

A Low-Bandwidth Encryption Protocol for AUVs

Owain D. Jones , Catherine A. Harris , Julia Bonat, James Kirk, Daniel Jones, Ashley Icton-Morris, Francesco Fanelli, Davide Fenucci, Alexander B. Phillips

Marine Autonomous and Robotics Systems, National Oceanography Centre, Southampton, United Kingdom

Index Terms—Autonomous Underwater Vehicles, information security, multi-party encryption, Iridium satellite communication, underwater acoustics, bandwidth-constrained messaging

Abstract—As a growing component of the Global Ocean Observing System (GOOS), MAS contribute to the prediction and monitoring of concerns with global impact, and should be viewed as a subsystem of critical infrastructure. At present, the communication security of MAS for civilian science applications is underdeveloped within the literature. In this paper, we explore the historical reasons for this, including strict power and bandwidth constraints, and a collaborative approach to scientific MAS deployments. We describe our open-source compact lightweight encryption protocol *Flatfish*, which uses a hybrid-encryption scheme and supports: one or more recipients, message authentication via an AEAD tag, and the splitting of larger messages for low bandwidth communications. We request comment and collaboration to further develop an open community standard for scientific MAS message encryption and related application areas, to increase the cybersecurity and resilience of the GOOS.

I. INTRODUCTION

Marine Autonomous Systems (MAS) are an increasingly important component of the Global Ocean Observing System (GOOS) [1], with autonomous networks (primarily Argo profiling floats and MAS gliders) driving recent expansion of the in-situ component of the GOOS [2]. Remote MAS have enabled the near-real-time collection and transmission of scientific measurements to open databases, accessible to scientists, policymakers and the public worldwide.

Whilst the importance of securing the integrity of MAS operations and data transfer has been long-established in domains such as defence, it is rarely publicly addressed for civilian science applications. In their 2025 report, the GOOS Secretariat et al. [2] state that despite the ocean observing system’s critical role in climate resilience, disaster risk reduction, economic stability and other globally important concerns, the GOOS ‘remains subcritical, with limited resilience to financial or geopolitical disruptions’. They call on the global community to view ocean observations as critical infrastructure, on par with weather or energy services. If we consider civilian scientific MAS in this way, the resilience of the entire system needs to be reviewed. The security of remote MAS communications is an area which, to the best of our knowledge, remains hugely under-represented in the literature.

We consider the integrity of civilian MAS scientific data transmissions in the context of a geopolitical cyber threat model: The corruption or falsification of MAS sensor data would have potential for widespread and undetected impacts to environmental research and the GOOS, with additional

financial and reputational impacts to operators and funding-bodies. We discuss the operational constraints preventing use of established internet encryption protocols and present our compact lightweight protocol *Flatfish*. We request community engagement to develop and adopt an open standard for civilian MAS data encryption.

II. HISTORICAL BARRIERS TO MAS ENCRYPTION (WHY BOATY COULDN’T ENCRYPT)

Encryption protocols are fundamental to the modern world, increasing security by ensuring sensitive data messages are readable only by their intended recipients. However, marine science is one civilian application in which operational security has historically been overlooked. As a result, scientific equipment has commonly transmitted data unencrypted over open channels such as serial radio links, underwater acoustic modems and low-bandwidth satellite connections. In this section we discuss the reasons that civilian scientific MAS, such as underwater gliders [3] and long-range Autonomous Underwater Vehicles (AUVs) [4] have historically been unable to prioritise encryption.

Severely restricted bandwidth: To conserve battery usage and enable multiple months of operation, long-range MAS typically use low-power and low-bandwidth hardware, including for communication with remote operators. Communication mediums include:

- *Iridium satellite modems - Short Burst Data* [5] which provide robustness and global coverage, with low energy consumption. The Iridium 9602 and 9523 transceiver chips can receive SBD messages with size ≤ 270 bytes and ≤ 1890 bytes, respectively.
- *Acoustic modems* [6], often as a secondary function of Ultra-Short Base-Line (USBL) positioning systems when within range of a ship or acoustic network. Some USBL systems allow use of the transducer as an acoustic modem in addition to position aiding, but doing so may limit ranging functionality and increase power consumption. Data payloads in commercial USBL systems range from 20 to 4095 bytes, depending on the telemetry scheme and environmental conditions.

