Chebyshev Polynomial and Private Blockchain Based Cross-Domain Authentication Protocol for IoD Networks

Cong Pu[‡] Kim-Kwang Raymond Choo[¶] Image Bhattarai[‡]
[‡]Oklahoma State University, USA. Email: cong.pu@outlook.com; image.bhattarai@okstate.edu
[¶]The University of Texas at San Antonio, USA. Email: raymond.choo@fulbrightmail.org

***Abstract*—With the maturity of Internet of Things (IoT), one of its descendants, Internet of Drones (IoD) has reached far beyond its proposers' vision in the recent decade. The IoD paradigm inherits the advantages of its predecessor, however, it also has its own unique challenges due to the drone's limited resources as well as the large scale deployment of services. As drones might be deployed for critical missions that span over a wide geographical area, the security and feasibility concerns are raised when drones are communicating with the ground stations located in different domains. Lately, blockchain has quickly become the preferred technique to realize the cross-domain communications in the IoD environment. Nonetheless, the current schemes either implement authentication and key agreement with resource-hungry operations, do not provide all required/vital security guarantees, or have inherent security flaws. To tackle the abovementioned issues, we propose a Chebyshev polynomial and private blockchain based authentication protocol (hereafter referred to as *polyBlock*) for cross-domain communications in the IoD environment. In the *polyBlock*, the Chebyshev polynomial technique is adopted to validate the identity of drone and negotiate the session key with the ground station, while the private blockchain is utilized to store the drone's cryptographic information. Through carrying out the security validation on *polyBlock* using the automated tool AVISPA, we claim that the *polyBlock* is completely free of security design flaws and is capable to operate safely in the adversarial settings. In addition, we implement the *polyBlock* and two benchmark schemes in the Eclipse-based simulation environment, and measure their performance in terms of execution time and communication overhead. Based on experimental results, we conclude that the *polyBlock* can provide more superior performance than its counterparts.**

***Index Terms*—Security, Privacy, Authentication Protocol, Chebyshev Polynomial, Blockchain, Internet of Drones**

I. Introduction

The popularity of drones, or officially known as unmanned aerial vehicles (UAVs), has quickly grown as the cost of drones has become more affordable in the civilian domain. Yet when drones were first invented and came into view, they were only used for battlefield surveillance missions. In the last decade, with the rapid development of lithium iron phosphate battery, ultra-thin microchip, advanced wireless communications, as well as carbon fiber materials, the types of drones are becoming more diverse and their capabilities have been significantly improved. Today, drones serve the military industry around the world, and have played a critical role in providing intelligence, surveillance, as well as delivering precision-guided munitions. For instance, the market size of global military drones is expected to reach \$16 billion in 2023 [1] due to the continuous government funding for military drones to enhance efficiency in military operations.

According to the concept of joint force in ''2018 Joint Concept for Integrated Campaigning'', drones have come to revolutionize warfare and are being pushed to carry out the mission from a global perspective, instead of focusing only on a specific and small geographic area. To realize the vision of joint force and fully exploit the potential of drones in modern warfare, the emerging Internet of Drones (IoD) paradigm [2] has to be adopted to integrate aerial and ground communications for military applications. Based on the idea of IoD paradigm, the military operation area can be virtually partitioned into a set of physical domains, where one or more ground stations (or mobile communications vehicles) are deployed to communicate with nearby drones for mission-specific operations [3]. Compared to the ancestor of IoD, vehicular network, where the movement of vehicles is constrained by terrain, the drones in the IoD paradigm are endowed with greater freedom of movement. In addition, the drones' activity arena is the airspace, thus, they are otherwise difficult to spot.

Although the IoD paradigm promises to transform the way drones are being used, security and privacy issues still continue to plague engineers and researchers when the IoD paradigm integrates with military applications. To be specific, security and privacy are not considered in the initial design of IoD paradigm, but regarded as add-on properties [4]. Consequently, the adversary can launch various security attacks (e.g., flooding attack [5]) to compromise the IoD networks through taking advantage of gaps in the IoD design. For example, nano ''bug'' drones can be deployed to spy targets (e.g., taking images and/or videos) up to 1.3 miles away in the battlefield. The images and videos captured by nano ''bug'' drones are deservedly regarded as critical information, thus, safely delivering these information to the operator or the ground station will determine a mission's success or failure. In addition, the military drone operation might be conducted in a wide geographical area. Thus, it is unavoidable that the drones need to communicate with the ground stations located in the different domains, and secure communication across domains becomes a reality that we need to face.

