Two-Way Real Time Multimedia Stream Authentication Using Physical Unclonable Functions

Mehrdad Zaker Shahrak[1], Mengmei Ye[1], Viswanathan Swaminathan[2], and Sheng Wei[1,2]

[1]Department of CSE, University of Nebraska-Lincoln, Lincoln, NE, USA
Email: {mehrdad, mye}@huskers.unl.edu, shengwei@unl.edu

[2]Adobe Research, Adobe Systems Inc., San Jose, CA, USA
Email: vishy@adobe.com

***Abstract*—Multimedia authentication is an integral part of multimedia signal processing in many real-time and security sensitive applications, such as video surveillance. In such applications, a full-fledged video digital rights management (DRM) mechanism is not applicable due to the real time requirement and the difficulties in incorporating complicated license/key management strategies. This paper investigates the potential of multimedia authentication from a brand new angle by employing hardware-based security primitives, such as physical unclonable functions (PUFs). We show that the hardware security approach is not only capable of accomplishing the authentication for both the hardware device and the multimedia stream but, more importantly, introduce minimum performance, resource, and power overhead. We justify our approach using a prototype PUF implementation on Xilinx FPGA boards. Our experimental results on the real hardware demonstrate the high security and low overhead in multimedia authentication obtained by using hardware security approaches.**

I. Introduction

Real time multimedia streaming systems, such as video surveillance systems, are gaining an increasing level of applications especially in the era of fast evolving internet of things (IoTs). For example, more and more video surveillance systems have been installed to monitor the real time security status of private residences, banks, highway traffic, etc. Since many of these systems are deployed for remote security monitoring, the video must be streamed to the monitoring portal (i.e., the remote receiving end) via network communications. In this case, the security and integrity of the multimedia stream become critical.

There are typically two potential threats that may compromise the surveillance video stream. First, it is possible for an adversary to switch the source of the stream or partially modify the video stream content before it is delivered to the receiver. Second, an unauthorized viewer may receive and watch the video stream that he/she is not supposed to view. The first threat may seriously compromise the security and thus the entire purpose of deploying such a surveillance system, and the second threat may raise privacy concerns for the objects being monitored.

The aforementioned threats raise two design goals for real time multimedia authentication, which we focus on in this paper: (1) *Multimedia Authentication*, which verifies whether the multimedia content could have been modified before playback at the receiver end; and (2) *Device Authentication*, which examines whether the receiver has the permission to play the streamed multimedia content. While both threats can be well addressed by employing a full-fledged video digital rights management (DRM) scheme widely used by Internet entertainment video streaming [6], it requires video encryption and thus sophisticated license and key management strategies that are too heavy weight for the real time video surveillance.

Therefore, most of the existing approaches have focused on non-encryption and non-DRM based video authentication solutions, such as signature-based [1][2][3] approaches, where the sender signs the stream using a public key and the receiver verifies it using the corresponding private key. There are many research efforts that focus on how to reduce the overhead of signing and verifying the stream by using a variety of ways for hash chaining [2][3][4][5]. However, the existing multimedia authentication approaches appear to be on another extreme compared to full DRM, as they lack the support of device authentication and thus have no control over who can view the content. This situation may be fine in a small constrained network. However, with the widespread IoT deployment, privacy issues have drawn significantly more attention than ever before, and it becomes necessary to deploy a lightweight device or user authentication scheme for the video surveillance scenario. In addition, the performance of the signature and the overhead of the authentication information exchange can still be improved.

In this paper, we develop a two-way real time multimedia authentication scheme that covers both device and multimedia content authentications. As different from the existing approaches, our idea is to employ a hardware security primitive, namely physical unclonable function (PUF) to achieve the authentication goals. As shown in Figure 1, a PUF is a hardware implementation of a secure one-way function, which generates a unique response string for each given input challenge, based on the unique hardware property of the PUF [8]. In particular, we employ the unique challenge response pairs (CRPs) corresponding to the hardware PUF to authenticate the

Fig. 1. Functionality of a basic PUF.

receiver device. Also, we adopt the hardware accelerated one-way function enabled by PUF to improve the existing hash chaining-based method in multimedia authentication. To the best of our knowledge, this is the first use case of hardware security primitives in real time multimedia authentication, and we expect the current and the continuing work could provide a new direction for improving the quality of experience (QoE) in multimedia authentication.