These restrictive bandwidth constraints, combined with limited awareness of the state of the remote MAS and its operating environment, result in compact, tightly specified messaging formats [7], [8].

Strict power constraints: Encryption requires additional processing, but long-range MAS typically utilise low-powered computers. Whilst newer MAS use more capable chips [4],

older models are still widely used, e.g. the Teledyne Webb Research Slocum G2 Glider, which has a single thread controller with 1MB of RAM [9]. The recent Slocum G3S uses a multi-threaded STM32 processor [10] but is still limited by 8MB RAM.

Unreliable communications: When operating in extreme ocean environments, it is common to lose messages. Therefore, any form of encryption or authentication that assumes a known messaging order or adopts handshaking protocols is unsuitable for operational MAS use.

A diverse range of users: Deploying and operating MAS is a highly multidisciplinary activity, with a diverse range of stakeholders. As MAS pilots are often technical experts rather than marine scientists, it is essential that multiple authenticated users have access to near-real-time MAS data, to assess the data against scientific objectives and operational constraints. Encryption methods that assume one-to-one transmission are insufficient.

Mixed-MAS deployments: Heterogeneous MAS deployments [11], where additional platforms provide greater sensing or communication capabilities (e.g. as communication relays), benefit from common communication languages [12]. In such broadcast environments, which may be vulnerable to eavesdropping and false data injection, MAS need to authenticate themselves.

Lack of open community-standard protocols: At present, the lack of an open community-standard MAS encryption protocol makes it difficult for MAS operators, often without comprehensive cryptographic knowledge, to individually successfully adopt appropriate encryption [13]. Community standard protocols encouraging interoperability and enabling oversight from a wide community of scientific stakeholders and security researchers, are therefore attractive.

III. REQUIREMENTS DEFINITION

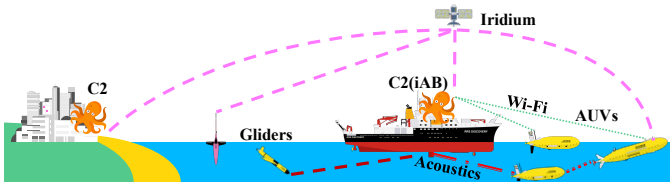


Fig. 1: A mixed-MAS deployment, each device considered an encrypted message ‘recipient’. Messages may be sent via multiple channels and addressed one-to-one, one-to-many, or many-to-many.

Considering MAS operations at the National Oceanography Centre as an exemplar, our primary aim is to secure Iridium satellite transmissions, identifying these as the greatest risk. Secondly, we consider the extremely-low-bandwidth constraints of acoustic communications. At this stage, our focus is not on novel cryptographic research, but instead on finding solutions to operational needs. A protocol must:

Be platform agnostic: Candidate encryption methods should not require modifications to the data they encrypt. It should be possible to insert the encryption method into the transmission chain without modifying the application. This enables its use on platforms where it is infeasible to modify software on

existing processors, but additional hardware may be added to the communication chain.

Minimise transmission overhead: Energy and financial budgets already drive efficient composition of data into transmissions of minimal size. Candidates should minimise number of additional bytes required to transmit messages, and not rely on message padding to improve security. They should be flexible in their message framing, allowing cryptographic metadata to be spread over multiple transmissions if necessary.

Be tolerant to lost messages, and not reliant on persistent shared state: Candidates should be resilient to loss of messages within a sequence, or messages transmitted out-of-order. All messages must be independently decryptable. Candidates should avoid use of any centralised or time-sensitive infrastructure (i.e. for key exchange) where possible. After initial distribution of keys, it should be possible for MAS to securely communicate with multiple recipients following months without communication.

Be multi-party: It must be possible for multiple recipients to decrypt messages, without centralised infrastructure or sharing a single decryption key - supporting scenarios such as illustrated in Figure 1.

Authenticate messages: The sender of the encrypted message must be authenticated, and it must be possible to validate that the message was not altered mid-transmission (protection from false data injection).

Protect from Replay Attacks: Candidates must enable detection and filtering of duplicate message transmissions without preventing the genuine re-transmission of identical data: Allowing authentic transmission of the same encrypted command multiple times, whilst resilient against message interception and retransmission by third parties.