In the recent past, research scientists from academia and industry have put forth a moderate effort to design authentication protocols for the IoD networks, where the drone and the ground station will first verify each other's identities and exchange information with the negotiated session key. However, the existent approaches either have inherent security design flaws (i.e., suffering from drone impersonation attack [6]), or do not provide the required/vital security guarantees (i.e., lack of anonymity feature [7]). Most importantly, they do not support cross-domain authentication that the drones authenticate with the ground stations located in different physical domains. Nonetheless, how to realize the process of authentication and key agreement between a drone and different ground stations in the IoD environment is a non-trivial problem. Recently, some researchers adopt blockchain technique to resolve the issue of cross-domain authentication [8], [9]. Unfortunately, these blockchain-based security protocols require the frequent update of cryptographic information stored in the blockchain, which incurs a very high communication and computation overhead. To sum up, what has been lacking in the current theory is a secure and lightweight cross-domain authentication protocol that adopts resource-friendly operations and meets all critical security requirements. The realization of such a novel security protocol would be unprecedented because the similar technique is not currently available in the IoD community, and the proposed work will fill this research gap.

Motivated by the above analysis, in this paper we propose a novel cross-domain authentication protocol to address the abovementioned challenging issues in the IoD environment. In summary, our major contribution is shortly summarized below:

- We propose a Chebyshev polynomial and private blockchain based authentication protocol (hereafter referred to as *polyBlock*) for cross-domain communications between the drones and the ground stations in the IoD environment. The *polyBlock* is realized by lightweight operations. In addition, the anonymity of drone will be changed after each communication session.
- We verify and prove the security of *polyBlock* through the security verification using the automated tool AVISPA [10]. Moreover, we build a simulation environment and conduct experimental study in terms of execution time and communication overhead. The *polyBlock* can meet all pre-defined security requirements while providing more superior performance than its counterparts.

II. Related Work

In [8], a blockchain assisted cross-domain authentication scheme is proposed for Internet of Drones (IoD) systems. In order to enable identity federation, the authors adopt threshold signature technique for the collaboration between different domains. When the entities from different domains want to communicate, the authentication can be realized through the smart contract. However, the authentication process is based on bilinear pairing which incurs a substantial computation overhead, especially to the resource-constrained drones. In addition, various kinds of information such as registration, cryptographic, and security audit information are stored in the blockchain, which will inevitably increase the size of each block and cause a throughput bottleneck. The authors in [11] propose a physical unclonable function (PUF) based authentication protocol for drone networks. The rationale behind the usage of PUF is to defend against drone tampering and cloning attacks. Moreover, the authors implement the authentication between drones from different domains. In the harsh environment, however, the reliability of PUF is in doubt because the PUF might not re-generate the same response with the same challenge. Thus, the pre-negotiated secret information based on the PUF response cannot be restored and the entire authentication process will fail. The proposed protocol supports the anonymity of drone and the adversary is unable to obtain the real ID of drone through monitoring the wireless communication. However, the drone's pseudonym is not being updated frequently. Thus, there is still a chance for the adversary to track the drone that uses the same pseudonym all the time. Recently cross-domain authentication for Internet of Things (IoT) networks has also been studied. For example, in [12] the authors argue that bilinear pairing is a high complexity technique and might not be suitable for IoT devices. Thus, they decide to use symmetric polynomial to realize the authenticated key agreement between the IoT device and the IoT server that are from different domain. However, the above approach does not endow IoT devices with any physical security features (e.g., tamper-resistant design) which make IoT devices vulnerable to physical attacks. This is because IoT devices are usually deployed in an unattended area and the adversary might approach and compromise them through probing attacks.

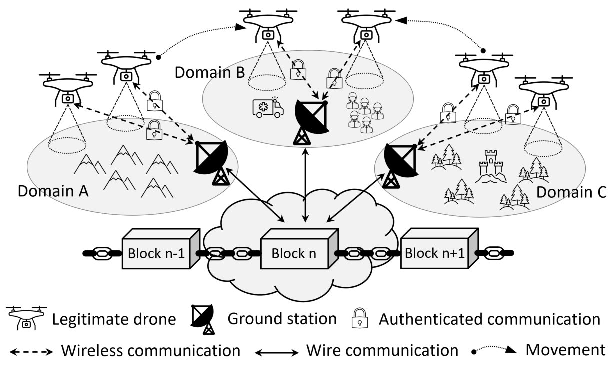


Fig. 1. System model.

