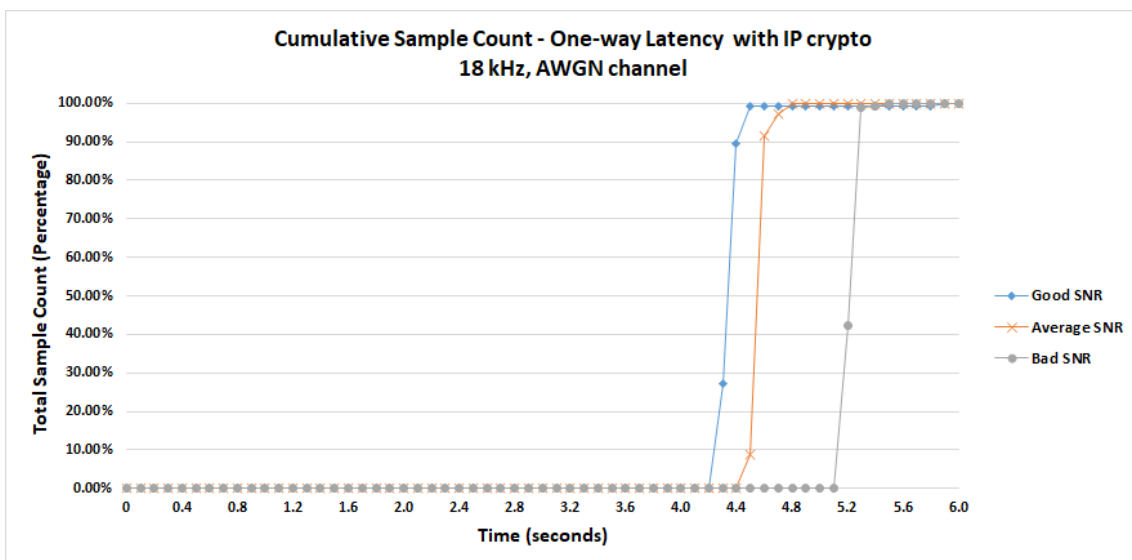


Figure B.3: Cumulative sample count: IP crypto One-way latency, 18 kHz, AWGN channel model

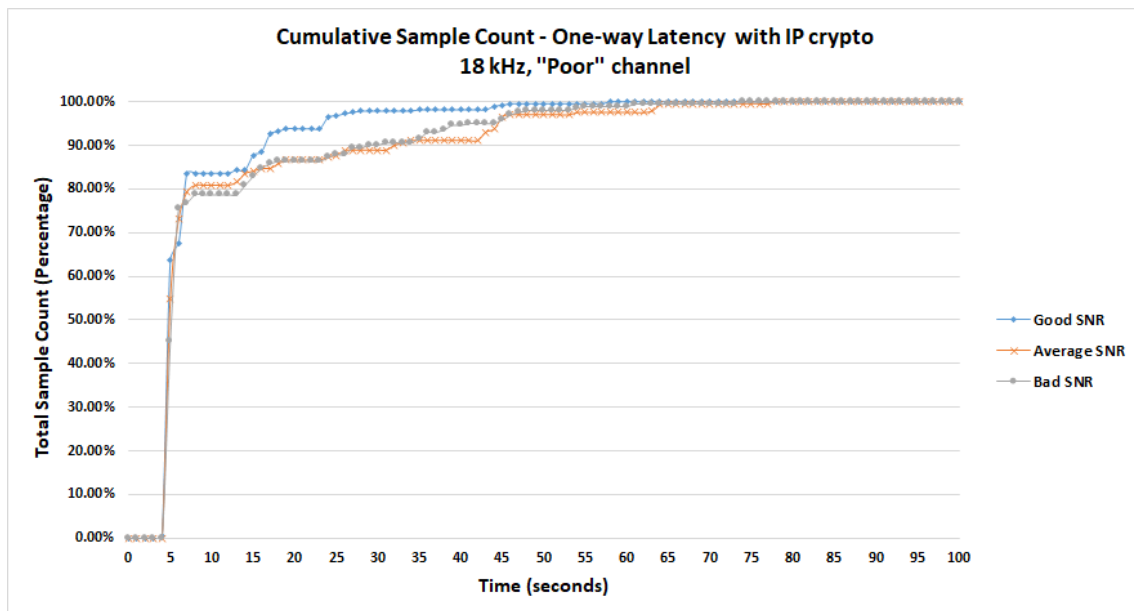


Figure B.4: Cumulative sample count: IP crypto One-way latency, 18 kHz, “Poor” channel model

B.3 TCP RTT

The five-number summary of the TCP RTT IP crypto results, together with the SD for ease of reference, is presented in Table B.3.