Be computationally efficient: Candidates must provide a sufficient level of security, but ideally still be practical to implement on micro-controllers.

Use standardised cryptographic primitives: Candidates must use cryptographic algorithms recommended by standards bodies such as the National Institute of Standards and Technology (NIST) and the Internet Engineering Task Force (IETF).

Be widely-used, well-supported, openly available, not restricted: Candidates must be free to use, under open-source licenses. The cryptography they use must not be patent-encumbered, nor have export restrictions preventing global use. Widely-used methods have increased likelihood of inclusion in well-tested, well-supported programming libraries, easing their use by non-experts.

IV. LITERATURE REVIEW

Dynamic Compact Control Language (DCCL) is a lightweight binary format with a focus on AUV acoustic and satellite communications. Previous versions of DCCL included encryption [12], but version 3 onwards removes any mention of that feature [8]. DCCL used the AES cipher [14] with a SHA-256-hashed pass-phrase as the secret key, and a hash of a timestamp in the message header used as an initialisation vector. This was reasonable for encryption of

real-time acoustic communication; allowing multiple devices to communicate with each other using a shared passphrase, but less suitable for over-the-horizon messaging: the SHA-256 hash of the pass-phrase is trivial to compute, making it vulnerable to offline brute-force attacks.

We found no other reproducible message encryption protocols specifically for AUVs, and limited work relating to application-layer encryption of Iridium satellite messaging [15]. We looked to related domains with similar properties to inform the design of a new protocol.

General Internet Protocols: Datagram Transport Layer Security (DTLS) [16] is well-documented, standardised, in widespread use, well-supported by software libraries, and very generalised. As such, it would require least effort to use. The Iridium SBD messaging protocol can be considered parallel to the User Datagram Protocol (UDP): datagrams sent from source to target addresses without any connection establishment process. DTLS applies TLS concepts to UDP. We found it unsuitable for our requirements due to its reliance on hand-shake messages and in-band encryption certificate transmission to establish encryption state. Additionally, TLS supports a wide range of cipher algorithms, which increases its complexity and inflates the amount of header metadata required beyond our constraints.

Group Chat Messaging Protocols: Protocols used by WhatsApp, Signal, Facebook Messenger, et al. appear promising, as these are optimised for small message payloads, encryption for multiple recipients, and some tolerance to lossy connections. Messaging Layer Security (MLS) [17] uses a ‘Double-ratchet’ key rotation algorithm, providing a level of forward-secrecy. However, this protocols’ reliance on trusted centralised key distribution and messaging overheads again make it too verbose for our application. The use of additional pair-wise, ephemeral encryption keys regularly rotated within each conversation between two users requires an additional 16 bytes (when 256-bit AES keys are in use) per key exchanged.

Mesh Communication over Long-Range radio: Protocols designed for use in long-range packet radio communication (‘LoRa’) are suitable for transmitting small data packets hundreds of kilometres over slow and lossy links with very low energy requirements, comparable to Iridium SBD. Meshtastic [18] is one such protocol in active development and use by international enthusiast communities. It supports one-to-one and broadcast messaging with optionally encrypted payloads, with open-source firmware for low-power ESP32 and nRF52 microcontrollers. It uses public key cryptography and AES-256 encryption in block counter (CTR) mode, operating as a stream cipher which does not require message padding. The protocol does not currently support integrity or message authentication checks, and this must instead be implemented by applications within the encrypted message payload. Messages can be encrypted for a single recipient or ‘channel’ for which the public key is known, but it does not natively support encrypting a single message addressed to multiple recipients simultaneously. Data packets include 23 bytes of overhead, the majority of which relates to addressing, which is already

handled by the Iridium link layer in our domain. Adding an authentication tag to each message incurs an additional 16 bytes.

Satellite Communication and Deep Space Networking: Deep Space Networking and ‘Delay-Tolerant Networks’ (DTNs) [19] share characteristics with our domain; operating in a challenging and remote environment with strict energy budgets, transmitting with routine disruptions. Latency can be very high, and there is little room for error with misconfiguration of devices due to their remoteness. The latency favours networks that use ‘store and forward’ communication. Menesidou et al. [19] note this makes it challenging to employ cryptographic protocols designed for internet, given many rely on round-trip messaging to establish secure connections. Secure management and distribution of keys is challenging with physically inaccessible devices.