III. System Model, Adversarial Model, and Security Requirements

A. System and Adversarial Models

As shown in Fig. 1, the operational area is partitioned into three adjacent physical domains (i.e., domain A, B and C). In each domain, there is one ground station deployed to serve as a radio communication facility. Multiple ground stations can also be deployed and inter-connected in each domain. In our system, another major participant is drones. Unlike the ground station that is considered as the credible entity, drones are regarded as untrusted participants. Thus, the communication between the drones and the ground stations requires mutual authentication and the follow-up critical information exchange should be conducted over an encrypted channel. In addition, the drones are designed with the built-in physical unclonable function (PUF) primitive so that the secret information can be dynamically calculated, rather than being cached directly in the storage unit. In order to become deployable, each drone needs to first register with one ground station to negotiate authentication information. Since drones might be responsible for the critical mission over multiple physical domains, a private blockchain is used to store drones' identities and authentication information so that the authenticated communication can be realized between different domains. The ground stations from all domains form a private blockchain network.

In this paper, we consider the well-known Dolev-Yao adversarial model [13]. In a nutshell, the primary goal of the adversary is to stealthy eavesdrop the communication, continuously attempt to obtain the secret information (e.g., cryptographic value or session key), and then maliciously compromise the communication. The wireless channel is an open-access medium, thus, the adversary can easily monitor the communication. However, this adversary behavior can be easily defeated if the critical information is being exchanged over an encrypted channel. In addition, the drone might enter into the adversarial zone, thus, there is a chance for the opponent to physically capture it with the special equipment and attempt to retrieve critical cryptographic information to compromise the future communication. But the adversarial attempt might not succeed at all. This is because the drone is built with the physical protection feature, which is PUF primitive, the physical probing attack will change or even destroy the PUF mapping which causes the same response not being re-generated with the same challenge. Thus, it is reasonable to assume that a maliciously compromised and re-programmed drone does not exist in the system. Other attacks such as jamming attacks [14], [15] can also be launched by the adversary, however, they are outside the scope of this paper.

B. Security Requirements

In this paper, we specify the following requirements to be met by the *polyBlock*. First, the *polyBlock* should allow a drone and a ground station to perform identity verification and negotiate a secret session key to establish a secure communication channel. Second, the *polyBlock* should use the pseudonym of drone, rather than the real identity, for the communication. In addition, the pseudonym of drone should be changed after each communication session so that the adversary cannot track the drone with the identification information. Third, the *polyBlock* should guarantee the confidentiality and integrity of exchanged messages between the drone and the ground station. Finally, the *polyBlock* is expected to outperform other benchmark schemes in terms of various performance metrics.

IV. The Proposed Authentication Protocol

A. The Design of Physical Unclonable Function

In this paper, we assume that the hardware-based security primitive is available to protect drones from physical attacks such as tampering and cloning attacks. One of the widely adopted hardware-based security primitives would be physical unclonable functions, or in short PUFs. Since the focus of this paper is the authentication protocol, rather than the actual design of PUF, we will simulate the PUF as a one-way function. To be specific, we represent the PUF as $res = F_{puf}(che)$, where the function input che is called challenge, and the

Algorithm 1: Response Generation Algorithm *rGen*

```
Input: Modulus n; Challenge che
1 Function rGen(n, che):
      /* $\overset{\circledast}{\longleftarrow}$ denotes sampling                         */
      /* $\oplus$ denotes exclusive OR function              */
      /* $\mathbb{Z}_n$ denotes the set of remainders in
         arithmetic modulo n                             */
2     $O = F_{puf}(che)$;
3     $res \overset{\circledast}{\longleftarrow} \mathbb{Z}_n$;
4     $S = O \oplus ECC(res)$;
5     return $\{res, S\}$;
```

Algorithm 2: Response Restore Algorithm *rRes*

```
Input: Challenge che; Helper string S
1 Function rRes(che, S):
2     $O' = F_{puf}(che)$;
3     $res = D_{er}(S \oplus O')$;
4     return res;
```

function output res is named response. The che along with the matching res is known as challenge-response pair, or just CRP. The most attractive characteristics of PUF is uniqueness, where the identical challenge will cause the PUF to output the same response in the normal situation. However, if a minor change is made in the challenge, we can expect a totally distinct response generated by the same PUF.

Recently, many researchers have chosen the PUF to produce the specific cryptographic information, so that the devices do not need to store the critical information directly in the storage unit. However, it is worth mentioning that the hash environment will make the PUF extremely unpredictable and unstable. Therefore, the same response might not be re-produced by the PUF with the same challenge. In order to make the PUF stably work in the severe circumstance, we develop an error correction code and a fuzzy extractor and integrate them with the PUF. First, an algorithm, called *rGen*, is created to generate the response. The *rGen* algorithm, as shown in Algorithm 1, is designed to produce a set $\{res, S\}$. Here, *res* is the response in the CRP, which is the value to be regenerated by the PUF. *S* is a specific string which is fed into the PUF and help to re-generate the CRP response *res*. The error correction code [16] is adopted to fix up to *x* bit errors in the response of CRP. Second, we design an algorithm $rRes$ to restore the same response, where the major operations are shown in Algorithm 2. With the $rRes$ algorithm, the PUF can re-generate *res* with the specific string *S* and the error decoding algorithm D_{er}, even if the PUF produces an output O' that differs from the original output O by at most *x* bits.

