Securely Controllable and Trustworthy Remote Erasure on Embedded Computing System for Unmanned Aerial Vehicle

Sieun Kim[1], Taek-Young Youn[2], Daesun Choi[3*], and Ki-Woong Park[4*]
[1]Sejong University SysCore Lab, Seoul, Republic of Korea
rlatldms2004@naver.com
[2]ETRI, Dea-jeon, Republic of Korea
taekyoung@etri.re.kr
[3]Kongju National University, Kong-ju, Republic of Korea
sunchoi@kongju.ac.kr
[4]Sejong University, Seoul, Republic of Korea
woongbak@sejong.ac.kr

Abstract

Unmanned Aerial Vehicle(UAV) plays an increasingly core role in modern warfare since powerful but tiny embedded computing system is actively applied in military field. Confidential data, such as military secrets, may be stored inside military devices such as UAVs, which can cause an exorbitant damage to the national security if the devices are kidnapped or lost. Therefore, a securely controllable and trustworthy erasure technique for military devices has been considered as a core technology. In this paper, we devise a securely controllable and trustworthy scheme for satisfying the reliable remote data erasure technology. The scheme allows the user to remotely erase data stored in the UAV even on loss of communication and returns proof of erasure to user after erasure.

Keywords: Unmanned Aerial Vehicle, Trustworthy Remote Erasure, Embedded Computing System

1 Introduction

An American Lockheed Martin RQ-170 Sentinel unmanned aerial vehicle (UAV) was captured by Iranian forces in 2011. The U.S. had lost control of the UAV during performing the mission of flying Afghanistan. Iran tried to extract information from the UAV captured and the attempt was successful. Iran released the video obtained from the captured UAV and constructed a copy of an RQ-170, the Saegheh [2]. Theft of a military UAV causes great damage by losing secret data, such as collected information.

To protect confidential data stored in a remote device, it can be safeguarded by safely deleting the data as needed. If the user sends erasure request, the remote device receives it and performs the erasure. However, in the case of the above-mentioned UAVs remotely performing mission, it may be confronted with a situation where it cannot receive the signal. Thus, if the confidential information need to be stored in UAV, the device have to provide a securely developed remote erasure mechanism.

Recognition that sensitive information has been leaked(erasure process has not been performed) is important as an indicator of future plans, so we need to confirm that erasure is really performed in a verifiable way (verifiable erasure). In other words, when data which must not be leaked is stored and erased remotely, the outcome of the erasure operation should be identified. Verifying the result of the

*Corresponding author: Daeseon Choi, Ki-Woong Park, Sejong University, 98, Gunja-dong, Gwangjin-gu, Seoul, Republic of Korea, Tel: +82-10-9165-1624, Email: woongbak@sejong.ac.kr erasure operation as one-bit response has low reliability [4]. Therefore, an obvious proof of erasure must be presented and a false proof must not be able to be generated.

By using conventional remote erasure system, we can try to prevent remote data theft. But two security issues arise as shown in Fig. 1. The issues and our approach are described as follows:

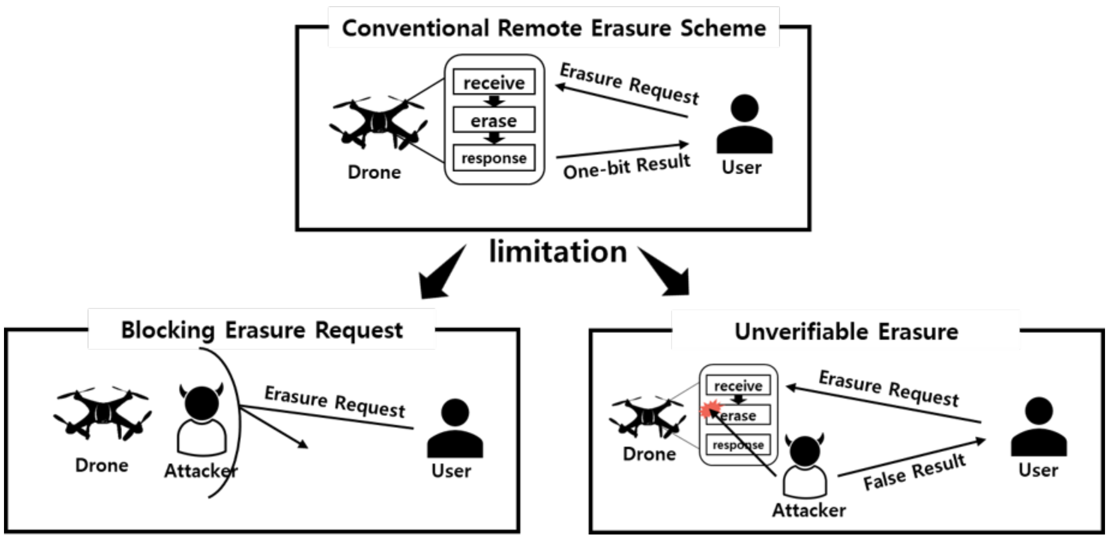


Figure 1: Limitations of conventional remote erasure scheme

- **Support for availability of remote erasure on loss of communication:** In case users lost devices supporting for remote erasure, users can delete data by sending a delete signal to lost devices. However, when communication is cut off the conventional remote erasure is not available. We, therefore, devised counter-based erasure which can delete data on uncontrollable environment. The counter-based erasure means the scheme that has a count value, increments the value one by one, and deletes data when the counter value reaches a certain value. The users can retain stored data only through periodically sending the specific value, h to the devices. If the user dose not send the specific value to the remote device, the device reaches a specific time and erases data. This mechanism enables to erase data by not providing the h even on devices lost control.

- **Support for a verifiable erasure:** Although it is essential to verify erasure of sensitive information such as military data, most mechanisms provide nothing or only a simple response (erase or not erase). Our remote erasure system provides a relatively complex but verifiable proof of erasure. Upon a process of erasure of data, the remote device generates a proof of erasure and sends that to the user. The user checks the proof received and verifies that remote data are assuredly erased.

The remainder of this paper is organized as follows: Section 2 reviews the related works of reliable remote data deletion. We present the system architecture of securely controllable and trustworthy remote erasure in Section 3 Section 4 describes testbed construction in progress. In Section 5, we present our conclusions.

2 Related Works

To resolve problem of erasure of remotely stored data with verifiability, the researchers have devised approaches which generate a provable value of erasure operation. We analyzed several of them and identified the limiting factors in applying a verifiable erasure for UAVs.

Daniele Perito and Gene Tsudik use a PoSE(Proof of Secure Erasure) for secure code update on embedded devices [8]. This scheme ensures that devices can securely update code by verifying erasure of memory. A verifier sends verifier-selected randomness of the size of memory to a prover and the prover's memory is filled with the received value. The prover returns the very same randomness written on the memory to the verifier. Because both the verifier and the prover send a random value of memory size to each other, this protocol has a huge overhead concerning communication.

Dziembowski et al. proposed a scheme which reduces communication complexity of PoSE [3]. A verifier sends a seed to a prover. The prover performs hash function using the seed and retains the sizable result value using all of prover's memory. The calculated hash value can only be generated once because it uses the secret key stored in memory, so it can be proof of erasure. This scheme reduces communication overhead of PoSE but a huge calculative overhead arises.

Mahmoud Ammar et al. introduced SPEED that guarantees erasure on embedded system [1]. The protocol is implemented in isolated memory using the Trusted Software Module. A verifier's distance is measured by the Distance Bounding(DB) protocol which establish an upperbound on the physical distance between a verifier and a prover. The prover authenticates the verifier through the DB protocol and erases data if verification passed successfully. To verify erasure, the prover sends the proof as the message authentication code value of the entire memory. The proposed scheme is limited in terms of distance so cannot be useful for UAV.