Bundle Protocol is an alternative to IP-based networking for DTNs, and BPsec defines a security layer on it. Birrane and McKeever’s [20] comprehensive review shows that encryption in deep space networking is in its infancy, a point reinforced by Dowling et al. [21], as BPsec has yet to undergo formal analysis. They propose a variant of BPsec which addresses vulnerabilities, but has not yet been standardised via a Request for Comment (RFC).

Underwater Acoustic Networks: Zhou et al. [22] cite recent breakthroughs in multi-modal security of underwater communication: combining security improvements at the hardware layer with cryptographic methods, allowing secure transmission of large datasets with a negligible error rate. However, we were unable to find information on what cryptographic protocols were used. They highlight encryption and authentication schemes which employ Elliptic Curve Cryptography, concluding the majority were unfit for the constraints of underwater acoustic communications or were theoretical, but finding PAuthKey [23] resilient to known cryptographic attacks and inconsistent communication links. We find PAuthKey too reliant on infrastructure for our purposes; it assumes a highly interconnected mesh network to relay information from centralised authorities.

V. THE FLATFISH ENCRYPTION PROTOCOL

Flatfish is a compact encrypted message protocol which aims to minimise transmission overhead whilst remaining practicably secure. The reference code is an open-source Python library¹. It uses cryptographic algorithms recommended by NIST and included in the IETF Transport Layer Security (TLS) standard, which are sufficiently lightweight to be used on modern embedded microcontrollers [24].

It employs a hybrid-encryption scheme [25] which uses asymmetric encryption (Elliptic Curve Diffie-Hellman (ECDH) [26] with 256-bit Curve25519 [27] keys) to generate a shared 128-bit key, which is then used to encrypt a 128-bit symmetric message key for one or more recipients. ECDH uses considerably shorter private and public keys compared to

¹<https://gitlab.com/nocacuk/mars-oss/libraries/flatfish>

Diffie-Hellman. The asymmetric keys are equivalent to SSH and GPG key pairs, i.e. long-lived identity keys which must currently be shared out-of-band. However, their short lengths make it feasible to do so within our constraints.

The symmetric message key is then expanded to 256 bits using the Hash-Based Key Derivation Function (HKDF) [28] with the SHA-256 algorithm and an optional ‘shared secret’ which can be of any length. This 256-bit key is then used to construct an encryption *session*. A one-time random value (‘nonce’) of 64 bits in length is generated and combined with an optional message count², to generate a longer 96-bit nonce via HKDF with SHA-256. The 256-bit symmetric key and the nonce are then used to initialise a ChaCha20 stream cipher, which encrypts the plaintext data and computes an AEAD tag using the Poly1305 hashing algorithm [29]. Each message must be encrypted with a new and unique nonce, but the symmetric key need only be generated and transmitted once per session. The duration of a session can be controlled by the application³.

Table I describes the binary message framing of Flatfish. The message key, encrypted once for each recipient, is prepended to the encrypted data. Each message is prepended with its Poly1305 hash (referred to as an Authenticated Encryption with Associated Data (AEAD) Tag) using the header information and the plaintext prior to encryption. This allows verification of the message integrity; the AEAD tag can be computed again at decryption time and must match the hash in the message header. The original 64-bit nonce, prior to its expansion to 96 bits, is also included in the message header. In addition to authenticating that a message has not been modified in transit, the AEAD tag can be used in the prevention of replay attacks [30]. A valid message sent multiple times will have a different nonce and thus a different tag, so messages received with previously-seen tags can be discarded. An optional timestamp can be included in the message header and as part of the Associated Data used for the tag. This enables the filtering of messages which are too old or too new, reducing the search space required in detecting whether a message AEAD tag has been seen before. This is beneficial for memory-constrained computing environments, as only the hashes for valid messages decrypted within a window around the current time need to be stored.