B. The Design of Private Blockchain

In [17], the authors develop a blockchain framework for resource-constrained IoT systems. Motivated by their work, this paper adopts a private blockchain system to store drones' cryptographic information and realize the cross-domain authentication. For the private blockchain, the nodes (i.e., the ground stations in this paper) who want to participate in the network need to obtain permissions first. In other words, each ground station is well-known by all other ground stations in the IoD network, and it is unnecessary to compete for generating and appending a block to the blockchain system. In each physical domain, the ground station has access to a copy of blockchain ledger which will provide the necessary information to authenticate the drone. When a ground station registers new drones to join the IoD network, it generates a block with the cryptographic information of new drones and waits for its turn to broadcast the block to other ground stations. After performing the block validation, the block is appended to the blockchain system by other ground stations.

C. The Proposed polyBlock Protocol

A drone ID_i is deployed for a military operation conducted in a large geographical area. The military operation area is further divided into a set of adjacent domains. For simplicity, one ground station G_j is located in each domain $\mathbb{D}_j$. When the drone ID_i collect enough data (i.e., the data storage unit is full), it submits the observational data to a nearby ground station. Note that the ground station that the drone ID_i submits the observational data to might be located in a different domain where the drone ID_i starts its mission. In summary, the *polyBlock* consists of three steps: (i) system initialization; (ii) drone registration; and (iii) authentication and key establishment.

1) System Initialization: In this step, a set of system parameters and functions are generated and published. Here, we assume that the system initialization is carried out by one ground station, e.g., G_j.

1) G_j selects a Chebyshev polynomial $T_{(n)}(x)$, where n is an integer and $x \in$ [-1, 1].
2) G_j specifies a one-way hash function H, where H:{0,1}*→{0,1}^{m}, where m indicates the number of bits.
3) G_j selects its private key PR_j^G and calculates its public key $PU_j^G = T_{(PR_j^G)}(x)$. G_j keeps PR_j^G secretly, and PU_j^G is shared with drones during the process of drone registration.
4) G_j publishes all system parameters as $\{T_{(n)}(x),\ H\}$ so that all participants can access them.

2) Drone Registration: In this step, the drone ID_i registers with the ground station G_j in the following steps.

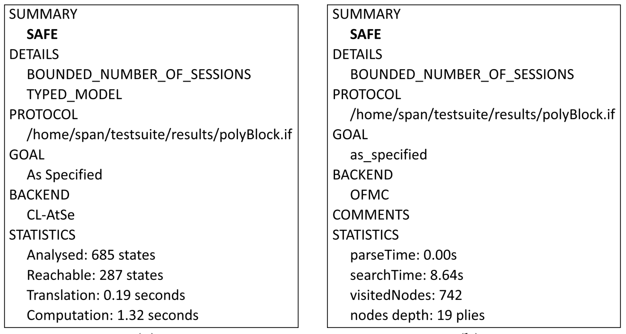
1) ID_i chooses its real identity RID_i and PUF challenge che_i arbitrarily, and then calculates the corresponding response $res_i = F_{puf}(che_i)$.
2) ID_i uses its PUF response res_i to calculate the public token $PT_i = T_{(res_i)}(x)$.
3) ID_i selects a random number $r_i^{ts_\triangledown}$. Here, $ts_\triangledown$ is the timestamp which is used to generate a pseudonym.
4) ID_i calculates its pseudonym $PID_i^{ts_\triangledown} = H(RID_i \parallel PT_i \parallel r_i^{ts_\triangledown})$.
5) ID_i shares $\{RID_i,\ PT_i\}$ with G_j via a secure channel.
6) ID_i stores its RID_i, che_i, and $r_i^{ts_\triangledown}$ in the memory. For security reasons, ID_i does not store the critical cryptographic value res_i directly. In order to save the storage space, ID_i can dynamically re-calculate res_i, PT_i, and $PID_i^{ts_\triangledown}$, instead of storing.
7) G_j puts $\{RID_i,\ PT_i\}$ into a block and adds it into the private blockchain. Here, a joint consensus mechanism [18] can be used to select the miner ground station. As a private blockchain, the ground station is the only participant that is allowed to access the ledger.