Table B.3: IP crypto results - TCP RTT (seconds) five-number summary with SD

Channel BW & Model	SNR	Successful Run Count	Min	Q_1	Median (Q_2)	Q_3	Max	SD (σ)
12 kHz AWGN	Good	4 of 5	8.591	8.613	8.621	8.633	8.790	0.026
	Average	5 of 5	9.115	9.162	9.421	11.321	32.375	3.847
	Bad	3 of 5	9.264	10.236	11.082	12.282	20.340	2.244
12 kHz "Poor"	Good	5 of 5	8.662	8.811	9.097	9.790	12.313	0.727
	Average	5 of 5	9.275	14.875	24.992	30.159	66.757	10.021
	Bad	4 of 5	10.304	11.769	12.890	14.407	63.212	6.776
18 kHz AWGN	Good	5 of 5	8.659	8.682	8.695	8.754	9.046	0.074
	Average	5 of 5	9.091	9.140	9.298	10.564	27.420	2.703
	Bad	5 of 5	10.387	10.397	10.404	10.438	11.551	0.180
18 kHz "Poor"	Good	5 of 5	8.888	12.785	15.269	18.102	24.608	3.397
	Average	3 of 5	9.894	22.732	31.818	38.290	82.219	13.466
	Bad	5 of 5	10.360	12.317	18.660	33.070	88.395	13.408

Figures B.5 to B.8 present cumulative sample count graphs for each test case, grouped according to the channel bandwidth and channel model pairing. These graphs visualise the outlier data, such as the maximum TCP RTT sample for each test, in relation to the rest of the measurement samples, and provide a way to determine what percentage of samples are contained within a time interval, e.g. the percentage of samples contained in one SD from the mean.

It is important to note that the magnitude of the outliers necessitate the use of an x-axis scale that can make it difficult to read specific values directly from the graph in some cases. To address this, some specific percentile ($P_{\%}$) values are given in Table B.4 for each test case.

Table B.4: IP crypto results - TCP RTT (seconds) percentile values

Channel BW & Model	SNR	Successful Run Count	$P_{75\%}$	$P_{80\%}$	$P_{85\%}$	$P_{90\%}$	$P_{95\%}$	$P_{98\%}$
12 kHz AWGN	Good	4 of 5	8.633	8.638	8.646	8.653	8.671	8.712
	Average	5 of 5	11.321	12.120	13.637	15.940	19.424	24.581
	Bad	3 of 5	12.282	12.886	13.680	14.998	16.957	18.223
12 kHz "Poor"	Good	5 of 5	9.790	9.956	10.188	10.498	10.860	11.378
	Average	5 of 5	30.159	31.654	32.610	34.259	37.594	46.918
	Bad	4 of 5	14.407	14.920	15.455	16.597	22.709	39.327
18 kHz AWGN	Good	5 of 5	8.754	8.775	8.803	8.842	8.893	8.947
	Average	5 of 5	10.564	11.281	12.363	13.975	16.399	18.656
	Bad	5 of 5	10.438	10.456	10.487	10.516	10.846	11.184
18 kHz "Poor"	Good	5 of 5	18.102	18.687	19.430	20.145	21.522	22.818
	Average	3 of 5	38.290	39.575	41.497	43.634	52.850	69.723
	Bad	5 of 5	33.070	35.675	38.081	40.833	45.000	54.641