The encrypted message can be split into multiple frame types, containing the header section and one or more of the other sections detailed in Table I. This allows the encrypted messages to be spread across multiple ‘frames’ when transmitted within the higher constraints of underwater acoustics, requiring a minimum of 26 bytes (header plus preamble) - recipient keys, when transmitted individually, require one 19 byte frame each (header plus key section).

²The message count is not transmitted in-band, and is only incremented from 1 when multiple messages are encrypted using the same session (i.e., using the same message key).

³For Iridium SBD, we consider each message an independent session, generating a new key each time.

Encryption *sessions* can be spread over one or more frames. Initially transmitted keys can be transmitted at the beginning of a session, and successive frames need only include header, preamble and data.

Recipients iterate over the set of keys transmitted in the header, decrypting each with their private key and then using that to attempt decryption of the message, computing the AEAD tag of the decrypted data on each attempt. When the computed tag matches the transmitted header tag, a valid *session key* has been found, and that key may be used to decrypt all subsequent message until a new session is established.

VI. EVALUATION OF TRANSMISSION OVERHEADS

Table II compares the theoretical performance *Flatfish* with multiple recipients, in relation to the data unencrypted, and Meshtastic [18]. We calculated overheads using historical data from two previous representative Autosub deployments with different Iridium modem models: Autosub 5 on the JC257 expedition, and two Autosub Long Range AUVs on the BioCarbon project. We compared the following methods:

Meshtastic: assuming a message overhead of 39 bytes, including 16 bytes of message authentication data proposed as an update to the protocol. Only a single recipient is supported.

Flatfish, 1 recipient, Public Key Encryption (PKE) only: Without generating a per-message key for each recipient, we encrypted messages for a single recipient by initialising the cipher directly using the key output from ECDH, similar to Meshtastic’s mode of operation.

Flatfish, with per-message keys, for 1/3/5 recipients: A message key is generated and encrypted multiple times, using each recipient’s public key. Requires an additional overhead of 16 bytes per recipient.

We use these projected overheads to estimate the Iridium costs, time, and battery energy for transmission. Energy usage estimates are based on the theoretical data rates and energy uses calculated by [31]. We assume a data rate of 97 bytes per second, assuming an Iridium modem draws a continuous 10 Watts during transmission and reception of a message.

VII. DISCUSSION

For the Iridium 9523 modems with larger buffers, our results show marginal differences in overheads between the methods, even allowing up to five different recipients to independently decrypt messages (a 12.9% increase in data sent over the unencrypted case). However, the increase in overheads is much larger (60.5% increase in total data sent for five recipients over the unencrypted case) for the Iridium 9602 modem with its smaller buffer. This is caused by repetition of the encryption header when input messages must be split over multiple encrypted messages; when we consider one recipient with per-message keys, the overhead increases by only 24.9% over the unencrypted case.

A. Current Limitations

Nonce Weakness: We opted to use the original ChaCha20 implementation, using a uniformly random 64-bit nonce, as

TABLE I: The binary message framing of the Flatfish encryption format

Section	Header (2 bytes)					Keys (1 to 4,097 bytes)		Preamble (24 bytes)		Optional	Payload	
Field	Format identifier	Format version	Frame type	Last frame?	Used in AEAD tag?	No. of keys (N)	Session keys (1..N)	Nonce	AEAD Tag	Time-stamp	Length	Encrypted Data
Size & Data	8 bits, ASCII 'E'	3 bits uint, 1-8	3 bits uint, 0-7	1 bit, True / False	1 bit, True / False	8 bits, 0-256	(128 * N) bits	64 bits, random value	128 bits, message hash	24 bits, UNIX time	16 bits uint	≤ 65,536 bytes

* Currently, this is always 1, or 'true' – all frames of a message should be used in the calculation of the message's AEAD tag. This bit is reserved for future work.