Algorithm 3: Authentication and Key Establishment

```
/* Drone initiates authentication */
Function DroneAuthReq($G_k$):
    $res_i = F_{puf}(che_i)$; $PT_i = T_{(res_i)}(x)$;
    $PID_i^{ts_\triangledown} = H(RID_i \parallel PT_i \parallel r_i^{ts_\triangledown})$;
    $r_i^{ts_\triangledown+1} \leftarrow$ RandNum(); $PU_i^{pol} = T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(x)$;
    $PR_i^{pol} = T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(PU_k^G)$;
    $H_{i,1} = RID_i \oplus H(PID_i^{ts_\triangledown} \parallel PU_i)$;
    $H_{i,2} = r_i^{ts_\triangledown+1} \oplus H(PID_i^{ts_\triangledown} \parallel PU_i \parallel RID_i)$;
    $H_{i,3} = H(PID_i^{ts_\triangledown} \parallel PU_i \parallel RID_i \parallel r_i^{ts_\triangledown+1} \parallel PR_i^{pol})$;
    $req_{i,k} = \{PID_i^{ts_\triangledown},\ PU_i^{pol},\ H_{i,1},\ H_{i,2},\ H_{i,3}\}$;
    Send($req_{i,k}$);
/* Ground station responds authentication */
Function GSAuthRep($req_{i,k}$):
    $RID_i' = H_{i,1}' \oplus H(PID_i^{ts'_\triangledown} \parallel PU_i')$;
    if $RID_i'$ is not valid then
        reject;
    else
        $r_i^{ts'_\triangledown+1} = H_{i,2}' \oplus H(PID_i^{ts'_\triangledown} \parallel PU_i' \parallel RID_i')$;
        $PR_k^{pol} = T_{(PR_k^G)}(PU_i^{pol'})$;
        $H_{i,3}' = H(PID_i^{ts'_\triangledown} \parallel PU_i' \parallel RID_i' \parallel r_i^{ts'_\triangledown+1} \parallel PR_k^{pol})$;
        if $H_{i,3}' \neq H_{i,3}$ then
            reject;
        else
            $r_k^{ts_\triangledown+2} \leftarrow$ RandNum(); $PU_k^{pol} =$
             $T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(PT_i)$;
            $H_{k,1} = H(G_k \parallel PID_i^{ts'_\triangledown} \parallel r_i^{ts'_\triangledown+1} \parallel PU_k^{pol})$;
            $rep_{k,i} = \{PID_i^{ts'_\triangledown},\ PU_k^{pol},\ H_{k,1}\}$;
            Send($rep_{k,i}$);
            $SK_{k,i} = T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(PU_i^{pol'})$;
        end
    end
/* Drone completes authentication */
Function DroneAuth($rep_{k,i}$):
    $H_{k,1}' = H(G_k \parallel PID_i^{ts_\triangledown} \parallel r_i^{ts_\triangledown+1} \parallel PU_k^{pol'})$;
    if $H_{k,1}' \neq H_{k,1}$ then
        reject;
    else
        $SK_{i,k} = T_{(r_i^{ts_\triangledown+1} \cdot ID_i)}(PU_k^{pol'})$;
    end
```

3) Authentication and Key Establishment: In this step, the drone ID_i and the ground station G_k validate each other's identities and negotiate a secure session key.

1) ID_i computes its PUF response $res_i = F_{puf}(che_i)$ and public token $PT_i = T_{(res_i)}(x)$.
2) ID_i calculates its pseudonym $PID_i^{ts_\triangledown} = H(RID_i \parallel PT_i$



```
SUMMARY
   SAFE
DETAILS
   BOUNDED_NUMBER_OF_SESSIONS
   TYPED_MODEL
PROTOCOL
   /home/span/testsuite/results/polyBlock.if
GOAL
   As Specified
BACKEND
   CL-AtSe
STATISTICS
   Analysed: 685 states
   Reachable: 287 states
   Translation: 0.19 seconds
   Computation: 1.32 seconds
```

(a)

```
SUMMARY
   SAFE
DETAILS
   BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
   /home/span/testsuite/results/polyBlock.if
GOAL
   as_specified
BACKEND
   OFMC
COMMENTS
STATISTICS
   parseTime: 0.00s
   searchTime: 8.64s
   visitedNodes: 742
   nodes depth: 19 plies
```

(b)

Fig. 2. Security verification results using AVISPA.

$\parallel r_i^{ts_\triangledown}$).