The above-mentioned works enhanced the reliability of erasure by returning the proof of erasure generated by the prover to the user. Our approach solves verifiable erasure through the proof of erasure that only the user can identify, as in the above researches. However, our approach creates a proof of erasure in such a way that the memory blocks each have a dependency on the block, which guarantees more robust evidence. The additional aim of our research is to achieve remote erasure of UAVs that are likely to be hijacked and disconnected. The above works don't consider the loss of communication environment that a specific device can be confronted with. Therefore, it is also our keynote to be able to remotely erase data stored in UAV from a hijack situation (or a situation in which the UAV can't be controlled).

3 System Architecture

We descript the proposed system architecture in this section. First, we present the overall process of erasure system devised. Next, we explain a communication protocol using a hash chain and describe a verifiable erasure process which is performed in UAVs.

3.1 Overall process of proposed scheme

We introduce the overall process of counter-based our approach. Incrementing the counter embedded in UAVs by one, the UAV wait the message from the user. The flow of our scheme is largely divided into two. If UAV can continuously receive the message from the user, the UAV works normally. If UAV cannot be provided the message from the user, the UAV erases stored data.

Fig. 2. shows the overall process of the securely controllable and trustworthy remote erasure system.

1. UAV's counter value keeps increasing. The UAV waits a message from the user.

2-a. If the UAV received a message, the UAV authenticates the user.

3-a. If the UAV validates successfully user, the UAV updates the stored hash value, initializes timer value to zero and again waits a message from user. In other words, UAV's state return to 1.

2-b. If the UAV didn't receive a message, the UAV just waits, increasing timer value.
3-b. If the UAV didn't receive a message until timer value reaches l, the UAV performs erasure process.
4-b. The UAV sends a proof of erasure to the user.

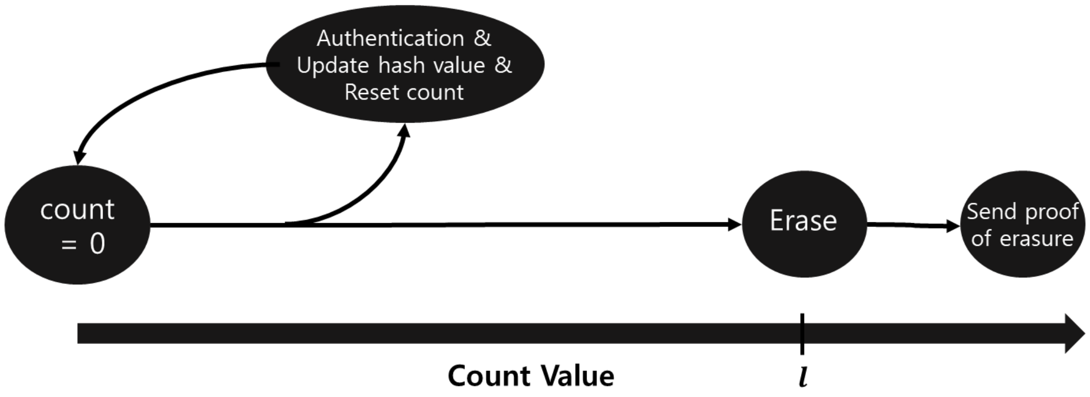
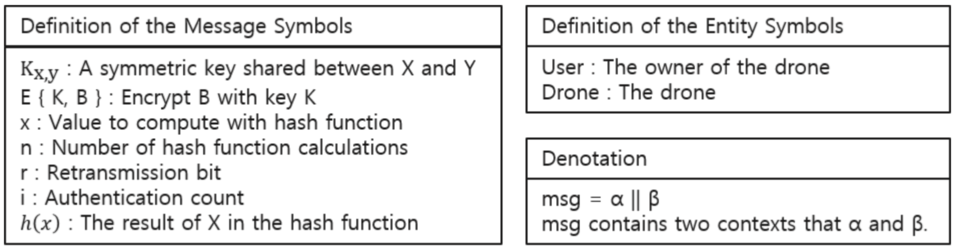


Figure 2: Overall process of the proposed remote erasure system

A more comprehensive description of the above communication and erasure process can be found in the Section 3.2 and 3.3.