TABLE II: Comparison of the projected overheads of the encryption methods, across two distinct deployments

Encryption Method	Per-Message Overhead ^a	Messages Sent ^b	Total Data Sent	Average Message Size	Usage fees (\$ USD)	Estimated Air-time	Estimated Energy (kJ)
JC257 (Autosub 5): Iridium 9602 modem, receive & send buffers of 270 & 340 bytes							
Unencrypted	0 B	2,306	405.42 KB	181 B	587.82	1:11:20	42.80
Flatfish (1 recipient, PKE only)	28 B	2,352	468.47 KB	204 B	599.55	1:22:26	49.46
Meshtastic (1 recipient, PKE only)	39 B	2,352	493.25 KB	215 B	599.55	1:26:48	52.08
Flatfish (1 recipient)	45 B	2,352	506.76 KB	221 B	599.55	1:29:10	53.50
Flatfish (3 recipients)	77 B	2,352	578.82 KB	253 B	599.55	1:41:51	61.11
Flatfish (5 recipients)	109 B	2,352	650.88 KB	284 B	599.55	1:54:32	68.72
BioCarbon (ALR-4, ALR-6): Iridium 9523 modems, receive & send buffers of 1,890 & 1,960 bytes							
Unencrypted	0 B	2,676	2.08 MB	814 B	3,254.78	6:14:09	224.49
Flatfish (1 recipient, PKE only)	28 B	2,676	2.15 MB	842 B	3,254.78	6:27:02	232.22
Meshtastic (1 recipient, PKE only)	39 B	2,676	2.18 MB	853 B	3,254.78	6:32:05	235.25
Flatfish (1 recipient)	45 B	2,774	2.19 MB	829 B	3,373.98	6:34:51	236.91
Flatfish (3 recipients)	77 B	2,774	2.27 MB	860 B	3,373.98	6:49:34	245.74
Flatfish (5 recipients)	109 B	2,774	2.35 MB	891 B	3,373.98	7:04:16	254.56

^a As the overhead of a full frame fits within the constraints of Iridium messaging, we only assess the overheads of frames containing everything but the optional timestamp (header, keys, preamble and payload).

^b Message sizes are capped to the maximum ingoing or outgoing message size for the Iridium modem installed on the respective AUVs: where messages would exceed the maximum payload size, we calculate how many additional iridium transmissions would be needed to transmit the same piece of data.

a calculated risk as RFC7539 [29] specifies a 96-bit nonce consisting of 32 bits of fixed data and a 64-bit counter. As a result, fewer messages can be sent using the same asymmetric key-pair (approximately 8,600 messages before the probability of an identical encryption output exceeds 0.0001%⁴). Nonce-reuse may allow an attacker to progressively recover plaintext from an encrypted stream via a dictionary attack. However, we are unlikely to reach 8,000 messages for a single deployment. A security review (see Section VIII) has since recommended adhering to the 96-bit nonce, at the expense of 4 bytes added overhead per message.

No Perfect Forward Secrecy: As no ephemeral keys are exchanged between parties during messaging, the protocol is vulnerable to private key leakage: If a recipient's private key is compromised, an attacker could decrypt all past and future messages sent to them, and authentically encrypt messages as them.

Key Distribution: As a result, pair-wise encryption is broken should any sender or recipient device be compromised. Securing private keys is a major operational challenge as security of the sender and recipient devices must also be accounted for. We recommend that asymmetric keys are routinely updated, and that private keys are not shared between multiple devices. Automated or manual key generation, distribution, rotation and revocation processes must be considered when using such encryption methods.

Computational Inefficiencies: Effort to decrypt messages scales linearly with the number of recipients, as each key in the header must be tried at the start of a session. For our purposes,

⁴ $\frac{\sqrt{2^{64}}}{0.0001 \times 50\%} = 8,589$ messages can be sent with a $\leq 0.0001\%$ probability of nonce re-use.

this is acceptable; the volume and rate of messages being maximum at once every few seconds (best-case acoustics) and 100 KB (best-case Iridium Certus).

B. Future Work

We have tested *Flatfish* in a lab setting, with messages transmitted between an Autosub and multiple Command and Control (C2) applications. Real-world testing in field trials is planned.

We have tested on modern ARM and X86 processors; re-implementation and performance testing on microcontrollers would inform whether *Flatfish* requires modifications for more constrained computing environments.

To increase security, future work may explore novel ways of using ephemeral keys, and in-band key management, whilst keeping overheads to a minimum.

Optimization of overheads can be made through more integration of the encryption protocol with the application data formats – metadata on device identity and message timestamps can be omitted from messages prior to encryption, where the encryption metadata provides that information adequately. For example, the Autosub AUVs use DCCL already; leveraging DCCL to provide the encryption header binary encoding and re-implementing field-level encryption of DCCL messages would be beneficial.