II. PUF Design and Implementation

In this section, we discuss our PUF design that can be implemented and tested on Field Programmable Gate Arrays (FPGAs). The basic concept of the PUF design, which was also introduced in [8][9], is to create a number of delayed path elements and use them randomly to generate random and unique responses.

A. PUF Design

There are different ways to generate such random delays in real hardware, such as using arbiters [9] and ring oscillators (ROs) [10]. In this work, we employ the RO-based design considering its ease of implementation on FPGAs. The RO PUF leverages delay loops that each oscillates at a specific frequency [10]. The main idea is that all delay loops are identical in design but, due to the random variations in manufacturing, each loop creates a particular frequency that is slightly different from other loops. As shown in Figure 2, we connect these loops to multiplexers, where the input challenge is being used as selectors that connect each loop to a counter. Each multiplexer's output serves as a counter clock, which in turn propagates a signal to a number of flip-flops. As a result, it produces unique response bits to a given input challenge. In this case, the counter produces the results sooner, i.e., the one with faster frequency, is responsible for generating the response.

We start to generate the response based on when one of the counters overflows. There are 16 ROs in this design which are connected to a 8-1 multiplexer respectively. The first 4 bits of challenge are used to choose the output of first multiplexer, and the next four bits are used as selector of the second multiplexer, and so on. Therefore, one input from each multiplexer is chosen to be connected to each counter to be used as counter clock. Once the clock is connected to the counter, it starts to count with positive edge of its clock until it overflows (i.e., geting all one in the output and restarts again). Based on the unique frequency of each clock, one of the counters overflows sooner than others. In our design, each RO has 74-stages which in theory create 175 ns frequency. In simulation, since the input paths have the same frequency, we make them different from each other by adding different hardcoded delays to each path.

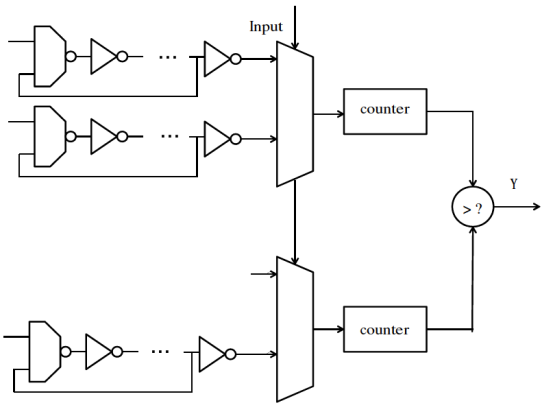

Fig. 2. Schematic of Ring Oscillator PUF

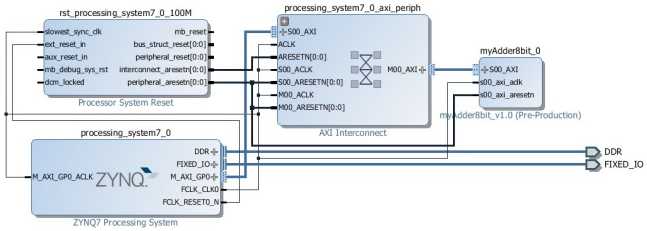

Fig. 3. Block diagram and custom IP connection using Xilinx Vivado toolchain.

B. PUF Implementation

We implement the designed RO PUF on Xilinx ZedBoard using the ViVado toolchain, following the design and implementation presented in [11]. Also, we package the PUF design as a custom Intellectual Property (IP) and connect it to the ZYNQ processing system using an AXI interconnection, as shown in Figure 3. Then, a hardware implementation is created from the design and, at the end, the bitstream is generated exported to Xilinx SDK.