3) ID_i selects a random number $r_i^{ts_\triangledown+1}$.
4) ID_i calculates the public Chebyshev polynomial $PU_i^{pol} = T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(x)$.
5) ID_i retrieves the public key of G_k, PU_k^G, from the storage.
6) ID_i calculates the secret Chebyshev polynomial $PR_i^{pol} = T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(PU_k^G)$.
7) ID_i calculates $H_{i,1} = RID_i \oplus H(PID_i^{ts_\triangledown} \parallel PU_i)$.
8) ID_i calculates $H_{i,2} = r_i^{ts_\triangledown+1} \oplus H(PID_i^{ts_\triangledown} \parallel PU_i \parallel RID_i)$.
9) ID_i calculates $H_{i,3} = H(PID_i^{ts_\triangledown} \parallel PU_i \parallel RID_i \parallel r_i^{ts_\triangledown+1} \parallel PR_i^{pol})$.
10) ID_i sends the authentication request $req_{i,k} = \{PID_i^{ts_\triangledown}$, PU_i^{pol}, $H_{i,1}$, $H_{i,2}$, $H_{i,3}\}$ to G_k over wireless channel.
11) G_k calculates $RID_i^{'} = H_{i,1}^{'} \oplus H(PID_i^{ts_\triangledown^{'}} \parallel PU_i^{'})$.
12) G_k checks whether $RID_i^{'}$ is valid through invoking the smart contract and retrieving $RID_i^{'}$ from the private blockchain. If not, $req_{i,k}$ is rejected. Otherwise, G_k retrieves ID_i's information from the private blockchain.
13) G_k computes $r_i^{ts_\triangledown^{'}+1} = H_{i,2}^{'} \oplus H(PID_i^{ts_\triangledown^{'}} \parallel PU_i^{'} \parallel RID_i^{'})$.
14) G_k calculates $PR_k^{pol} = T_{(PR_k^G)}(PU_i^{pol^{'}})$, where $PR_k^{pol} = PR_i^{pol}$. The proof is as follows,

$$\begin{aligned} PR_k^{pol} &= T_{(PR_k^G)}(PU_i^{pol^{'}}) \\ &= T_{(PR_k^G)}(T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(x)) \\ &= T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(T_{(PR_k^G)}(x)) \\ &= T_{(r_i^{ts_\triangledown+1} \cdot res_i \cdot ID_i)}(PU_k^G) \\ &= PR_i^{pol}. \end{aligned}$$

15) G_k computes $H_{i,3}^{'} = H(PID_i^{ts_\triangledown^{'}} \parallel PU_i^{'} \parallel RID_i^{'} \parallel r_i^{ts_\triangledown^{'}+1} \parallel PR_k^{pol})$.
16) G_k verifies whether $H_{i,3}^{'} = H_{i,3}$. If the verification fails, $req_{i,k}$ is rejected. Otherwise, G_k continues as follows.
17) G_k selects a random number $r_k^{ts_\triangledown+2}$ and calculates the public Chebyshev polynomial $PU_k^{pol} = T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(PT_i)$.
18) G_k calculates $H_{k,1} = H(G_k \parallel PID_i^{ts_\triangledown^{'}} \parallel r_i^{ts_\triangledown^{'}+1} \parallel PU_k^{pol})$.
19) G_k sends the authentication response $rep_{k,i} = \{PID_i^{ts_\triangledown^{'}}$, PU_k^{pol}, $H_{k,1}\}$ to ID_i over wireless channel.
20) G_k calculates $SK_{k,i} = T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(PU_i^{pol^{'}})$.
21) ID_i computes $H_{k,1}^{'} = H(G_k \parallel PID_i^{ts_\triangledown} \parallel r_i^{ts_\triangledown+1} \parallel PU_k^{pol^{'}})$, and verifies $H_{k,1}^{'} = H_{k,1}$. If the verification fails, ID_i discards $rep_{k,i}$. Otherwise, ID_i calculates the secure session key $SK_{i,k} = T_{(r_i^{ts_\triangledown+1} \cdot ID_i)}(PU_k^{pol^{'}})$. Here

$$\begin{aligned} SK_{i,k} &= T_{(r_i^{ts_\triangledown+1} \cdot ID_i)}(PU_k^{pol^{'}}) \\ &= T_{(r_i^{ts_\triangledown+1} \cdot ID_i)}(T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(PT_i)) \\ &= T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(T_{(r_i^{ts_\triangledown+1} \cdot ID_i)}(PT_i)) \\ &= T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(T_{(r_i^{ts_\triangledown+1} \cdot ID_i)}(T_{(res_i)}(x))) \\ &= T_{(r_k^{ts_\triangledown+2} \cdot PR_j^G \cdot G_k)}(PU_i^{pol}) \\ &= SK_{k,i}. \end{aligned}$$

At this moment, the drone ID_i and the ground station G_k have completed the mutual authentication and set up a secure session key for the follow-up communication over wireless channel. The authentication and key establishment algorithm is shown in Algorithm 3.