3.2 Communication Protocol

The communication protocol of proposed system is introduced in more detail in this section. Fig. 3. defines the entity symbols and the message symbols.



Definition of the Message Symbols
$K_{x,y}$: A symmetric key shared between X and Y
E { K, B } : Encrypt B with key K
x : Value to compute with hash function
n : Number of hash function calculations
r : Retransmission bit
i : Authentication count
$h(x)$: The result of X in the hash function

Definition of the Entity Symbols
User : The owner of the drone
Drone : The drone

Denotation
msg = α || β
msg contains two contexts that α and β.

Figure 3: Notations of the entity and messages

Fig. 4. shows the message flow when communication is conducted without a problem and when communication is cut off. First, the User encrypts the number of hash function calculations(n) and the value to compute with hash function(x) using symmetric key already shared between the User and the Drone and sends x and n to the Drone. After receiving the values from the User, the Drone calculates $h^n(x)$ and sends that to the User. The User checks whether the Drone receives correct x and n from the User. The User and the Drone, that verified each other, start to exchange the authentication message using a hash chain. After the Drone received the hash value from the User, the Drone calculates hash value of stored value and compares computed value and received value. If two values are equal, the Drone resets counter and updates stored value. These transactions are repeated as n times if communication is good. When the number of authentication(i) exceeds n which means maximum number of hash function calculations, the User and the Drone again exchange new n and x. If the Drone doesn't receive the hash value from the User until the particular time, the Drone proceed erasure process as shown on the right.

If the User doesn't receive a message from the Drone, the User again sends the same hash value after waiting the specified number of seconds. At the time, *r* bit which means a retransmit is contained and set in the message. The Drone which received the message checks *r* bit, recognizes resending, and confirms that the hash value stored in the Drone and received value are equal. If two values are perfectly equal, the Drone again sends the value(*n-i*) which means a request for next hash value. At the time, the Drone doesn't update the counter value and the hash value stored. Afterward, the User and the Drone again start normal communication.