The SDK creates an address map for ZedBoard processor, namely Cortex-A9. The address map is critical, as it gives the architecture the addresses necessary to load different registers and memories to CPU and cache. To evaluate the PUF performance, we feed the base address of AXI with the challenge so the challenge can be stored into first slave register of AXI, which in turn connects to the PUF input. The response of the PUF is being stored in the second slave register of AXI, which is accessed by the base address plus an offset. Therefore, by printing the second slave register to the terminal we can see the response of PUF to each corresponding challenge.

III. PUF-based Multimedia Authentication Protocol

In this section, we discuss the details of the multimedia authentication protocol based on the PUF, which authenticates both the device and the multimedia stream in a surveillance video streaming scenario.

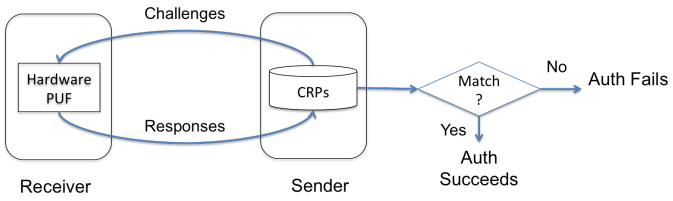

Fig. 4. Flow of the device authentication protocol.

A. Device Authentication

In device authentication our goal is to let the sender determine whether the receiver device, which is requesting the video stream, is a registered device that has permission to view the surveillance video. Figure 4 illustrates the authentication protocol to achieve the goal leveraging PUFs. There is one PUF involved for each pair of sender and receiver, and the PUF is physically possessed by the receiver. The sender hosts a secure database that contains a sufficient set of challenge response pairs (CRPs) corresponding to the receiver's PUF. There are several ways in which the sender could initialize the CRP database. For example, the sender can collect a large set of CRPs from the actual PUF before delivering it to the receiver, or the sender can maintain a software model of the PUF, obtained from the manufacturer, and generate the CRP in the real time. For the discussion of the device authentication protocol, we do not differentiate these detailed CRP generation mechanisms.

In an authentication session, the sender randomly picks K CRPs from the database, where K is a parameter determined by the sender for different levels of security, and send only the challenges to the receiver. Upon receiving the challenges, the receiver apply them one by one to the hardware PUF, collect the responses, and send them back to the sender. Then, the sender verifies if the received responses match with the records in the database. If they do, the sender confirms that the device is authenticated and starts the video stream. Otherwise, the sender will claim an authentication failure and do not initiate the stream.

B. Multimedia Stream Authentication

For multimedia content authentication, i.e., verifying the integrity of the stream, we adopt the hash chaining method proposed by Gennaro and Rohatgi [2] with our improvement based on PUF. We first split the video stream into continuous streams, which is compliant with the existing HTTP video streaming mechanisms. Then, for each segment i, [2] proposed to embed a one-time public key and the one-time signature of itself with respect to the key contained in segment $i-1$. In this way, there will be a chained signing and verifying procedure as shown in Figure 5.

Although the chaining-based signature scheme eliminates the need of the whole stream to verify the signature, it requires storing the one-time public key in each segment, which increases the segment sizes as well as the complexity for processing. In our PUF-based scheme, since both the sender and receiver holds either a software model or hardware PUF, they can both use the PUF to generate the same public key (e.g., using a hash or a portion of the segment as input challenges) and do not need to exchange the key at the runtime.

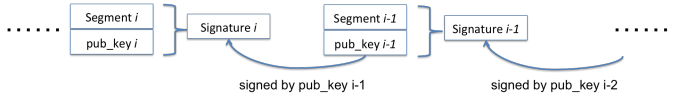

Fig. 5. Flow of the chaining-based signature scheme for multimedia authentication [2].