V. Security Analysis

AVISPA [19] is an automated tool which has been widely adopted to validate the security of Internet protocols. Thus, in this paper we also use AVISPA to evaluate the *polyBlock* and intend to prove that the *polyBlock* is free of security design flaws as well as is immune to cyber attacks. We first install Virtual Box, and then download the virtual machine image which contains a fully-functional SPAN+AVISPA [10] environment. Then, we implement the *polyBlock* in High-Level Protocol Specification Language (or called HLPSL) [10] which is required by AVISPA for protocol verification. Finally, we execute the *polyBlock* HLPSL programs in AVISPA. During the process of evaluation, AVISPA will check whether the *polyBlock* has potential vulnerabilities which could be exploited by masquerading attacks, replay attacks, and other unknown attacks in the adversary setting. If AVISPA discovers the security flaws in the design of *polyBlock*, it will produce a sequence diagram showing the vulnerable scenario. Otherwise, AVISPA will attest to the truth that the *polyBlock* does not have any security design flaws. The outputs of AVISPA are presented in Fig. 2, where the *polyBlock* is certified as a safe security protocol in the adversarial setting.

VI. Performance Evaluation

In this section, we evaluate and analyze the performance of *polyBlock*, Feng et al. [8], and Tan et al. [9] in terms of execution time, energy consumption, and communication overhead. On a Windows 10 PC (4th Generation Intel Core i5-4690K CPU, 6M Cache, up to 3.90 GHz), we set up an Eclipse programming environment, implement the *polyBlock* and two counterparts in Java programming language, and then conduct simulation-based experiments. The execution time is the amount of time required to run the protocol. The

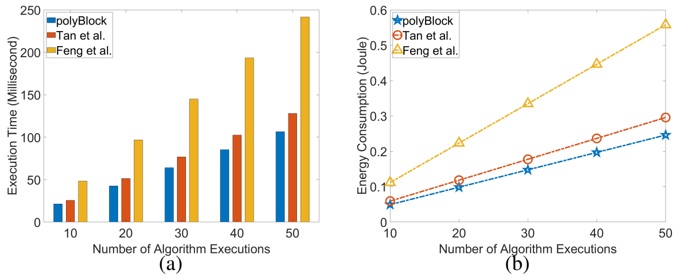


Fig. 3. Execution time and energy consumption against the number of algorithm executions.

energy consumption indicates how much energy the protocol will consume while running. The communication overhead is measured with regard to the number of sent messages and the energy consumption of sending those messages.

The execution time of *polyBlock*, Feng et al. [8], and Tan et al. [9] are measured by changing the number of algorithm executions, and the results are presented in Fig. 3(a). Overall, the lowest execution time is provided by our approach *polyBlock* as the number of algorithm executions is increased from 10 to 50. This is because the *polyBlock* is realized with lightweight function and operation such as Chebyshev polynomial and hash function. As a result, less amount of time is taken to run the *polyBlock*, and a smaller execution time is obtained. Feng et al. [8] and Tan et al. [9] choose bilinear pairing and elliptic curve cryptography respectively, which are widely regarded as resource-hungry cryptographic operations. Thus, a higher execution time is observed with both Feng et al. [8] and Tan et al. [9] compared to our approach *polyBlock*. The execution time of Feng et al. [8] is higher than that of Tan et al. [9] because bilinear pairing cryptographic scheme is more time-consuming than elliptic curve cryptographic scheme. We also obtain the results of algorithm energy consumption with varying number of algorithm executions in Fig. 3(b). Clearly, our approach *polyBlock* outperforms Feng et al. [8] and Tan et al. [9] because the *polyBlock* takes lesser amount of time to run, resulting in lower energy consumption.

The communication overhead of *polyBlock*, Feng et al. [8], and Tan et al. [9] are shown in Table I. In the *polyBlock*, four (4) messages are required for the entire process of authentication and key agreement, where one (1) message is needed for the authentication request, two (2) messages are required by the ground station to obtain the cryptographic information of drone, and one (1) message is used for the authentication response. However, according to Tan et al. [9] and Feng et al. [8] communication sequence diagrams, they will need to exchange six (6) messages and four (4) messages to finally establish a secure session key, respectively. We also obtain the energy consumption of communication based on the number of sent messages for all three protocols. The energy consumption of communication for *polyBlock*, Feng et al. [8], and Tan et al. [9] are 0.451627×10^{-3}, 0.677441×10^{-3}, and 0.451627×10^{-3} joule, respectively.

