Non-interactive Identity-based Key Establishment—the best choice for Underwater Wireless Sensor Networks

Qian Li, Husen Muhammad Umer, Rui Feng

1 Underwater Wireless Sensor Networks

Wireless sensors are low-power wireless motes with a tiny amount of CPU and memory [1]. Batteries are the sole power supply units to a sensor. Hence, energy conservation is one of the most important aims of a wireless sensor network. Both computation and communication consume power, but the latter becomes vital when the wireless sensor network is put underwater. Since water attenuates radio waves significantly, a radio frequency transceiver is no longer practical for underwater wireless sensor networks. Instead acoustic modems are used in underwater environment. However, this kind of modems consumes much more energy than radio transceivers when they transmit data. As a result, limiting the amount of data transmitted among sensors becomes crucial for underwater wireless sensor networks.

2 Non-interactive Identity-based Key Establishment (NIKE)

To secure an underwater wireless sensor network, certain cryptographic algorithms should be deployed. A desired algorithm should have a property that consumes as less power as possible. At the same time, it should also have scalability, efficiency, resilience, and extensibility. NIKE is a good candidate. It represents a family of protocols, which compute the pair-wise key by only exchanging each other’s identities. NIKE is a new kind of key agreement protocol for symmetric cryptography. It is built on the mathematical foundation of bilinear map or pairing.

Bilinear pairings or simply pairings is a nick name of bilinear maps. A bilinear map is a function of two arguments that is linear in each. Integer multiplication is an example of bilinear maps [2]. A formal definition follows.

Let $R$ be a ring, and let $M$, $N$ and $P$ be modules over $R$.
A function $f: M \times N \rightarrow P$ is said to be a bilinear map if
1) for each $m \in M$ the function $f_m: N \rightarrow P$ defined by $f_m(n) = f(m, n)$ for all $n$. 


\(\epsilon N\) is linear, and

2) for each \(n \in N\) the function \(f_n : M \to P\) defined by \(f_n (m) = f (m, n)\) for all \(m \in M\) is linear [3].

In other words, if we hold the first entry of the bilinear map fixed, while letting the second entry vary, the result is a linear operator and vice versa.

A bilinear map \(\mathbf{e} : G \times G \to G_1\) has two properties

- Bilinearity: \(\mathbf{e}(g^a, g^b) = \mathbf{e}(g, g)^{ab}\) for all integers \(a, b\)
- Distorted: \(\mathbf{e}(g, g) \neq 1\) in \(G_1\)

Where \(G = \langle g \rangle\) is a cyclic group of order \(q\) for prime \(q > 3\)

The basic idea of NIKE is that there is a Key Generation Center (KGC) in a wireless sensor network, usually the base station. The base station computes the secret keys for each node based on their unique string IDs and sets up the whole network.

SOK (Sakai, Ohgishi, and Kasahara) is the earliest non-interactive identity-based key agreement protocol. When node A wants to communicate with node B, it first sends its ID to B, and B replies this message with B’s ID. After A received B’s ID, it computes the pair-wise key, \(K_{AB}\), using the following formula

\[ K_{AB} \leftarrow \mathbf{KDF} \left( \mathbf{e} \left( H \left( \text{id}_B \right), \text{sk}_A \right) \right) \]

Where, \(\text{id}_B\) is B’s ID, \(\text{id}_A\) is A’s ID, \(\text{sk}_A = H \left( \text{id}_A \right)^z\) is A’s secret key generated by the base station, \(H\) is a hash function, \(z\) is the master secret key only known to the base station, \(\mathbf{e}\) is a bilinear map, and \(\mathbf{KDF}\) is a Key Derivation Function, which can be implemented with SHA-160 for example. Due to the bilinearity of \(\mathbf{e}\), node B can obtain \(K_{AB}\) by doing the same using its own secret key \(\text{sk}_B\) and A's ID \(\text{id}_A\). This can be proved as follows.

\[ \mathbf{e} \left( H \left( \text{id}_B \right), \text{sk}_A \right) = \mathbf{e} \left( H \left( \text{id}_B \right), H \left( \text{id}_A \right)^z \right) = \mathbf{e} \left( H \left( \text{id}_B \right)^z, H \left( \text{id}_A \right) \right) = \mathbf{e} \left( \text{sk}_B, H \left( \text{id}_A \right) \right) = \mathbf{e} \left( H \left( \text{id}_A \right), \text{sk}_B \right) \]

With SOK, a node only needs to send one packet to its peer to compute the pair-wise key, \(K_{AB}\), shared between them. The packet sent contains the node’s ID, protocol ID, message ID, checksum, packet headers and footers, which add up to 384 bits in total. In the meanwhile the node receives a packet in the same format from its peer. This incurs 191.99 mJ’s communication cost when a UWM2000 modem is deployed. On the other hand, the hash and pairing computations make up a 309.39 mJ’s energy cost.
3 NIKE V.S. traditional KMS frameworks

Since asymmetric-key algorithms are computationally intensive and extremely power consuming, they are prohibited in wireless sensor networks. Then the only concern left is to choose a proper key agreement protocol or Key Management System (KMS) for symmetric-key algorithms, so that it expends the least possible energy. Traditional KMS frameworks can be classified into two categories — Public Key Cryptography (PKC) based frameworks, and Symmetric Key (SKE) based frameworks. And the latter can further be divided into three categories: mathematical-based, key pool-based, and negotiation-based frameworks, as shown in figure 1.