The next generation of Iridium satellites offer faster data rates of up to 1408 Kb/s downlink and larger message sizes of up to 100 KB via Iridium Certus, relaxing our encryption overhead constraints. As use of this service becomes more common, longer keys and nonces can be used, and using existing transport layer security protocols becomes more feasible. That said, recent publications show that SBD continues

be used widely, and both IMT and SBD are vulnerable to eavesdropping and false-data injection [32].

Rather than attempting to accommodate both satellite and acoustic encryption within a single protocol, we may explore having multiple variants of this encryption method; Iridium and higher-bandwidth acoustics allow for stronger encryption with greater overheads, whilst another variant could optimise for very constrained acoustics (i.e., buffers of ≤ 80 bytes).

Finally, we see value in working within an intersectional community of MAS platform developers, manufacturers of satellite and acoustic communications technology, and security researchers; developing encryption as part of an interoperable communications suite suited for AUV operations and onward. Given the collective work on Delay-Tolerant Networking resulted in RFC 5050 Bundle Protocol for use in Spacecraft, there is scope to collaborate on terrestrial-to-satellite and underwater communications protocols also becoming published RFC standards.

VIII. CONCLUSION

Encryption is one of the foundations of information security, reducing operational risks by ensuring messages are readable only by their intended recipient. For remote MAS, a balance must be made — encryption increases security but introduces computational and bandwidth overheads which impact strict battery budgets. We presented *Flatfish*, a compact lightweight encryption protocol with per-message keys that supports multiple recipients. For a single recipient, *Flatfish* requires only 45 bytes of overhead per message, enabling its use with even extremely limited bandwidths. Open research questions remain, including how to efficiently and securely distribute the long-term asymmetric encryption key pairs within collaborative scientific environments in a way which scales to many MAS and many operators. We encourage the community to collaborate with us to develop an open standard for MAS encryption, increasing the resilience of the critically important GOOS.

ACKNOWLEDGEMENTS

We thank Chris Leppard (CISSP, MCIIS), Callum Bryant and Niall Buckley from Digital XRAID Limited, who we contracted to independently verify the security of our protocol design and to identify limitations. Their report will be made available with our reference *Flatfish* implementation.