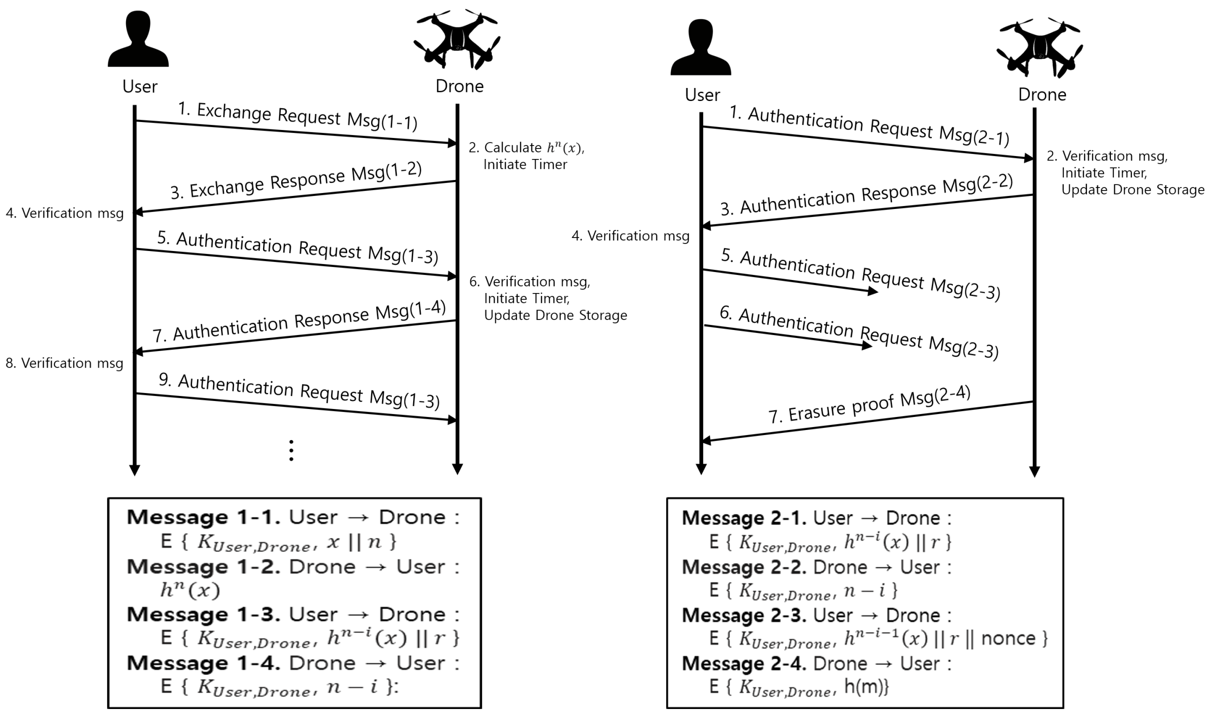

User
Drone
1. Exchange Request Msg(1-1)
2. Calculate $h^n(x)$, Initiate Timer
3. Exchange Response Msg(1-2)
4. Verification msg
5. Authentication Request Msg(1-3)
6. Verification msg, Initiate Timer, Update Drone Storage
7. Authentication Response Msg(1-4)
8. Verification msg
9. Authentication Request Msg(1-3)
Message 1-1. User → Drone : E { $K_{User,Drone}$, $x \,||\, n$ }
Message 1-2. Drone → User : $h^n(x)$
Message 1-3. User → Drone : E { $K_{User,Drone}$, $h^{n-i}(x) \,||\, r$ }
Message 1-4. Drone → User : E { $K_{User,Drone}$, $n-i$ }:
User
Drone
1. Authentication Request Msg(2-1)
2. Verification msg, Initiate Timer, Update Drone Storage
3. Authentication Response Msg(2-2)
4. Verification msg
5. Authentication Request Msg(2-3)
6. Authentication Request Msg(2-3)
7. Erasure proof Msg(2-4)
Message 2-1. User → Drone : E { $K_{User,Drone}$, $h^{n-i}(x) \,||\, r$ }
Message 2-2. Drone → User : E { $K_{User,Drone}$, $n-i$ }
Message 2-3. User → Drone : E { $K_{User,Drone}$, $h^{n-i-1}(x) \,||\, r \,||$ nonce }
Message 2-4. Drone → User : E { $K_{User,Drone}$, h(m)}

Figure 4: The message flow depending on communication environment

We analyzed that our protocol is safe from replay attack and MITM attack. We assumed that the used hash function and encryption algorithm is invulnerable.

This scheme is safe from *replay attacks*.

- **User** $\Rightarrow$ **Drone** An attacker can try to attack by resending the authentication request message captured. In case of resending, attacker cannot recognize resending and interpret authentication message because the user encrypts message using shared key between the user and the UAV. Thus, the attacker no longer cannot perform authentication request mechanism.

- **Drone** $\Rightarrow$ **User** An attacker can try to attack by resending the authentication response message captured. However, the attacker cannot know the remaining number of hash chain because the UAV encrypts remaining number of hash chain using already exchanged key. Therefore, the attacker no longer cannot perform authentication response mechanism.

This scheme is safe from *MITM(Man In The Middle) attacks*.

- **User** $\Rightarrow$ **Drone** Authentication mechanism uses the generated hash chain in the reverse direction. Thus, an attacker cannot forge authentication request message of the user.