![KMS Frameworks Diagram](image)

**Fig. 1. KMS frameworks for WSN**

In the Mathematical-based framework, the mathematical concepts are used to calculate the common pair-wise key. In the Key Pool-based framework, each node stores a subset of keys (the key-chain) retrieved from a large number of pre-calculated keys (the key-pool). And in the Negotiation-based framework, sensor nodes exchange information related to their pair-wise keys after the deployment of the network.

3.1 NIKE V.S. PKC-based frameworks

Public Key Cryptography based key agreement frameworks, such as Elliptic Curve Menezes-Qu-Vanstone (ECMQV), are less computationally intensive, but
have more communication overhead. ECMQV works as follows. When node A wants to communicate with node B, it first sends its certificate together with its ephemeral key \( E_A \) to node B, upon receiving B’s certificate and ephemeral key \( E_B \), A first verifies B’s certificate and then does the following.

1: \( m \leftarrow \lfloor \log_2(n) \rfloor / 2 \) \{ \( m \) is the half bitlength of \( n \) \}
2: \( u_A \leftarrow (u_x \mod 2^m) + 2^m \) \{ \( u_x \) is the \( x \)-coordinate of \( E_A \) \}
3: \( s_A \leftarrow (y_A + u_A x_A) \mod n \)
4: \( v_A \leftarrow (v_x \mod 2^m) + 2^m \) \{ \( v_x \) is the \( x \)-coordinate of \( E_B \) \}
5: \( z_A \leftarrow s_A u_A \mod n \)
6: \( K_{AB} \leftarrow KDF(E_B^{s_A} \cdot p_k A \mod n) \)

Where, \( n, g \) are input parameters, \( x_A, y_A \) are A’s secret keys, \( x_B, y_B \) are B’s secret keys, \( p_k_A = g^{x_A} \) and \( p_k_B = g^{x_B} \) are A’s and B’s public keys, respectively, \( E_A = g^{y_A} \) and \( E_B = g^{y_B} \) are the ephemeral keys of A and B, respectively, and KDF is the Key Derivation Function. \( K_{AB} \) is the pair-wise key shared between A and B.

B can get \( K_{AB} \) by doing the same except replacing \( x_A, y_A \) with \( x_B, y_B \) and \( E_B, p_k_B \) with \( E_A, p_k_A \).

To exchange certificates and ephemeral keys as well as other necessary information, a node has to send and receive 1410 bits during the key establishment process. This means a 704.98 mJ’s communication cost when UWM2000 modem is used. On the computational side, a node has to verify its peer’s signature and do one multi-exponentiation computation, \( mexp(2) \), one exponentiation computation, \( exp \), and two square roots computation, \( sqrt \). So, the total cost is

\[
2mexp(2) + 1exp + 2sqrt (+trans. 1410 bits + recep. 1410 bits)
\]

Whereas the energy and transmission cost of SOK for one node amounts to:

\[
1expG + 1pairing (+trans. 384 bits + recep. 384 bits)
\]

Here we could see that totally 384 bits are used for the packets in SOK which is much less than the ECMQV. The energy cost of ECMQV is less than that of SOK (107.26mJ and 309.39mJ respectively). But the communication overhead of ECMQV in underwater wireless sensor networks is much higher than that of SOK (2291.23 mJ and 623.99 mJ, respectively, when using UWM4000). And the total cost of SOK is less than that of ECMQV when it works in underwater environment, as shown in table 1.
### 3.2 NIKE V.S. SKE-based frameworks

Most of the KMS belonging to the three main SKE-based frameworks must exchange certain information to derive a pair-wise key, which calls for more bandwidth and energy usage than NIKE protocols for underwater environments. Although there are some optimized protocols could be used by the SKE-based frameworks to reduce the communication overheads, like even only the ID of a node needs to be exchanged, these protocols have certain disadvantages to discourage their usage.

For example, in the Key Pool-based KMS, the connectivity and the resilience are not good as well as the scalability and the extensibility of the mathematical schema are not satisfied in the Mathematical-based KMS. There are also some security problems.

In contract with these factors listed above, the NIKE protocols like SOK could offer better key connectivity, scalability, network resilience as well as no size restrictions. Also, if an adversary captures a sensor node, he/she could only get information related to that node, and is unable to eavesdrop the communications between any other nodes. So NIKE outperforms other KMS frameworks in the security field too.

### Conclusion

In conclusion, Non-interactive Identity-based Key Establishment protocols are computationally intensive due to pairing computations, but require less communication among sensors. It is this attribute that makes NIKE protocols very suitable for underwater wireless sensor networks.
5 References