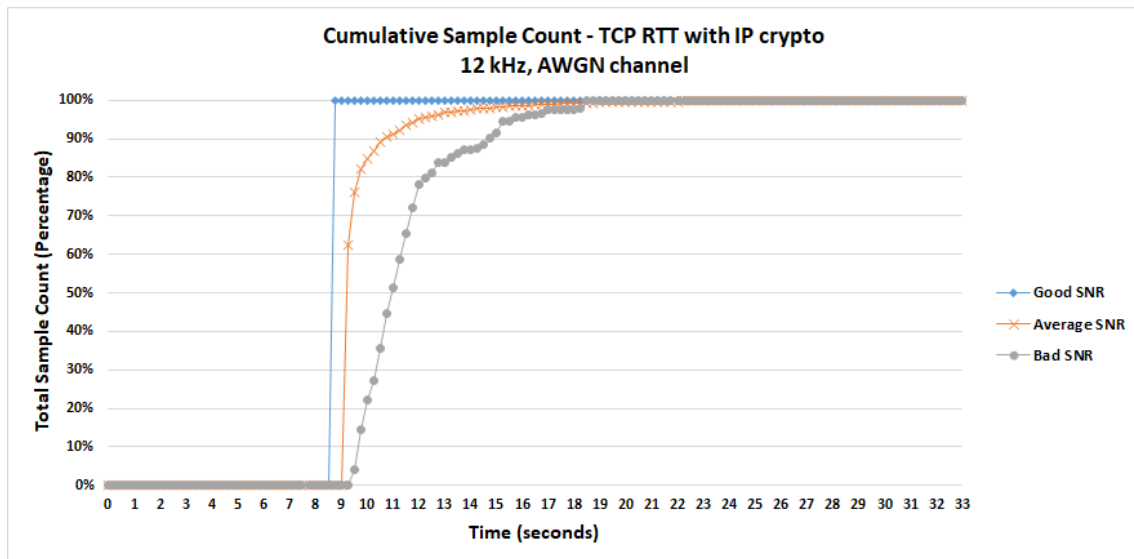


Figure B.5: Cumulative sample count: IP crypto TCP RTT, 12 kHz, AWGN channel model

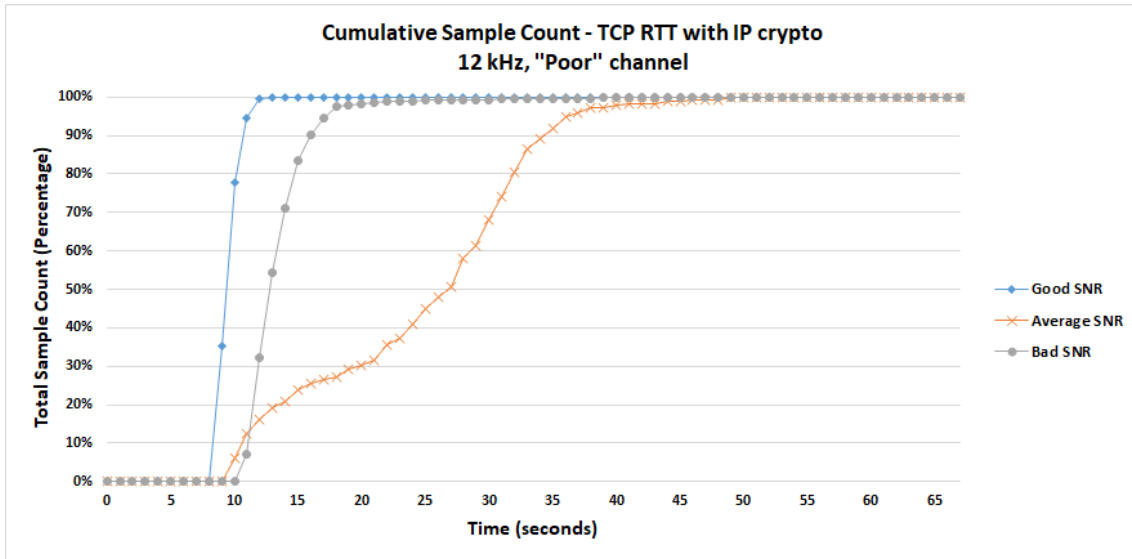


Figure B.6: Cumulative sample count: IP crypto TCP RTT, 12 kHz, "Poor" channel model