- **Drone** ⇒ **User** Before sending message to the user, the UAV encrypts remaining number of authentication using symmetric key already shared between the user and the UAV. Thus, the attacker cannot forge authentication request message of UAVs.

3.3 Verifiable Erasure Protocol

In this section, we describe the process by which the UAV erases data and generates proof of erasure. If the UAV cannot initialize the inserted counter because the user doesn't provide the h, the counter will reach the specified time and delete the data stored in UAV. The UAV performs an XOR operation of the $\mathbf{m}_0$ block of memory and the last hash value received from the user and wipes the $\mathbf{m}_0$ block with the calculated value. Next, the $\mathbf{m}_1$ block is wiped with the value obtained by performing the XOR operation of the wiped $\mathbf{m}_0$ block and the $\mathbf{m}_1$ block. We perform wiping repeatedly memory in the same way as above until wiping the $\mathbf{m}_n$ block which means last block of memory. We define the whole process mentioned above as a round.

To prevent forensic, we proceed multiple rounds. We don't focus on delete mechanism to make data impossible to recover it. In this paper, we assume that the repeatedly wiped memory cannot be recovered, so it is impossible to extract information from the memory. In the second round, we start the round process, wiping the $\mathbf{m}_0$ block with the value performing XOR operation of the block $\mathbf{m}_n$ and the $\mathbf{m}_0$ block. After proceeding the round several times, as a proof of erasure, the block $\mathbf{m}_n$ value is calculated using a hash function and UAV send the proof to the user as Morse code. The user who received the proof of erasure from the UAV checks that the erasure is really proceeded by predicting the proof. Fig. 5. shows our approach about verifiable erasure.

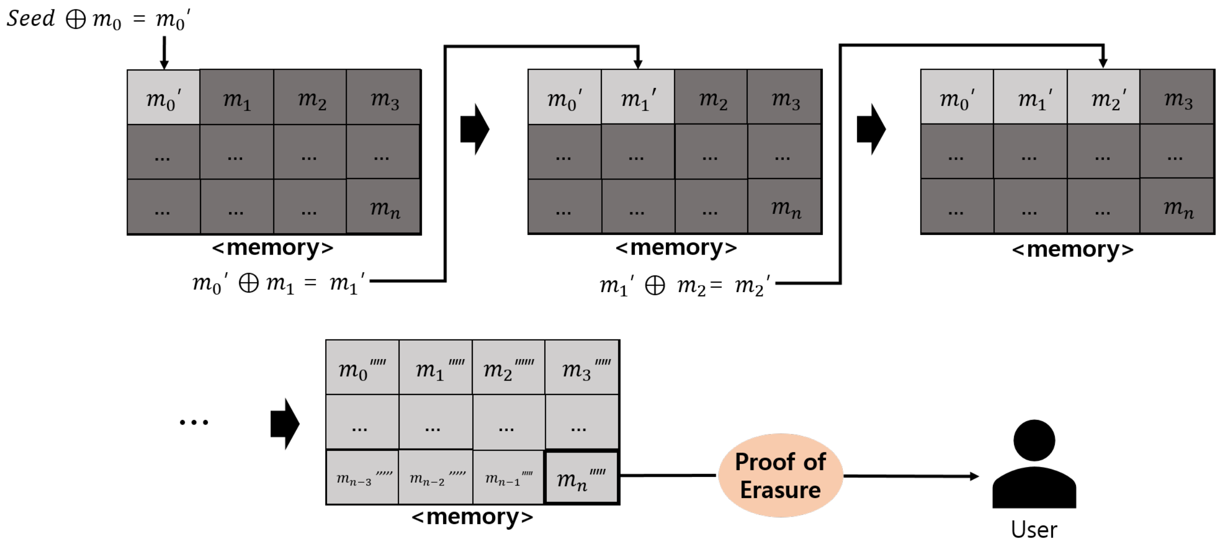


Figure 5: Process of the Verifiable Erasure

In the first part of the erasure process, the last hash value sent by the user is used as a seed in the erasure process to enhance security. The hash value used as the seed is an encrypted value, so an attacker cannot determine this value. Each value calculated by the above erasure process has a dependency on the previous memory data. Therefore, even if the attacker knows some blocks of memory, the false proof cannot be generated.