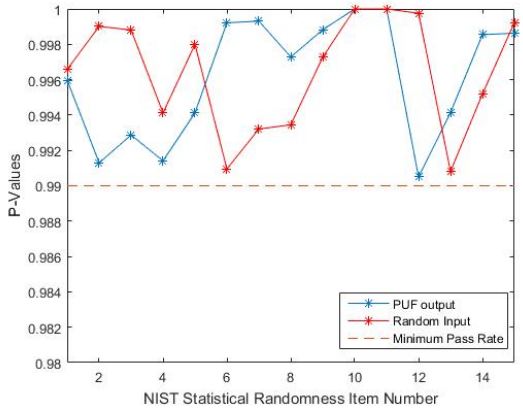

Fig. 6. NIST randomness test results.

IV. Experimental Results

A. PUF Security Evaluation

1) Evaluation Methods: There are two different categories of tests designed to evaluate the security of PUF. First, for two very similar challenges (e.g., challenges which have only one bit different) we should get very different responses, which demonstrates the uniqueness of the PUF responses. This difference is often measured by computing the hamming distance (i.e., number of flipped bits between two responses). Second, for a randomly distributed sets of challenges, the distribution of the responses should also appear random, so that it is almost impossible to guess the bit responses correctly.

2) Randomness Test: We evaluate the randomness of the PUF responses using NIST randomness tests [7], which is a common benchmark used for entropy analysis. NIST evaluates response sequences of PUF using 15 different tests. As shown in Figure 6, the results obtained from our PUF implementation closely resemble the results of a randomly generated set, showing an indiscernible similarity between PUF responses and random bits set. The minimum pass rate for each of the 15 NIST tests is 99% and both cumulative p-values and proportions are 99% pass.

3) Hamming Distance Test: In hamming distance test, a single PUF is given 5000 pairs of randomly generated challenges where each pair has one hamming distance. For an ideal PUF, we expect to get 50 percent hamming distance between pairs which is equal to a hamming distance of 4 for

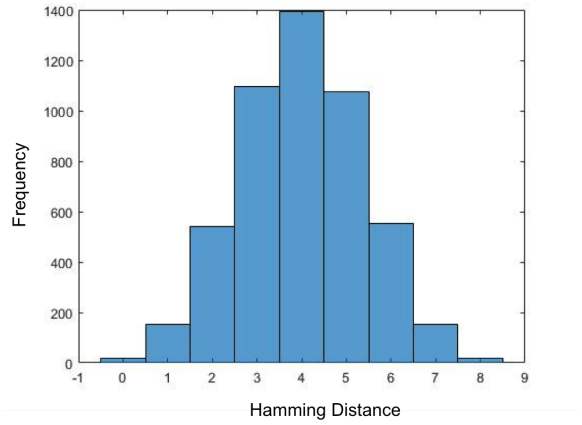

Fig. 7. Response distribution of hamming distance

an 8-bit PUF. Figure 7 shows the hamming distance histogram for 5000 CRPs. We observe that the most frequent hamming distances are 4, 3, and 5. On average, we have 4.001 bits change in the response for one bit change in challenge. In other words, we have 99.6% different responses once one bit is changed in the input challenge, and only 0.4% of responses are similar due to the one bit change in input.

B. PUF Performance and Overhead Evaluation

We further evaluate the performance of the PUF, in terms of the delay between it receives the input challenge and it generates the output response. To accommodate for the random noise and possibly different delays in processing different input challenges, we run 5 groups of timing experiments with each feeding the PUF with a random set of input challenges. Figure 8 shows the timing evaluation results. We observe that the total delay increases at a slope of around 5 us per challenge/response, which is considered as a short time in the context of multimedia signal processing.