REFERENCES

- [1] A. Fischer, J. Gunn *et al.*, “Global Ocean Observing System 2030 strategy,” UNESCO IOC GOOS, Tech. Rep., 2020. [Online]. Available: <https://goosocean.org/document/24590>
- [2] GOOS Secretariat, OceanOPS, and GOOS Observations Coordination Group, “Global Ocean Observing System status report 2025,” GOOS, Tech. Rep. GOOS-310, 2025.
- [3] P. Testor, G. Meyers *et al.*, “Gliders as a component of future observing systems,” in *Proceedings of OceanObs’09: Sustained Ocean Observations and Information for Society*, vol. 2, 01 2010.
- [4] A. B. Phillips, R. Templeton *et al.*, “Autosub long range 1500: A continuous 2000 km field trial,” *Ocean Engineering*, vol. 280, p. 114626, 2023.
- [5] C. Gomez, S. M. Darroudi *et al.*, “On the energy performance of Iridium satellite IoT technology,” *Sensors*, vol. 21, no. 21, 2021.
- [6] G. West and E. Ceurstemont, “Optimising performance of a long range ultra short baseline tracking and telemetry system,” in *OCEANS 2018 MTS/IEEE Charleston*, 2018, pp. 1–6.
- [7] *APEX Profiler User Manual*, Teledyne Webb Research, 2017.
- [8] T. Schneider, S. Petillo *et al.*, “The dynamic compact control language version 3,” in *OCEANS 2015 - Genova*, 2015, pp. 1–7.
- [9] S. McCammon, T. P. Welch *et al.*, “Onboard autonomy system for the Slocum Glider,” in *OCEANS 2019 MTS/IEEE Seattle*, 2019, pp. 1–7.
- [10] *Slocum Glider G3S New Processor Guide*, Teledyne Webb Research, 2021.
- [11] G. Salavasidis, T. Saeharaseelan *et al.*, “Squads of Adaptive Robots (SoAR): an architecture for operating heterogeneous fleets of marine robots,” in *OCEANS 2023 - Limerick*, 2023, pp. 1–9.
- [12] T. Schneider and H. Schmidt, “The Dynamic Compact Control Language: A compact marshalling scheme for acoustic communications,” in *OCEANS’10 IEEE Sydney*, 2010, pp. 1–10.
- [13] S. Ruoti, J. Andersen *et al.*, “Why Johnny still, still can’t encrypt: Evaluating the usability of a modern PGP client,” 2016.
- [14] National Institute of Standards and Technology (US), “Advanced encryption standard (AES),” National Institute of Standards and Technology (U.S.), Washington, D.C., Tech. Rep., May 2023.
- [15] T. Morgan and J. Lukowski, “Secure satellite communication & beacon tracking device,” in *2009 IEEE Conference on Technologies for Homeland Security*, 2009, pp. 188–192.
- [16] E. Rescorla and N. Modadugu, “Datagram Transport Layer Security Version 1.2,” RFC 6347, Jan. 2012.
- [17] R. Barnes, B. Beurdouche *et al.*, “The Messaging Layer Security (MLS) Protocol,” RFC 9420, Jul. 2023.
- [18] “Meshtastic documentation,” [Accessed 11-06-2026]. [Online]. Available: <https://meshtastic.org/docs/>
- [19] S. Menesidou, V. Katos, and G. Kambourakis, “Cryptographic key management in delay tolerant networks: A survey,” *Future Internet*, vol. 9, p. 26, 06 2017.
- [20] E. J. Birrane and K. McKeever, “Bundle Protocol Security (BPsec),” RFC 9172, Jan. 2022.
- [21] B. Dowling, B. Hale *et al.*, “Cryptography is rocket science: Analysis of BPsec,” *IACR Communications in Cryptology*, vol. 1, no. 4, 2025.
- [22] Q. Zhou, Q. Ye *et al.*, “Cryptography-based secure underwater acoustic communication for UUVs: A review,” *Electronics*, vol. 14, no. 12, 2025.
- [23] P. Porambage, C. Schmitt *et al.*, “Pauthkey: A pervasive authentication protocol and key establishment scheme for wireless sensor networks in distributed iot applications,” *International Journal of Distributed Sensor Networks*, vol. 2014, 07 2014.
- [24] K. Dokic, T. Mesic, and M. Cobovic, “Analysis of available microcontroller computing resources for the application of some cryptographic hash functions on different internet of things platforms,” in *Multimedia Communications, Services and Security*, A. Dziech, W. Mees, and A. Czyżewski, Eds. Cham: Springer International Publishing, 2020, pp. 168–181.
- [25] I. A. Shoukat, I. A. Shoukat *et al.*, “A generic hybrid encryption system (hes),” *Research Journal of Applied Sciences, Engineering and Technology*, vol. 5, no. 9, pp. 2692–2700, 2013.
- [26] E. Barker, L. Chen *et al.*, “Recommendation for pair-wise key-establishment schemes using discrete logarithm cryptography,” National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep., Apr. 2018.
- [27] D. J. Bernstein, “Curve25519: New diffie-hellman speed records,” in *Public Key Cryptography - PKC 2006*, M. Yung, Y. Dodis *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 207–228.
- [28] H. Krawczyk, “Cryptographic extraction and key derivation: The HKDF scheme,” Cryptology ePrint Archive, Paper 2010/264, 2010. [Online]. Available: <https://eprint.iacr.org/2010/264>
- [29] Y. Nir and A. Langley, “ChaCha20 and Poly1305 for IETF Protocols,” RFC 8439, Jun. 2018.
- [30] A. D. Jurcut, T. Coffey, and R. Dojen, “Design guidelines for security protocols to prevent replay & parallel session attacks,” *Computers & Security*, vol. 45, pp. 255–273, 2014.
- [31] A. Bodenmann, D. O. B. Jones *et al.*, “Remote awareness of image quality for multiweek shore-launched AUV surveys,” *IEEE Transactions on Field Robotics*, vol. 2, pp. 147–164, 2025.
- [32] E. Jedermann, P. Kulpinski *et al.*, “Systematic security analysis of the Iridium satellite radio link,” 2026.