4 Testbed Construction in progress

To test performance of the devised protocol, we constructed a testbed as shown in Fig. 5. Communication between the user and the UAV has been linked to experiment with our approach. The board used for UAV is the T2080. The T2080 can function as UAV flight computer and has 128 MB NOR flash, SD connector to interface, and SATA interface.

Through this testbed, we will measure erasure latency to investigate how practical the devised scheme is. To evaluate the latency of the verifiable erasure, we conduct experiments on SD cards inserted in the T2080. SD cards can read and write as 512-byte block unit through commands provided by default. We measure erasure latency according to the size of memory to be erased and the memory block size to be calculated with XOR operation. The block size used in the XOR operation is determined based on how many 512-byte blocks are sequentially read at a time. The erasure latency in this experiment means the time from when the embedded timer reaches l to when the user receives the proof of erasure. Since the experiment is performed on the assumption that the user already knows the data stored in the SD card, the user can determine whether the proof of erasure is valid when receiving the proof.

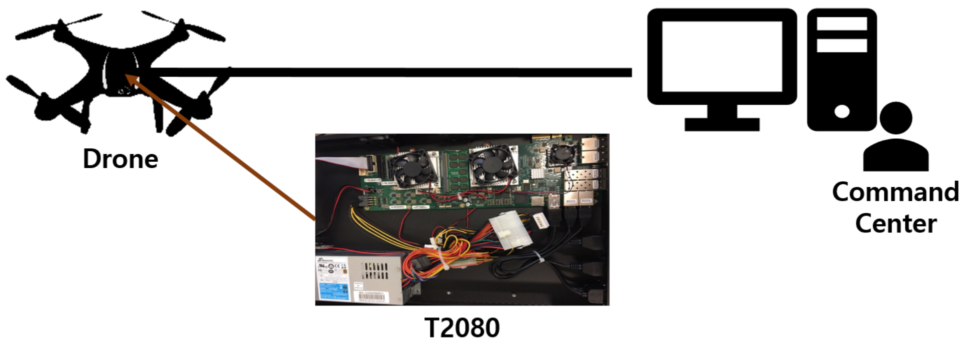


Figure 6: Testbed Construction

5 Conclusion

Our approach assumes that the user already possesses proof of erasure. If the data stored in the UAV is static data that does not change, the user can predict the contents of the remote memory and calculate the proof of erasure. However, the data stored in the memory of a UAV performing a mission has high possibility of being dynamically changed. Therefore, even if the user knows the last hash value that the attacker cannot know, it is very difficult for the user to predict the data in the remote memory and calculate the proof of erasure. We need to study how to allow users to accurately predict the proof while maintaining the proof of erasure that we devised in the future work.

We presented the protocol to achieve securely controllable and trustworthy remote erasure for UAV in this paper. If a user doesn't provide a specific value to UAV, stored data in UAV is erased. This scheme allows the user to prevent data theft on uncontrollable devices. After erasing by sending the proof as calculated memory hash value not one-bit response to the user, the user can recognize that data is assuredly deleted.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (NRF-2017R1C1B2003957, NRF-2016R1A4A1011761) and supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00420, Development of Serverless Cloud Computing technology for resource allocation and usage measurement by API call unit)

References

[1] M. Ammar, W. Daniels, B. Crispo, and D. Hughes. Speed: Secure provable erasure for class-1 iot devices. In *Proc. of the 8th ACM Conference on Data and Application Security and Privacy (CODASPY'18), Tempe, Arizona, USA*, pages 111–118. ACM, March 2018.