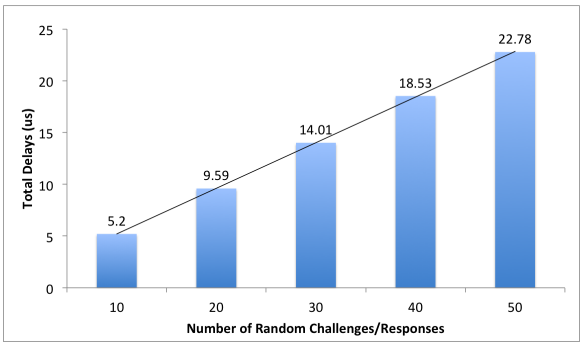

Fig. 8. PUF timing evaluation results.

Furthermore, Table I shows the resource usage and power consumption of the PUF implementation obtained from the power simulation report using the Xilinx Vivado toolchain. We observe that the PUF uses very small portion of resource on the FPGA board, i.e., 5.5% for I/O and 0.06% for LUT. Also, the dynamic power of generating the responses is less than 0.001 W, and the major portion of the power consumption is from the static power, which is 0.120 W. The low overall power consumption enables the PUF to be adopted by power constrained IoT devices for multimedia streaming and authentication.

TABLE I
POWER AND RESOURCE USAGE OF THE PUF IMPLEMENTATION ON XILINX ZEDBOARD.

Resource Type	Power (W)	Used	Utilization
LUT as Logic	< 0.001	33	0.06%
Signals	< 0.001	2	—
I/O	< 0.001	11	5.5%
Static Power	0.120	—	—
Total Power	0.121	—	—

V. CONCLUSION

We have developed a two-way real time multimedia authentication mechanism using hardware-based approach, namely physical unclonable functions (PUFs). We presented how PUF could accomplish both device and multimedia content authentications. Also, we implemented the PUF approach on real hardware (Xilinx FPGA), and our experimental results show that PUF ensures strong security while maintaining a low overhead in terms of propagation delay, resource usages, and power consumption. To the best of our knowledge, this is the first use of hardware security primitives to address real time multimedia authentication problems.

REFERENCES

[1] M. Hefeeda, K. Mokhtarian, Authentication Schemes for Multimedia Streams: Quantitative Analysis and Comparison. ACM Trans. Multimedia Comput. Commun. Appl. 6, 1, Article 6, 24 pages, 2010.
[2] R. Gennaro, P. Rohatgi, How to Sign Digital Streams, CRYPTO, pp. 180-197, 1997.
[3] P. Golle, N. Modadugu, Authenticating Streamed Data in the Presence of Random Packet Loss, Network and Distributed Systems Security Symposium (NDSS). pp.13-22, 2001.
[4] Z. Zhang, Q. Sun, W. Wong, A Proposal of Butterfly-Graph based Stream Authentication over Lossy Networks. IEEE International Conference on Multimedia and Expo (ICME), pp. 784-787, 2005.
[5] C. Wong, S. LAM, Digital Signatures for Flows and Multicasts. IEEE/ACM Transactions on Networks, Vol 7, No. 4, pp. 502-513, 1999.
[6] R. Wang, Y. Shoshitaishvili, C. Kruegel, and G. Vigna. Steal this movie: Automatically bypassing DRM Protection in Streaming Media Services. In USENIX Security Symposium, 2013.
[7] Rukhin et al., A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST, 2010, available online: http://csrc.nist.gov/groups/ST/toolkit/rng/documents/SP800-22rev1a.pdf
[8] B. Gassend, D. Clarke, M. Van Dijk, S. Devadas, Silicon Physical Random Functions,' ACM Conference on Computer and Communications Security (CCS), pp. 148–160, 2002.
[9] B. Gassend, D. Clarke, M. Van Dijk, S. Devadas, Delay-based Circuit Authentication and Applications, iACM symposium on Applied computing, pp. 294–301, 2003.
[10] G. E. Suh and S. Devadas, Physical Unclonable Functions for Device Authentication and Secret Key Generation, Design Automation Conference (DAC), pp. 9–14, 2007.
[11] Ivan Jimenez, PUF Implementation, available online: https://www.google.com/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=ivan%20jimenez%20puf%20report