VII. Conclusion

In this paper, we proposed a cross-domain authentication and key agreement protocol (also called *polyBlock*) for IoD applications. The major advantage is that the *polyBlock* is realized by lightweight operations such as Chebyshev polynomial and hash function. In terms of evaluation, the *polyBlock* was first validated by the automated protocol verification tool AVISPA. Then, the *polyBlock* along with two counterparts were implemented and compared for performance evaluation. Experimental results indicate that the *polyBlock* not only is a secure protocol without any design flaws, but also provides more superior computation and communication performance.

TABLE I
Communication Overhead

Scheme	No. of Sent Messages	Energy Consumption
polyBlock	4	0.451627×10^{-3}
Tan et al. [9]	6	0.677441×10^{-3}
Feng et al. [8]	4	0.451627×10^{-3}

References

[1] *Military Drones Global Market Report 2023*, https://www.researchandmarkets.com/reports/5735293/military-drones-global-market-report.

[2] P. Boccadoro, D. Striccoli, and L. Grieco, “An extensive survey on the Internet of Drones,” *Ad Hoc Networks*, vol. 122, no. 8, p. 102600, 2021.

[3] C. Pu, A. Wall, and K. Choo, “Bilinear Pairing and PUF Based Lightweight Authentication Protocol for IoD Environment,” in *Proc. IEEE MASS*, 2022, pp. 115–121.

[4] C. Pu, A. Wall, K. Choo, I. Ahmed, and S. Lim, “A Lightweight and Privacy-Preserving Mutual Authentication and Key Agreement Protocol for Internet of Drones Environment,” *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 9918–9933, 2022.

[5] C. Pu and P. Zhu, “Defending against Flooding Attacks in the Internet of Drones Environment,” in *Proc. IEEE GLOBECOM*, 2021, pp. 1–6.

[6] M. Zhang, C. Xu, S. Li, and C. Jiang, “On the Security of an ECC-Based Authentication Scheme for Internet of Drones,” *IEEE Systems Journal*, vol. 16, no. 4, pp. 6425–6428, 2022.

[7] M. El-Zawawy, A. Brighente, and M. Conti, “SETCAP: Service-Based Energy-Efficient Temporal Credential Authentication Protocol for Internet of Drones,” *Computer Networks*, vol. 206, p. 108804, 2022.

[8] C. Feng, B. Liu, Z. Guo, K. Yu, Z. Qin, and K. Choo, “Blockchain-Based Cross-Domain Authentication for Intelligent 5G-Enabled Internet of Drones,” *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 6224–6238, 2021.

[9] Y. Tan, J. Wang, J. Liu, and N. Kato, “Blockchain-assisted distributed and lightweight authentication service for industrial unmanned aerial vehicles,” *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 16 928–16 940, 2022.

[10] *SPAN*, Last accessed: Mar 12, 2023, http://www.avispa-project.org/.

[11] C. Tian, Q. Jiang, T. Li, J. Zhang, N. Xi, and J. Ma, “Reliable PUF-based mutual authentication protocol for UAVs towards multi-domain environment,” *Computer Networks*, vol. 218, p. 109421, 2022.

[12] B. Gong, G. Zheng, M. Waqas, S. Tu, and S. Chen, “LCDMA: Lightweight Cross-domain Mutual Identity Authentication Scheme for Internet of Things,” *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[13] D. Dolev and A. Yao, “On the Security of Public Key Protocols,” *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[14] C. Pu, “Jamming-Resilient Multipath Routing Protocol for Flying Ad Hoc Networks,” *IEEE Access*, vol. 6, pp. 68 472–68 486, 2018.

[15] C. Pu and P. Zhu, “Mitigating Routing Misbehavior in the Internet of Drones Environment,” in *Proc. IEEE VTC2022-Spring*, 2022, pp. 1–6.

[16] C. Pu, “A Featherweight Authentication and Key Agreement Scheme for Internet of Drones Applications,” in *Proc. IEEE PIMRC*, 2023, pp. 1–6.

[17] A. Dorri *et al.*, “LSB: A Lightweight Scalable Blockchain for IoT security and anonymity,” *Journal of Parallel and Distributed Computing*, vol. 134, pp. 180–197, 2019.

[18] C. Pu, A. Wall, I. Ahmed, and K. Choo, “SecureIoD: A Secure Data Collection and Storage Mechanism for Internet of Drones,” in *Proc. IEEE MDM*, 2022, pp. 83–92.

[19] *Automated Validation of Internet Security Protocols and Applications*, http://www.avispa-project.org.