[2] D. Cenciotti. Iran unveils new UCAV modeled on captured U.S. RQ-170 stealth drone. `https://theaviationist.com/2016/10/02/iran-unveils-new-ucav-modeled-on-captured-u-s-rq-170-stealth-drone/` [Online; accessed on August 15, 2018], October 2016.

[3] S. Dziembowski, T. Kazana, and D. Wichs. One-time computable self-erasing functions. In *Proc. of the 2011 Theory of Cryptography Conference (TCC'11), Providence, Rhode Island, USA*, volume 6597 of *Lecture Notes in Computer Science*, pages 125–143. Springer, Berlin, Heidelberg, March 2011.

[4] F. Hao, D. Clarke, and A. F. Zorzo. Deleting secret data with public verifiability. *IEEE Transactions on Dependable and Secure Computing*, 13(6):617–629, April 2016.

[5] K. Hartmann and C. Steup. The vulnerability of uavs to cyber attacks-an approach to the risk assessment. In *Proc. of the 5th International Conference on Cyber Conflict (CyCon'13), Tallinn, Estonia*, pages 1–23. IEEE, June 2013.

[6] G. O. Karame and W. Li. Secure erasure and code update in legacy sensors. In *Proc. of the 2015 International Conference on Trust and Trustworthy Computing (Trust'15), Heraklion, Greece*, volume 6229 of *Lecture Notes in Computer Science*, pages 283–299. Springer, Cham, August 2015.

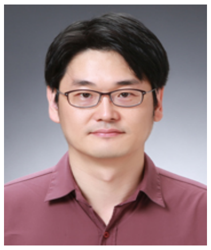
[7] K.-W. Park, J. Han, J. Chung, and K. H. Park. THEMIS: A mutually verifiable billing system for the cloud computing environment. *IEEE Transactions on Services Computing*, 6(3):300–313, January 2013.

[8] D. Perito and G. Tsudik. Secure code update for embedded devices via proofs of secure erasure. In *Proc. of the 2010 European Symposium on Research in Computer Security (ESORICS'10), Athens, Greece*, volume 6345 of *Lecture Notes in Computer Science*, pages 643–662. Springer, Berlin, Heidelberg, September 2010.

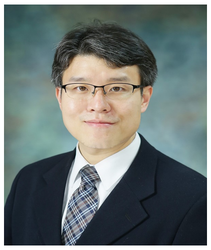
Author Biography

Sieun Kim received the B.S. degree in Information Security from Sejong University in 2018. Currently she is taking a master's course at Graduate School of Information Security, Sejong University. Her research interests include Embedded System Security.

Taek-Young Youn received his BS, MS, and Ph.D from Korea University in 2003, 2005, and 2009, respectively. He is currently a senior researcher at Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. From 2016, he serves as an associate professor in University of Science and Technology (UST), Daejeon, Korea. His research interests include cryptography, information security, authentication, data privacy, and security issues in various communications.



Daeseon Choi received the B.S. degree in computer science from Dongguk University, Korea, in 1995, the M.S. degree in computer science from Pohang Institute of Science and Technology(POSTECH), Korea, in 1997, and the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Korea, in 2009. He is currently a Professor at Department of Medical Information, Kongju National University, Korea. His research interests include information security and identity management.



Ki-Woong Park received the BS degree in computer science from Yonsei University in 2005, and the MS and PhD degrees in electrical engineering from KAIST in 2007 and 2012, respectively. He is currently an assistant professor in the Computer and Information Security Department at Sejong University. He worked as a researcher at the National Security Research Institute in 2012. His research interests include system security issues for cloud and mobile computing systems as well as the actual system implementation and subsequent evaluation in a real computing system. He received a 2009-2010 Microsoft Graduate Research Fellowship. He is a member of the IEEE and the ACM.