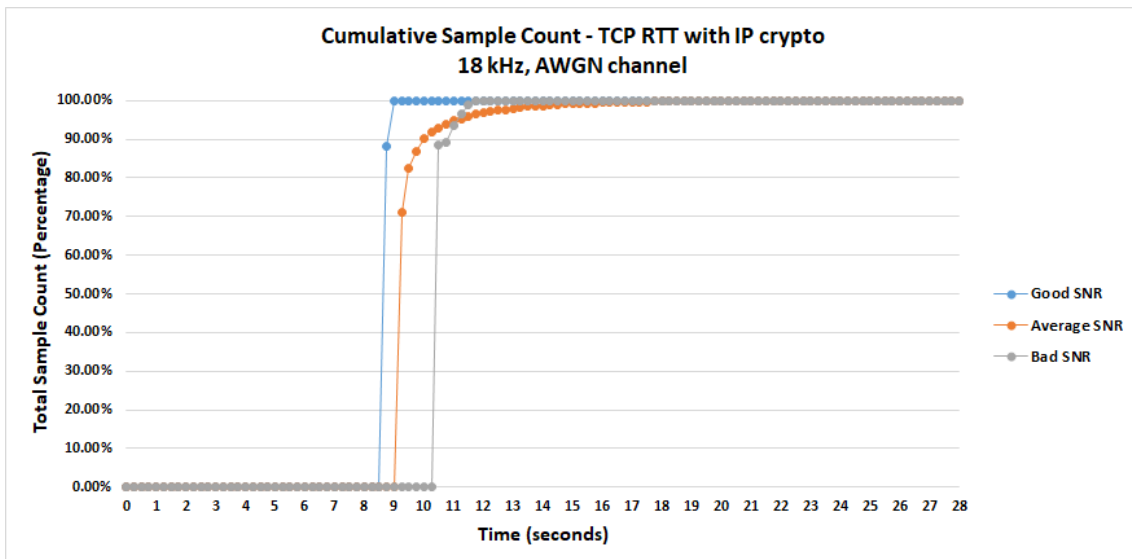


Figure B.7: Cumulative sample count: IP crypto TCP RTT, 18 kHz, AWGN channel model

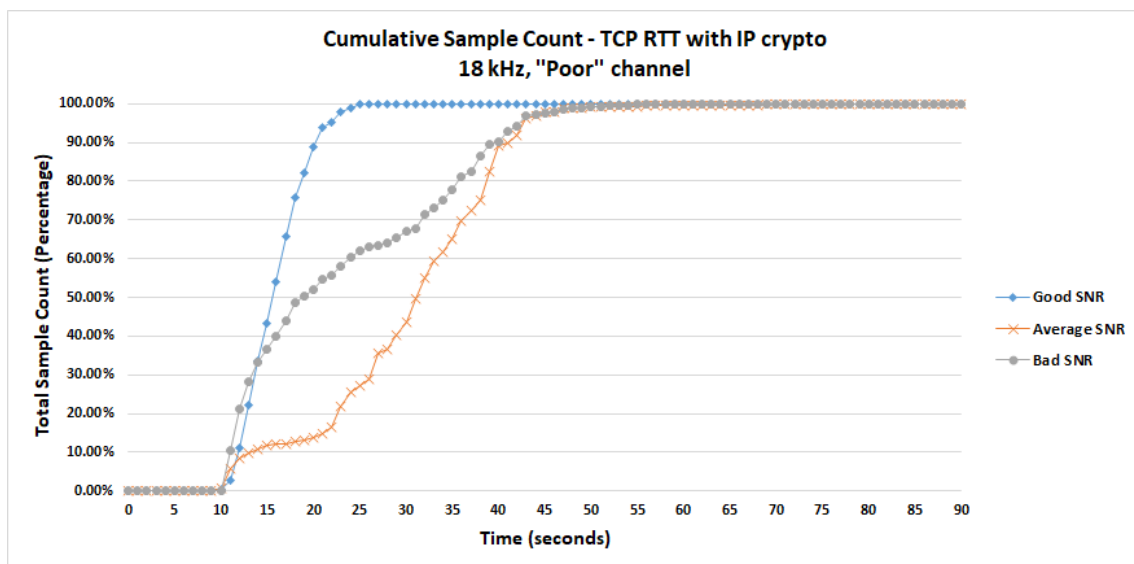


Figure B.8: Cumulative sample count: IP crypto TCP RTT, 18 kHz, “Poor” channel model